Journal of the Eastern Asia Society for Transportation Studies, Vol. 6, pp. 1426 - 1438, 2005

A SHORTEST PATH ALGORITHM FOR REAL ROAD NETWORK BASED ON PATH OVERLAP

Yongtaek LIM
Assistant Professor
Division of Transportation and
Logistics System Engineering
Yosu National University
San 96-1, Dundeok-dong, Yosu city
Chunnam, 550-749, KOREA
Fax:+82-061-659-3340
E-mail:limyt@yosu.ac.kr

Hyunmyung KIM
PhD Student
Department of Civil Engineering
Institute of Transportation Studies
University of California, Irvine
USA
Fax: (949) 824-8385
E-mail: hyunmyuk@uci.edu

Abstract : Existing k-shortest path algorithms has some weaknesses such as path similarity among determined paths and network expansion for describing turn prohibitions. Path similarity represents that many of the alternative paths derived from the k-shortest path algorithm are likely to share a lots of links, so they could not represent heterogeneity. The turning restrictions popularly adopted in real road may violate Bellman's principle of optimality in searching shortest path, so network expansion technique is widely used to avoid such difficulty. But, this method needs additional works to expand the network.
This paper proposes a link-based shortest path algorithm to generate dissimilar paths for the travel information in real road network where exists turn prohibitions. The main merit of proposed model is to provide efficient alternative paths under consideration of overlaps among paths to alleviate the path similarity. Another merit is that it does not require extra nodes and links for expanding the network. Thus it is possible to save the time of network modification and of computer running. The algorithm is tested with some example networks and then will be expanded to a dynamic case.

Key Words : dissimilar path, path overlap, turn prohibition, link-based algorithm, k-shortest path

1. INTRODUCTION

A shortest path problem is for finding a path with minimum travel cost from one or more origins to one or more destinations through a connected network. It is an important issue because of its wide range of applications in transportations. In some applications, it is also beneficial to know the second or third shortest paths between two nodes. For instance, in order to improve the effectiveness of travel information provision, there is a need to provide some rational alternative paths for road users driving in real road network. To meet it, k shortest path algorithms have been used in general. Yen (1971) first proposed k shortest path searching method, which could generate several additional paths by deleting node on the shortest path. Since Yen, Several k-shortest algorithms have been suggested. Although k shortest path algorithm can provide several alternative paths, it has inherent limit of heavy overlapping among derived paths, which may lead to wrong travel information to the users. This is that significant proportion of links on the first path is overlapped by second and third path calculated from the method, so the drivers on those links may suffer severe congestions if they follow the travel information. This is the same problem of IIA (Independence of

Irrelevant Alternatives) in logit-based stochastic assignment.

On the other hand existing shortest path algorithms such as Dijkstra, Moore build the optimal path based on the node they reach. However, in the case of considering the network consisting of several turn prohibitions such as restricting left-turn, which are popularly adopted in real world network, it makes difficult for the traditional network optimization technique to deal with. Banned and penalized turns may be not described appropriately for in the standard node/link method of network definition with intersections represented by nodes only. The reason is that Bellman`s 'Principle of Optimality` does not hold in that case. Several approaches have been proposed for solving the problem. Among all methods currently available, the most widely used method is network expansion for describing turn penalties, adding extra nodes and extra links to original network and modify the network to be easily implemented the conventional shortest path algorithms. The principal advantage of this method is that it can describe the turn prohibitions perfectly. The method, however, has limitation of expanding network as the size of network increases, so this method could not apply to large networks as well as dynamic case due to its overwhelming additional works.

This paper proposes a link-based shortest path algorithm for the travel information in real road network where exists turn prohibitions. When penalized turns are dealt with without explicitly expanding the network, node-based existing shortest path algorithm is inappropriate, because the Bellman's optimality condition has the property that no node may be approached from the origin for less cost than the chosen preceding node which is also reached by a shortest cost path. But link-based shortest path algorithm makes possible to reflect all turns effectively because it holds Bellman's optimal condition in searching step.
The main merit of proposed model is to provide efficient alternative paths for route guidance under consideration of overlaps among paths. The algorithm builds new path based on both the degree of overlapping between each path and travel cost, and stops building when the degree of overlapping ratio exceeds its criterion. Because proposed algorithm generates the shortest path based on the link-end cost instead of node cost and constructs path between origin and destination by link connection, the network expansion does not require. Thus it is possible to save the time of network modification and of computer running. The algorithm is tested with some example networks and then will be expanded to dynamic case.
This paper has been organized as follows. In the next section, two problems of existing shortest path algorithms are given. A detailed description of the proposed algorithm is defined in section 3, and the comparison with conventional algorithm and performance of the algorithm are illustrated in section 4 with some numerical examples. Finally, conclusions are drawn in section 5.

2. TWO PROBLEMS IN REAL ROAD NETWORK FOR FINDING SHORTEST PATHS

2.1 K shortest paths and similarity

In single objective route-planning problems, it may be adequate to choose a single best path from an origin to a destination. In contrast, there may be some cases that several paths are required for specific purposes such as route guidance, road construction and hazardous

material transportation. It is possible to accomplish this via a k-shortest path algorithm by selecting a sufficiently large k. A straightforward way to define a set of k-paths is to successively select the 1^{st}, 2^{nd}, 3^{rd}…, k^{-th} shortest path. In some applications, it is also beneficial to know the second or third shortest paths between two nodes. There are a number of algorithms to find the paths classified by two categories such as simple paths without repeated nodes and arcs, and as looping paths with repeated node and arcs. (Yen, 1971; Shier,1979). However, many of these alternative paths derived from the k-shortest path algorithm are likely to share a large number of links. It represents spatial similarity between generated paths and may not be representative for the heterogeneity that can be found in the set of all paths. In order to obtain a path set that is more representative for the variety of choice that are available, overlapping alternatives should be excluded. Heuristic alternatives to the k-shortest paths method are based on link elimination and link penalty rules. Both methods consist of modifying the network after identifying the shortest path. A lot of approaches have been presented regarding this problem. Barra et al. (1993) proposed a link penalty method such that the network is modified by increasing the cost of all links on the shortest path. After modifying the network a new shortest path is computed according to the increased costs and the process continues until no more paths are required or no more paths can be determined. Scott et al.(1997) proposed an approach that would find the best k-similar paths that have at most k links in common with the shortest path. Akgun et al.(2000) determined a dissimilar path set by measuring the spatial dissimilarity between any two paths in the past set. The dissimilar paths are calculated by choosing some paths from the paths set in a way that the minimum of distance between any two paths is maximized.

Although we adopt the link penalty method of Barra et al.(1993), the method proposed in this paper is somewhat different from that of Barra et al. in that the alternative paths are found under the maximum degree of overlap among paths determined previously. The merit of this method is that it allows path cost and path overlaps simultaneously for searching the k-paths.

2.2 Turn prohibitions

On the other hand, some kinds of turning restrictions such as turn prohibitions, P-turn or U-turn which are popularly adopted in real world network analysis, make difficult for the traditional network optimization technique to deal with. Banned and penalized turns are not described appropriately for in the standard node/link method of network definition with intersections represented by nodes only. Caldwell(1961) proposed a method for incorporating them in the basic road network by transforming it into a much larger network in which each node represents a link in the original network and each link is a permitted turning movement. This approach adopted on a network-wide basis as in the TRRL method of network definition, is however, not suitable for networks which are already large when defined by the standard node/link method.

It is possible to handle banned or penalized turns without explicitly expanding the network; If the turn from link(i,j) into link(j,k), ie. turn i→j→k, is penalized, the network data table for link(j,k) is extended to include node i and the turning penalty. Data for any number of penalized turns into link(j,k), or any other link, may be stored in this way at the expense of increased run time spent searching for and taking account of the relevant penalties.

When penalized turns are dealt with without explicitly expanding the network, tree-building algorithm is inappropriate, because of not allowing turning penalties. Existing vine-based algorithms, however, have limitations to find shortest path in some cases of that P-urn, U-turn,

turn prohibitions are included. In real road network P-turn and U-turn are introduced frequently in order to prevent left-turn for the purpose of reducing the number of signal phases at intersection. In such case conventional vine algorithm has the difficulty in searching optimal path. The reason also comes from the fact that optimality condition is not valid any more. Bellman's optimality condition has the property that no node may be approached from the origin for less cost than the chosen preceding node which is also reached by a shortest cost path. However, introduction of turn delay and bans means that it is sometimes necessary, in practice to violate this condition. The 'P-turn' or 'U-turn' shown in Figure 1 provides common examples.

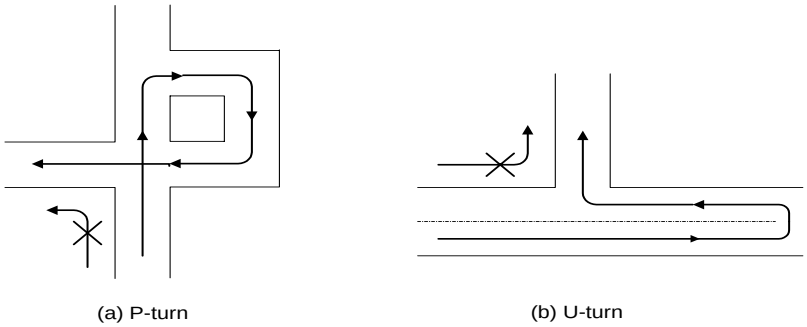


Figure1. Examples for P-turn and U-turn

The solution to model the situations correctly is to build trees in a form known as vines. Vines have the property that a node may occur more than once in a path. Vines are built by expanding the network so that each turning movement at a node is represented as an arc. The vine-based algorithm, however, also has the limitation of finding shortest path in the case of considering successive banned turns. Kim (1998) showed that in some cases vine-based method could not search optimal path. A case is shown in Figure 2. Two successive banned left-turns are existed on the network which has one origin-destination pair from node r to node s. The banned left-turns are represented with relatively big turning penalties. The number of links and link travel times are on the link in the Figure, the values in parenthesis are link travel times. In this case, conventional vine-based algorithms may lead to wrong result; not finding optimal path. The reason comes from that the vine algorithms, satisfying the optimality condition, restrict the feasible searching nodes.

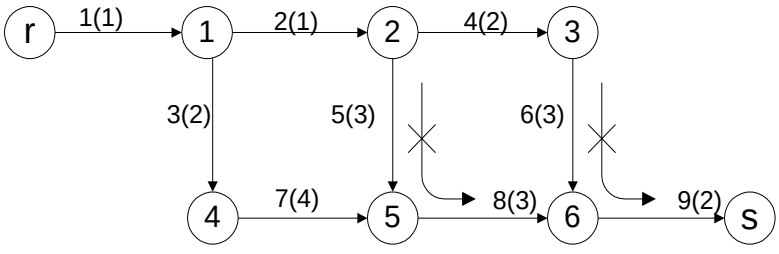


Figure 2. Example network by Kim (1998)

To overcome the limitations above-mentioned, in this paper we adopt a link-based shortest path algorithm, which was originated by Potts and Oliver (1972). This method does not

require the network expansion, thus enable to save the time of network modification and of computer running. The algorithm builds the shortest path based on the link-end cost instead of node cost. Thus, it makes possible to reflect all turns such as left-turn, right-turn, U-turn, P-turn and turn prohibition effectively when we search for the shortest path in real world.

3. DESCRIPTION OF THE ALGORITHM

3.1 Link-based shortest path algorithm

Let a network consist of a set of nodes and a set of links connecting the nodes. The nodes are also referred to as vertices or points. The links are also called arcs, edges, and branch. A network can be represented by the notation G=(N,A) , where N is the set of nodes and A the set of links of the network G . Let LC(i,j) be the nonnegative link cost required to travel from node i to node j and LEC(o,i) be the link end cost, or minimum path cost from origin to node i through link(o,i) which refer to the directed link leading from node o to node i .TP[link(o,i),link(i,j)] is the turn penalty which implies the additional cost at node i between from link(o,i) to link(i,j) when turning prohibitions exist as shown in Figure 3. With these notations we can define the link-based shortest path optimality condition with turn penalty as follows.

$$LEC(o,i)+TP[link(o,i),link(i,j)]+LC(i,j)\leq LEC(i,j)\ , \qquad \text{FORALL } o,i,j \in N \qquad (1)$$

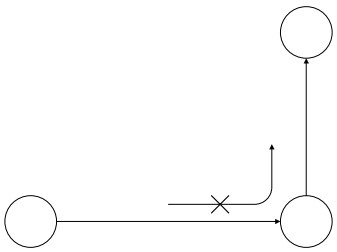


Figure 3. Basic concept of the link-based shortest path algorithm

The optimality condition in equation (1) has the unique solution, because we can easily take over the optimality theorem already proved for node-based case by simply replacing link costs by the sum of link costs and turn penalty, TP. The optimality theorem for node-based searching method is explained fully in Potts & Oliver(1972). In the paper, instead of preceding nodes in conventional shortest path algorithms, preceding links are used to memorize the track of shortest path. The preceding link from node i to node j , PL(i,j) , of Figure 3 is defined as

$$PL(i,j)=link(o,i) \qquad (2)$$

ie. PL(i,j) is the link immediately before link(i,j) on the shortest path.

Based on the equation (1) and (2), the steps of link-based shortest path algorithm are as lists. Let R be the set of all labeled links, Rºset of unlabelled links and O the set of all connected nodes with origin.

[Step 1] Label the link(h,i) , connecting origin node h with node i , $\in$ O

enter link(h,i) into set R , i.e. R= { link(h,i) }
set LEC(h,i)=LC(h,i) and LEC(h,j)=INF $\forall j \neq i$
PL(h,i)= Φ

[Step 2] Find an unlabelled link
If LEC(i,j)+TP[link(i,j),link(j,k)]+LC(j,k)≤LEC(j,k)
Then, LEC(j,k)=LEC(i,j)+TP[link(i,j),link(j,k)]+LC(j,k)
PL(j,k)=link(i,j)

[Step 3] Label the link(i,j)
Add the link(i,j) to the set R , and delete it from the set R^o

[Step 4] If R^o= Φstop, otherwise go to [Step 2].

The main advantage of the algorithm described above is that it is easy to code and possible to consider the turn restrictions on the way of shortest path searching. The algorithm is very similar to the conventional tree building algorithms. The only difference is that it is link-based searching, not node-based. Thus with a little modification, the conventional algorithms can be used.

Now consider computational experience of the algorithm. A time taken by an algorithm, which is also called the running time of the algorithm, depends on the topology of the network. A time requirement for an algorithm is a function of the problem size and specifies the largest amount of time needed by the algorithm to solve any problem instance of a given size. In other words, the time requirement measures the rate of growth in solution time as the problem size increases. To compare the efficiency of the shortest path algorithms, we use the notation,O(n) , with the network parameter n . O(n) means the maximum time requirement to solve the problem and the complexity of the algorithm (see Ahuja,et.al.,(1993) in more detailed).
With the notation O(n) , we can compare the efficiency of the algorithms. The conventional tree-based algorithm has the time requirement of $O(n^2)$ with parameter n, number of nodes. The vine-based algorithm, however, has $O(n^3)$. On the other hand, if the example network composed of n node and *l* links has bi-directional rectangular topology, there exists a relationship between n and *l*; $l=4[\, n-\sqrt{n}\,]$. Because the link-based algorithm presented in the paper is based on links, not nodes, it has $O(l^2=[\, 4(n-\sqrt{n})\,]^2)$ of time requirement, which is greater than tree-based algorithm but less than the vine-based one. This implies that the algorithm dose not requires too much running time to find the path, compared with the existing algorithms. In special when it applies to a network with turn prohibitions or to an integrated network with several transportation modes on it, less computing time is required because it does not need to expand the network.

3.2 Formulation of a shortest path algorithm with path overlap

The shortest path algorithm proposed in this paper generates several dissimilar paths by adopting both the link-based shortest path technique of section 3.1 and the link penalty method for finding dissimilar ones. The algorithm builds new path based on the degree of overlapping between each path and travel cost, and stops building when the degree of overlapping ratio exceeds its criterion. First let some new variables be denoted.

l_n^{rs} length of n-th path for origin-destination pair *rs*

p_n^{rs} n-th path for origin-destination pair rs

P^{rs} path set for origin-destination pair rs; $P^{rs}=\{ p_n^{rs}, l_n^{rs} \}$

$ol_{k/n}^{rs}$ overlapping length of n-th path to the length of k-th path for origin-destination pair rs, $\forall P_k^{rs} \in P^{rs}$, $k = n-1, n-2, ..., 1$

$Op_{k/n}^{rs}$ degree of overlap between the length of n-th path and the length of k-th path for origin-destination pair rs; $Op_{k/n}^{rs} = \frac{ol_{k/n}^{rs}}{l_n^{rs}}, \forall P_k^{rs} \in P^{rs}, k = n-1, n-2, ..., 1$

Op maximum degree of overlap

Oz link penalty; $Oz = [\frac{1}{Op}]^{\alpha}$

α positive parameter

With these variables, we can describe the shortest path algorithm as,

[Step0] Initialization

set Op and $n = 1$

[Step1] With the link-based shortest path algorithm, find the first shortest path

and add l_n^{rs}, p_n^{rs} to path set; $P^{rs} = \{P_n^{rs}, l_n^{rs}\}$

[Step2] Link cost update (Link penalty)

new link costs of the n-th path = link costs of the n-th path * Oz

[Step3] Path search and Overlapping ratio calculate

(3.1) $n = n+1$

(3.2) with the link-based shortest path algorithm, find the n-th shortest path; $\{ P_n^{rs}, l_n^{rs} \}$

(3.3) degree of overlap : $Op_{k/n}^{rs} = \frac{ol_{k/n}^{rs}}{l_n^{rs}}, \forall P_k^{rs} \in P^{rs}, k = n-1, n-2, ..., 1$

[Step4] Convergence test

if $Op_{k/n}^{rs} > Op$, then stop

otherwise, add $\{ P_n^{rs}, l_n^{rs} \}$ to P^{rs} and proceed [Step2]

In Step 2 of the algorithm, the costs of links belonging to the shortest path are modified by multiplying link penalty of Oz, which is a function of degree of path overlap(Op). Thus only one path is generated if all paths are allowed of overlapping each other ($Op = 1.0$), while more than one path are generated if the value of Op is below 1.0.

4. NUMERICAL EXAMPLES

Some numerical examples are presented to illustrate the performance of the algorithm; the first is for verifying whether the proposed link-based algorithm in section 3.1 obtain optimal path or not, and the others for comparing and assessing the proposed algorithm.

4.1 Successive turn-prohibited network

The first example network is the one proposed by Kim (1998) as shown in Figure 2, this network includes 9 directed links, 8 nodes and 2 banned left-turns which impose turn penalties. The number of links and link travel times are on the link in Figure 2, values in parenthesis are link travel time. Turning penalties are given in Table 1.

Table 1. Turn penalties of the first example

turn	penalty
②→⑤→⑥	900
③→⑥→ⓢ	900

Table 2 describes in detail the procedure for searching shortest path from origin r to destination s. Figures in the table show that we obtain exact optimal path and its cost of 12. From the result, we find that link-based algorithm can easily search the shortest path under considering banned turns, which may not be accounted in conventional vine-based algorithms

Table 2. Results of the first example

Number of link	i	j	LC(i,j)	LEC(i,j)	PL(i,j)	labelled link set(R)	link set for path
1	r	1	1	1	0	1	1
2	1	2	1	2 (1+1)	1	1,2	1,2
3	1	4	2	3 (1+2)	1	1,2,3	1,3
4	2	3	2	4 (1+1+2)	2	1,2,3,4	1,2,4
5	2	5	3	5 (1+1+3)	2	1,2,3,4,5	1,2,5
6	3	6	3	7 (1+1+2+3)	4	1,2,3,4,5,6	1,2,4,6
7	4	5	4	7 (1+2+4)	3	1,2,3,4,5,6,7	1,3,7
8	5	6	3	908 (1+1+3+900+3)	5	1,2,3,4,5,6,7,8	1,2,5,8
	5	6	3	10 (1+2+4+3)	7	1,2,3,4,5,6,7,8	1,3,7,8
9	6	s	2	909 (1+1+2+3+900+2)	6	1,2,3,4,5,6,7,8,9	1,2,4,6,9
	6	s	2	12 (1+2+4+3+2)	8	1,2,3,4,5,6,7,8,9	1,3,7,8,9
link set for shortest path				1 – 3 - 7 - 8 – 9	shortest path cost (r → s)		12

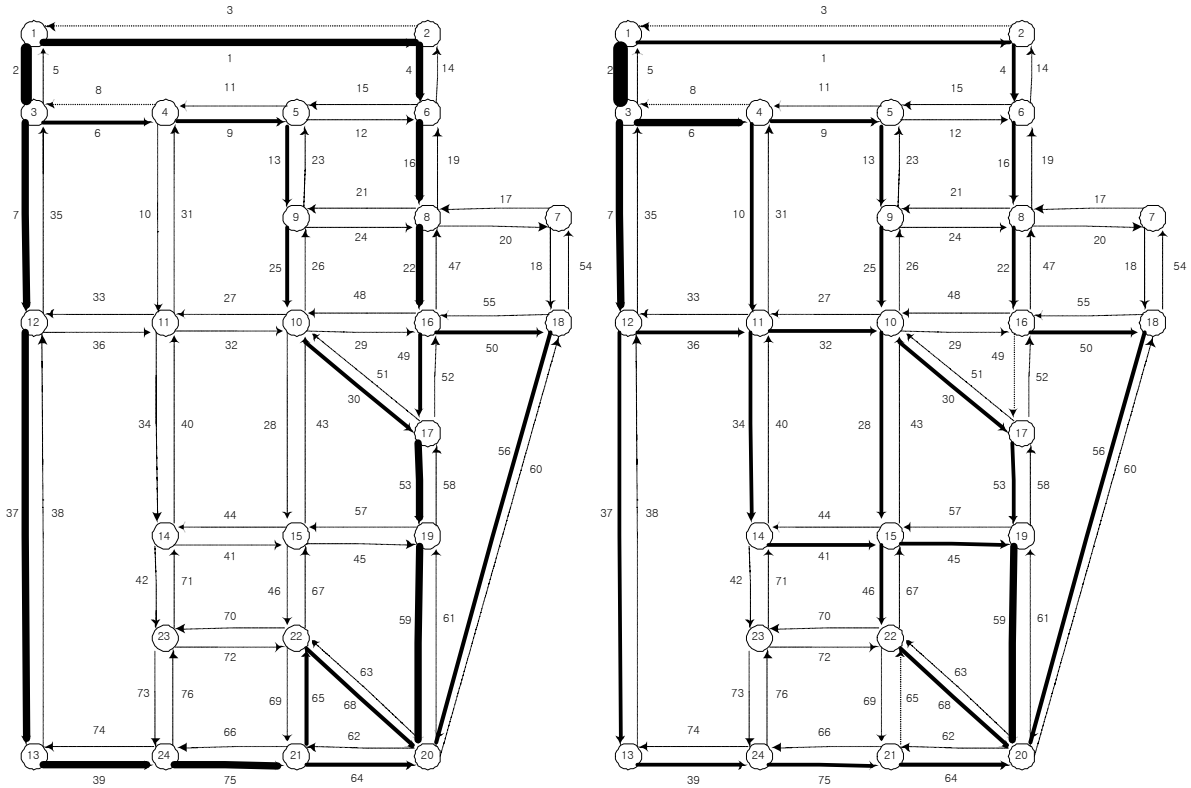
4.2 Sioux Falls network

The second test is performed with the Sioux Falls network, which has 24 nodes and 76 links. This paper just considers only one origin-destination pairs from node 1 to node 20 for clear comparison between the method proposed in this paper and existing k-shortest path method. The specifications for the network are shown in [Appendix A].

Figure 4 shows explicit difference between the k-shortest path algorithm and the proposed algorithm. Both algorithms generate 5 different paths. The k-shortest path algorithm finds very similar paths as we expect, while the proposed algorithm produce relatively dissimilar paths because it considers the degree of path overlap in searching steps. So the paths derived

from the k-shortest algorithm are concentrated, but the paths from the proposed method are scattered over the whole network.

Table 3 summarizes the values of path cost and links on the path. The first two paths have the same values of path costs for all method, but the rest three path costs of the proposed algorithm are higher than those of k-shortest path one because it generates the paths in order of degree of path overlap, not in order of path costs. From the point of view in travel information for drivers, the similar paths may occur to vehicle concentration to some specific routes, which leads to oversaturation of traffics. In such environments dissimilar paths generated from the proposed method may be more efficient to spread vehicles over the network.



(a) Paths from k-shortest path method (b) Paths from the proposed method

Figure 4. Comparison between the methods

Table 3. Generated paths and each path costs

Paths	k-shortest path method		The proposed method	
	Link set	Path cost	Link set	Path cost
1st path	2-7-37-39-75-64	1260	2-7-37-39-75-64	1260
2nd path	1-4-16-22-50-56	1320	1-4-16-22-50-56	1320
3rd path	1-4-16-22-49-53-59	1320	2-6-9-13-25-30-53-59	1440
4th path	2-6-9-13-25-30-53-59	1440	2-7-36-34-41-46-68	1500
5th path	2-7-37-39-75-65-68	1440	2-6-10-32-28-45-59	1680

Table 4, Table 5 and Figure 5 show some results of the proposed method when left-turn

prohibitions at node 11, node 14 and node 23 exist, with the value of 50% of path overlap. Table 4 demonstrates that five dissimilar paths are also generated, and that last 2 paths are different from those of paths in Table 3 due to turning restrictions. We note here that the 5th path has U-turn movement at node 24 because of left-turn prohibition at node 23. Path overlapping ratios among paths calculated from the method are shown in Table 5. Each figure in the table is below 0.5 except for diagonal elements, since the maximum degree of path overlap sets 50%.

Figure 5 depicts the number of paths produced from the method with varying the degree of path overlap (Op) from 0.1 to 1.0. There are maximum 5 paths through Op=0.4 to 0.7 and only one path when Op=1.0 that all paths are possible to be overlapped, as we expect. Figure 6 shows the number of paths with varying the value of parameter (α) in the link penalty (Oz).

Table 4. Generated paths and path cost (when $Op = 0.5$ and α=1.8)

Paths	Link set	Path cost	
1st path	2-7-37-39-75-64	1260	
2nd path	1-4-16-22-50-56	1320	
3rd path	2-6-9-12-25-30-53-59	1440	
4th path	2-7-36-32-28-46-68	1560	
5th path	2-6-10-34-42-73-76-72-68	1860	U-turn at node 24

Table 5. Matrix of degree of path overlaps (when $Op = 0.5$ and α=1.8)

Paths	1	2	3	4	5
1	1.000	0.000	0.167	0.333	0.167
2	0.000	1.000	0.000	0.000	0.000
3	0.125	0.000	1.000	0.125	0.250
4	0.286	0.000	0.143	1.000	0.286
5	0.111	0.000	0.222	0.222	1.000

Figure 5. Number of paths with varying Op

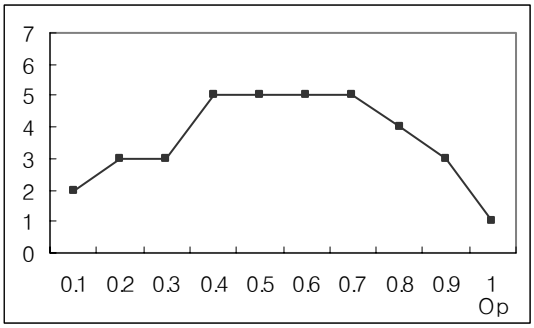
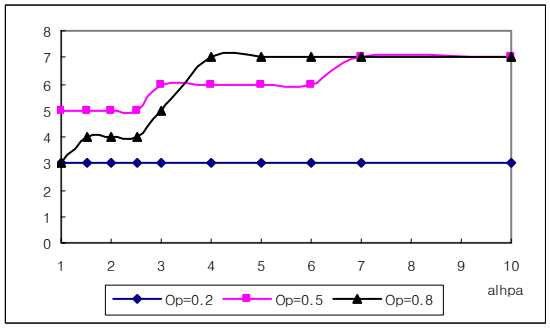


Figure 6. Number of paths with varying α



4.3 Expansion to dynamic case

The proposed algorithm here expands to a dynamic one, in which each link cost is variable as time elapses. This dynamic shortest path algorithm adopts the method of Jayakrishnan et al. (1995), who presented a link-based time-expanded network technique corresponding to static

network. Step 1 and Setp 3 in section 3.2 are replaced with this dynamic algorithm for generating dynamic shortest paths. A numerical result applying to Sioux Fall network is given in Appendix B, where each link cost is generated at random with mean and variance.

5. CONCLUSION

In this paper, a new shortest path algorithm capable of considering path overlap and turn prohibitions was proposed and tested, which provided efficient dissimilar alternative paths. This algorithm builds paths based on the degree of overlapping ratio and does not need network expansion to find the shortest path because it searches the paths with link-end cost, not node cost. After verifications of the algorithm with some examples, we also expanded it to dynamic one and tested. From the results of some numerical examples, we may conclude that the algorithm is more efficient for finding several dissimilar shortest paths than others.
The algorithm proposed in the paper may be applied to a variety of network analyses. In first, it could be adopted in the field of travel information such as providing diverse routes for drivers. Secondly, the algorithm generates dissimilar paths so that it could alleviate the problem of IIA (Independence of Irrelevant Alternatives) in logit-type stochastic assignment. Lastly, it can be also used for solving easily multi-mode traffic assignment, which should consider transfer behaviors of users.

REFERENCES

a) Books and Books chapters

Ahuja,R.K., T.L. Magnanti, J.B. Orlin(1993) **Network Flows**: Theory, Algorithms and Applications, Prentice Hall
Bellman,R.E.(1957) **Dynamic programming**, Princeton University Press, Princeton
Potts,R.B., R.M. Oliver(1972) **Flows in transportation networks**, Academic press

b) Journal papers

Akgun,A.,Erkut,E.,Batta,R. (2000) On finding dissimilar paths, European **Journal of Operational Research 121**, 232-246
Caldwell,T.(1961) On finding minimum routes in a network with turn penalties, **Commum, ACM4**, 107-108
Dijkstra, E. W.(1959) “A note on two problems in connection with graphs”, **Numer.Math.1**, 269-271
Jayakrishnan,R.,W.K.Tsai, A.Chen (1995) A dynamic traffic assignment model with traffic-flow relationships, **Transportation Research (C), Vol3**, 51-72
Kim,I.(1998) Development of a modified vine building shortest path algorithm for ATIS, **Journal of Korean society of transportation, Vol.16, No.2**, 157-167
Shier, R.D.(1979) On algorithms from finding the k shortest paths in a network, **Networks, V ol.9**, 195-214
Yen J.Y.(1971) Finding the K shortest loopless paths in a network, **Management Science, Vo l 17, No. 11**, 712-716

c) Papers presented to conferences

Barra, T. et al.(1993) Multidimensional Path Search and Assignment, **21st PTRC Summmer Annual Conference**

Scott,K., Pabon-Jimenez,G.,Bernstein,D. (1997) Finding alternatives to the best path, **Presented at the 76th Annual Meeting of the Transportation Research Board**, Washington,DC

[Appendix A] Specifications of Sioux Fall network

Link	From node	To node	Link cost	Link	From node	To node	Link cost
1	1	2	360	39	13	24	120
2	1	3	120	40	14	11	240
3	2	1	360	41	14	15	240
4	2	6	120	42	14	23	180
5	3	1	120	43	15	10	240
6	3	4	300	44	15	14	240
7	3	12	300	45	15	19	180
8	4	3	300	46	15	22	180
9	4	5	180	47	16	8	120
10	4	11	300	48	16	10	180
11	5	4	180	49	16	17	120
12	5	6	180	50	16	18	180
13	5	9	120	51	17	10	180
14	6	2	120	52	17	16	120
15	6	5	180	53	17	19	180
16	6	8	180	54	18	7	300
17	7	8	180	55	18	16	180
18	7	18	300	56	18	20	360
19	8	6	180	57	19	15	180
20	8	7	180	58	19	17	180
21	8	8	180	59	19	20	240
22	8	16	120	60	20	18	360
23	9	5	120	61	20	19	240
24	9	8	180	62	20	21	180
25	9	10	120	63	20	22	240
26	10	9	120	64	21	20	180
27	10	11	300	65	21	22	120
28	10	15	240	66	21	24	180
29	10	16	180	67	22	15	180
30	10	17	180	68	22	20	240
31	11	4	300	69	22	21	120
32	11	10	300	70	22	23	240
33	11	12	180	71	23	14	180
34	11	14	240	72	23	22	240
35	12	3	300	73	23	24	120
36	12	11	180	74	24	13	120
37	12	13	360	75	24	21	180
38	13	12	360	76	24	23	120

[Appendix B] Generated dynamic paths and cost (when $Op = 0.5$ and α =1.8)

Paths	Time intervals	Link set	Path cost
1st path	1	2	1247
	3	7	
	8	37	
	13	39	
	15	75	
	18	64	
2nd path	1	1	1258
	6	4	
	8	16	
	11	22	
	13	50	
	16	56	
3rd path	1	1	1493
	6	4	
	8	15	
	11	13	
	13	25	
	15	30	
	18	53	
	21	59	
4th path	1	2	1647
	3	6	
	8	9	
	13	12	
	17	16	
	20	22	
	22	50	
	26	56	
5th path	1	2	1666
	3	6	
	8	10	
	13	34	
	17	42	
	20	73	
	22	75	
	26	64	