Journal of the Eastern Asia Society for Transportation Studies, Vol. 6, pp. 2688 - 2701, 2005

CONSTRUCTION OF REALISTIC ROUTE CHOICE SETS USING MULTI-LABEL VINE-BUILDING FOR REAL-TIME TRAVELER INFORMATION PROVISION

Ikki KIM
Department of Transportation Engineering
Hanyang University
1271 Sa-1-dong, Sangrok-gu, Ansan,
Gyunggi-do, South Korea 425-791
Fax: +82-31-406-6290
E-mail: ikkikim@hanyang.ac.kr

Jeong Whon YU
The Korea Transport Institute
2311 Daewha-dong, Ilsan-gu, Goyang-si,
Gyeonggi-do, South Korea 411-701
Fax: +82-31-910-3229
Email: jeong@koti.re.kr

Keemin SOHN
Seoul Development Institute
391 Seocho-dong, Seocho-ku, Seoul, South
Korea 137-071
Fax: +82-2-2149-1120
Email: kmsohn@sdi.re.kr

ABSTRACT

This paper proposes an algorithm to identify multiple alternative routes with the multi-label vine-building shortest path algorithm while incorporating turning penalties and prohibitions. The used shortest path algorithm keeps several node labels through the vine search and improves the backward tracing method by excluding all illogical movements at intersections. It also adopts the concept of the rational upper boundary in evaluating the realism of alterative routes considered. The multiple node labels used in the algorithm can foster more effectiveness of the routing advisories and information through analyzing alternative route characteristics in various ways. A simple network is used to analyze the applicability of the proposed algorithm to generate realistic route choice sets. The results show that the proposed algorithm is capable of incorporating various driver behavior classes. The reasonable multiple paths could be effectively used especially in application of ATIS operation which requires sophisticated alternative paths for each traveler.

Key Words: alternative path set, shortest path, ATIS

1. INTRODUCTION

A critical aspect for effectively providing real-time routing advisories and information via Advanced Traveler Information Systems (ATIS) is the correct identification of realistic route

choice sets perceived by drivers. This is because an incorrect identification of alternative routes can result in irrelevant routing advisories and information, which potentially degenerates the effectiveness of ATIS deployment. At the same time most stochastic traffic assignment (STA) literatures have also recognized the identification of multiple alternative routes as a critical issue in the real-world applications of the associated STA models. The STA models assuming alternative routes *a priori* employ several approaches to enumerate alternative routes such as the k-shortest path algorithm, heuristic approaches, simulation method, and combinations thereof. If the cost of an alterative path is not much different from that of the minimum cost path, drivers do not perceive the difference between two paths. In this context, STA can be more realistic by incorporating the variations in drivers' perception on alternative paths due to several factors such as taste heterogeneity, imperfect information, omitted independent variables, inconsistent decision over time, and observation/measurement errors. However, STA has a major weakness that the alternative paths should be given *a priori*. Past studies suggested the methods for enumerating alternative paths or K-shortest path algorithm such as link elimination approach (Azevedo et al., 1993; Bellman and Kalaba, 1968), link penalty approach (De la Barra et al., 1993; Park and Rilett, 1997) and simulation method (Sheffi and Powell, 1982).

In this study, we propose a new approach to realistically identify multiple alternative routes that drivers consider for their pre-trip and en-route routing decisions. The proposed approach finds multiple paths close to the minimum cost path without link removal and link penalty increase. It can generate all possible alternative paths directly from the multi-label vine-building shortest path algorithm (MVA) (Kim, 2001). MVA finds the minimum cost path while incorporating all turning penalties and prohibitions and can also search the shortest path involving the U-turn or P-turn, which is not possible by simply using the traditional node labeling shortest path algorithm. Hence, the path generating algorithm proposed in this study is able to find all possible alternative paths in a more realistic manner.

2. METHODOLOGY

2.1 Multi-label Vine-building Shortest Path Algorithm

MVA can be described as a modification of the algorithms developed by Kirby and Potts (1969) or Ziliaskopoulos and Mahmassani (1996). Although the basic concepts of these three algorithms are quite similar, MVA can differentiate itself from the Kirby's algorithm by employing its unique way of labeling nodes. The node-labeling approach used in MVA is computationally efficient and intuitively simple to understand because it does not require any converting procedure from node labels to link labels or vice versa. Also, it is easier to trace

path using MVA than the link-labeling approach. The algorithm is also different from the Ziliaskopoulos's algorithm in way of labeling a node.

MVA keeps several node labels represented by incoming links and uses the vine search instead of the tree search. It employs the vine-building algorithm with multiple node labels in each direction and then searches two consecutive downstream nodes simultaneously. MVA is capable of finding realistic paths by realistically considering all turns and more than two left-turn prohibitions in a row. Even though the algorithm could find correctly the shortest path including P-turn and U-turn, it will never include any loops because all impedances are positive value and the algorithm uses multiple labels for a node. If the link impedances are additive link by link, the algorithm could be applicable effectively. It also improves the backward tracing method using the final node labels obtained from the conventional vine-building algorithm and does not generate illogical movements at intersections along a path.

The steps of MVA are as follows:

STEP 0: Initialize the labels for all nodes

$C_{rr}^1 = 0$, $b_{rr}^1 = 0$, $bb_{rr}^1 = 0$, $\forall$ r

$C_{ra}^1 = t_{ra}$, $b_{ra}^1 = r$, $bb_{ra}^1 = 0$, $\forall$ $a \in A$

(A is a set of nodes directly connected to the origin node r)

$C_{ri}^d = \infty$, $b_{ri}^d = 0$, $bb_{ri}^d = 0$, $\forall i, d$ ($i \neq r, i \neq a$)

Put the origin node r and a in A into set E, the scan eligible list.

STEP 1: Choose node i from E by order, and remove it from E. Then, find the node set $J_i = \{j_i^1, j_i^2, ...\}$, which are connected to i via a direct link. If the set E is empty, then go to Step 5.

STEP 2: Choose node j from set J_i by order, and remove it from set J_i. Then, find the node set $K_{ij} = \{k_{ij}^1, k_{ij}^2, ...\}$, which are connected to j via a direct link. If the set J_i is empty, then go to Step 1.

STEP 3: Choose node k from set K_{ij} by order, and remove it from set K_{ij}. Assign a directional code d for the incoming node j at node k. If the set K_{ij} is empty, then go to

Step 2.

STEP 4: Calculate the total cost from *r* to *k* and correct node labels accordingly.

(1) Find the directional code *m* such that $i = b_{rj}^{m}$

(2) Calculate $T_{rk}^{d(j)} = C_{rj}^{m(i)} + t_{jk} + p_{ijk}$ for the path coming from *j*

(3) If $T_{rk}^{d} \prec C_{rk}^{d}$, then correct node labels such as $C_{rk}^{d} = T_{rk}^{d}$, $b_{rk}^{d} = j$, $bb_{rk}^{d} = i$.

First, put *j* into the list of set *E* if *j* is not in *E*. Second, if *k* is not in *E*, then put *k* into the list of set *E*.

(4) If $T_{rk}^{d} \succ C_{rk}^{d}$, do not correct any node labels.

(5) Go to Step 3.

STEP 5: Back-tracing the shortest path by using the final node labels

(1) Find the directional code *d* such as $C_{rs}^{d} = \min\{C_{rs}^{1},...,C_{rs}^{D}\}$ at the destination node *s*.

Then, find the predecessor node $k = b_{rs}^{d}$ and the back-back node $j = bb_{rs}^{d}$.

(2) Save node *j*, *k* and *s* in the path set *R* such as *R* = {*j*, *k*, s}.

(3) Find the directional code *d* such as $b_{rk}^{d} = j$ at node *k*.

Then, find the back-back node *i* of node *k* with the directional code *d* such as $i = bb_{rk}^{d}$.

(4) Put the node *i* into *R* such as *R*={*i*, *j*, *k*, ... , *s*}.

(5) If *i* = *r*, then stop and trace the shortest path by using the set *R*.
If *i* ≠ *r*, set *k* = *j* and *j* = *i*. Then, go to (3) in Step 5.

where

b_{rk}^{d} = predecessor node of node *k* for the path coming from the direction *d* to node *k* in a path from the origin *r*.

bb_{rk}^{d} = back-back node of node *k* (predecessor node to b_{rk}^{d}) for the path coming from the direction *d* to node *k* in a path from the origin *r*.

C_{rk}^{d} = minimum path cost from the origin node r to any node k coming from node j recorded as the direction code d.

p_{ijk} = the turn penalty of turn i-j-k, where i, j and k are from-node, turn-node and to-node, respectively.

r = origin node

s = destination node

t_{ij} = generalized link cost from node i to node j

2.2 Path Generation

Alternative paths can be identified through simply tracing back because MVA keeps multiple labels in each direction at a node. They are generated by checking three consecutive nodes (or two links), from-node, at-node and to-node. In other words, alternative paths from several from-nodes reaching to the to-node via the at-node are compared to the shortest path to the to-node with consideration of turning penalty. In path generation, the concept of the rational upper boundary is employed to decide the realism of alternative paths. The concept is similar to that of rational boundary proposed by Mahmassani and Peeta (1993). The rational upper boundary used in this study assumes that drivers do not distinguish a path from the minimum path if the cost difference between two paths is within the rational upper boundary. The rational upper boundary can have an absolute or relative value. The absolute value indicates that the boundary is irrelevant to path length while the relative value implies that the boundary can be set in proportion to the minimum cost path. If a rate is used to define the boundary, the number can be unrealistically small for short distance trips. This is the reason that a fixed small number could be applied arbitrary for reality. The specific value, absolute or rate, of the rational upper boundary could be studied through empirical studies. The steps of the proposed path generation algorithm are as follows:

Step 1: Pick a specific O-D pair from origin r to destination s

Step 2: Initialization

$flag(p)=0$ for all p

$np = 1$

$path(p,n) = 0$ for all p and n

$ppath = 1$

Step 3: Read the multi-labels obtained by MVA.

Step 4: Find possible paths reaching at the to-node *s*

(1) Pick the back-node b_{rs}^{d} which is not a part of the minimum cost path.

(2) If $C_{rs}^{d} - \min\{C_{rs}^{1},...,C_{rs}^{D}\} \le \varepsilon^{path}$,

then $np = np+1$, cost(np)= C_{rs}^{d} $path(np,1) = s$, $path(np,2) = b_{rs}^{d}$ and at(np)=2.

Otherwise, go to (4)

(3) If there is any other possible paths to reach to *s* except the minimum cost path, then, go back to (1).
Otherwise, go to (4)

(4) Read the back-node b_{rs}^{d} and back-back node bb_{rs}^{d} which are on the way of the minimum cost path, then set $path(1,1)=s$, $path(1,2)=b_{rs}^{d}$, path(1,3)= bb_{rs}^{d}, $pto = s$, $pat=b_{rs}^{d}$, $pfrom = bb_{rs}^{d}$ and m=3

Step 5: Back-trace the current path (*ppath*) reaching to node *pto via pat*.

(1) Select a BB_{rs}^{d}, which is not bb_{rs}^{d}, the predecessor node of pat.

(2) Compare the path cost from BB_{rs}^{d} to the *pto* via *pat* with the minimum path cost to reach *pto*.

$$\Delta C_{pto} = C_{r,pat}^{d(BB)} + t_{pat,pto} + p_{BB,pat,pto} - C_{r,pto}^{d(pat)}$$

$$\Delta C_{rs}^{ppath} = \cos t(ppath) + \Delta C_{pto} - C_{rs}^{d(\min)}$$

If $\Delta C_{pto} \le \varepsilon^{node}$ and $\Delta C_{rs}^{ppath} \le \varepsilon^{path}$,

Then, $np = np+1$

$cost(np) = cost(ppath) + \Delta C_{pto}$

$path(np, m) = BB_{rs}^{d}$

$path(np,i)=path(ppath,i),\ i=1, m-1$

$at(np)=m$

Otherwise, go to (3)

(3) If all BB_{rs}^{d} except *pfrom* was checked, then go to (4)

Otherwise, go back to (1).

(4) Choose $bbb_{rs}^{d(\min)}$ which is the predecessor of bb_{rs}^{d} (= *pfrom*) and is on the minimum cost path from the origin *r* to node *pat* via *pfrom,*

Then, set *m=m+1,* *path(ppath,m)=* $bbb_{rs}^{d(\min)}$, *pfw=pat,* *pat=pbw,*

pbw=path(ppath,m)

Step 6: Check whether the backward tracing for *ppath* ends at the origin *r*.

(1) If *pfrom=r,* then, *flag(ppath)=1.*
Otherwise, go back to Step 5.

(2) Select another path to be back-traced.
If there is *flag(p)=0, p=np, 2*, then choose the first *p* in inverse order. And then, set *ppath=p*
Otherwise, go to Step 7.

(3) Find the node to be traced from for the path *ppath*
m=at(ppath), pto=path(ppath, m-1), pat=path(ppath, m)

(4) Find BB_{rs}^{d}, the predecessor node of *pat,* on the way of the minimum cost path to the node *pto.* Set *pfrom =* BB_{rs}^{d}, m=m+1, and *path(ppath, m)=pfrom.*

Then go to Step 5.

Step 7: Reordering all paths which has *flag*(*p*) = 1, and save them into the set of possible alternative paths.

where

$C_{rn}^{d(m)}$ = Minimum path cost from origin r to the node n via node m

cost(*p*) = total generalized cost from origin to destination for the p^{th} path

ε^{node} = rational upper boundary for reaching to a node

ε^{path} = rational upper boundary of a path *p* from origin r to destination s

flag(*p*) = 0 if path *p* is never back-traced

= 1 if path *p* is back-traced successfully and is set as an alternative path

path(*p*, *n*) = n^{th} node inversely from the destination in the p^{th} path.

3. EXPERIMENTS

The proposed algorithms are tested using a simple network with turn penalty, turn prohibition, P-turn and U-turn as shown in Figure 1. The numbers in circles and squares denote node numbers and the generalized costs of the corresponding links, respectively. The turn penalties specified in the test network are summarized in Table 1. The final results of MVA with the multi-labels for each node are summarized in Table 2. There exist six possible paths from *r* to *s* in the test network. The results show that the proposed algorithm correctly finds the real shortest path.

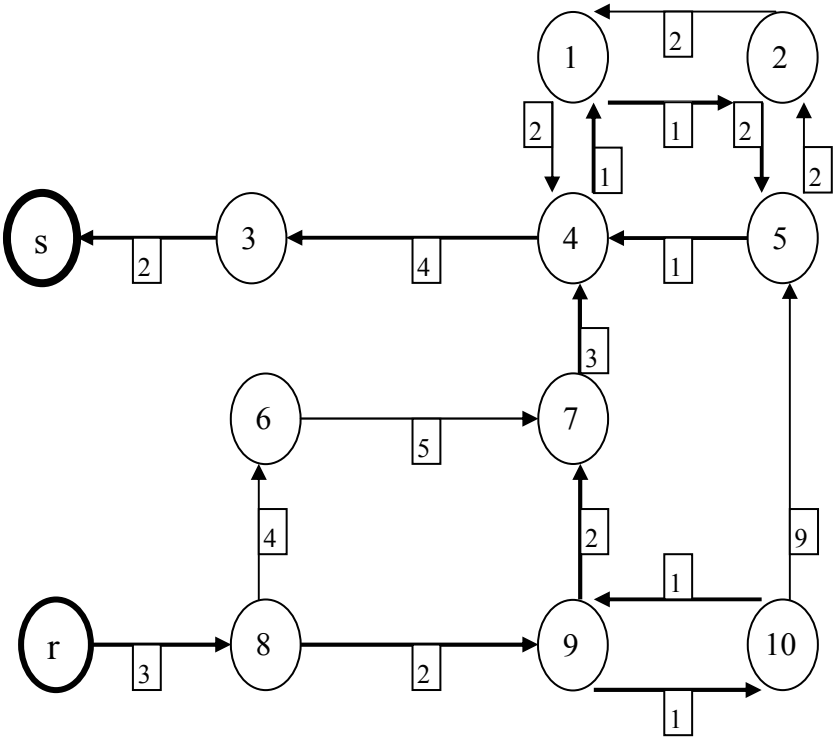

Figure 1. Test Network

Table 1. Turn Penalties and Prohibitions Specified in the Test Network

From-node	Turn-node	To-node	Turn Penalty
R	8	6	6
2	5	2	900 (U-turn prohibition)
7	4	3	6
8	9	7	900 (left-turn prohibition)
9	10	5	3
10	5	4	3

Table 2. Results of MVA

Node number	Label ($C_{rk}^d, b_{rk}^d, bb_{rk}^d$)	Node number	Label ($C_{rk}^d, b_{rk}^d, bb_{rk}^d$)
1	(13, 4, 7) (22, 2, 5)	7	(18, 6, 8) (9, 9, 10)
2	(20, 5, 10) (14, 1, 4)	8	(3, r, 0)
3	(21, 4, 5)	9	(5, 8, *r*) (7, 10, 9)
4	(12, 7, 9) (17, 5, 2) (24, 1, 2)	10	(6, 9, 8)
5	(18, 10, 9) (16, 2, 1)	*r* (origin)	(0, 0, 0)
6	(13, 8, *r*)	*s* (destination)	(23, 3, 4)
Shortest Path : *r* – 8 – 9 – 10 – 9 – 7 – 4 – 1 – 2 – 5 – 4 – 3 – *s* Shortest Path Cost : 23			

The concept of the proposed path generation algorithm can be easily understood through the simple example presented in Figure 2. The key concepts of the proposed algorithm are to select reasonable paths among merging alternative paths at an intersection and to back-trace the multiple paths until reaching the origin. Figure 2 illustrates the processes to back-trace and to decide whether merged paths at an intersection are included in a set of alternative paths considered by drivers..

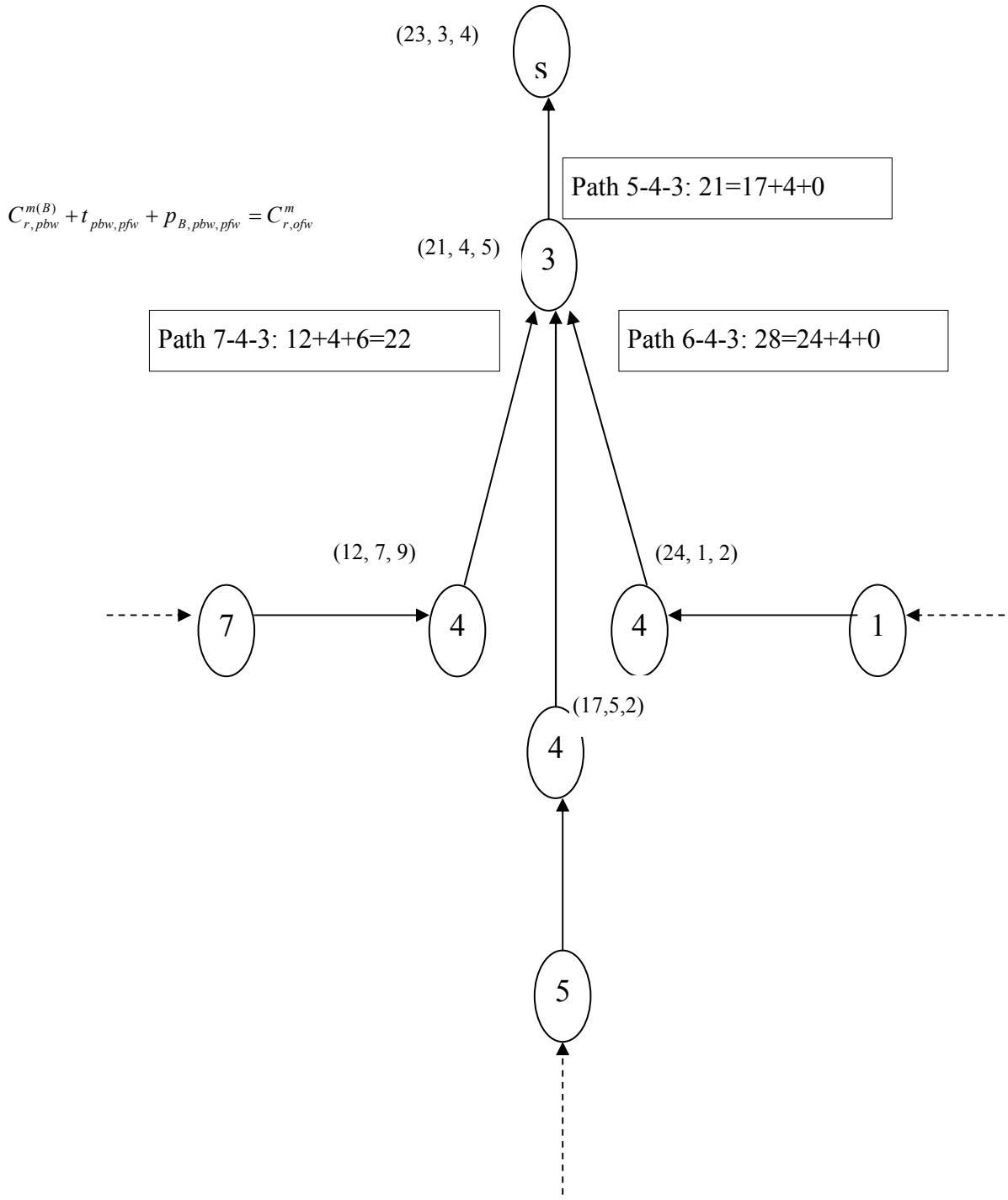

Figure 2. Back-tracing and Checking Possible Paths at an Intersection

In the path generation process, we first need to check the minimum cost path (*r*-….-5-4-3 shown in the example) to get a node (to-node of a link) like node 3 via the at-node (backward node of the link) like node 4 as shown in Figure 2. Also, other potential paths merged at the at-node (node 4) from other directions (7-4-3 and 1-4-3 in the example, from several other from-nodes) other than the minimum cost path should be checked. The proposed algorithm

compares the cost of each potential path with the minimum cost. In the above example, *cost* (22) of path 7-4-3 is compared with the minimum *cost* (21) of the path 5-4-3 to get node 3. Similarly *cost* (28) of path 1-4-3 is compared with the minimum cost. If the difference (22-21=1, or 28-21=7) is less than ε^{node}, which can be interpreted as the rational upper boundary, the path is kept in the set of temporal possible paths. If the difference is bigger than ε^{node}, the path is not considered (discarded from the set of possible paths). ε^{node} can be an absolute value or a relative value to the minimum cost. If ε^{node} = 5 which is set as an absolute value, then path 1-4-3 is not selected as a possible path, while path 7-4-3 can be accepted. In case that a relative value is used and the rate to the minimum cost is 0.2 or 20% ($\varepsilon^{node}=0.2\times21=4.2$), path 1-4-3 is kept as a possible path while path 7-4-3 is not. The path (7-4-3) kept as a possible path is back-traced from node 4 by searching the minimum cost path leading to node 4 via node 7. The back-tracing process to find the minimum cost path for node 3 via node 4 is continued until the origin node is reached. After completing the search for a path, the proposed algorithm picks up a possible path most recently branched from an intersection. The algorithm continues until no incomplete path exists to be back-traced further. Finally, if a path has any cycle or the difference between its cost and the minimum path cost is more than ε^{path}, it will be removed from the set of the possible paths.

Table 3. Possible Paths with a Large Rational Upper Boundary

Possible Path	Sequential Nodes in Possible Paths (Origin)													Path Cost
Path 1	*r*	8	9	10	9	7	4	1	2	5	4	3	*s*	23
Path 2	*r*	8	9	10	9	7	4	3	*s*					24
Path 3	*r*	8	9	10	5	4	3	*s*						25
Path 4	*r*	8	9	10	5	2	1	4	3	*s*				30
Path 5	*r*	8	6	7	4	1	2	5	4	3	*s*			32
Path 6	*r*	8	6	7	4	3	*s*							33

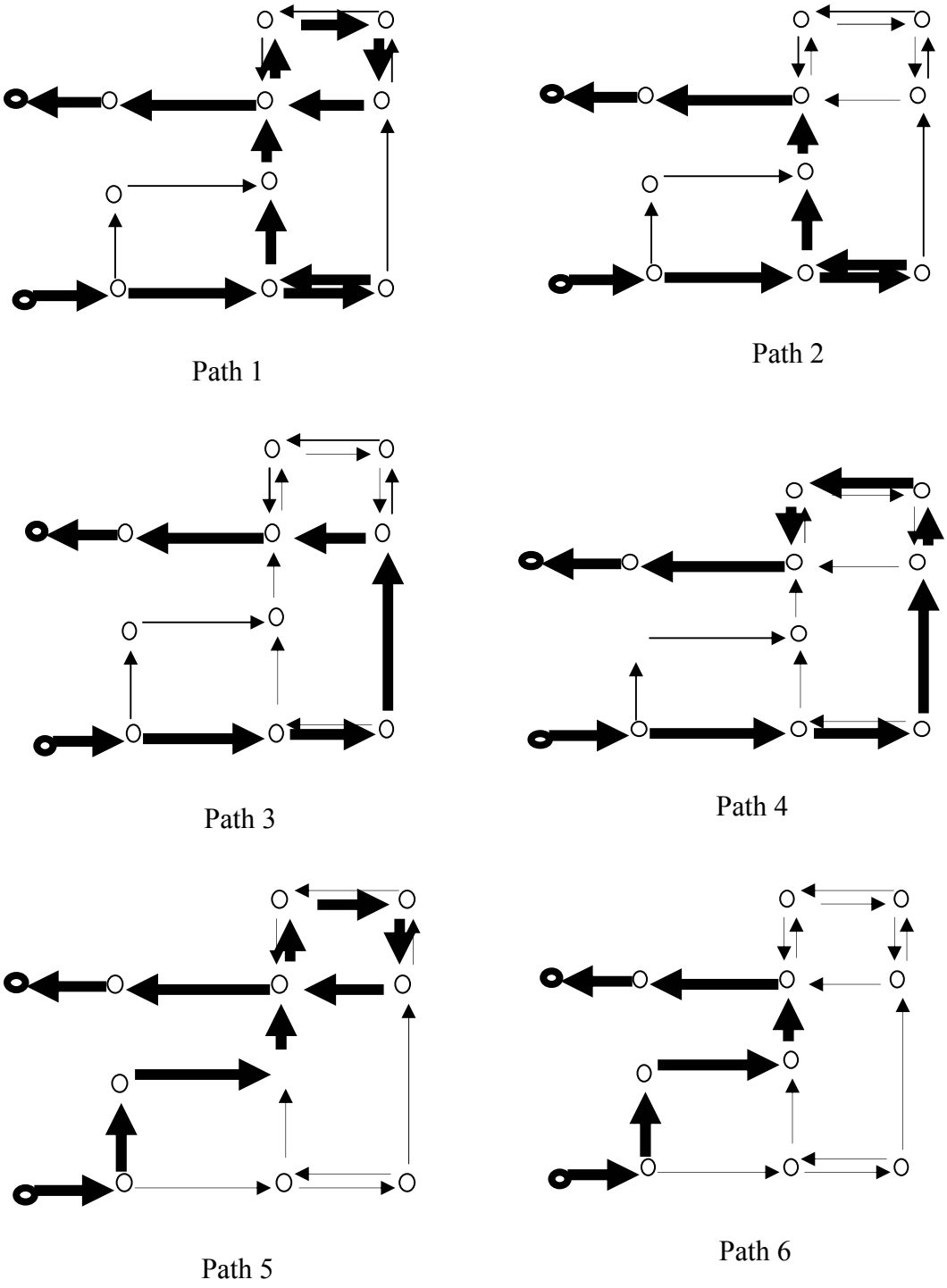

Figure 3. Path Generation

Table 3 and Figure 3 show the generation of possible paths. Paths are constructed according to their costs with ε^{node} >9 or rate>0.75 at an intersection and with ε^{path} >10 or rate >0.44. Paths 2 and 4 are branched from path 1 at merging node 4 to go node 3 in the back-tracing and path generation processes, while paths 3 and 5 are branched from path 1 at merging nodes 5 and 7, respectively. Path 6 is branched from path 2 at merging node 7. If ε^{node} =3 or rate=0.15 (15%)

and ε^{path}=3 or rate=0.15 are applied for the minimum cost at an intersection and the total path cost, respectively, only paths 1, 2, and 3 can be selected as possible paths while paths 4, 5, and 6 are not. Hence, the proposed algorithm enables to get various sets of possible paths by changing the values of ε^{node} and ε^{path}, and rates for the path cost to a specific node and the total path cost.

4. CONCLUSION

This paper proposes an approach to generate multiple possible paths from an origin to a destination. The improved proposed in this study is developed to efficiently search multiple and realistic alternative routes considering turning penalties and prohibitions. MVA keeps several node labels at each node through the vine search instead of tree search. It keeps multiple node labels in each direction, and each node label records the cumulative path costs and two consecutive predecessor nodes. The proposed MVA procedures also improve the backward tracing method compared to the conventional vine-building algorithm. The suggested backward tracing method, so called “one-link overlapping back-trace” in this study, could exclude all illogical movements at intersections. It is also important to note that the proposed approach adopts the concept of the rational upper boundary in evaluating the realism of alterative routes considered. The rational upper boundary used in this study means that drivers consider a route as an unrealistic alternative if the travel cost difference between the route and the minimum cost route is larger than the rational upper boundary. The rational upper boundary can be demarcated by absolute or relative values. The study investigates the properties of both of criteria, absolute and relative values, in order to figure out the direction of further empirical study.

A simple urban traffic network is used to analyze the applicability of the MVA procedures to generate realistic route choice sets used for real-time information provision. The MVA provides the flexibility to construct route choice sets that can be tailored to the specific driver behavior classes being considered. Traditional path enumeration methods can be restrictive in this context due to their inflexible structures to incorporate driver routing behavior. An attractive aspect of the MVA is that multiple node labels at intersecting nodes can be beneficially utilized to analyze alternative route characteristics in various ways, thereby fostering more effectiveness of the routing advisories and information provided via ATIS. Its computational process is illustrated with a small test network. The computational needs of constructing realistic route choice sets are especially relevant in the ATIS operation where significant computational burden is added to the associated implementation procedures for online processing of sophisticated real-time data. The further study will apply the proposed algorithm to large size of real network to find whether it is computationally effective.

REFERENCES

Azevedo, J. A., Santos Costa, M. E. O., Silvestre Maderira, J. J. E. R. and Vieira Martins, E. Q. (1993) An algorithm for the ranking of shortest paths, **European Journal of Operational Research**, **Vol. 69**, 97-106.

Bellman, R. and Kalaba, R. (1968), On K^{th} best policies, **Journal of the Society for Industrial and Applied Mathematics**, **Vol. 10**, 582-588.

de la Barra, T., Perez, B., and Anez, J. (1993) Multidimensional path search and assignment, Proceedings of the 21st PTRC Summer Meeting, 307-319.

Kim, I. (2001) A multi-label vine-building algorithm for the shortest path problem with turn penalties and prohibitions, Transportation Research Board 80th Annual Meeting, Proceeding CD-ROM

Kirby, R. F. and Potts, R. B. (1969), The minimum route problem for networks with turn penalties and prohibitions, **Transportation Research**, **Vol. 3**, 397-408.

Mahmassani, H. S. and Peeta. S. (1993) Network performance under system optimal and user equilibrium dynamic assignments: Implications for advanced traveler information system, **Transportation Research Record 1408**, 83-93.

Park, D. and Rilett, L. R. (1997) Identifying multiple and reasonable paths in transportation networks: A heuristic approach, **Transportation Research Record 1607**, 31-37.

Ramming, M. S. (2002), Network knowledge and route choice, MIT, Ph.D. Thesis.

Sheffi, Y. and Powell, W. B. (1982) An algorithm for the equilibrium assignment problem with random link times, **Networks**, **Vol. 12**, **No. 2**, 191-207.

Ziliaskopoulos, A. K. and Mahmassani, H. S. (1996), A note on least time path computation considering delays and prohibitions for intersection movements, **Transportation Research B**, **Vol. 30**, **No. 5**, 359-367.

.