VECTOR LABELING APPROACH FOR IDENTIFYING MULTIPLE REASONABLE ALTERNATIVE ROUTES IN TRANSPORTATION NETWORKS

Dongjoo PARK
Assistant Professor
School of Civil Engineering
Asian Institute of Technology
P.O.Box No. 4, Klong Luang
Pathumthani, 12120
Bangkok, Thailand
Email: djpark@ait.ac.th
Tel: 66-2-524-5681
Fax: 66-2-524-5509

Shyam Lal SHARMA
M. Eng. Student
School of Civil Engineering
Asian Institute of Technology
P.O. Box No.4, Klong Luang
Pathumthani, 12120
Bangkok, Thailand
Email: tra999762@ait.ac.th

Abstract: The existing multiple path algorithms such as k-shortest path algorithms have several limitations from transportation engineering view point. They often identify dominated paths and that multiple routes identified are too similar in terms of link used. The objective of this research is to develop an efficient algorithm for identifying non-dominated multiple reasonable alternative routes in transportation networks and to compare the results obtained from the algorithm with those from the existing algorithms.

In the proposed algorithm, a network is pruned using resource constraint and route constraint is used for maintaining route uniqueness of the identified routes. Dominance checking concept is introduced to avoid the dominated paths being identified. The vector labeling approach is used for labeling of each node. The algorithm was tested using the transportation network of Texas, Austin, U.S.A. It has been found that the algorithm can efficiently identify non-dominated multiple reasonable alternative routes in real transportation networks.

Key words: Efficient Vector Labeling, Route Similarity, Multiple Reasonable Paths, Transportation Network, Network Reduction

1. INTRODUCTION

Multiple reasonable alternative paths in transportation networks have received a considerable interest in recent days. The development in intelligent transportation system (ITS) along with in-vehicle route guidance system (RGS) has added an extra demand for identifying alternative paths from origin to destination. In-vehicle RGS and centralized RGS require identifying multiple alternative paths. The operations of emergency vehicles like emergency medical services (EMS) and emergency rescue services may require more than one alternative path for the higher reliability of earlier arrival.

The paper first introduces the definition of reasonable routes from transportation engineering perspective. Potential problems of traditional k-shortest path algorithms that have been used to identify multiple paths in transportation engineering application are explained and the limitation of a heuristic approach is illustrated. This research then develops an efficient algorithm for identifying non-dominated multiple reasonable alternative routes in transportation networks and compares the results of the algorithm with the existing algorithms. An appropriate methodology is developed here to identify multiple reasonable paths in real transportation networks. The proposed algorithm is tested using the transportation network of Texas, Austin, U.S.A. The results obtained from the proposed algorithm are compared with the results of traditional k-shortest path algorithm and the heuristic algorithm.

2. LITERATURE SURVEY

2.1 K- Shortest Path Algorithms

Shier's algorithm (Shier, 1979) and Yen's algorithm (Yen, 1971) are the two commonly used traditional k- shortest path algorithms. However, both of these algorithms do not use the concept of route similarity while identifying k-shortest paths. At least one link different between two paths is the sufficient condition to be the alternative paths.

2.2 Reasonable Paths

Dreyfus, S.E., (1969) mentioned that the two paths that do not visit the same nodes precisely in the same order may be considered as the alternative paths from the mathematical view point. This definition can accept loop (visit the same node more than once) in networks and hence will be problematic from transportation engineering context. This may yield in a large number of alternative paths in many real transportation networks even within the travel time limit slightly higher than that of the shortest path (Park and Rillet, 1997). Park (1998) put forward a formal definition for the reasonable alternative path as," a path is a reasonable alternative path if it not only has acceptable attribute value(s) but is also dissimilar in terms of the links used with respect to previously identified routes."

2.3 Identifying Multiple Reasonable Alternative Paths

Scott et al. (1997) presented a concept of "□- similar Path" and proposed an algorithm that can identify only single alternative path. When multiple alternative paths are to be identified, it becomes a computational burden. Park and Rillet (1997) and Park (1998) developed a heuristic algorithm that can identify the multiple reasonable paths. But the algorithm uses an empirical dispersion factor, which depends upon the network topology. Besides, it often identifies the dominated paths with respect to one attribute (e.g. travel time).

3. MATHEMATICAL FORMULATION AND PROPOSED ALGORITHM

In the proposed algorithm, the two constraints, namely travel time constraint (□) and route similarity constraint (□) are used to satisfy the route reasonableness of the alternative routes. The travel time of the alternative paths shall not exceed a certain percentage (M) of that of the fastest path and the route overlap of one route with all other competent routes shall not exceed a specified percentage of length of the shortest path (U). In other words, it uses the concept of the multiple constrained shortest path problems.

The approach used in this proposed algorithm was conceived from the algorithms of Aneja et al. (1983) and Park (1998). Resource based network reduction technique proposed by Aneja et al. (1983) is used to reduce the network size and the reasonable path concept of Park (1998) is used to define the route constraint parameter that forms the part of the route reasonableness.

3.1 Problem Formulation

Let (N,A) be a directed network with nodes N and links A. Let the other notations be as follows:

O	Origin
D	Destination
M	Maximum travel time of the reasonable alternative paths (percent with respect to fastest path)
U	Minimum desired uniqueness of the route
α	Maximum travel time of the reasonable alternative paths (i.e. resource constraint)
β	Maximum allowable length of paths overlapped between paths
L	Set of efficient paths which are temporarily labeled
P	Set of efficient paths which are permanently labeled
t_{ij}	Travel time on link i,j
d_{ij}	Distance of link i,j
i^k	k-th (pareto optimal) path from O to node i
e(i)	Number of pareto optimal paths from the origin O to node i
$\tau^k(i)$	Travel time of the k-th path from the origin to node i
$\gamma^k(i)$	Travel time of the k-th path from the node i to destination
$\delta^k(i)$	Length of the portion of the path which is shared with the k-th reasonable path from the origin to node i
$\theta^k(i,j)$	1 if link ij consist of the k-th path, 0 otherwise
$l^k(i)$	$(\tau^k(i), \delta^1(i), \ldots\ldots \delta^{k-1}(i))$ Label for the k-th path from the origin O to node i when identifying the "k" reasonable path

The mathematical formulation of the problem for identifying k-th reasonable alternative path using the notation of the constrained shortest path problem is given by

$$\text{P1:} \quad \min \sum_{(i,j)\in A} t_{ij} x_{ij} \tag{1}$$

$$\sum_{(i,j)\in A} t_{ij} x_{ij} \le \alpha$$

$$\sum x_{ij}^k x_{ij}^l d_{ij} \le \beta, \quad \forall l, \quad \forall k, \quad l \ne k$$

s.t.

$$\sum x_{ij} - \sum x_{ji} = \begin{cases} 1 & for \ \ i=1 \\ 0 & for \ \ i \in N-\{1,n\} \\ -1 & for \ \ i=n \end{cases}$$

$$x_{ij} = 0 \quad or \quad 1 \text{ for all } (i,j)\in A$$

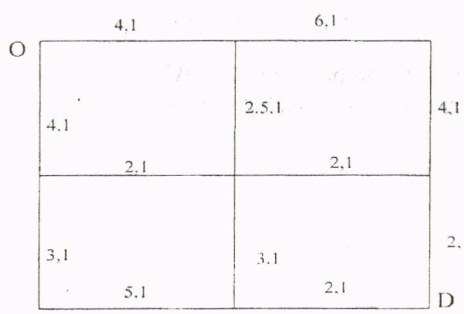
Note that P1 is a multiple constrained shortest path problem with not only the resource constraint (□) but also the route constraint (□).

3.2 Proposed Algorithm

The algorithm consists of two parts. In the first part, the network is reduced based upon the resource constraint and in the second part, nodes are labeled vectorially. The distance from origin O to an intermediate node j (say, h_j) and the distance from the intermediate node j to the destination D (say, □$_j$) is evaluated using Dijkstra's shortest path algorithm. The nodes that have the total distance of h_j+ □$_j$ greater than the specified (□) are deleted from the network. With the potentially reduced network, the algorithm enters into the second part.

The second part consists of the process of enumerating the shortest paths implicitly and vector labeling of analyzed nodes. Two kinds of label have been adopted, namely the tentative label and the permanent label. The labeling is of the type $\theta^r(y)$ which represents r^{th} label at node y and it has the structure of $\left[x^l;\alpha^r(y)\right]$ where x^l is the label index indicating that $\alpha^r(y)$ comes from the l^{th} label at node x using the relation $\alpha^l(x)+c(x,y)=\alpha^r(y)$. $c(x,y)$ is the travel time of the link. Hence $\alpha^r(y)$ is a vector of m+1 elements so that $\alpha^r(y)=(d^r(y), q^r(y))$ where $d^r(y)$ is scalar and $q^r(y) =(q^r_1(y), q^r_2(y), q^r_3(y),\ldots\ldots\ldots q^r_m(y))$. The tentative label gets the permanent label only if the path passing through that node satisfies all the constraints. Once a node gets the permanent label, the tentative label of the node is eliminated from the set of temporal label.

Following four figures (Figure 1 to Figure 4) are used to show the pruning of network more clearly using the resource constraints. The travel time from origin (O) to an intermediate node j termed as τ^1(j) and from node j to destination (D), termed as γ^1(j) along the shortest path for each node is first evaluated. The total travel time (τ^1(j) + γ^1(j)) is then used to decide whether the alternative paths can pass through that node or not considering only the travel time constraint. If the total travel time through the node(s) under consideration is higher than that specified by the constraint, the node(s) is/are deleted. Figure 1 shows the travel time (t) for each link and the distance (d) of each link in appropriate units for a sample network.



t,d

Figure 1. A Segment of Network Showing Travel Time and Distance

Figure 2 shows the travel time from origin to node j using the shortest path (τ^1(j)) and travel time from node j to the destination node ($\gamma^1(j)$) along shortest path along each node j.

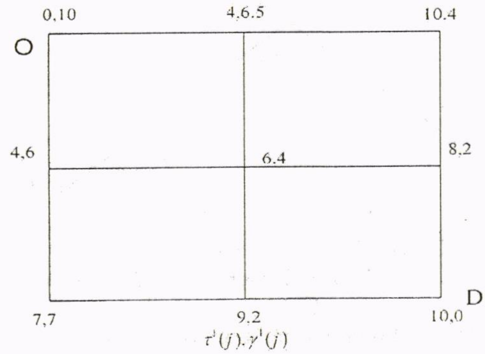


Figure 2. A Segment of Network Showing Travel Times at Nodes from Origin and from Destination through Fastest Path

Figure 3 shows the node to be deleted considering the travel time from origin to node j and node j to destination through the shortest path along each node j. Figure 4 shows the relationship for deletion criteria (values given are for example).

The expansion of nodes is performed from every possible node and the feasibility of the path is checked for every path. If travel time of the path considered is more than that of the constraint specified or if the overlapping of the path is more than that specified, the path is rejected. But the paths that satisfy both of the constraint are kept and these paths are eligible for dominance checking. The dominance check is performed to assess whether the path being analyzed is found dominated by previously identified path or not. If it is found dominated, the dominated path is rejected. Besides, it also checks whether the previously identified temporal paths are dominated or not by current path. The essence of dominance checking is that the rational drivers will choose the non-dominated paths. The non-dominated paths or efficient paths are those, which have similar, or at least one better attribute than other competent paths. The process is repeated for other paths.

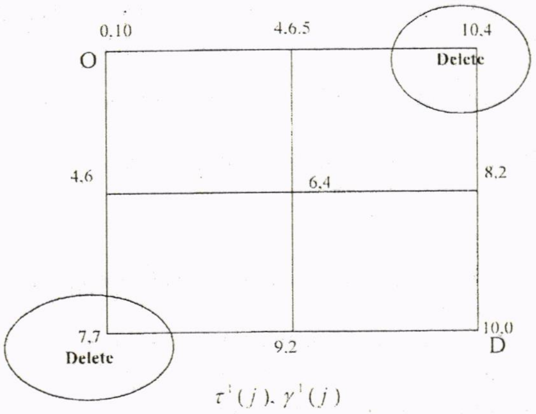
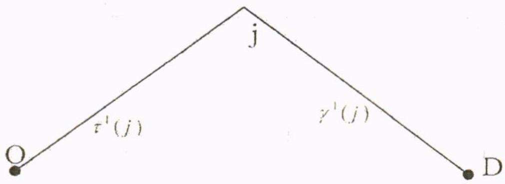


Figure 3. A Segment of Network Showing Nodes for Pruning



If □^{1} (j)+□^{1}(j)>□, then delete node j
If □^{1} (j)+□^{1}(j)≤□, then do not delete node j
□ =11

Figure 4. Pruning Condition for a Node

The steps of the efficient vector labeling (EVL) algorithm are summarized as follows:

STEP 1: Constraints Initialization

1.1 Minimum travel time from O to D: τ^1(D)
  Label setting algorithm
1.2 Resource constrain(s)
  $\alpha = \tau^1(D) \times M\ (\%)$
1.3 Uniqueness constrains
  β = distance of 1st path $\times$ (1-U(%))

STEP 2: Pruning by Resource Constraint(s): (Travel time is the resource constraint in this paper)

2.1 Identifying minimum travel times from origin to node j, τ^1 (j), and from node j to destination, γ^1(j)
  Label setting algorithm twice (one is from origin, the other is from destination)
  When the label (i.e. travel time) of current nodes is greater than α, stop and set all other non-permanently labeled nodes' label into very large value
2.2 Network reduction by pruning nodes
  For all nodes, if $\tau^1(j)+\gamma^1(j)>\alpha$, then delete node j in the network

STEP 3: Multi-Criteria SPP (vector labeling) and Feasibility Checking

3.1 Initialization
  K=2
3.2 K-th Reasonable Path Search with a Vector Labeling Approach
  3.2.1 For all paths of nodes, let
    $l^k(j)=(\infty, \infty, \infty \ldots)$ for enough number of k values (array size is same as that of value of k) e(j)=0, and P = ϕ and L= ϕ
  3.2.2 Choose Origin node and initialization
    Insert O into L (i.e. L={O}) and let l^1 (O)=(0,0,....)
  3.2.3 Determine Next Path (for permanent labeling)

If L= ϕ, STOP (K-th path is infeasible)
Choose any node, i, in L, which has the smallest travel time
If node i =Destination, GOTO STEP3.3.2 (we obtained the K-th path)
Insert node i into P(i.e P=P $\cup$ {i})
Delete node i from L (i.e. L=L-{i})

3.2.4 Label Updating
For all nodes j connected with node i by link i,j
e(j)=e(j)+l
$\tau^{e(j)}(j)=\tau^{e(j)}(i)+t_{ij}$
For all previously identified paths (1~ K-1st path)
$\delta^{k-l}(j)=\delta^{k-l}(i)+d_{ij}\times\theta^{k-l}(ij)$

3.2.5 Feasibility Checking
3.2.5.1 Travel Time Constraints
If $\tau^{e(j)}+\gamma^{1}(j)>\alpha$, then delete current path
3.2.5.2 Overlapping Constrains
For all K-1st paths, if one of these is true, then delete current path:
$\delta^{K-l}(j)>\beta$

3.2.6 Dominance Checking
3.2.6.1 Dominance checking with respect to travel time and overlapped length
3.2.6.2 Decision
If new path is not dominated by existing paths, insert the path into L.
If new path is dominated, delete it
If any existing temporal efficient path/s is/are dominated, delete the path from the L, and let e(j)=e(j)-l

3.3 Stop Rule
3.3.1 If K is equal to the required number of reasonable paths, STOP.
3.3.2 K=K+l and GOTO STEP 3.2

4. EXPERIMENTAL STUDY DESIGN

The traditional k-shortest path algorithm are the common algorithms used for identifying multiple alternative paths in networks although they can not identify the alternative paths from the point of view of transportation engineering application. Shier's k-shortest path algorithm mentioned in section 2 is one of them. This study compares the results obtained from the proposed EVL approach with those from the Shier's algorithm and the heuristic algorithm developed by Park and Rillet (1997).

The transportation network of Texas, Austin is used for the testing of the algorithm. The Austin network consists of 4463 nodes and 7760 links. One thousand O-D pairs were randomly selected for the analysis. To simulate the traffic congestion in network as practically as possible, three levels of congestion, levels I, II and III were assumed as shown in Table 1. It may be noted that the traffic congestion could be higher in the higher functional classification during peak hours, hence a higher level of congestion has been assumed in these classes. The link type hierarchy is classified by numbers 1, 2, 3 and 4 which represents freeway, major arterial, minor arterial and local street respectively. Each link was associated with its link capacity and randomly generated link volumes in the specified level of congestion.

Table 1.V/C Ratios for Different Link Types

Congestion Levels	V/C ratio in link			
	Freeway	Major Arterial	Minor Arterial	Local
I (light)	0.6 - 0.9	0.5 – 0.7	0.4 – 0.6	0.3 – 0.5
II (medium)	0.9 – 1.3	0.7 – 1.1	0.6 – 0.9	0.5 – 0.7
III (Severe)	1.3 – 1.9	1.1 – 1.6	0.9 – 1.3	0.7 – 1.0

For comparing the results of the proposed approach with those from existing algorithms, two measures of effectiveness (MOE) elements are considered. They are total travel time ratio (TTTR) and route similarity (RS). The TTTR is the ratio of the travel time of alternative path i to the travel time of the shortest path j. The route similarity is the ratio of the length of the route i repeatedly used while traversing through the alternative path j. The TTTR of the routes i and j may be given as

$$TTTR_{ij} = \frac{TT_i}{TT_j} \tag{2}$$

Where,

$TTTR_{ij}$ = TTTR between route i and route j

TT_i = travel time on route i

TT_j = travel time on route j

The TTTR shows how similar the two routes i and j are when they are compared with their respective travel times. The TTTR close to one means that the two routes are very similar in terms of their travel time.

The RS is defined as

$$RS_{ij} = \frac{\sum_{(i,j)\in A}^{n} d_a x_{ij}^a}{L_{ij}^1} \tag{3}$$

Where,

RS_{ij} =RS of route i to route j

d_a = distance of link a

L_{ij}^1 = distance of the shortest route (k=1)

x_{ij}^a = 1 if link a is used by route i and route j

0 otherwise

n= number of links in the network

Route similarity, average TTTR and average number of identified paths are compared for various O-D distances for heuristic and EVL approaches.

To compare the three algorithms, following sensitivity analyses were performed for the first five reasonable alternative paths as available for congestion levels I, II and III shown in Table 1.

- Average route similarity versus O-D distance
- Average TTTR versus O-D distance
- Average number of reasonable paths identified versus O-D distance

For the efficient vector labeling approach itself following sensitivity analysis were also carried out for three different levels of congestion:

- Trade off between number of nodes deleted versus travel time constraint (□) to evaluate the average percentage reduction in various congestion levels.
- Trade off between route similarity, average TTTR and number of average paths identified versus various route similarity and travel time constraint.

5. ANALYSIS OF RESULTS

For sensitivity analyses, the volume to capacity ratio in each link in a specified congestion level was randomly generated and average number of paths with specified route similarity limit were identified for each level of congestion. The figures of results for the congestion level III (i.e. severe congestion condition) are only shown here due to the space limitation.

5.1 Traditional k-Shortest Path Algorithm Results

In analyzing the results obtained from traditional k-shortest path algorithm, it was found that when k was set to five, the average TTTR as compared to the fastest path was 1.04. This indicates that the travel times of the four paths (second through fifth paths) are on average within four percent higher than that of the fastest path. Apart from this, the average RS was 0.77 indicating that the second through fifth fastest paths have an average of 77 percent of traversed links in common with the fastest path. Besides, the average number of paths (out of a possible four) that are 25, 50 and 75 percent different from the fastest path were 1.35,0.67 and 0.38 respectively. This shows that when the alternative paths were fairly acceptable based on travel time consideration, the links common to each other path were also higher not satisfying the dissimilarity condition of reasonable alternative routes. This reveals the deficiency of the traditional k-shortest path algorithm from the application viewpoint of transportation engineering.

5.2 Comparison of the Three Algorithms

In this section, the results are analyzed by disaggregating the O-D distance for each O-D pair. For evaluation of two MOE's, a constant dispersion factor of 100 was assumed for the heuristic algorithm. Let travel time of the fastest path be T_1 and travel distance of the same path be D_1 for simplicity in refering several times. For the comparison with existing approaches, the travel time limit is kept to be 110 % of the fastest path (□=1.10 T_1) and route similarity constraint is restricted to 70 % (□=0.70 D_1) with every alternative candidate paths.

5.2.1 Route Similarity versus O-D Distance

It has been found that average RS compared to the fastest path increases as the O-D distance does for the traditional k- shortest path algorithm. This pattern is expected because as the O-D distance increases the probability that similar paths to the fastest path exist also increases. For example, the O-D pairs which were 15 km apart had RS of 66 percent where as O-D pairs that were approximately 35 km apart, the average RS was in the range of 80 percent. The paths identified by the heuristic algorithm do not follow this pattern as evidenced by the relatively flat relationship between the RS and O-D distance. However, this relationship is even flatter for the EVL approach as portrayed in Figure 5.

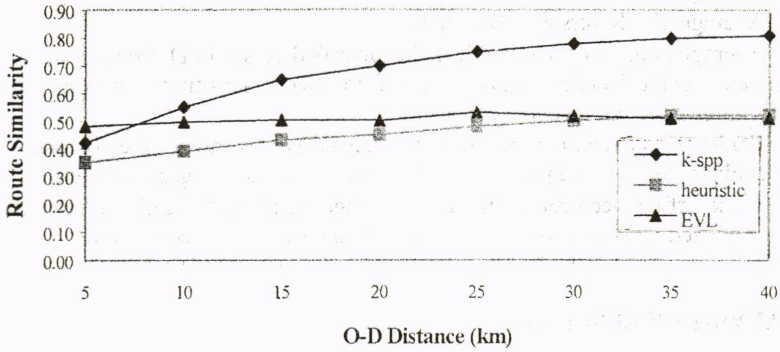


Figure 5. Route Similarity versus O-D Distance

5.3.2 TTTR versus O-D distance

The average TTTR for the k-SPP and the heuristic algorithm were found to decrease at a decreasing rate as the O-D distance increases as shown in Figure 6. This is due to the fact that as the O-D distance increases, there is higher chance of being more paths that have similar travel time to that of the fastest path. For the shorter O-D pairs, there are few alternative routes and these routes are, by nature, comparatively longer than that of the fastest path. For example, while the average TTTR for the traditional k-shortest path algorithm is 1.04, the average TTTR is approximately 1.02 for O-D pairs that are longer than 35 km. The average TTTR for the heuristic approach is comparatively higher than that for traditional k-shortest path algorithm. However, for the EVL approach, the TTTR does not change significantly with the O-D distance. This can be due to the fact that the probability of getting dominated for candidate paths with higher travel times is higher in shorter O-D pairs and probably there are fewer number of alternative routes. Those paths that do not get dominated can have lesser travel time and hence the TTTR is consistent for all O-D pairs.

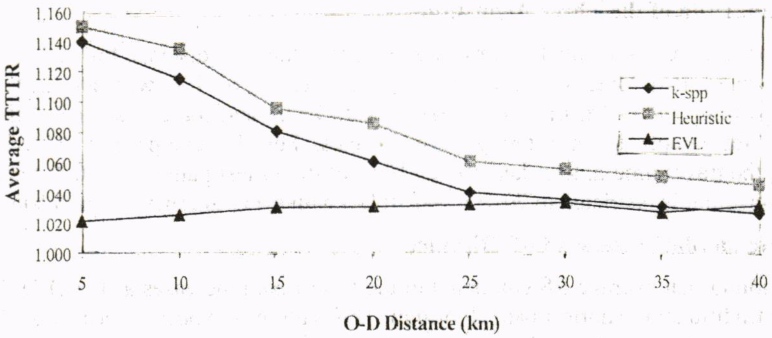


Figure 6. Average Total Travel Time ratio versus O-D Distance

5.3.3 Number of Reasonable Paths versus O-D Distance

Figure 7.illustrates the average number of paths that are at least 50 percent different from the fastest path. The result obtained from the heuristic algorithm is such that the number of paths identified does not increase irrespective of the the O-D distance. A little higher number of paths were obtained in the heuristic approach compared to the EVL approach for a shorter O-D distances. This may be due to the dominated paths included in the heuristic approach. In the EVL approach, however, the number of reasonable paths identified are fairly consistent (the number of paths increases consistently together with O-D distance). The number of distinct paths tends to decrease as the O-D travel time increases for the traditional κ-shortest path algorithm. This illustrates that for the longer routes the probability of finding the routes that are unique in terms of links used using the traditional k-shortest path algorithm is even less than that indicated by the aggregate analysis.

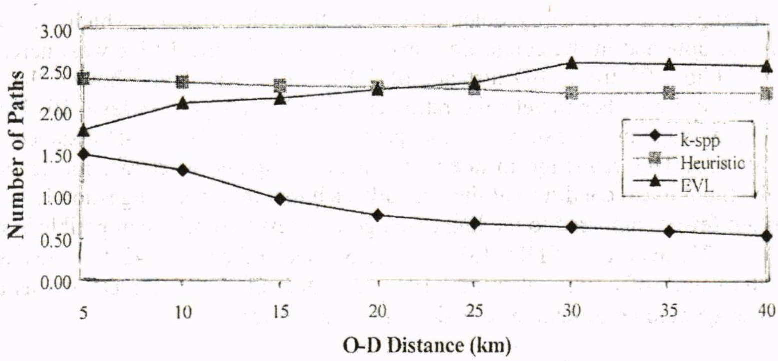


Figure 7. Average Number of Reasonable Paths that are 50 Percent Different from Shortet Path versus O-D Distance

5.4 Sensitivity Analysis for the Efficient Vector Label approach

Sensitivity analyses for the EVL approch were also performed for various route overlaping limits and travel time limits for the three congestion levels.

5.4.1 Congestion Level III

Analyses show that when the travel time limit (□) and the route similarity limit (□) were 1.10 T_1 and 0.5 D_1 respectively, the average route similarities between alternative routes were found to be on the order of 0.37. The TTTR was on the order of 1.02 for O-D distances higher than 15 km while it was on the order of 1.01 for O-D distances less than 15 km. Similarly, the number of reasonable alternative paths was on the order 2.3. It was found that the number of nodes not deleted in the network were on the order of 239. When the value of □ and □ was choosen as 1.10 T_1 and 0.25 D_1 respectively, the route similarity between routes was increasing from 0.16 to 0.19 for O-D distances between 5 km and 40 km. The TTTR was similar as in the case of □=1.10T_1,and □=0.5 D_1, where as the number of reasonable alternative paths identified was from 1.47 for O-D distance of 5 km to 1.9 for O-D distance of 40 km. This result indicates that the number of reasonable paths significantly decreases as the route similarity limit does. In other words, the routes that have similar route travel time but use significantly different routes are very few.

It was found that when □ was reduced to $1.05T_1$ and □ was kept 0.7 D_1, the average route similarity and TTTR obtained were very similar to those obtained for □ = $1.05T_1$ and □ =0.7 D_1. However, the average number of paths obtained was ranging from 1.9 to 3. This little decrease in paths shows that most of the paths obtained have lower TTTR and those which have higher TTTR get dominated by other competent paths. But it was observed that the average computation time for the lesser value of □ was noticed significantly lesser than that for the higher value of □. This is because many of the nodes may have been deleted from the network that did not require any comparison further. It was found that the average number of nodes not deleted was on the order of 173.

5.4.2 Congestion Level II

In congestion level II, when the value of □ was kept to be $1.10T_1$ and value of □ was kept to be 0.7 D_1, the average route similarity obtained was on the order of 0.53, which was very similar to the result obtained in the congestion level III. However, the TTTR was increased gradually from 1.01 to 1.03 from O-D distance of 5 Km to 40 Km. This showed that the alternative paths have the higher travel time ratios compared to congestion level III. It was also found that the average number of reasonable paths were from 2.21 at O-D distance of 5 Km to 3.12 at 40 Km. This reduction in alternative paths compared to congestion level III may be due to the topological condition of the network such that in lower congestion level the altenative paths are fewer compared to the higher congestion level. Similar compatible results (Route similarity=0.35, average TTTR= 1.01 to1.03, Number of paths=1.42 to 2.42) were obtained when the values of □ was taken as $1.10T_1$ and value of □ was 0.5 D_1. It was also found that the average number of undeleted nodes was on the order of 215.

5.4.3 Congestion Level I

When □ was set to $1.10T_1$ and □ to 0.5 D_1, similar RS and TTTR were obtained compared to those under congestion level II. However, the average number of paths was 1.64 for O-D distance of 5 km to 2.42 for O-D distance of 40 km as compared to 1.42 and 2.42 of the congestion level II, respectively. This slightly higher number of paths in congestion level I than that obtained in congestion level II can be attributed to the topological condition of the network. It was found that the average number of undeleted nodes was on the order of 214.

A comparison of the three different congestion levels show that the number of reasonable alternative paths are highest at congestion level III and lowest at congestion level II. The number of undeleted nodes does not depend upon the congestion level of the network but depends upon the travel time limit.

6. COMPUTATIONAL COMPLEXITY OF THE ALGORITHM

The computational complexity of efficient vector labeling algorithm consists of the complexity of the two separate algorithms. These two algorithms are

- Dijkstra's Label Setting Algorithm
- Efficient Vector Labeling Approach

The computational complexity of Dijkstra's label setting algorithm is rather straightforward. Let G=(N,A) be the network under consideration, with n nodes in set N. In the worst-case condition, the terminal node is the nth node to receive a permanent label. Assume that at any given time of the execution of the algorithm there are m nodes with permanent labels, and n-m nodes with temporary labels. To identify the $(m+1)$st node that must be given a permanent label, the n-m temporary labels have to be updated by performing an addition and then one comparison for each label (Philips and Garcia-Diaz, 1981). Once the temporary labels are updated, one more comparison is needed for minimization to identify the label that becomes permanent. This shows that the number of elementary operations required to ass'gn a permanent label to one more node, with m nodes of the network being already permanently labeled becomes approximately $3(n\text{-}m)$.

Thus, the overall complexity under-worst case is given by

$$\begin{aligned}\sum_{m=1}^{n} 3(n-m) &= 3(\sum_{m=1}^{n-1}(n-m) \\ &= 3[(n-1)+(n-2)+(n-3)+\ldots\ldots\ldots\ldots+1] \\ &= 3\frac{(n-1)(n-1+1)}{2} \\ &= 3\frac{n(n-1)}{2} \\ &= O(n^2) \qquad (4)\end{aligned}$$

When shortest path problem is called from destination to origin, the computational complexity is the same as above i.e.

$$O(n^2)$$

The determination of computational complexity of the EVL approach is relatively complex. The algorithm will require the following major operations for each step of the algorithm.

Step 1 Initialization

Dijkstra's label setting algorithm will be executed once to find the shortest path between origin and destination. The computational complexity of the algorithm is given separately above.

Step 2 Pruning of Nodes

2.1 It is required to run Dijkstra's label setting algorithm second times from destination to origin. The major number of operations required will be the same as given above. The total of the operations in executing twice of this algorithm will be

$$2\times 3\frac{n(n-1)}{2}, \text{ which gives } 3n(n-1)$$
$$=O(n^2) \qquad (5)$$

2.2 The pruning of network requires N comparisons to check whether the nodes satisfy the travel time constraint or not.

Step 3 Multi-criteria Shortest Path Problem (Vector Labeling) and Feasibility Checking

3.2.4 For updating the labels of the temporary nodes

Let k be the maximum number of reasonable alternative paths to be identified, l be the maximum number of local paths that come to a node during labeling of each node, n and m be as explained for Dijkstra's label setting algorithm, then the numbe of additions required is given by

$$\sum_{m=1}^{m=n} q\,l(n-m)(k-1) \qquad 0<q\leq 1$$

Where l = number of maximum local paths at any node
k=2,3,4,5.....

Local paths can be understood as the sub paths that come to a node for labeling (temporary or permanent) from a node's all possible previous nodes during enumeration of paths. These paths later become part of the k route if they lie in the particular route identified.

If the number of local paths incoming to a node is considered as a finite value, then the number of operations can be summed as

$$q(k-1)l\sum_{m=1}^{m=n}(n-m) \qquad 0<q\leq 1$$

$$=\frac{n(n-1)}{2}q(k-1)l$$

$$\approx O(n^2kl) \qquad (6)$$

3.2.5.1 The comparisons for the check of travel times whether it satisfies the constraint or not is given by

$$\sum_{m=1}^{m=n} q\,l(n-m)(k-1) \qquad 0<q\leq 1$$

$$=\frac{n(n-1)}{2}q(k-1)l$$

$$\approx O(n^2kl) \qquad (7)$$

3.2.5.2 The comparisons for route overlap checking is given by

$$\sum_{m=1}^{m=n} q\,l(n-m)(k-1) \qquad 0<q\leq 1$$

$$=\frac{n(n-1)}{2}q(k-1)l$$

$$\approx O(n^2kl) \qquad (8)$$

3.2.4 The number of comparisons for dominance checking of the local paths coming to a node is given by

$$\sum_{m=1}^{m=n}[3\times 3\times q_1\,k\,(l-1)(n-m)(k-1) \qquad 0<q_1\leq 1$$

$$=9\frac{n(n-1)}{2}q_1(k-1)k(l-1)$$

$$\approx (n^2-n)(k^2-k)(l-1)$$

$$\approx O(n^2k^2l) \qquad (9)$$

The minimization operation required for permanent label setting is given by

$$\sum_{m=1}^{m=n}[q_3\,(n-m)]l(k-1) \qquad 0<q_3\leq 1$$

$$= \frac{n(n-1)}{2} q_3(k-1)l$$
$$\approx O(n^2 kl) \quad (10)$$

The total computatational time of the algorithm is the sum of the computational time required for all the above mojor operations outlined above. Hence, the overall computational complexity of the algorithm is given by

$$O(n^2) + O(n^2) + O(n^2 kl) + O(n^2 kl) + O(n^2 kl) + O(n^2 k^2 l) + O(n^2 kl)$$
$$= 2O(n^2) + 4O(n^2 kl) + O(n^2 k^2 l)$$
$$= O(n^2) + O(n^2 kl) + O(n^2 k^2 l) \quad (11)$$

Since the value of constant k, is very small compared to the value of n and l, it can be said that the computational time of the algorithm is mostly dependent on the number of nodes and the number of maximum local paths of each node under consideration. Since the number of local paths that of each node depends upon the network characteristics, it is difficult to precisely evaluate the worst case complexity of the algorithm. It can be concluded that the higher the reduction in the network size, the lesser is the number of local paths that come in to a node.

The value of l (number of local paths) in the worst case can be non-deterministically large with an exponential growth. In that situation no exact quantification of complexity is possible and hence the problem is non-deterministic polynomial (NP- complete).

7. CONCLUDING REMARKS

The algorithm proposed in this paper can be viewed as an exetension of algorithm for solving multiple constrained shortest path problems. The resource constraint is used to prune the network as suggested by Aneja et al. (1983) and a route constraint is used to identify the paths dissimilar in terms of links used as sugested by Park and Rillet (1997). The concept of dominance checking during labeling is introduced to identify only the paths non-dominated paths.

Results from the experimental study show that the EVL approach is superior to traditional k-shoetest path algorithms in that it identifies the reasonable routes applicable in many transportation engineering applications. The heuristic approach by Park and Rillet (1997) has the limitation that the approach can not be applied in every general transportation networks without changing the value of dispersion factor judiciously. The fewer number of reasonable alternative paths identified by the heuristic approach than those identified using the EVL approach for similar constraint values shows that the heuristic algorithm can not identify the alternative paths in many situations.

Although the EVL approach can identify the reasonable alternative paths in real transportation networks, still more works remains to be done. The computational efficiency of the EVL approach mostly depends upon the network size and characteristics. When travel time constraint is very close to the travel time of the fastest path, the algorithm is more efficient because of the higher number of deleted nodes from the network. But when the travel time constraint is wider, the number of local paths to be identified increases thus making the algorithm relatively inefficient. Hence, an appropriate methodology for reducing the number of local paths without deteorating the accuracy of the results should be proposed.

REFERENCES

Ahuja, R.K, Magnanti, T.L and Orlin, J.B (1993) **Netwotks: Theory, algorithms and Applications**, Prentice Hall ,Englewood, New Jersey.

Aneja, Y.P, Aggarwal, V. And Nair, K.P.K.(1983) "Shortest chain subject to side constraints." **Networks, Vol. 13**, 295-302.

Dreyfus, S.E, An appraisal of some shortest path algorithms, **Operations Research Vol. 17**, 1969, 395-412

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

Park, D.(1998) "Multiple path base vehicle routing in dynamic and stochastic transportation networks." **Ph. D. Dissertation**, Texas A&M University.

Philips, Don T. and Garcia-Diaz, Alberts (1981) **Fundamentals of Network Analysis**, Prentice-Hall Inc., Englewood clieffs, New Jersey

Scott, Kelley,Pabon-Jimenez, Glarycelis and Bernstein, David (1997) Finding alternatives to the best path. **Presented at the 76th Transportation Research Board Annual Meeting**, Paper No. 970682,1-13

Shier, D.R (1979) "On algorithms for finding the K- shortest paths in a network." **Networks, Vol. 9**, 195-214

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