대한산업공학회/한국경영과학회 2002 춘계공동학술대회
한국과학기술원(KAIST) 2002년 5월 3일-4일

복수최단경로의 새로운 해법 연구

장 병 만

서울산업대학교 산업정보시스템공학과

A Study on New Algorithm for K Shortest Paths Problem

ByungMan Chang

Seoul National University of Technology, E-mail:bmchang@plaza1.snut.ac.kr

Abstract

This article presents a new algorithm for the K Shortest Paths Problem which develops initial K shortest paths, and repeat to expose hidden shortest paths with dual approach and to replace the longest path in the present K paths. The initial solution which comprises K shortest paths among shortest paths to traverse each arc is made from bidirectional Dijkstra algorithm. When a crossing node that have two or more inward arcs is found at least three time by turns in this K shortest paths, one inward arc of this crossing node, which has minimum detouring distance, is chosen , and a new path is exposed with joining a detouring subpath from source to this inward arc and a spur of a feasible path from this crossing node to sink. This algorithm requires worst case time complexity of $O(Kn^2)$, and $O(n^2)$ in the case $K \leq 3$.

Keywords : K shortest paths problem, Shortest aborescence, Dijkstra method, Dual approach

1. Introduction

This paper presents a problem of finding K shortest simple paths which has minimum weight in a graph $G=(N, A)$ which has node-set N of cardinality n and arc-set $A=(c_{ij})$ of cardinality m with positive length. In the K shortest paths problem, for a given positive integer $K \leq n$ and a given source-destination pair in directed graphs. The paths should be simple means that no node can be repeated.

This K shortest paths problem is useful to calculate the all pair of K shortest paths for the automatic vehicle guidance system in the Intelligent Transport System (ITS) [14], transportation planning analysis, and shipping goods through a distribution network, and is a well-studied graph optimization problem that is encountered in numerous application in telecommunications , VLSI design [9] and so on.

There are some methods which have been proposed for solving this problem.

Lawler [10, 11] presented a search tree type algorithm with complexity $O(Kn^3)$.

Dreyfus[4] presented these K shortest paths from a source node to each of the other $n-1$ nodes with time complexity $O(Kn^3)$.

Yen [16] developed an $O(Kn^3)$ algorithm that repeats to search candidate shortest paths with breaking arc and merging the root and the minimum spur in each iteration, and to select the shortest one of them on the direct network and the nondirect network.

Hadjiconstantinou and Christofides [7], and Katoh et al. [8] presented an $O(Kn^2)$ algorithm that gets the shortest path from origin s to destination t, searches three types of shortest detouring path from a node in the shortest path to destination, selects the shortest path among all detours, and iteratively repeats the above procedure.

Almost studies till now have made one shortest path and after then they repeat to search the next shortest path one by one with their own methods. But this article presents another algorithm which builds near optimal K shortest paths initially, and improves the initial solution.

This section provides the initial solution procedure for the K shortest paths problem.

We require the following notation :

- $T(s)$ foreward shortest arborescence from s to every node ;
- $T(t)$ reverse shortest arborescence from t to every node ;
- $T(s, t)$ a double shortest arborescence made by merging $T(s)$ and $T(t)$;
- π_i the length from s to node i in $T(s)$;
- δ_i the length from t to node i in $T(t)$;
- f_i the predecessor of node i in $T(s)$;
- h_i the successor of node i in $T(t)$;
- P^k_l the K- th shortest path from s to t in the l-th improved feasible solution $(l \geq 1)$;
- $FP(g, i, j)$ the shortest subpath from node g through node i to node j ;
- $FP(g,(i,j))$ the shortest subpath from node g through *arc* (i, j) to node j ;
- $SP(u, v)$ the shortest path from s to t through *arc* (u, v) ;
- $HP(g, i, j)$ the shortest path from node s to node t through *arc*(g, i) and *arc*(i, j) or *subpath*(i, j) ;
- $LP(P^k_l)$ the length of P^k_l ;
- $P(i)$ the shortest path from node s to node i in $T(s)$;
- $P'(j)$ the shortest path from node j to node t in $T(t)$;
- $v^k_l(i)$ the i-th node in P^k_l ;
- KSP_l the l-th improved solution (set) of K shortest paths, $KSP_l = \{P^1_l, P^2_l, \cdots, P^k_l\}$
- EP_l the l-th exposed hidden path which is shorter than P^k_l ;
- $IA(j, r)$ the *r-th* inward arc of node j ;
- $OA(j, r)$ the *r-th* outward arc or outward sub path from node j ;

2. Initial Solution Procedure

It's well known that the shortest paths from node s to all other nodes in G can be represented by the shortest path arborescence $T(s)$, and the unique path from s to t in this arborescence tree, denoted by $s \xrightarrow{T(s)} t$, represents the shortest path from s to t. Therefore,

$$P^1 = s \xrightarrow{T(s)} t = s \xrightarrow{T(t)} t$$

We can find $T(s)$ and $T(t)$ using Dijkstra's method forward from s and backward from t separately. With the bidirectional Dijkstra's algorithm, we can simultaneously apply the forward Dijkstra's algorithm from node s and the reverse Dijkstra's algorithm from node t.

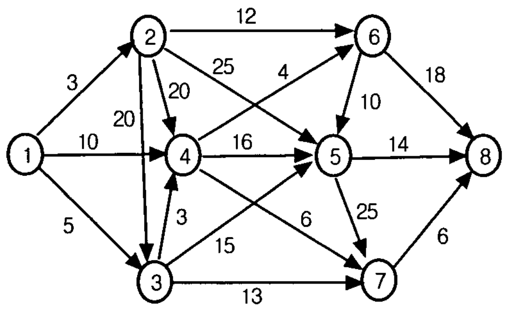
When $T(s)$ and $T(t)$ can be merged, $T(s, t)$ is produced, and in which we can get the informations about each shortest path $SP(u, v)$. from s to t to pass through each *arc* (u, v), and the length $LP(SP(u, v))$ of the shortest path $SP(u, v)$.

In $T(s, t)$, the shortest path from node s to node t through node i is formed with the path $P(i) \cup P'(i)$ and the shortest path from node s to node t through an *arc* (i, j), node i in $T(s)$ and node j in $T(t)$ is formed using the path $P(i) \cup (i, j) \cup P'(j)$. This $T(s,t)$ can be called a Double Shortest Arborescence(DSA).

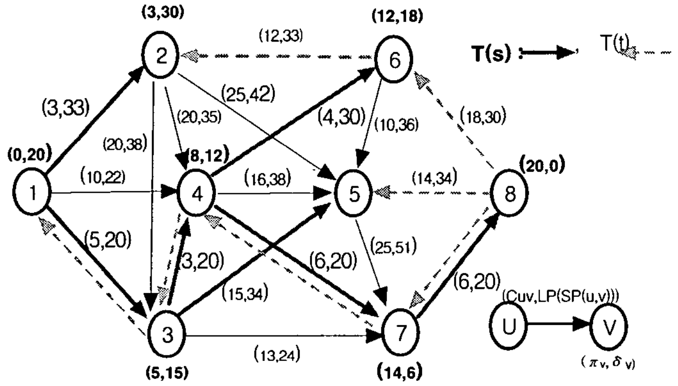
Then $SP(u, v)$ can be described with $s \xrightarrow{T(s)} u \longrightarrow v \xrightarrow{T(t)} t$.

The length of $SP(u, v)$, $LP(SP(u, v))$, is

$$LP(SP(u, v)) = \pi_u + c_{uv} + \delta_v \qquad (1.1)$$



(a) Example network



(b) T(s, t), Double Shortest Arborescence

[Figure 1] An example network and its $T(s,t)$

Therefore an initial solution of a K shortest paths problem can be found by searching for the k shortest paths from $T(s,t)$ in ascending order of the length of the paths $SP(u,v)$s. This solution provides an upper bound on the K shortest paths solutions values.

The $KSP_I = \{P^1, P^2, \cdots, P^5\}$ for $K=5$ shortest paths problem in the network of [Figure 1] is as follows:

P^1_I : 1 - 3 - 4 - 7 - 8, $LP(P^1_I) = 20$
P^2_I : 1 - 4 - 7 - 8, $LP(P^2_I) = 22$
P^3_I : 1 - 3 - 7 - 8, $LP(P^3_I) = 24$
P^4_I : 1 - 3 - 4 - 6 - 8, $LP(P^4_I) = 30$
P^5_I : 1 - 2 - 6 - 8, $LP(P^5_I) = 33$

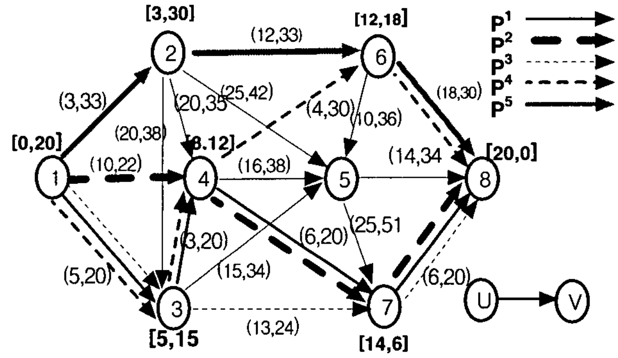
This KSP may not be an optimal solution. KSP comprises $\{P^1, P^2, \cdots, P^k\}$ which is the set of k apparent shortest paths that pass from s to t through each specified arc. Some paths in the KSP are comprised in the optimal solution and the others may not be.

3. Improvement Procedure

3.1 Crossing Nodes and Hidden Paths

Let an apparent path be one of $SP(u, v)$, $\forall (u, v) \in A$, which path has at least one first passing *arc* (u, v) in KSP. Let a hidden path be composed of all arcs which are passed and covered entirely with some other apparent path in KSP. Let an exposed path be a new appeared hidden path by breaking one of arcs of apparent path.

Then all initial K shortest paths appeared in KSP are the apparent paths. All hidden paths are not appeared in $T(s, t)$ whether the length of some hidden paths are shorter than the length of some apparent paths in KSP. Generally hidden paths are longer than apparent paths. The hidden path is not appeared in KSP, because all arcs of hidden path are already



[Figure 2] KSP_I for $K = 5$ shortest paths problem in the network of [Figure 1]

passed and covered by the other apparent paths.

To obtain the optimal solution, we need to develop an algorithm to search and expose the hidden paths shorter than some paths of the KSP, and to replace some apparent paths by these exposed hidden paths until there are not any shorter hidden path in the KSP.

The algorithm in this article presents to expose some hidden paths that are shorter than P^k_I, and to replace some of $P^i_I, \forall i \in K$, in KSP_I by the exposed shortest paths.

Let an intersection node i, IN_i be a node that has at least two inward arcs and at least one outward arc, excluding node s, node t and f_t which has one outward arc. Let a crossing node i, CN_i be an intersection node which at least two inward arcs are appeared by turns and which is appeared at least three times in the KSP_I. Let IN be a set of intersection nodes, and CN be a set of crossing nodes.

In initial solution KSP_I or feasible solution KSP_I, if there are three or more apparent paths, $HP(g,i,j)$, $HP(g,i,k)$, and $HP(h,i,j)$, then this node i is a crossing node (see [Figure 3]), and there could be a hidden path shorter than P^k_I, which is $HP(h,i,k)$. Because *arc* (h, i) was passed by $HP(h,i,j)$, and *arc* (i, k) was passed by $HP(g,i,k)$, so path $HP(h,i,k)$ is hidden and disappeared by the two apparent shorter path.

Theorem 1 : If node i is CN_i and has two inward arcs and two outward arcs, a path, $SP(IA(i,2),OA(i,2))$ is appeared fourthly at earliest and hidden by second or third path.

Proof : The shortest path which passes CN_i is $SP(IA(i,1),OA(i,1))$, and second or third path is $SP(IA(i,1),OA(i,2))$ or $SP(IA(i,2),OA(i,1))$. $SP(IA(i,2),OA(i,2))$ is disappered and hidden by second or third path., because $FP(s,IA(i,2))$ is passed by $SP(IA(i,2),OA(i,1))$, and $FP(OA(i,2),t)$ by $SP(IA(i,1),OA(i,2))$. Therefore, $SP(IA(i,2),OA(i,2))$ is appeared fourthly at earliest and hidden by second or third path.

And the crossings may be produced not only on intersection nodes, but also on intersection arcs and on intersection subpaths which have two or more inward arcs, like *arc* (g, i) and *arc* (h, i) in the [Figure 4].

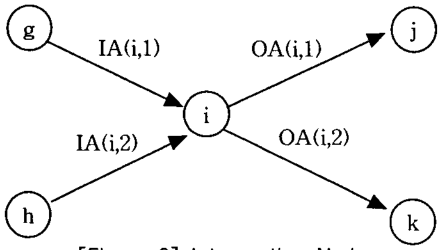
When a crossing occurs at IN_i in the KSP_I in the [Figure

3] or the [Figure 4], IN_i should be appeared at least 3 times in the KSP_l .

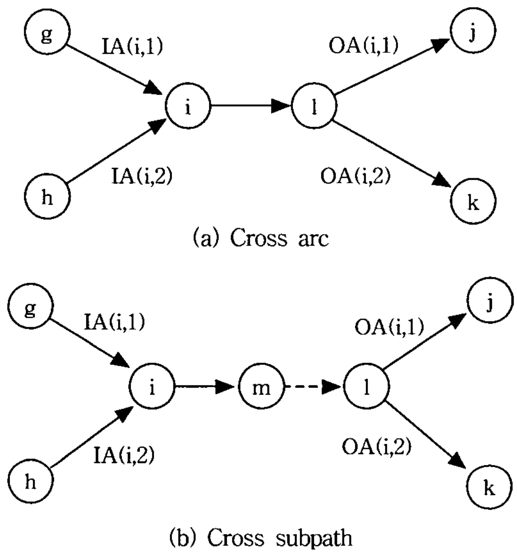
But even though this INi is appeared only two times in the KSPl , or the INi is appeared three or more times and only same inward arc is appeared continuously, the crossing does not occur.

A crossing at IN_i occurs necessarily in the case that $IA(i, 1)$ is appeared two or more and $IA(i, 2)$ is appeared at least once in the KSP_l.

We can find out the candidate list of crossing nodes in the method of check the number of nodes



[Figure 3] Intersection Node



[Figure 4] Cross arc and Cross subpath

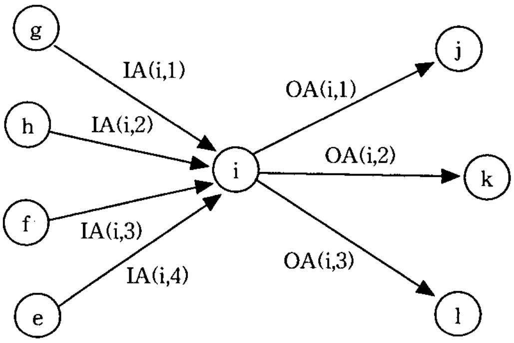
which are appeared three and more times in the KSP_l within at most complexity $O(Kn)$.

KSP_l comprises $\{P^1_l, P^2_l, \cdots, P^k_l\}$ which is the set of K apparent shortest paths that pass from s to t through each specified arc, but may not be the set of an optimal K shortest paths. There may be hidden paths that are shorter than P^k_l. When a crossing occurs at IN_i, a hidden path EP_l is disappeared by second and next paths which pass through an intersection node. EP_l which may pass the second or next inward arc and the second or next outward arc of the intersection node is not appeared in [Figure 3], because at least three or more paths passed all arcs of EP_l in advance.

Therefore, if we find out a crossing node from the $KSP_l = \{P^1_l, P^2_l, \cdots, P^k_l\}$, we search and expose the hidden paths shorter than P^k_l centering around the crossing node, and replace P^k_l and some shorter paths in the KSP_l by the exposed hidden paths till there is not any crossing node or not any exposed path shorter than P^k_l in the KSP_l . But it is not easy to find out crossing nodes efficiently and to expose hidden shorter paths.

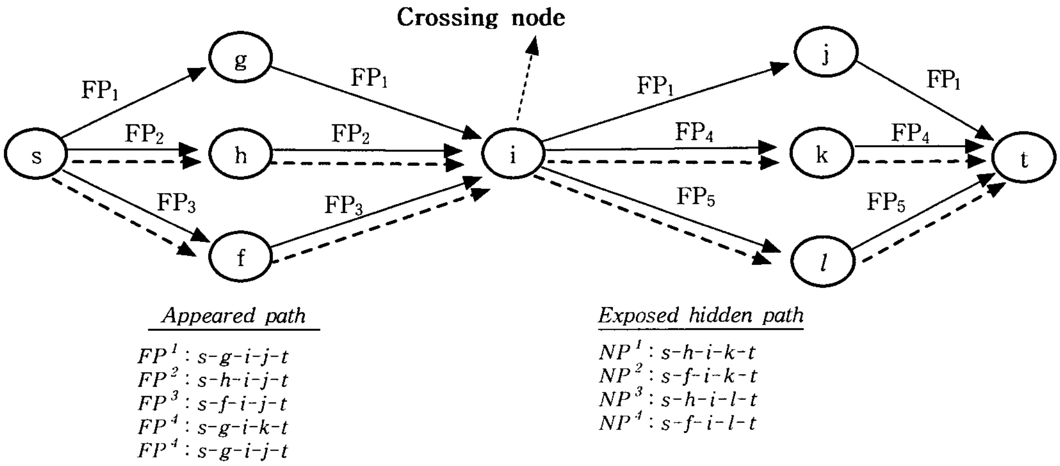
When three shortest paths pass through a node i, and $IA(i, 1)$ are appeared at least twice and $IA(i, 2)$ are appeared once in KSP, then this node i is a crossing node, CN_i , and a hidden path that is the next path to pass this CN_i is covered and hidden by the former three paths. Because the hidden path passes through $IA(i, 2)$ and $OA(i, 2)$, we need to break temporarily the path to pass through $IA(i, 1)$ of CN_i in order to expose this hidden path.

Therefore, we can make it a rule to break temporarily this $IA(i, 1)$ of CN_i in order to expose a hidden path EP_l which passes from node s through $IA(i, 2)$ and $OA(i, 2)$ or a second subpath from CN_i to node t.



[Figure 5] CN_i with multiple inward arc

In case that the crossing node have many inward arcs [Figure 5], we can expose hidden shortest paths with breaking some inward arcs one by one from the first inward arc of the crossing node. In the [Figure 5], with breaking $IA(i, 1)$, that is, $arc(g, i)$, we can expose hidden paths HP



[Figure 6] Appeared paths and Exposed hidden paths

(h, i, k) and $HP\ (h, i, l)$, with adding to break $IA\ (i, 2)$, that is, $arc\ (h, i)$, we can expose hidden paths $HP\ (f, i, k)$ and $HP\ (f, i, l)$, and with adding more to break $IA\ (i, 3)$, that is, $arc\ (f, i)$, we can expose hidden paths $HP\ (e, i, k)$ and $HP\ (e, i, l)$.

Therefore, if a node is appeared at least three times and two inward arcs of the node are appeared by turns, then there may be a hidden shortest path in the KSP_l, and with breaking the first inward arc, a hidden shortest path can be exposed. If this exposed hidden path, EP_l, is shorter than the path P^k_l, this EP_l is entered and the present P^k_l is left out, and an improved KSP_{l+1} is produced in ascending order of the path length with $\{P^1_l, P^2_l, \cdots, P^{k-1}_l, EP_l\}$.

We can expose the hidden paths which pass CN_i in the way of connecting the subpath passing from s through the second inward arc and the next inward arcs to CN_i, $FP\ (s, IA\ (i,a))$, where $a \geq 2$, with the subpath passing from CN_i through the second outward arc and next outward arcs to node t, $FP\ (OA\ (i,b), t)$, where $b \geq 2$. In case that CN_i has three or more inward arcs and outward arcs in the KSP_l [Figure 6], we can connect $FP\ (s, IA\ (i,a))$, where $a \geq 2$, that is, $s \sim f \rightarrow i$, and $s \sim h \rightarrow i$, with $FP(OA(i,b), t)$, where $b \geq 2$, that is, $i \rightarrow k \sim t$, and $i \rightarrow l \sim t$, to expose $HP\ (h, i, k)$, $HP\ (h, i, l)$, $HP\ (f, i, k)$, and $HP\ (f, i, l)$.

$LP(EP_l)$, the length of the new exposed path EP_l, $HP\ (h, i, l)$, that is, $s \sim h \rightarrow i \rightarrow l \sim t$ can be easily calculated from the information about π_h and δ_l in $T\ (s, t)$.

$$LP\ (EP_l) = \pi_h + c_{hi} + c_{il} + \delta_l \quad (2.1)$$

Lemma 1. *If an intersection node becomes definite to be the first crossing node on the r - th path in the* KSP_l, *then* $\{P^{1*}, P^{2*}, \cdots, P^{r*}\} = \{P^1_1, P^1_2, \cdots, P^1_r\}$, *and* $r \geq 3$.

Proof. *In a* KSP_l, *when at least three shortest paths intersect centering around a crossing node, there may be hidden shortest paths that are shorter than* P^k_l. *But there is not any hidden path until the first crossing node is appeared three times. If the first crossing node is checked on the r-th path in the* KSP_l, *the crossing node is appeared three times by the r-th path. The hidden shortest paths to pass through the crossing node in the fourth is appeared after the r-th path. Therefore* $r \geq 3$, *and there is not any hidden path from the first path to the r-th path. The hidden shortest path should be longer than the r-th path.*
Therefore $\{P^{1*}, P^{2*}, \cdots, P^{r*}\} = \{P^1_1, P^2_1, \cdots, P^1_r\}$.

Lemma 2. *In a* KSP_l, *if there is no crossing node, then the* KSP_l *is the optimal solution.*

Proof. *If there is not any crossing node in the* KSP_l, *then* $r \geq K$ *and by Lemma 1, the* KSP_l *is the optimal solution.*

If the first crossing node is detected on the k - th path, P^k_l, there is no hidden path in the KSP_l, because the first hidden path can be existed in the next of the k - th path at the earliest. Therefore the KSP_l is the optimal solution.

Lemma 3. *In the* $K = 3$ *shortest path problem,* $KSP_{l=1}$ *is the optimal solution.*

Proof. *It becomes clear that a first crossing node is appeared at the earliest on the 3rd path in the* $KSP_{l=1}$. *It follows from Lemma 1 that* $r \geq K = 3$, *then* $\{P^{1*}, P^{2*}, P^{3*}\} = \{P^1_1, P^2_1, P^3_1\}$. *Therefore if* $K = 3$, $KSP_{l=1}$ *is the optimal solution.*

3.2 Dual Approach

The method to find crossing nodes and to break inward arc is useful for a small-size K shortest paths problem, but for big-size K shortest paths problem, this method may not guarantee to solve in a reasonable time, because it may be difficult to find out crossing nodes and to expose hidden shortest paths in the KSP_l.

In order to check some crossing nodes and to expose hidden shortest paths systematically, we can couple a dual approach to this improvement procedure.

Let π_i be the label value on node i of the shortest path tree $SPT(N_T, A_T)$, which is $T(s)$ rooted at node s by Dijkstra method.

Then $\pi_s = 0$

$$\pi_j = \min\{\pi_i + c_{ij}\},\ \forall j \in N \setminus s.$$

Let $\Pi = \{\pi_1, \pi_2, \cdots, \pi_n\}$.

Then Π is a dual vector and satisfies the complimentary slackness condition (C. S. C) related to SPT, and its dual are rewritten like below ;

$$\pi_j \leq \pi_i + c_{ij},\ \forall (i,j) \in A \ \cdots\cdots \text{(Dual Feasibility)}$$
$$\pi_j = \pi_i + c_{ij},\ \forall (i,j) \in A_T \ \cdots\cdots \text{(C.S. Conditions)}$$

Therefore, $(i,j) \notin A_T \Rightarrow \pi_j \leq \pi_i + c_{ij}$

Generally, $\overline{c}_{ij}$, a reduced cost relative to SPT, is

$$\overline{c}_{ij} = \pi_i + c_{ij} - \pi_j > 0, \quad \text{on } (i, j) \notin A_T$$
$$\overline{c}_{ij} = 0, \quad \text{on } SPT\ (N_T, A_T)$$

Let a detouring incremental cost be $LP(SP(i, j)) - LP(SP(i^*, j))$, $(i, j) \notin A_T$, $(i^*, j) \in A_T$. Because $SP(i^*, j)$ is the shortest path passing through $IA\ (i, 1)$ and $SP(i,j)$ is the path passing through $IA(i,a)$, where $a \geq 2$. The length of a shortest path $SP(i^*,j)$, which passes through arc (i^*,j), $(i^*,j) \in A_T$, is $\overline{c}_{ij}$ shorter than the length of a path $SP(i, j)$ which passes an inward arc $(i,j) \notin A_T$.

If $\overline{c}_{ij} = 0$, then the $arc\ (i, j)$ is an arc of SPT, and is an arc of a path of KSP_l.

Theorem 2. *If* $\overline{c}_{ij} > 0$, *then* $\overline{c}_{ij}$ *is a detouring incremental cost*

Proof. *If* $\overline{c}_{ij} > 0$, *then the arc (i, j) is not any arc of SPT,*

$$LP\ (SP\ (i, j)) = \pi_i + c_{ij} + \delta_j$$
$$LP\ (SP\ (i^*, j)) = \pi_j + \delta_j.$$

And then, $LP\ (SP\ (i, j)) - LP\ (SP\ (i^*, j))$

$$= (\pi_i + c_{ij} + \delta_j) - (\pi_j + \delta_j)$$
$$= c_{ij} + \pi_i - \pi_j$$
$$= \overline{c}_{ij}$$

Therefore, $\overline{c}_{ij}$ *is a detouring incremental cost.*

Let $LP\ (HP\ (i, j, m))$ be the length of a hidden shortest path which detours through an inward $arc\ (i, j)$ and an outward $arc\ (j, m)$, $OA(j,b)$, $b \geq 2$, of a crossing node j.

And then, $LP\ (SP\ (j, m)) = LP\ (SP\ (r, j, m))$, $(r, j) \in A_T$, $SP\ (r, j, m)$ is a second or a next path which passes through $arc\ (r, j)$ in the KSP_l.

Lemma 4. $LP\ (HP\ (i,j, m))$

$$= LP\ (SP\ (j, m)) + \overline{c}_{ij},\ \forall j \in N_T.$$

Proof. $LP\ (HP\ (i, j, m)) = \pi_i + c_{ij} + c_{jm} + \delta_m$

$$= \pi_i + (\overline{c}_{ij} + \pi_j - \pi_i) + c_{jm} + \delta_m$$
$$= (\pi_j + c_{jm} + \delta_m) + \overline{c}_{ij}$$
$$= LP(SP(j, m)) + \overline{c}_{ij},\ \forall (j,m) \neq OA(j, 1) \quad (2.2)$$

Therefore, $LP\ (HP\ (i, j, m)$ which passes through second or next inward $arc\ (i, j)$ and second or next outward $arc\ (j, m)$ of CN_i can be calculated by equation (2,2) in case that $\overline{c}_{ij} > 0$. That is, in order to expose hidden shortest paths, it is needed to check the value of $\overline{c}_{ij}$ of inward $arc\ (i, j)$ toward CN_j, and the value of $LP\ (SP\ (j, m))$ on the all outward $arc\ (j, m)$, $\forall (j, m) \neq OA\ (j, 1)$, then we can compute the value of $LP(HP(i,j, m)$, $\forall (j,m) \neq OA(j, 1)$ by equation (2.2).

Lemma 5. *If arc (i, j) is an arc of an exposed hidden path, then* $LP(HP(i, j, m)) = \bar{c}_{ij} + LP(SP(j, m)) < LP(P^k_l)$.

Proof. *Exposed hidden paths must be shorter than* P^k_l *in the* KSP_l.
Then $LP(HP(i, j, m)) = LP(SP(j, m)) + \bar{c}_{ij} < LP(P^k_l)$.

Therefore, if $\bar{c}_{ij} + LP(SP(j, m)) \geq LP(P^k_l)$, then $HP(i, j, m)$ is not shorter than P^k_l and it is not exposed.

Let $LP(HP(i, j, m^*)) = \min_m \{LP(HP(i,j,m))\}, \forall (j, m) \neq OA(j, 1)$, the exposed hidden path can be produced in a method of joining $FP(s, (i, j))$ and $FP((j, m^*), t)$, which is the spur of $HP(r, j, m^*)$.

But this equation (2.2) can not easily be applied to a KSP_l which have cross arcs and cross subpaths, because the outward *arc (j, m)* is only one in KSP_l and *arc (j, m)* $\in A_T$. We can't calculate and check the hidden paths which detour through inward *arc (i, j)*, $\forall i$, and outward *subpath (j, m, n)*, $\forall n$.

So, we find out all apparent paths $HP(r, j, m), \forall (j, m) \neq OA(j, 1)$, which pass CN_j in KSP_l and their length $LP(HP(r, j, m))$, and then calculate the length of detouring hidden paths, $LP(HP(i, j, m))$ by equation (2.3), after then choose detouring hidden paths, $HP(i, j, m)$, which is shorter than $LP(P^k_l)$, and replace P^k_l by $HP(i, j, m)$.

Lemma 6. *In this improvement algorithm, If* $\bar{c}_{ij} \geq LP(P^k_l) - LP(P^1), \forall (i, j) \notin A_T$, *the present* KSP_l *is the optimal solution* (KSP^*).

Proof. *In the case that* $\bar{c}_{ij} \geq LP(P^k_l) - LP(P^1_l)$ *then,* $LP(HP(i, j, m)) = LP(SP(j,m)) + \bar{c}_{ij} > LP(P^1_l) + \bar{c}_{ij} \geq LP(P^k_l)$, *for* $LP(SP(j, m)) > LP(P^1_l)$. *Therefore, if* $\bar{c}_{ij} \geq LP(P^k_l) - LP(P^1_l)$, $\forall (i, j) \notin A_T$, *the present solution,* KSP_l *is the optimal condition.*

When we need to check inward arcs of a crossing node to expose a hidden path, the above three lemmas are useful to reduce the number of computational iteration.

Let *arc (r, q)* be $IA(q, 1)$ of CN_q, and *arc (p, q)* be $IA(q, 2)$ of CN_q, and also there are some inward *arc (i, q)*, $\forall i$.

Lemma 7. *If* $LP(HP(p, q, m)) \geq LP(P^k_l)$, *then* $LP(HP(i, q, m)) \geq LP(P^k_l), \forall i$.

Proof. $LP(HP(p, q, m)) = LP(SP(q, m)) + \bar{c}_{pq}$, *and* $LP(HP(i, q, m)) = LP(SP(q, m)) + \bar{c}_{iq}, \forall i$.
Let $GAP = \bar{c}_{iq} - \bar{c}_{pq}, \forall i$. *Then arc (p, q) is* $IA(q, 2)$ *of* CN_q, *so* $LP(HP(p, q, m))$ *is GAP shorter than* $LP(HP(i, q, m))$.
Therefore if $LP(HP(p, q, m)) \geq LP(P^k_l)$, *then* $LP(HP(i, q, m)) \geq LP(P^k_l), \forall i$.

Therefore, if $LP(SP(p, q, m)) \geq LP(P^k_l)$, there is no more candidate hidden path passing through CN_q, which can enter into the KSP_l.

In order to get the optimal solution $KSP^* = \{P^{1*}, P^{2*}, \cdots, P^{k*}\}$,

1. Make the ascending order set LIST of $(\bar{c}_{ij}), \forall j \in$ IN, which are $0 < \bar{c}_{ij} < LP(P^k_l) - LP(P^1_l)$, in the KSP_l.
2. Find out arc (p, q) whose $\bar{c}_{pq} = \min(\bar{c}_{ij}) > 0$, and arc $(r, q) \in A_T$ in the KSP_l.
3. Find out paths $HP(r,q,m), \forall m, (r,q) \in A_T$ in the KSP_l, which contains $FP(1, q)$.
4. If $LP(HP(r, q, m)) + \bar{c}_{pq} < LP(P^k_l)$, then we expose new shortest path $HP(p, q, m)$ with joining $FP(1, (p, q))$ and $FP((q, m), t)$. Otherwise go to 2, and select next $\bar{c}_{pq}$. $FP(1, (p, q))$ is a forepart of the chosen path $FP(1, (p, q), t)$, and $FP((q, m), t)$ is a spur part of $HP(r, q, m)$.
5. $HP(p,q,m)$ replaces P^k_l, and rearrange paths of KSP_l in ascending order of its length in order to get KSP_{l+1}.
6. We should repeat the above routine till LIST of $\bar{c}_{ij} > 0$ is empty, or there is not any hidden path shorter than P^k_l.

In case that K may be more or less larger than the number of shortest paths which we can get, in the initial solution, in the improvement procedure we can add some exposed hidden paths, and get K shortest paths.

In the optimal condition, we can reach one of the following cases in the KSP_l,

Case 1 ; $\bar{c}_{pq} \geq LP(P^k_l) - LP(P^1)\ \forall (p,q)$.
- *That is,* $(LP(HP(r, q, m)) + \bar{c}_{pq}) \geq P^k_l$.
- *All of the exposed paths are longer than* P^k_l.

Case 2 ; *There is no crossing node till the (K-1)th path.*
- *Each node is appeared at most two times or only same inward arcs are appeared several times.*
- *There is no intersection node in* KSP_l.

Case 3 ; *LIST of* $(\bar{c}_{ij}) > 0$ *is empty.*

Case 4 ; $K \leq 3$.

4. New K Shortest Paths Algorithm

We focus on the K shortest paths problems in a directed network that may contain positive length arcs. Our goal is to provide a new algorithm that globally sharp for these problems.

This method, which we call *KSP-DSA*, may be described as follows.

4.1 KSP-DSA

Step 0. Initialization.
Given a network $G = (N, A)$, input the network structure and distance data.
$l = 1$, $KSP_l = \phi$.

Step 1. Produce $T(s)$ and $T(t)$ with the Dijkstra method, and let them be merged and make *DSA*, $T(s,t)$.

Step 2. Compute $SP(u, v)$ and $LP(SP(u, v)), \forall (u,v) \in A$, and select K shortest paths in ascending order of $LP(SP(u,v))$ *from* $T(s,t)$.
$KSP_l = \{P^1_l, P^2_l, \cdots, P^k_l\}$.
If $K \leq 3$, then KSP_l is the optimal solution set. STOP.

Step 3. Calculate $(\bar{c}_{ij})$, which is $j \in IN$, $(i, j) \in KSP_l$.
$\bar{c}_{ij} = c_{ij} + \pi_i - \pi_j$.

Step 4. Make *LIST* of inward arc (i,j) which has $\bar{c}_{ij}$, $0 < \bar{c}_{ij} < LP(P^k_l) - LP(P^1_l)$, and arrange $(\bar{c}_{ij})$ in ascending order.

Step 5. Check the possibility for improvement.
1. IF $LIST = \phi$, then STOP. KSP_l is optimal.
2. Select a next $\bar{c}_{ij}$ in the *LIST*, $LIST = LIST - (i, j)$, $JJ = 1$. $\bar{c}_{pq} = \bar{c}_{ij}$

Step 6. Expose the hidden detouring paths and improve present solution.
1. Pick out all paths which contain *arc (r, q)*, $(r, q) \in A_T$ in the KSP_l, excepting the shortest one and P^k_l.
→ $HP(r, q, m), \forall m \in KSP_l$.
2. Choose JJ-th shortest path $HP(r,q, m*)$, whose length is $(LP(HP(r, q, m*)) + \bar{c}_{pq}) < LP(P^k_l)$, and $JJ = JJ + 1$. Otherwise, go to step 6-5.
3. Expose hidden shortest paths $HP(p, q, m^*)$ by joining $FP(1, (p, q))$ and $FP((q, m^*), t)$, the spur of $HP(r, q, m^*)$.
4. Replace P^k_l by $HP(p, q, m^*)$, and make and rearrange KSP_{l+1} in ascending order,

and $l = l + 1$, and go to step 6-2.

5. If $JJ = 1$, then remove $\bar{c}_{pq}$, $\forall p$ from *LIST*, and arrange *LIST*.
Otherwise remove only $\bar{c}_{pq}$ from *LIST*.
Then go to step 5-1,

<u>Lemma 8.</u> *KSP-DSA is an algorithm with complexity* $O(Kn^2)$.

<u>Proof.</u> *The major operations required by the new algorithm is as follows. For the computational complexities in step 1 ~ step 4, to make* $T(s, t)$ *is required* $O(n^2)$ *because the complexity for Dijkstra algorithm is at most* $O(n^2)$, *to compute* $SP(u, v)$ *and to select K shortest path in ascending order is* $O(n^2)$, *to compute* $(\bar{c}_{ij})$, $\forall(i, j)$ *and to check* $0 < \bar{c}_{ij} < LP(P^k_1) - LP(P^1_1)$ *and to arrange in ascending order is* $O(n)$. *To improve* KSP_1, *in step 5, requires at most* $O(n)$, *and in step 6, requires* $O(Kn)$ *to find out all paths containing arc* (r, q) *in present K shortest paths and requires at most* $O(n)$ *to expose hidden paths. We need to repeat the improvement procedure at most n times because one node among interaction nodes should be checked if it is passed by a hidden path. Therefore the total complexity bound is* $O(Kn^2)$ *in this algorithm as follows ;*

$$O(n^2) + O(n^2) + O(n) + (n)*\{O(n) + O(Kn) + O(n)\} = O(Kn^2).$$

In the case of $K \le 3$, this algorithm works within time complexity $O(n^2)$, because the improvement procedure (step 3~6) is not required and the initial solution, KSP_1, is optimal.

5. Application

To see how the algorithm *KSP-DSA* works, we consider an example network given in [Figure 1], where the $T(s)$, $T(t)$, and $T(s, t)$ of the network has been shown.
We will solve the $K = 10$ shortest path problem of the network in the [Figure 1].

Step 0. Initialization
Input $G = (N, A)$, $l = 1$.
$IN = \{3, 4, 5, 6\}$

Step 1. Produce $T(s, t)$ like in [Figure 1].

Step 2. Compute and select 10 shortest paths from $T(s,t)$.

P^1_1 : 1-3-4-7-8, $LP(P^1_1) = 20$
P^2_1 : 1-4-7-8, $LP(P^2_1) = 22$
P^3_1 : 1-3-7-8, $LP(P^3_1) = 24$
P^4_1 : 1-3-4-6-8, $LP(P^4_1) = 30$
P^5_1 : 1-2-6-8, $LP(P^5_1) = 33$
P^6_1 : 1-3-5-8, $LP(P^6_1) = 34$
P^7_1 : 1-2-4-7-8, $LP(P^7_1) = 35$
P^8_1 : 1-3-4-6-5-8, $LP(P^8_1) = 36$
P^9_1 : 1-3-4-5-8, $LP(P^9_1) = 38$
P^{10}_1 : 1-2-3-4-7-8, $LP(P^{10}_1) = 38$

Step 3. Calculate $(\bar{c}_{ij})$, $j \in IN$, $(i, j) \in KSP_1$.
$\bar{c}_{ij} = c_{ij} + \pi_i - \pi_j$
$\bar{c}_{14} = 2$, $\bar{c}_{65} = 2$, $\bar{c}_{26} = 3$, $\bar{c}_{37} = 4$, $\bar{c}_{24} = 15$.
Node $7 \notin IN$

Step 4. List $(\bar{c}_{ij})$, $0 < \bar{c}_{ij} < LP(P^k_1) - LP(P^1_1)$, and arrange in ascending order.
$LP(P^k_1) - LP(P^1_1) = 38 - 20 = 18$.
$LIST = \{(1,4), (6,5), (2,6), (2,4)\}$.

(Iteration 1)

step 5. Checking the improvement.
2. $\bar{c}_{14} = 2$. $LIST = \{(6, 5), (2, 6), (2, 4)\}$.
$JJ = 1$, $(p, q) = (1, 4)$.

step 6. Expose a hidden detouring path.
1. $(r,q) = (3,4)$, Pick out paths containing *arc* (3, 4).

2. Choose $P^4_1 = HP(3, 4, 6)$.

JJ	Path No. in KSP_1	Route	$LP(P^j_l)$	Remark
	P^1_1	1-3-4-7-8	20	Excepting the shortest.
1	P^4_1	1-3-4-6-8	30	
2	P^8_1	1-3-4-6-5-8	36	
	P^9_1	1-3-4-5-8	38	Excepting the longest.

$LP(P^4_1) + \bar{c}_{14} = 30 + 2 = 32 < 38$.
$JJ = 2$.

3. Expose $HP(1,4,6)$ by joining $FP(1, (1, 4))$ and $FP((4, 6), 8)$
$HP(1, 4, 6) = EP$: 1-4-6-8, $LP(EP) = 32$.

4. Improve present solution ; KSP_2

P^1_2 : 1-3-4-7-8, $LP(P^1_2) = 20$
P^2_2 : 1-4-7-8, $LP(P^2_2) = 22$
P^3_2 : 1-3-7-8, $LP(P^3_2) = 24$
P^4_2 : 1-3-4-6-8, $LP(P^4_2) = 30$
P^5_2 : 1-4-6-8, $LP(P^5_2) = 32$
P^6_2 : 1-2-6-8, $LP(P^6_2) = 33$
P^7_2 : 1-3-5-8, $LP(P^7_2) = 34$
P^8_2 : 1-2-4-7-8, $LP(P^8_2) = 35$
P^9_2 : 1-3-4-6-5-8, $LP(P^9_2) = 36$
P^{10}_2 : 1-3-4-5-8, $LP(P^{10}_2) = 38$

Then go to step 6-2.

2. Choose $JJ = 2$, $P^8_1 = HP(3, 4, 6)$.
$LP(P^8_1) + \bar{c}_{14} = 36 + 2 = 38$
$= LP(P^{10}_1)$.
Go to step 6-5.

5. $JJ = 2$, then go to step 5-1.

(Iteration 2)

step 5. Checking the improvement.
2. $\bar{c}_{65} = 2$, $LIST = \{(2, 6), (2, 4)\}$.
$JJ = 1$
$(p, q) = (6, 5)$.

step 6. Expose a hidden detouring path.
1. $(r, q) = (3,5)$, Pick out paths containing *arc* (3, 5).

JJ	Path No. in KSP_1	Route	$LP(P^j_l)$	Remark
	P^7_2	1-3-5-8	34	Excepting the shortest.

4. $JJ = 1$, then remove $\bar{c}_{p5}$, $\forall p$, and go to step 5-1.

(Iteration 3)

step 5. Checking the improvement.
2. $\bar{c}_{26} = 3$, $LIST = \{(2, 4)\}$.
$JJ = 1$
$(p,q) = (2,6)$.

step 6. Expose a hidden detouring path.
1. $(r,q) = (4,6)$, Pick out paths containing *arc* (4,6).

JJ	Path No. in KSP_1	Route	$LP(P^j_l)$	Remark
	P^4_2	1-3-4-6-8	30	Excepting the shortest.
1	P^8_2	1-3-4-5-6-8	36	

3. Choose $P^8_2 = HP(4, 6, 5)$.
$LP(P^8_2) + \bar{c}_{26} = 36 + 3 = 39 > 38$.
Then go to 6-4.

4. $JJ = 1$, then remove $\bar{c}_{p6}$, $\forall p$, and go to step 5-1.

(Iteration 4)

step 5. Checking the improvement.
2. $\bar{c}_{24} = 15$, $LIST = \{\phi\}$.
$JJ = 1$, $(p, q) = (2, 4)$.

step 6. Expose a hidden detouring path.
1. $(r,q) = (3,4)$, Pick out paths containing *arc*(3,4).

2. Choose $P^4_2 = SP\ (3, 4, 6)$.
$LP(P^4_2) + \bar{c}_{24} = 30 + 15 = 45 > LP(P^{10}_2)$
Then go to step 6-4.

JJ	Path No. in KSP_1	Route	$LP\ (P^j_i)$	Remark
	P^1_2	1-3-4-7-8	20	Excepting the shortest.
1	P^4_2	1-3-4-6-8	30	
2	P^9_2	1-3-4-6-5-8	36	
	P^{10}_{12}	1-3-4-5-8	38	Excepting the longest.

4. $JJ = 1$, then remove $\bar{c}_{p4}, \forall p$, and go to step 5-1.

(Iteration 5)

step 5. Checking the improvement.

1. $LIST = \emptyset$, then KSP_2 is optimal. STOP.

The optimal solution KSP^*

P^{1*} : 1-3-4-7-8, $LP(P^{1*}) = 20$
P^{2*} : 1-4-7-8, $LP(P^{2*}) = 22$
P^{3*} : 1-3-7-8, $LP(P^{3*}) = 24$
P^{4*} : 1-3-4-6-8, $LP(P^{4*}) = 30$
P^{5*} : 1-4-6-8, $LP(P^{5*}) = 32$
P^{6*} : 1-2-6-8, $LP(P^{6*}) = 33$
P^{7*} : 1-3-5-8, $LP(P^{7*}) = 34$
P^{8*} : 1-2-4-7-8, $LP(P^{8*}) = 35$
P^{9*} : 1-3-4-6-5-8, $LP(P^{9*}) = 36$
P^{10*} : 1-3-4-5-8, $LP(P^{10*}) = 38$

6. Future Research

This article presents a new algorithm for K Shortest Paths Problem which has time complexity $O\ (Kn^2)$. Especially in the case of $K \leq 3$, this algorithm works within time complexity $O\ (n^2)$.

In this algorithm we can make an initial solution with K paths among shortest paths from s to t through each node, and improve and reach an optimal solution with dual approach which gets detouring incremental distance, $\bar{c}_{ij}$, and applies a concept of breaking inward arcs, merging sub-path, and exposing hidden shortest paths around crossing nodes.

In the near future, all pair of K shortest paths problem algorithm development and numerical comparisons will be the aim of our research for the application in the real fields like as ITS (Intelligent Transport Systems), transportation planning analysis, and transportation goods through a distribution network in the logistics management, telecommunications , VLSI design [9] and so on.

REFERENCES

[1] Ahuja, R.K., T.L. Magnanti, and J.B. Orlin, Network Flows, Prentice Hall, Englewood Cliffs, NJ,1993.
[2] Chang, B.M., A study on the new algorithm for K shortest paths problem, Korean Management Science Review, 15, 2(1998), pp. 229-237.
[3] Dijkstra, E.W., A note on two problems in connection with graphs, Numerische Mathematik, 1(1959), pp.269-271.
[4] Dreyfus, S., An appraisal of some shortest path algorithms, Oper Res, 17, 2(1969), pp. 395-412.
[5] Eppstein, D., Finding the K shortest paths, SIAM J. Comput., 28, 2 (1998), pp.652-673.
[6] Glover, F., R. Glover, and D. Klingman, Computational study of an improved shortest path algorithm, Networks, 14 (1984), pp.25-36.
[7] Hadjiconstantinou, E., and N. Christofides, An efficient implementation of an algorithm for finding K shortest paths, Networks, 34 (1999), pp.88-101.
[8] Katoh, N., T. Ibaraki, and H. Mine, An efficient algorithm for K shortest simple paths, Networks, 12(1982), pp.411-427.
[9] Lalgudi, K.N. and M.C. Papaefthymiou, Computing strictly-second shortest paths, Information Processing Letters, 63 (1997), pp. 177-181.
[10] Lawler, E., A procedure for computing the K best solutions to discrete optimization problems and its application to the shortest path problem, Management Sci, 18, 7(1972), pp.401-405.
[11] Lawler, E., Combinatorial Optimization : Networks and Matroids, Holt Reinhart and Winston, New York, 1976.
[12] Shier, D., On algorithm for finding the K shortest paths in a network, Networks, 9 (1979), pp.195-214.
[13] Yen, J. Finding the K shortest loopless paths in a network, Management Sci, 17 (1971), pp.712-716.
[14] Ziliaskopoulos, A., D. Kotzinos, and H. Mahmassani, Design and implementation of parallel time-dependent least time path algorithm for intelligent transportation systems applications, Transportation Res-C, 5, 2 (1997), pp.95-107.