經營科學
第15卷 第2號
1998年 11月

복수 최단 경로 문제의 새로운 해법 연구*

장 병 만**

A Study on the New Algorithm for K Shortest Paths Problem*

Byung Man Chang**

■ Abstract ■

This paper presents a new algorithm for the K Shortest Paths Problem which is developed with a Double Shortest Arborescence and an inward arc breaking method.

A Double Shortest Arborescence is made from merging a forward shortest arborescence and a backward one with Dijkstra algorithm, and shows us information about each shorter path to traverse each arc. Then K shorter paths are selected in ascending order of the length of each short path to traverse each arc, and some paths of the K shorter paths need to be replaced with some hidden shorter paths in order to get the optimal paths.

And if the cross nodes which have more than 2 inward arcs are found at least three times in K shorter path, the first inward arc of the cross node is broken and a new shorter path is exposed.

If this exposed path is shorter than the Kth shorter path, the exposed path replaces the Kth shorter path. This procedure is repeated until the cross nodes are not found in K shorter paths, and then the K shortest paths problem is solved exactly.

This algorithm are computed with complexity $O(n^3)$ and especially $O(n^2)$ in the case $K=3$.

I. 서 론

본 연구에서는 복수최단경로문제(K-Shortest Path Problem)의 새로운 해법을 제시하고자 한다. 복수최단경로문제는 ITS(Intelligent Transport System)내의 차량안내시스템 내에서 모든 지점간의 복수최단경로를 구하는 문제 등의 요긴한 해법 연구에 사용될 수 있다. 이 복수최단경로문제는 확

* 본 연구는 서울산업대학교 대학학술연구비에 의하여 지원되었음.
** 서울산업대학교 산업공학과

을 가지는 경로를 허용하지 않는 경우를 다룬다. 무방향과 유방향의 네트워크 G(V,E)상에서 양방향의 호를 가지며, V는 n개 마디의 집합이고, E는 m개 호의 집합이며, m≥n의 경우에서 비음의 호의 길이를 가지며 호(u,v)∈E 에서 d(u,v) = d(v,u)를 가정한다.

Yen[8]은 유방향이나 무방향 네트워크에서 후보 경로들을 구하고 그 중 가장 짧은 경로를 찾는 과정을 반복하는 $O(Kn^3)$의 계산 복잡도를 가지는 해법을 개발하였다. Katoh외 2인 [4]은 무방향 네트워크상에서 두 지점간 최단경로를 구한 뒤, 매번 특정마디와 최종마디까지 각 마디와 호를 지나는 최단경로 가운데 3가지의 짧은 후보경로들을 구하고 그 중 가장 짧은 경로를 찾는 방법을 반복하는 $O(Kn^2)$의 계산 복잡도를 가지는 우수한 해법을 개발했다. 또한 장병만[9]이 K=3의 최단경로를 $O(n^2)$에 구하는 이중최단거리나무해법을 발표하였다.

본 연구에서는 출발지에서 모든 마디까지의 최단 경로와 도착지에서 모든 마디까지의 최단 경로를 각각 구한 후 결합시켜서, 모든 호에 대하여 각 호를 지나가는 최단 경로의 정보를 얻고, 이 경로들 가운데서 길이가 짧은 K개의 경로를 선정하여 초기해로 삼고, 이 K개의 경로들 보다 짧은 잠재경로(Hidden Path)들을, 짧은 경로들이 교차하여 지나감으로 교차가 생긴 교차발생마디를 중심으로 계속 찾아내어 해를 개선시키면서 최적해를 찾는 $O(n^3)$의 새로운 해법을 제시하고자 한다.

K≤3이면 초기해가 바로 최적해가 된다. 출발지를 제외하고 각 마디에서의 진출호가 각각 1개씩인 경우도 초기해가 최적해로 된다. 이 때의 계산 복잡도는 $O(n^2)$이 된다.

Ⅱ. 초기해의 형성

출발지 S에서 모든 마디까지의 전방향 최단 경로나무(Forward Shortest Arborescence)를 Dijkstra법으로 구한 최단 경로 나무를 T(s)라 하고, 이 때 S에서 각 마디 i까지의 최단 경로는 P(s,i), 이 경로의 길이는 a_i라 하자. 출발지 S에서 도착지 t까지의 최단 경로는 P(s,t), 최단 길이는 a_t이다. 각 마디 i의 직전 마디를 f_i라고 하며, 각 마디 i의 최단 경로 정보의 꼬리표는 (a_i, f_i)로 표시한다. 마찬가지로 t에서 모든 마디까지의 역방향 최단 경로나무(Backward Shortest Arborescence)는 T(t)라고 하고, 각 마디 j까지의 최단 경로는 P(t,j), 이 경로의 길이는 b_j 라고 하자. t에서 s까지의 최단 경로는 P(t, s)이며, 최단 길이는 b_s 이다. 각 마디 j의 직후 마디를 h_j 라고 하며, 각 마디 j의 최단 경로 정보의 꼬리표는 (b_j, h_j)로 표시한다. $a_t = b_s$이다.

$P^k(s,t)$는 S에서 t까지의 k번째 최단 경로라 하고, 최단 경로가 아니지만 구한 해에서 k번째로 짧은 경로는 $FP^k(s,t)$, 또는 FP^k라고 하자. 그 경로는 다음과 같이 표현한다.

$$P^k(s,t) = [v^k(1), v^k(2), \cdots, v^k(q_k)]$$

단, $v^k(1) = s$, $v^k(q_k) = t$이다.

그리고 $P'(s,t) = P' = s \xrightarrow[T(s)]{} t = s \xrightarrow[T(s)]{} t$ 로 표현한다.

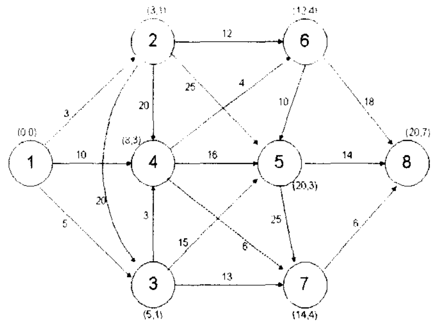
T(s)와 T(t)를 연결시키면 각 호 (u,v)를 통과하는 s에서 t까지의 최단 경로와 거리를 구할 수 있다. T(s)와 T(t)를 결합시켜 각 호를 통과하는 s에서 t까지의 최단 경로와 거리에 관한 정보를 알려주는 나무를 이중 최단 거리 나무(Double Shortest Arborescence:DSA) T(s,t)라고 하자.

T(s,t)에서 호(u,v)를 통과하는 s에서 t까지의 최단 경로를 SP(u,v)라 하고, LP(u,v)는 그 경로의 길이라고 하자. 그러면

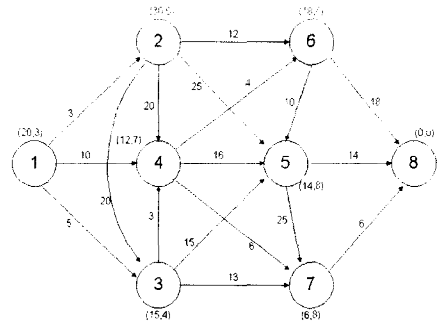
SP(u,v)는 $P'(s,u) \longrightarrow P'(v,t)$ 이며,

$s \xrightarrow[T(s)]{} u \longrightarrow v \xrightarrow[T(s)]{} t$ 이다.

LP(u,v)는 $a_u + c_{uv} + b_v$이다.

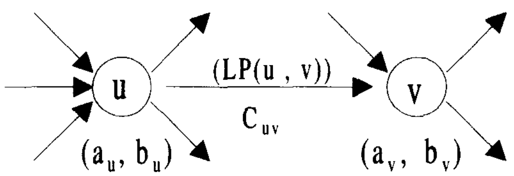


(a_i, f_i) : a_i - 출발지 1에서 마디 i 까지의 최단 거리
f_i - 한 단계 전 마디
(1) 전방향 최단 거리 나무 T(s)



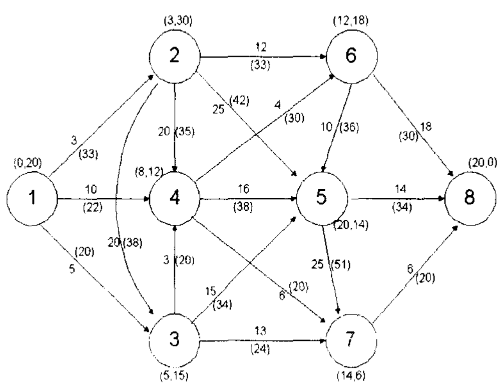
(b_j, h_j): b_j - 도착지 8에서 마디 j까지의 최단 거리
h_j - 한 단계 후 마디
(2) 후방향 최단 거리 나무 T(t)

〈그림 1〉 T(s)와 T(t) 최단 거리 나무



(a_u, b_u) : a_u - 출발지에서 마디 u까지의 최단거리
b_u - 도착지에서 마디 u까지의 최단거리

〈그림 2〉 이중 최단 거리 나무 T의 표시



〈그림 3〉 이중 최단 거리 나무 T(s,t)

T(s,t)에서 모든 호에 대해서 각 호(u,v)를 지나가는 최단 경로 SP(u,v)를 구한 다음, 이 경로들을 경로 길이의 오름차순으로 나열하면 K개의 최단 경로의 초기해를 구할 수 있다.

K=5인 사례 문제에 대한 초기해는 다음과 같다.

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

Ⅲ. 해의 개선

이중 최단 거리 나무 T(s,t)에서 각 호(u,v)를 지나는 각 최단 경로 SP(u,v)를 구하고, 이 경로들을 길이의 오름차순으로 나열하여 구한 k개의 경로 집합 KSP={FP^1, ..., FP^k}는 k개의 최단 경로가 아니므로, 이 경로들 중의 일부를 더 짧은 경로들을 찾아내어 대체하여서 해를 개선시키는 과정을 반복하여 최적해를 구해야 한다.

정의 1. 노출 경로 (Exposed Path)와 잠재 경로 (Hidden Path)

노출 경로는 T(s,t)상에서 적어도 하나의 호(u,v)를 처음으로 통과하는 $s \xrightarrow[T(s)]{} u \rightarrow v \xrightarrow[T(t)]{} t$의 경로이며 T(s,t)에서 구한 SP(u,v), ∀(u,v)에 의해 만들어진 경로들이다.

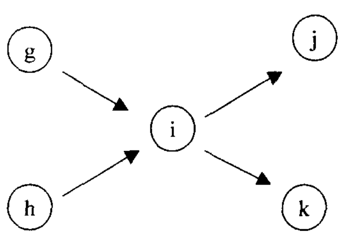
각 노출경로 상에서는 적어도 1개 이상의 호가 처음으로 통과된다. 초기해로 나온 K개의 경로들은 모두 노출 경로들이다.

잠재 경로는 T(s,t)에서 구한 더 짧은 길이의 노출 경로들에 의해서 이미 통과되어진 호들만으로 이루어진 경로이며, 경로의 길이가 FP^k보다 짧아도 T(s,t) 상에서는 나타나지 않는 경로들이다.

이 FP^k보다 짧은 모든 잠재 경로들을 찾아내어 FP^k와 같은 긴 노출 경로를 대체 하는 방법으로 최적의 복수 최단 경로들을 찾아내는 방법을 찾아내고자 한다.

정의 2. 교차 마디 (Cross Node)

교차 마디는 진입호(inward arc)가 2개 이상인 마디 가운데서, 2개 이상의 진입호들이 K개의 노출경로 속에 3회 교차되어 나타나는 마디를 가리킨다. KSP에서 그림에서 처럼 3회 이상 마디 i가 나타나고, 부분 경로 g→i→j, g→i→k, h→i→j가 들어 있는 노출 경로가 3개 이상 나타나는 경우, 마디 i는 교차가 발생한 교차발생마디이다.

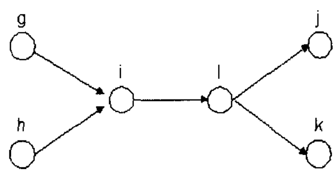


호(g, i) : 진입호 중 제일 짧은 경로와 연결되는 호
호(h, i) : 진입호 중 2번째 짧은 경로와 연결되는 호
호(i, j) : 진출호 중 제일 짧은 경로와 연결되는 호
호(i, k) : 진출호 중 2번째 짧은 경로와 연결되는 호

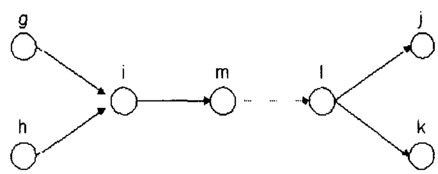
〈그림 4〉 교차 마디

상기 그림 4에서처럼 교차 마디 i에서 교차가 일어났으면, FP^k보다 짧으면서 부분 경로 h→i→k를 포함하는 잠재 경로가 발생할 수 있다. 호(h,i)는 부분 경로 h→i→j, 호(i,k)는 g→i→k를 포함하는 더 짧은 노출 경로에 의해 통과되었기 때문에 부분 경로 h→i→k는 잠재 경로속에 있어서 나타나지 않을 수 있다.

그리고 상기의 교차는 교차 마디에서처럼 다음 그림 5와 같은 교차호, 교차 부분 경로가 있는 경우에서도 발생한다.



(1) 교차 호



(2) 교차 부분 경로

〈그림 5〉 교차 호, 교차 부분 경로

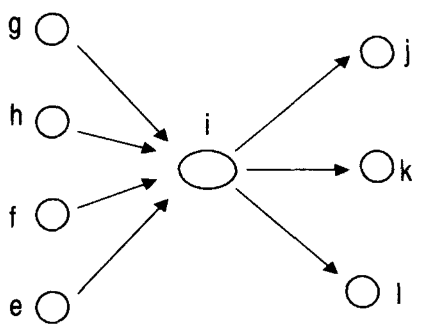
교차 마디, 교차 호, 교차 부분 경로가 K개의 경로 중 나타나는 경우, 교차 마디 i가 g→i→j와 g→i→k 또는 교차호, 교차 부분경로 g→i⋯j와 h→i⋯j 속에서만 나타난다면, h→i⋯j 또는 g→i⋯k라는 노출경로가 FP^K보다 길기 때문에 K개 경로 속에 포함되지 않았으며, 이 마디 i에서는 교차가 발생하지 않았다. 이러한 교차마디(호, 부분경로)를 지나면서 다른 부분경로를 지나는 경로는 잠재 경로가 아니지만 FP^K보다 길므로 나타나지 않는다.

본 연구에서는 이러한 교차가 발생하는 교차 마디를 T(s,t)에서 구한 k개의 경로들 중에서 찾아내고, 각 교차마디 별로 FP^k보다 짧은 잠재 경로들을

찾아내어 내체하는 과정을 K개의 경로 중에서 잠재 경로를 모두 찾을 때까지 반복하면서 초기해를 개선시켜 최적해를 구하는 해법을 제시하고자 한다.

잠재 경로인 h→i→k나 h→i⋯l→k를 나타나게 하려면 교차 마디 i로 가장 먼저 진입하는 호 g→i를 통과중지(Breaking)시켜서, 잠재 경로의 부분 경로 h→i→k 가운데 i→k나 l→k가 노출이 되게 하여, h→i→k나 h→i⋯l→k를 지나는 잠재 경로가 나타나게 하면 된다.

그리고 그림 6처럼 마디 i로의 진입호가 많으면 먼저 나오는 진입호부터 하나씩 통과중지(Breaking)시키면서 차례로 새로운 잠재 경로들(h→i→k, h→i→l, f→i→k, f→i→l, e→i→k, e→i→l)을 찾아낼 수 있다.



〈그림 6〉 다수의 진입호와 진출호를 가지는 교차 마디 i

여기서 교차 마디나 호, 부분 경로를 찾아내는 작업이 계산 복잡도를 높이게 된다. 교차 마디, 교차 호, 교차 부분 경로를 일일이 점검하여 찾지 않고 노출 경로들에 의해 숨겨진 잠재경로들을 찾는 해법을 찾아야 한다. 본 연구에서는 KSP내의 K개 경로 집합 중에서 어떤 교차마디에서 서로 다른 2개의 진입호들이 3회 교차하여 나오면, 그 다음번째 경로들 사이에는 잠재 경로가 있는 것으로 가정하고, 이 교차마디로 2개의 진입호 가운데 먼저 들어가는 진입호를 통과 중지시켜서 숨어 있던 잠재 경로들을 노출시키고, 이 경로가 FP^k보다 짧으면 FP^k와 대체하여 KSP내에 포함시킨 후 KSP를 경로 길이의 오름차순으로 나열하여 k개의 새로운 경로를 구할 수 있게 된다. 물론 서로 다른 2개 이상의 진입호들이 3회 교차하여 나오는 마디라도, 현재 k개의 경로들 사이에서 잠재경로가 없을 수 있으나, 교차마디, 교차호, 교차 부분 경로를 찾고 또 통과 중지시켜 새롭게 드러나는 잠재 경로를 찾는 계산 복잡도가 매우 높기 때문에, 단순히 교차마디 i로 진입하는 호를 통과 중지시키고 드러나는 잠재 경로를 찾는 방법을 사용하기로 한다.

잠재 경로를 찾기 위해서는 진입호 (g, i)를 Breaking 시킨 후에는, 진입호 (h, i)까지의 부분경로와 2번째로 진입호 (g, i)를 포함하는 경로의 진출호 (i, k)에서 도착지 t로 이어진 부분 경로를 이어주면 된다. 새로운 노출된 h→i→k를 포함하는 잠재 경로 SP(i, k)의 길이는 LP(i, k)= $a_h + c_{hi} + c_{ik} + b_k$이며, 기존의 T(s, t)의 각 마디와 호의 정보를 이용하여 쉽게 계산할 수 있다.

정리 1. 처음 교차가 일어난 교차마디(호, 부분경로)가 두 가지 종류의 진입호를 가지고 3번째 나타나기까지의 경로 수가 Q개이면, 이 Q개의 경로는 모두 최적해의 확정된 일부이다.

증명) T(s,t)에서 나온 K개 경로에서 가장 먼저 나오고 교차가 발생한 교차 마디(호, 경로) i에 대해서는 3개의 부분경로 (i→k⋯l, i→k⋯m, j→k⋯l)가 각각 포함된 3개의 노출 경로가 발생한다. 잠재 경로는 교차가 발생한 후에 생기므로 처음으로 교차가 발생한 교차 마디(호, 경로)가 3번 나타나기까지의 FP^1, ⋯, FP^Q의 Q개 경로까지에는 잠재 경로가 없다. 그러므로 이 Q개의 경로는 최적해의 일부이다.

그러므로 KSP에서 같은 교차가 3번 나타나는 경우가 없으면 KSP는 K개의 최단 경로의 최적해이다. 같은 교차 마디가 3번 나타나도 FP^K가 마지막 경로인 3번째 교차마디를 포함하고 있다면 현재의 KSP가 최단 경로의 해이다.

정리 2. KSP집합 중에 교차가 발생한 교차마디(호, 부분경로)가 없으면 KSP = {FP^1, …, FP^K}가 K개의 최단 경로이다.

증명) KSP내에서 교차가 발생하지 않았다면 Q ≥ K 이다. 그러므로 정리 1에 의하면 K개의 경로가 최적해인, K개의 최단 경로이다.

정리 3. K ≤3이면 T(s,t)에서 구한 KSP가 최적 경로이다.

증명) 정리 1에 의하면 초기해 내에 잠재 경로가 숨어있지 않는 최적의 최단 경로수는 Q개 이며, Q는 교차마디(호, 부분경로)가 처음 3회 나타날 때까지의 경로들의 수이다.
그러므로 적어도 KSP에서 나오는 처음 3개의 경로는 최적해에 포함되는 최단 경로 이다. K ≤3 이면 T(s,t)에서 구한 초기해가 최적해이다.

FP^k보다 짧은 잠재경로가 생길 수 있는 경우는 같은 마디를 3번 이상 교차해서 통과하는 교차마디가 발생하고 난 다음에 생길 수 있다. 그런데 3번째 교차가 발생하는 마디는 빨라야 3번째 경로상에서 발생하므로 잠재경로는 빨라야 3번째 경로 다음, 4 번째 경로 앞에서 발생할 수 있다. 그러므로 FP^1, FP^2, FP^3 사이에는 잠재경로가 없다.

T(s,t)를 구하고 각 호를 지나는 K개의 짧은 경로의 오름차순인 K개의 경로 집합 KSP={FP^1, FP^2, …, FP^K}을 구한 후 최적의 K개 최단 경로 집합 KP={P^1, P^2, …, P^K}를 구하기 위해서는, FP^K보다 짧은 잠재 경로들을 모두 찾아내어 KSP를 개선하는 과정을 반복하면서 최적의 K개의 최단 경로 KP를 구해 나가면 된다.

이 잠재 경로들은 KSP 중에서 가장 먼저 나오는 교차 마디(호, 부분 경로)를 찾아 가장 짧은 진입호를 못 가게 Breaking시킴으로 가장 짧은 진입호를 포함하는 경로에 의해서 파묻혔던 짧은 새 노출 경로들을 찾아내고, 현재의 K개의 경로들보다 짧으면 대체시켜 넣고, KSP를 다시 각 경로 길이의 오름차순으로 정리한 후, 다시 KSP중에서 가장 먼저 나오는 교차 마디(호, 부분경로)를 찾아내어 가장 짧은 진입호를 Breaking하여 새 노출 경로들을 찾아내어 대체시키는 상기 과정을 KSP내에서 더 이상 교차 발생 마디가 없거나 FP^k보다 짧은 잠재경로가 없는 최적의 K개 최단 경로를 구할 때까지 반복한다.

최적의 조건에 도달하면 다음과 같은 상태 중 하나가 나타난다.

경우 1. 정리 2에 따라 KSP내에서 교차마디 (호, 부분경로)가 없는 경우

경우 2. KSP내에서 같은 교차 마디가 2회까지만 나타나는 경우

경우 3. KSP내의 FP^K에서 첫 번째 교차 마디가 3번째 나타나는 경로인 경우
경우 3에서는 교차 발생 마디의 진입호를 Breaking시켜 새로운 노출 경로를 구하여도 이 경로는 FP^K보다 길이가 길므로 KSP내에 포함시킬 수 없다.

경우 4. KSP내에서 3회 이상 나오는 교차 마디들이 나타나더라도 Breaking 시킨 후 노출된 잠재 경로의 길이가 FP^K보다 모두 긴 경우

이 최적의 조건이 나타날 때까지 KSP내에서 가장 먼저 나오는 교차 마디(호, 부분경로)를 찾아 가장 짧은 진입호를 Breaking시켜 새 노출 경로를 찾아 대체하는 과정을 최적 조건에 도달할 때까지 반복해야 한다.

Ⅳ. 이중 최단경로나무(DSA)에 의해 복수 최단 경로 해법

4.1 복수 최단 경로 탐색 DSA법

단계 1. 교차마디의 List를 작성한다.
- 진입호가 2개이상인 마디 집합 : CN

$|CN|$: 교차마디의 수, (최종 마디 제외)

ICN_j : 교차마디 j의 진입호 수,

KSP= ϕ : K개 경로 집합

단계 2. DSA법으로 각 호를 지나는 최단 경로와 그 길이를 구한다.

(1) Dijkstra법으로 전방향 최단나무 T(s)를 구한다.

(2) Dijkstra법으로 역방향 최단나무 T(t)를 구한다.

(3) 전방향 최단나무와 역방향 최단나무를 결합시켜 이중최단나무 T(s,t)를 만든다.

(4) 각 호별로 그 호를 지나는 출발지와 도착지 간의 최단경로(SP_i), 길이(LP_i)들을 구함

단계 3. K개의 짧은 경로를 선정하여 오름차순으로 정리한다.

$$KSP = \{SP_i \mid i=1, \cdots, K\}$$

(1) $K \le 3$ 이면 K개의 최단 경로가 선정되었음 → STOP.

(2) KSP내에서 CN중에서 교차가 일어난 교차발생마디 HCN를 찾는다.

(3) HCN = 0 이거나, 원 네크워크에서 $|CN|$ = 0 이면, K개 최단 경로가 선정되었음 → STOP.

단계 4. 새로운 교차 발생 마디를 탐색한다.

(1) 1번째부터 K-1번째까지의 K-1개 경로 중에서 2개 이상의 진입호가 3회 이상 교차 되어 연결되는 교차 발생 마디 HCN_j를 선택한다. 단, 교차가 발생한 마디가 없으면 현재의 K 개 경로가 최단 경로임 → STOP

(2) HCN_j가 포함된 경로들($Path(HCN_j)$)을 모두 선정한다. I=2

단계 5. 새로운 잠재 경로를 탐색한다.

(1) $Path(HCN_j)$의 경로들에서 HCN_j의 1번째에서 I-1번째 진입호까지의 부분경로를 제거한다.

(2) I번째 진입호를 통하여 HCN_j에 들어오는 부분경로와 I-1번째 진입호가 2번째 나오는 경로의 HCN_j 이후의 부분경로를 연결하여서 새로운 잠재경로를 생성한다.

단계 6. K개의 짧은 경로 해를 개선한다.

(1) 새로운 경로가 KSP내의 최장 경로(FP^K)보다 짧으면 FP^K는 제거한 후, K개 경로 중에 포함시키고, KSP 내의 K개의 경로 집합을 오름차순으로 다시 정리한다.

(2) I = I + 1

$I \le ICN_j$ 이면, 새로운 잠재경로를 탐색하려 단계 5로 간다.

$I > ICN_j$ 이면, $HCN = \{HCN / HCN_j\}$,

$|HCN| = |HCN| - 1$

$|HCN| = 0$ 이면, 최종해에 도달했다. STOP.

$|HCN| > 0$이면, 새로운 교차발생마디를 탐색하려 단계 4로 간다.

4.2 계산 복잡도 계산

DSA법의 계산 복잡도는

전방향 최단나무계산에 $O(n^2)$,

역방향 최단나무계산에 $O(n^2)$,

K개 경로를 오름차순으로 정리하는데 $O(n^2)$,

새로운 교차 마디 탐색에 $O(n^2)$,

새로운 잠재경로 탐색에 $O(n)$,

K개 짧은 경로 해 개선에 $O(n)$이므로

전체 계산 복잡도는, $K < n < m$ 인 경우 $O(n^2)+O(n^2)+O(n^2)+(n-3) \cdot \{O(n^2)+O(n)+O(n)\} = O(n^3)$이다. 단, K=3인 경우는 $O(n^2)$이 된다.

4.3 사례 연구

<그림 1>의 복수최단 경로문제를 확장하여 K=10인 복수최단 경로문제에 대해서 사례를 풀어보도록 하자.

단계 1. CN = {3, 4, 5, 6}, $ICN_3 = 2$, $ICN_4 = 3$, $ICN_5 = 4$, $ICN_6 = 2$

단계 2. 그림 3의 복수최단 경로문제의 이중 최단 거리 나무 T(s,t)

단계 3. (1) 10개의 짧은 경로

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

(2) HCN = {4, 5, 6}

(3) |HCN| = 3

단계 4. (1) HCN_1 = 4

(2) 마디 4가 포함된 경로들
Path (4) = {FP^1, FP^2, FP^4, FP^7, FP^8, FP^9, FP^{10}}
I = 2

단계 5. (1) 호(3,4) Breaking

(2) 2번째로 호(3,4)가 있는 FP^4 선정
FP^4 → HSP1 = 1-4-6-8, LP = 30 - 8 + 10 = 32
FP^8 → HSP2 = 1-4-6-5-8, LP = 36 - 8 + 10 = 38

단계 6. (1) HSP1을 포함시키고 , P^{10}을 제거함

FP^1 : 1-3-4-7-8 (20)
FP^2 : 1-4-7-8 (22)
FP^3 : 1-3-7-8 (24)
FP^4 : 1-3-4-6-8 (30)
FP^5 : 1-4-6-8 (32)
FP^6 : 1-2-6-8 (33)
FP^7 : 1-3-5-8 (34)
FP^8 : 1-2-4-7-8 (35)
FP^9 : 1-3-4-6-5-8 (36)
FP^{10} : 1-3-4-5-8 (38)

(2) I=3=ICN4

단계 5. (1) 호(3,4), 호(1,4) Breaking

(2) 2번째로 호(1,4)가 있는 FP^5 선정
FP5 → HSP4 = 1-2-4-6-8 , LP = 32-10+23
= 45 〉 LP(FP^{10}) = 38 → 버림

단계 6. (1) HSP = 0 . k개 경로 중에서 개선 없음

(2) I=4 〉 $1CN_4$ = 3
HCN = {5,6} |HCN| = 2

단계 4. (1) HCN2 = 6

(2) 마디 6이 포함된 경로들
Path (6) = {FP^4 , FP^5 , FP^6 , FP^9}
I = 2

단계 5. (1) 호(4,6) Breaking

(2) 2번째로 호(4,6)이 있는 FP^5 선정
FP^5 → HSP5 = 1-2-6-8 , 노출경로 → 버림
FP^9 → HSP6 = 1-2-6-5-8 , LP = 36-12+15
= 39 〉 LP(FP^{10}) = 38 → 버림

단계 6. (1) HSP =0, K개 경로 중에서 개선 없음

(2) I=3 〉 ICN6 = 2
HCN = {5}, | HCN | = 1

단계 4. (1) HCN3 = 5

(2) 마디 5가 포함된 경로들
Path (5) = {FP^7, FP^9} , I = 2

단계 5. (1) 호(3,5)를 Breaking

(2) 2번째로 호(3,5)가 포함된 경로가 없음

단계 6. (1) HSP = 0 , K개 경로 중에서 개선 없음

(2) I=3 〈 ICN5 = 4

단계 5. (1) 호(3,5), (6,5) Breaking

(2) 2번째로 호(6,5)가 포함된 경로가 없음

단계 6. K개의 경로 중에서 개선 없음
I = 4 = ICN5

단계 5. (1) 호(3,5), 호(6,5), 호(4,5)를 Breaking

(2) 2번째로 호(4,5)가 나오는 경로 없음

단계 6. K개의 경로 중에서 개선 없음
I = 5 〉 ICN5 = 4
| HCN | = 0, → STOP

〈표 1〉 K = 10 최단경로의 해

순서	경로	경로길이
P1	1-3-4-7-8	20
P2	1-4-7-8	22
P3	1-3-7-8	24
P4	1-3-4-6-8	30
P5	1-4-6-8	32
P6	1-2-6-8	33
P7	1-3-5-8	34
P8	1-2-4-7-8	35
P9	1-3-4-6-5-8	36
P10	1-3-4-5-8	38

Ⅴ. 결론 및 추후 방향

본 연구에서는 복수최단 경로문제에 대해서 $O(n^3)$의 새로운 해법을 제시하였다.

또한 K=3인 경우의 복수최단 경로문제에는 $O(n^2)$의 계산 복잡도를 가진다. 이 해법은 전방향의 최단 경로 나무와 후방향의 최단 경로 나무를 구하고 이것을 결합시켜 각 호를 거쳐 출발지에서 목적지까지 가는 최단 경로 정보를 제공하는 이중 최단 나무 T(s,t)를 구하고, 초기 가능해인 K개의 짧은 경로를 길이의 오름차순으로 구한 후, 교차 마디로 인하여 발생하는 짧은 잠재 경로를 찾아내어 K개의 짧은 경로를 개선하는 과정을 K개의 짧은 경로들 속에 더 이상 잠재 경로가 없을 때까지 반복하여 K개의 최단 경로를 구하는 해법으로 발견적 방법을 이용한 최적해법이다.

추후 연구 과제로는 기존의 최단 경로 문제처럼 알고리즘의 실행을 보다 효율적으로 할 수 있으면서 계산 복잡도가 낮은 보다 효율적인 해법을 찾는 것과, 실제 도로망이나 교통·수배송 등 활용 영역에서의 검증이 필요하다. 또한 ITS 등과 같은 실제 활용 영역에서의 특성상 요구되는 모든 마디간의 복수 최단 경로 문제 해결을 위한 효율적인 해법의 연구가 계속적으로 요망된다.

참 고 문 헌

[1] S. Clarke, A. Krikorian, and J. Rausen, "Computing the N best loopless paths in a network," J. Soc. Indust. Appl. Mathematics 11(1963), pp.1096-1102.

[2] S. Dreyfus, "An Appraisal of Some Shortest Path Algorithms," Oper. Res. 17(1969), pp.395-412.

[3] F. Glover, R. Glover, and D. Klingman, "Computational study of an improved shortest path algorithm," Networks 14 (1984), pp.25-36.

[4] N. Katoh, T. Ibaraki, and H. Mine, "An efficient algorithm for K shortest simple paths," Networks 12(1982), pp.411-427.

[5] E. Lawler, Combinatorial Optimization: Networks and Matroids. Holt, Rinehart, and Winston, New York, 1976.

[6] A. Perko, "Implementation of Algorithms for K shortest Loopless Paths," Networks 16 (1986), pp.149-160.

[7] D. Shier, "On algorithms for finding the K shortest paths in a network," Networks 9 (1979), pp.195-214.

[8] J. Yen, "Finding the K shortest loopless paths in a network," Manag. Sci. 17(1971), pp.712-716.

[9] 장병만, "첨단도로교통체계 차량안내용의 시간종속 복수최단경로 해법", 대한산업공학회/한국경영과학회, 98춘계공동학술대회(1998) C.03.6.