■論 文■

교통망에 적합한 K 비루프 경로 탐색 알고리즘

Finding the First K Shortest Loopless Paths in a Transportation Network

신 성 일
(서울시정개발연구원 도시교통연구부 부연구위원)

목 차

Key Words : 다수경로, 네트워크확장, 비루프경로, 링크표지, 경로삭제방법, 경로분할방법

요 약

다수경로 알고리즘은 비루프경로(Loopless Path: 또는 Simple Path)와 루프경로(Loop Path)를 탐색하는 방안으로 대별된다. 알고리즘의 난이도 측면에서 일반적으로 비루프경로를 탐색하는 방안이 루프경로를 탐색하는 방안보다 많은 노력이 소요된다. 바꾸어 말하면, 루프경로 탐색방안이 알고리즘의 이해 및 활용성 측면에서 용이하다는 장점이 존재한다.

루프경로탐색방안에서 경로삭제방식(Path Deletion Method)이 가장 효율적인 알고리즘으로 알려져 있다. 경로삭제방식은 K번째의 최적경로를 탐색하기 위하여 K-1번째의 경로의 탐색금지 상황설정이 필요하며, 이를 네트워크의 변형된 확장형태(Enlarged Transform)를 통하여 추구하는 방식이다. 그러나 이 알고리즘은 경로상에 노드 및 링크의 반복이 허용되는 루프를 생성시켜 교통망에 적용하기에는 한계가 존재하는 단점이 있다.

본 연구에서 링크표지를 활용하여 루프를 제거하는 방안을 개발한다. 이를 위해 K-1번째 확장네트워크에서 링크표지를 갱신하는 과정에서 대상링크와 전 링크의 부분경로와의 관계를 고려하여 루프가 생성되지 않도록 링크표지를 확정하여 원천적으로 루프의 생성을 방지한다. 본 연구에서 제안하는 비루프 알고리즘은 노드비루프와 링크비루프로 구분되며, 노드비루프는 경로 상에서 노드의 반복이 존재하지 않는 일반적인 단순경로(Simple Path)를 의미하며, 링크비루프는 경로 상에 링크의 반복이 존재하지 않는 경로를 의미한다. 특히 링크비루프는 도시 교차로에서 발생하는 U-턴, P-턴의 덩굴망 통행행태를 설명하기 위한 중요개념으로 확대 정의된다. 사례연구를 통하여 제안된 알고리즘의 활용성을 검증한다.

I. 서론

첨단여행자정보체계(ATIS)의 주요목적 중 하나는 여행자가 요구하는 최적의 경로정보를 제공하는 것이다. 그러나 최적정보는 여행자의 환경과 여건에 따라 인지적으로 결정되므로 선택의 폭이 다양한 여행정보의 제공은 첨단여행자정보체계(ATIS: Advanced Traveler Information System)의 실현을 위해서 매우 중요하다. 다양한 경로정보 제공을 위한 방안으로 다수의 대안경로열거를 통한 개선시도가 이루어지고 있다. 복수의 대안경로정보를 산출하는 주요 방안은 (K)개의 경로탐색 알고리즘을 활용하는 것이다.

경로삭제방법(Path Deletion Method)은 출발지와 도착지가 정해진 네트워크에서 두 지점 간 (K)개의 경로를 탐색하는 기법이다 (Yen, 1971; Martins, 1984; Avezedo et al, 1993). 이 방법은 순차적으로 탐색된 (K-1)개의 경로정보를 (K)번째 경로를 발견하기 위하여 이용하는 것이다. 기존의 최적경로 알고리즘(Dijkstra, 1959; Moore, 1957)을 직접 활용할 수 있기 때문에 (K)개의 경로를 동시에 탐색하는 방안(Pollack, 1961; Bellman & Kalaba, 1968, Shier, 1979)보다 이해와 구축이 용이하다.

경로삭제방법을 활용하여 (K)개의 경로를 탐색하는 알고리즘은 크게 두 가지로 분류된다: 하나는 경로상에 노드 또는 링크의 중복이 존재하지 않는 비루프경로(Loopless Path; Simple Path)를 탐색하는 방안(Yen, 1971)이고 다른 하나는 루프경로(Loop Path)까지 포함하여 탐색하는 방안(Martins, 1984; Avezedo et al, 1993)이다. 두 알고리즘의 수행측면에서, 전자의 Yen(1971) 방법은 경로의 일부분을 삭제하는 방안(Path Partition Algorithm)에 기반을 두고있다. 이 방안은 (K-1)개의 링크와 관련된 부분링크조합을 모두 고려하여 최적경로탐색을 수행하기 때문에 네트워크가 커질수록 (K)번째 경로를 선정을 위한 알고리즘의 수행속도가 크게 저하된다. 반면, 후자의 Avezedo et al(1993)- Martins(1984)이 제안한 방안보다 개선된 -알고리즘은 네트워크의 변형(Enlarged Network)을 통해 경로의 전체를 삭제하는 기법(Entire Path Deletion)에 기반을 두고있다. 이 방법은 최적경로 알고리즘의 한번 수행으로 (K)번째 경로를 발견하기 때문에 네트워크의 규모에 따라 수행속도는 매우 효율적이다 (Yang & Chen, 2003).

이러한 장점에도 불구하고, 전체의 경로를 삭제하는 방법은 불필요한 루프까지 모두 고려해야 하는 한계성이 존재한다. 루프는 개념적으로 링크나 노드의 무제한적인 반복이 포함되어 있으며, 따라서 경로를 탐색함에 있어 무제한적인 루프의 포함은 알고리즘의 활용성에 있어서 한계에 도달할 가능성-특히 네트워크가 커질수록-이 크게 존재한다. 또한 요구되는 경로정보를 획득하기 위하여 (K)의 수를 증가시키는 것은 긍극적인 수행속도의 저하도 함께 초래된다. 따라서 불필요한 루프를 방지하는 방안에 대한 고려가 알고리즘의 현실 교통망에서 활용되기 위해서 필요하다.

본 연구의 목적은 Avezedo et al(1993)가 제시한 전체경로삭제방법에서 불필요한 루프를 제거하는 방안을 개발하는 것이다. 이를 위해 (K)번째 경로탐색을 위해 수행되는 (K-1)번째 확장네트워크에 링크표지(Link Labeling)을 활용하는 방안을 제안한다. 링크표지활용의 장점은 경로탐색 시 두 인접링크를 동시에 고려하기(Kirby & Potts, 1969; Potts & Oliver, 1972) 때문에 루프가 생성되는 방향의 탐색을 생략하는 것이 가능하며, 두 링크간의 탐색관계를 고려하여 노드의 반복이 존재하지 않는 노드비루프(No Node Repeated Path)와 링크반복이 존재하지 않는 링크비루프 (No Link Repeated Path)로 구분하여 수행이 가능하다. 특히 링크비루프는 (특히 도시) 교통망에서 발생하는 운전자의 합리적 통행행태를 반영하는 장점을 제공한다(임용택, 2004).

II. 이론적배경

본 연구는 경로를 삭제하여 (K)개의 경로를 발견하는 알고리즘에 링크표지를 적용하는 것이다. 1절에서 전체경로삭제방법을 중심으로, 2절에서는 링크표지의 최적경로 및 다수경로탐색알고리즘의 적용사례에 대해 검토한다.

1. 경로삭제방법

출발지와 도착지가 정해진 경우 경로삭제방법(Path Deletion Method)은 (K)개의 경로를 탐색하기 위하여 (1) 비루프경로(Loopless Path)를 탐색하는 방안(Yen, 1971)과 (2) 루프경로(Loop Path)를 포함하여 탐색하는 방안(Martins, 1984; Avezedo et al,

1993)으로 구분된다.

Yen(1971)은 경로를 링크단위의 부분삭제방안(Path Partition Algorithm)에 기반을 두고 비루프 경로탐색을 위한 효율적인 알고리즘을 제안했다. 이 방안의 수행시간은 일반적인 $|V|$ 노드수의 일반적인 네트워크에서 $O(K|V|^3)$이다. (K-1)개의 링크와 관련된 부분링크조합을 고려하여 최적경로탐색을 수행하기 때문에 네트워크가 커질수록 ($|V|$) 수행시간은 3의 지수만큼 큰 폭으로 증가한다.

루프경로를 포함하여 탐색하는 방안으로서, Martins (1984)은 전체경로삭제방안으로서 최대수행시간이 $O(K^3|V|)$인 알고리즘을 제안했다. 유사한 방법을 활용하여 Avezedo et al(1993)는 최대수행시간 $O(K^2|V|)$의 Martins(1984)의 방법보다 개선된 효율적인 알고리즘을 제안했다.

Martins(1984)와 Avezedo et al(1993)이 제안한 알고리즘은 네트워크의 변형(Enlarged Network)을 통해 경로의 전체를 삭제하는 기법(Entire Path Deletion)에 기반을 두고있다. 이들 방법은 최적경로 알고리즘의 한번 수행으로 (K)번째 경로를 발견하기 때문에 효율적으로 수행시간을 갖는 것으로 알려져 있다 (Yang & Chen, 2003).

두 알고리즘에 대해 보다 상세히 설명하면, 우선 Martins(1984) 알고리즘은 네트워크 N에 대하여 2단계의 세부알고리즘을 수행한다: (1) 두 지점간 최적경로p 발견을 위한 최적경로알고리즘과, (2) 새로운 네트워크 N'을 생성시키기 위한 경로삭제알고리즘으로, 개략적 개념은 우선 최적경로 p가 탐색되면, 그 p경로를 네트워크 N에서 삭제하여 다시 최적경로알고리즘을 수행한다는 것이다. 이때 새로운 노드와 링크를 네트워크 N에 추가하여 구성된 확장된 네트워크 N'은 경로 p를 제외한 모든 경로의 탐색이 가능하게 구축된다. N_1을 기본네트워크로 하여 순차적인 (K)개의 경로를 탐색한다는 것은 $\{N_1, N_2, \Lambda, N_K\}$의 순차적인 네트워크를 구축함을 의미하며, 이 경우 j번째 네트워크 N_j로부터 j번째 경로 p_j 가 탐색된다. 이 알고리즘에서 최적경로알고리즘을 K번 경로삭제알고리즘은 K-1번 수행된다.

경로삭제알고리즘의 수행과정을 나타내기 위한 표식(Notations)으로서 네트워크 N의 경로 p는 다음과 같은 노드의 순서로 표현된다. $p=\{v_0, v_1, \Lambda, v_{m-1}, v_m\}$, 여기서 $v_0=R$는 출발지 $v_m=S$은 도착지 노드이며 $m\geq 3$. 어느 노드 u와 연결된 유출링크(Outgoing Links) 집합은 $O(u)=\{(u,v)\in A | u,v\in N\}$, 유입링크(Incoming Links) 집합은 $I(u)=\{(v,u)\in A | v,u\in N\}$로 정의한다.

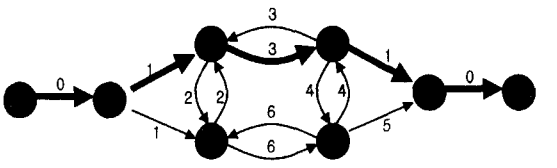
경로 p를 삭제하여 네트워크 N'을 생성하기 위하여 Martins 알고리즘은 새로운 노드를 추가하고, 이들 노드에서 유출링크를 연결하며, p의 첫번째 경로를 삭제하는 것이다. 알고리즘의 수행과정은 다음과 같다.

단계1: $N=N\cup\{v_1', \Lambda, v_{m-1}'\}$

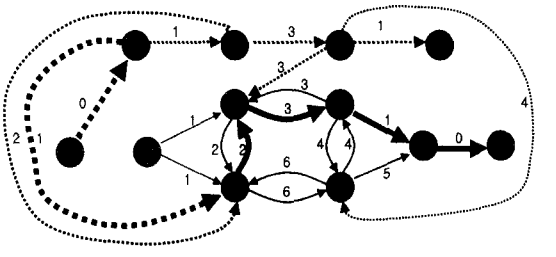
단계2: $O(v_0)=O(v_0)-\{(v_0,v_1)\}\cup\{(v_0,v_1')\}$

단계3: $O(v_j')=\{(v_j',u)|(v_j,u)\in O(v_j); u\neq v_{j+1}\}$
$\cup\{(v_j', v_{j+1}')\}$, $\forall j\in\{1,\cdots,m-2\}$
$O(v_{m-1}')=\{(v_{m-1}',u)|(v_{m-1},u)\in O(v_{m-1})\}$

간단한 예제 네트워크(<그림 1>)를 통하여 위의 알고리즘을 추적하여 보면, 최적경로는 $p=\{R,1,2,4,6,S\}$의 노드순서로 네트워크 N에 굵게 표시되어 있다. 알고리즘으로 생성된 네트워크 N'은 <그림2>와 같다. **단계1**의 새로운 노드집합 $\{1',2',4',6'\}$가 추가되었고, **단계2**의 p의 첫번째 링크 $(R,1)$가 삭제되어 링크$(R,1')$로 대체되



<그림 1> 네트워크 N



<그림 2> Martin알고리즘의 네트워크 N'

었고, **단계3**의 유출링크집합 $\{(R,1'),(1',3),(2',3),(4',2),(4',5),(1',2'),(2',4'),(4',6')\}$이 추가되었다. 〈그림 2〉의 네트워크 N'을 기반으로 최적경로는 알고리즘을 수행하여 $p'=\{R,1',3,2,4,6,S\}$는 두 번째 탐색된 경로가 된다.

Avezedo et al(1993) 알고리즘은 Martins(1984) 알고리즘의 수행능력의 향상을 목적으로 제안되었다. Avezedo et al (1993) 알고리즘이 Martins (1984)이 제안한 방법과의 가장 큰 차이는 경로 p의 마지막 링크를 삭제하고 새로 추가된 노드의 유입링크집합을 네트워크 N에 추가한다는 것이다. Martins (1984) 알고리즘이 K번의 최적경로알고리즘과 K-1번의 경로삭제 알고리즘이 수행되나, p의 마지막 링크를 삭제하고 새로 추가된 노드의 유입링크집합을 추가하여 N'을 생성시키면, Avezedo et al (1993) 알고리즘은 N의 링크 및 노드의 영구표지는 N'에서 영구표지로 남아있으며, N에서 삭제된 p의 부분경로(Subpath)가 N'의 최적경로설정에 포함된다는 사실에 근거하여 (K-1)번의 최적경로알고리즘 수행을 절약할 수 있다. 따라서, Avezedo et al (1993) 알고리즘의 최대수행시간이 $O(K^2|V|)$로 절감된다. 알고리즘의 수행과정은 다음과 같이 설명된다.

단계1: $N = N \cup \{v_1', \Lambda, v_{m-1}'\}$

단계2: $I(v_1') = \{(u,v_1') | (u,v_1) \in I(v_1); u \neq v_0\}$

$I(v_j') = \{(u,v_j') | (u,v_j) \in I(v_j); u \neq v_{j-1}\} \cup$

$\{(v_{j-1}',v_j')\}$, *for any* $j \in \{2,\cdots,m-1\}$

단계3: $I(v_m) = I(v_m) - \{(v_{m-1},v_m)\} \cup \{(v_{m-1}',v_m)\}$

Avezedo et al (1993) 알고리즘을 요약하면 크게, 1) 최적경로탐색알고리즘과 2) 네트워크 확장알고리즘과 3) 확장네트워크의 추가 노드 및 링크표지확정 알고리즘으로 구분되며 알고리즘의 구성은 다음과 같다.

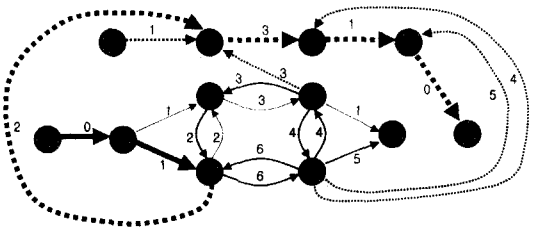
단계 1: 최적경로알고리즘의 수행으로 p_1의 발견

단계 2: k=2부터 K까지 반복

네트워크 확장알고리즘으로 N에서 N'의 구축

N'에 추가된 노드 및 링크표지확정

출발지에서 도착지까지 최적경로탐색



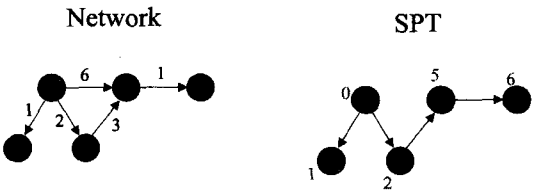
〈그림 3〉 Avezedo et al 알고리즘의 네트워크 N'

〈그림1〉의 네트워크를 알고리즘을 통하여 수행한 과정 네트워크 N'은 〈그림3〉에 도식되어 있다. **단계1**의 새로운 노드집합 $\{1',2',4',6'\}$가 추가되었고, **단계2**의 유입링크집합 $\{(3,2'),(4,2'),(5,4'),(1',2'),(2',4'),(4',6'),(5,6'),(6',S)\}$이 추가되었다. **단계3**의 p의 마지막 링크 $(6,S)$가 삭제되어 링크$(6',S)$로 대체되었다. 네트워크 N의 노드 및 링크표지는 N'에 영구적으로 고정되었고, 삭제된 p의 부분경로는 N'의 표지검색만으로 결정될 수 있기 때문에 최적경로알고리즘의 수행과정 없이 $p'=\{R,1,3,2',4',6',S\}$의 탐색이 가능하다.

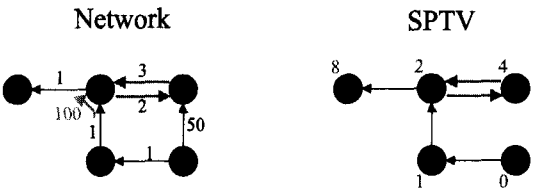
2. 링크표지기법과 최적경로알고리즘

최적경로알고리즘(Shortest Path Algorithm: SPA)은 두 지점의 최적통행시간 또는 비용을 갖는 경로를 발견한다. 최적경로는 최적경로알고리즘에서 채택하고 있는 경로탐색방법에 의해 결정된다. 경로탐색방법은 최단경로가지(Shortest Path Tree: SPT)를 구축하는 방법과 최단경로가지와 덩굴망(Shortest Path Tree and Vine)을 함께 구축하는 방법으로 구분된다. 탐색된 경로를 일련의 노드순서로 표현할 때, SPT와 SPTV의 차이는 SPT는 경로상에 노드가 반복이 허용되지 않는 반면, SPTV는 반복이 허용된다.

〈그림 4〉 네트워크와 SPT의 예를 보여준다. 〈그림 4〉에서 네트워크를 SPT로 표현하면, 출발지 노드1에서 노드4까지의 최단경로는 1->3->2->4로서 경로에 노드의 중복이 존재하지 않는다. SPT를 이용한 탐색방법은 가지기반 최적경로알고리즘(Tree-Based SPA)의 기초가 되었으며, 일반적인 최적경로알고리즘으로 알려진 노드기반 표지확정/갱신(Node-Based Label Setting/ Correcting) 방법이 있다 (Dijkstra, 1959; Moore, 1957). 식(1)은 가지기반 SPA의 최적경로비



〈그림 4〉 네트워크와 최단경로가지 구축



〈그림 5〉 네트워크와 최단경로가지와 덩굴망 구축

용을 계산하는 과정을 나타낸다. 두 인접노드 i와 j간의 탐색이 고려된다.

$$\pi^{rj} = \min_{i \neq j} \{\pi^{ri} + c_{ij}, \pi^{rj}\} \quad (1)$$

이 경우 r, i, j : 노드, 특히 r은 출발지노드

π^{ri} : 출발지 r에서 노드i까지의 최적경로비용

c_{ij} : 링크(i, j)의 통행비용

SPTV는 두 인접링크 a와 b의 고려가 가능하다. 따라서 인접링크의 중심에서 발생하는 회전비용에 대한 고려가 가능하며, 회전지체나 금지 등으로 인한 최적경로의 출발지와 도착지가 같은 덩굴망(Vine)의 발생에 대한 설명이 가능하다. 〈그림 5〉 네트워크와 SPTV의 예를 보여준다. 노드3을 중심으로 진행방향 2->3->5로 회전지체가 존재하며, 이 회전지체의 비용까지 고려해서 SPTV로 나타내면, 출발지 1과 도착지5까지 최적경로는 1->2->3->4->3->5로 노드3이 U-턴 통행표현을 위해 경로에 두 번 나타난다. 이 방법은 덩굴망 기반 최적경로알고리즘 (Vine-Based SPA)의 기반이 되었다. 링크를 표지로 이용하는 방법은 Kirby & Potts (1969)에 회전지체와 금지가 존재하는 네트워크에 처음 시도되었고, Potts & Oliver는 P-턴을 포함하는 네트워크에 적용하였다.

$$\pi^{rb} = \min_{a \neq b} \{\pi^{ra} + c_b + d_{ab}, \pi^{rb}\} \quad (2)$$

이 경우 a, b : 링크

π^{ra} : 출발지 r에서 링크a까지의 최적경로비용

c_a : 링크a의 통행비용

d_{ab} : 두 인접링크 a와 b사이에서 발생하는 회전비용

식(2)의 Dijkstra(1959) 방식에 의한 링크표지확정 최단경로알고리즘은 다음과 같다. 초기 노드표지에 근거한 알고리즘과 차이는 1) 링크 수 만큼 알고리즘을 반복과, 2) 링크탐색과정에서 두 인접링크에서 발생하는 회전지체 또는 금지의 특성을 반영 (Step 3)과, 3) 알고리즘 종료 후 각 링크표지에 최적경로비용을 노드표지로 계산과정의 포함이다 (Step 2).

L : 링크집합

Ξ : 노드집합

A_r : 출발노드가 r인 링크집합

B_i : 도착노드가 i인 링크집합

D_a : 출발링크가 a인 방향별 회전링크집합

Q : 탐색링크집합

[Step 1] 초기화

$\pi^{ra} = \infty, \forall a \in L, r \in \Xi$

$\pi^{ra} = c_a, \forall a \in A_r, r \in \Xi$

$Q = Q \cup \{a\}$

[Step 2] 다음탐색링크결정

If : $Q = \phi$: $\pi^{ri} = \min\{\pi^{ra}, \forall a \in B_i\}, \forall i \in \Xi$; 종료

Else : $\min\{\pi^{ra}, \forall a \in Q\}$을 만족하는 a의 선정

$Q = Q - \{a\}, \ a \in Q$

[Step 3] 다음링크$(a \rightarrow b)$로 확장

If : $\pi^{ra} + d_{ab} + c_b < \pi^{rb}$: $\pi^{rb} = \pi^{ra} + d_{ab} + c_b$

$\forall b \in D_a ; Q = Q \cup \{b\}$; Go to Step 2

Else : Go To Step 2

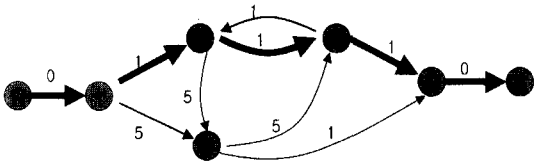
Ⅲ. 알고리즘

본 연구는 Avezedo et al(1993)알고리즘에서 경로

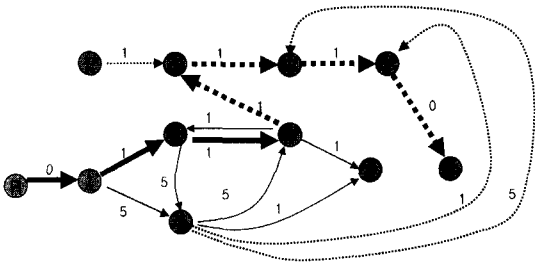
에 루프가 발생되는 것을 교통망에 적용가능 하도록 노드비루프와 링크비루프로 제한하는 알고리즘으로 개선하는 방안에 대해 제안한다.

1. 경로삭제방법과 루프

루프가 경로p에 포함되면 노드 또는 링크의 반복이 발생한다. Avezedo et al(1993) 알고리즘은 루프가 p에 포함되는 본질적인 탐색구조를 채택하고 있다. 루프의 발생은 네트워크 N에서 탐색된 경로p의 비용이 타 경로의 비용보다 작을수록 빈도가 높아진다. 재 설명하면, 루프를 포함되지 않는 다음 경로의 비용과 비교하여 루프를 발생시켜 비용이 커질 때 까지 계속적으로 루프를 발생시킨다. 이 과정에 대한 설명으로서, 〈그림 6〉은 네트워크 N에서 $p=\{R,1,2,4,5,S\}$이며 비용이 3으로 다음 비용경로 $\{R,1,3,4,5,S\}$보다 비용이 작다. 이 경우 N'에서 최적경로는 $\{R,1,3,4,5,S\}$가 발견되지 않고 $p'=\{R,1,2,4,2',4',5',S\}$로 노드2와 4의 중복방문이 발생할 뿐만 아니라, 링크(2,4)를 두 번 방문하는 비합리적 통행현상을 나타내는 루프를 생성시킨다. 이 과정은 루프생성으로 더 해지는 추가비용이 경로$\{R,1,3,4,5,S\}$보다 클 때 까지 루프는 계속해서 반복된다. 〈그림7〉은 N'에서 탐색된 루프를 포함한 최적경로 $p'=\{R,1,2,4,2',4',5',S\}$를 보여주고 있다. 이 과정은 루프를 발생시켜도 다음 비루프 경로인



〈그림 6〉 네트워크 N



〈그림 7〉 노드 및 링크의 루프를 포함한 최적경로의 발견(N')

$p'=\{R,1,3,5',S\}$의 경로비용보다 적을 경우 지속되며, 두 경로의 비용차이가 클수록 비용의 차이가 줄어들 때 까지 추가적인 노드 및 링크루프를 발생시키는 빈도수가 커지게 된다.

2. 링크표지기반 알고리즘

본 연구에서 네트워크 N'에서 수행되는 최적경로 알고리즘에서 회전지체를 포함한 루프가 존재하는 경로의 탐색을 금지하는 방안을 강구한다. 기존의 가지기반 최적경로 알고리즘은 2개의 노드를 탐색하는 방법을 이용하므로, 회전지체나 금지가 포함된 교차로에서 발생하는 루프의 존재-특시 링크 비루프-에 대한 고려가 불 가능했다.

루프의 발생을 방지하는 방법은 최적경로알고리즘에서 적용된 링크탐색과정에 대한 추가적인 고려로서 해결 가능하다. 두 링크(a와 b)의 탐색과정은 a의 서브경로와 b의 링크가 더해지는 과정이며, 이때 루프를 발생시키는 경로의 생성을 삭제하는 방법으로 링크표지기반 최적경로탐색의 활용으로 해결될 수 있다.

우선 노드 또는 링크루프의 생성을 방지하는 과정이 포함되는 링크표지기반 최적경로알고리즘의 Step3가 아래에 나타나 있다. Step 3의 (1)은 노드의 반복이 존재는 루프, (2)는 링크의 반복이 존재하는 루프를 각각 제거하는 과정을 각각 나타낸다. 세부수행에 대하여 관찰하여 보면, 먼저 표식 $P_r^a \oplus b$는 링크a의 부분경로P_r^a와 연결된 링크를 의미하므로, 두 링크의 연결과정에서 링크a의 부분경로를 탐색하여 링크b의 노드 또는 링크가 존재여부를 점검하여 링크표지확정과정을 생략을 결정하는 방안이다. 이는 링크a의 표지가 이미 확정되어 있기 때문에 링크a의 전 경로를 계속 따라서 출발지까지 역 추적(Backward Search)하면 링크b의 존재여부를 확인할 수 있다. 이를 나타내기 위한 표식을 정의하면 다음과 같으며, 링크b의 탐색에 의하여 생성 될 새로운 경로 $P_r^a \oplus b$에 대하여 $\vartheta(i,j)$은 노드비루프, $\psi(b)$는 링크비루프를 각각 나타내기 위하여 정의되었다.

P_r^a : 출발지 r에서 링크a까지 최적경로

$P_r^a \oplus b$: P_r^a의 마지막 링크a의 도착노드와 링크

b의 출발노드 연결된 경로

$\vartheta(i,j)$: 링크(i,j)의 출발노드i와 도착노드j의 반복이 없는 경로집합

$\psi(b)$: 링크b의 반복이 없는 경로집합

[Step 3] 다음링크$(a \rightarrow b)$로 확장

If : $\pi^{ra} + d_{ab} + c_b < \pi^{rb}$ {

(1) If : $P_r^a \oplus b \in \vartheta(i,j)$: $\pi^{rb} = \pi^{ra} + d_{ab} + c_b$

(2) If : $P_r^a \oplus b \in \psi(b)$: $\pi^{rb} = \pi^{ra} + d_{ab} + c_b$

}

위에서 Step3을 포함하는 링크표지기반 최적경로알고리즘에서 노드비루프와 링크비루프를 탐색하는 방법을 한 번만 수행하고 나머지 (K-1)번의 네트워크확장을 통한 추가노드 및 링크의 표지를 확정을 통해 최적경로를 발견하는 것이 본 연구에서 제안하는 알고리즘의 특징이다. 따라서, 본 연구에서 제안하는 알고리즘을 요약하면 크게, 1) 노드 또는 링크비루프 탐색을 위한 링크표지기반 최적경로탐색알고리즘과 2) 네트워크 확장알고리즘과 3) 확장네트워크 확정에서 추가로 늘어난 링크 및 노드표지를 확정하는 알고리즘으로 구분되며 알고리즘의 전체구성은 다음과 같다.

단계 1 : N을 기반으로 링크표지확정 최적경로알고리즘의 수행으로 P_1의 발견

단계 2 : k=2부터 K까지 반복

Avezedo et al. (1993)의 네트워크 확장알고리즘으로 N에서 N'의 구축

N'에 추가된 링크표지 및 노드표지확정

출발지에서 도착지까지 최적경로탐색

여기서 알고리즘의 단계 1은 위에서 설명되었으므로 알고리즘의 탐색과정을 추가로 수행하지 않고 N'에 추가된 링크표지 및 노드표지를 확정하는 방안에 대해 집중적으로 검토하면 알고리즘에 대한 전체적인 설명이 가능하다. 네트워크 N과 N'을 구분하기 위해 링크집합과 노드집합에 대한 표식과 관계를 정의하여 다음과 같다.

L^N : 네트워크 N의 링크집합

Ξ^N : 네트워크 N의 노드집합

$L^{N'}$: 네트워크 N'에 추가된 링크집합

$\Xi^{N'}$: 네트워크 N'에 추가된 노드집합

$L^N \cap L^{N'} = \{\}$; $\Xi^N \cap \Xi^{N'} = \{\}$

제안된 알고리즘에서 네트워크 N'에 추가된 링크와 노드의 표지를 확정하기 위한 링크표지 및 노드표지 확정방안을 설명하면, 우선 링크표지 확정방안으로서, N'에 포함된 링크 중에서 출발노드가 N에 포함되어 있는 링크(L^N)와 N'에만 포함되어 있는 링크($L^{N'}$)로 구분된다. 이때 $L^{N'}$의 링크표지를 구축하기 위해서는 이미 링크표지가 구축된 N의 링크(L^N)를 이용해야 한다. 이 과정에 의해 출발노드가 N에 존재하고 도착노드가 N'에 존재하여 링크표지가 구축되며, 구축된 링크표지를 기반으로 다시 출발노드와 도착노드 모두 N'에만 존재하는 링크의 표지를 구축하게 된다. 이와 같은 과정을 통하여 구축되는 노드비루프와 링크비루프의 2단계 링크표지확정과정알고리즘은 아래와 같다.

N에서 N'으로 추가된 링크표지확정

(1) 노드비루프 링크표지확정

(step1) N에 포함된 링크(a)와 N'에 포함된 링크(b)

$$\pi^{rb} = \min\left\{\pi^{ra} + d_{ab} + c_b \mid \forall a \in L^N, P_r^a \oplus b(i,j) \in \vartheta(i,j), i \in \Xi^N, j \in \Xi^{N'}\right\} \forall b \in L^{N'}$$

(step2) N'에 포함된 링크(a,b)

$$\pi^{rb} = \min\left\{\pi^{ra} + d_{ab} + c_b \mid \forall a \in L^{N'}, P_r^a \oplus b(i,j) \in \vartheta(i,j), i \in \Xi^N, j \in \Xi^{N'}\right\} \forall b \in L^{N'}$$

(2) 링크비루프 링크표지확정

(step1) N에 포함된 링크(a)와 N'에 포함된 링크(b)

$$\pi^{rb} = \min\left\{\pi^{ra} + d_{ab} + c_b \mid \forall a \in L^N; P_r^a \oplus b \in \Psi(b)\right\} \forall b \in L^{N'}$$

(step2) N'에 포함된 링크(a,b)

$$\pi^{rb} = \min\left\{\pi^{ra} + d_{ab} + c_b \mid \forall a \in L^{N'}; P_r^a \oplus b \in \Psi(b)\right\} \forall b \in L^{N'}$$

링크표지의 확정과정이 완료되면, 네트워크 N'에

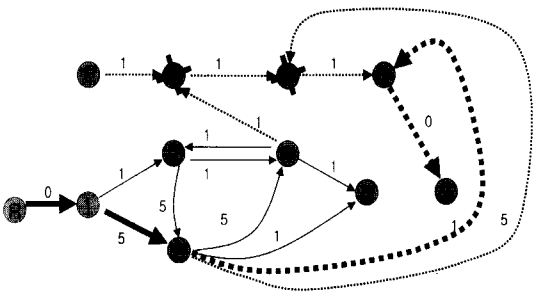
추가된 노드표지를 확정하면 표지확정과정이 완료된다. 이때 역방향 추적으로 최종적인 최적경로탐색이 가능해진다. 노드표지확정과정은 새롭게 구축된 링크표지를 기반으로 도착노드가 j인 링크 중에서 최소의 비용을 갖는 링크의 비용을 노드표지 비용으로 확정하게 된다.

N 에서 N' 으로 추가된 노드표지확정

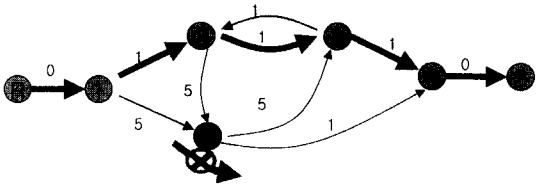
$$\pi^{rj} = \min\left\{\pi^{ra} \mid \forall a \in B_j; a \in \left(L^N \cup L^{N'}\right)\right\}, \forall j \in \Xi^{N'}$$

〈그림 8〉은 〈그림 6〉의 네트워크 N'에 대하여 노드 비루프 경로 $p'=\{R,1,3,5',S\}$를 탐색한 것이다. 이때 링크(2->4에서 링크(4->2')의 탐색은 노드비루프를 발생시켜, 즉 $P_r^{(2->4)} \oplus (4->2') \notin \vartheta(4,2')$, 탐색과정에서 생략된다.

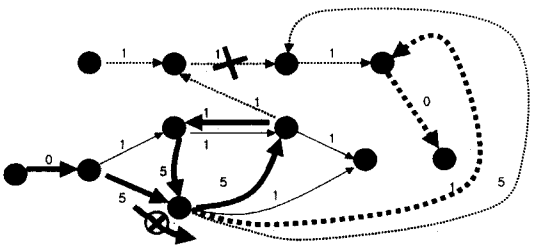
〈그림 9〉는 N 에 링크(1->3)에서 (3->5)방향으로 회전제한이 포함된 경우를 나타낸 것이다. 링크(2->4)에서 링크(4->2')는 $P_r^{(2->4)} \oplus (4->2') \in \Psi(4,2')$ 를 만족하는 링크비루프이므로 탐색과정이 계속 진행된다. 그러나, 링크(4->2')에서 (2'->4')로의 탐색과정은 $P_r^{(4->2')}$에 링크(2->4)가 포함되어 있으므로 링크루프 ($P_r^{(4->2')} \oplus (2'->4') \notin \Psi(2',4')$)이므로 탐색과정에서 생략된다. 또한 링크(1->3)에서 (3->5)로는 통행규제가 존재하므로 〈그림 9〉에 대한 N'의 최적경로는



〈그림 8〉 노드 비루프 최적경로의 발견(N')



〈그림 9〉 네트워크 N 과 통행규제



〈그림 10〉 링크 비루프 최적경로의 발견(N')

〈그림 10〉과 같이 $p'=\{R,1,3,4,2,3,5',S\}$로 나타나며, 이는 노드루프이나, 링크비루프 경로이다.

위의 예제를 통해 제안된 알고리즘에서 링크표지 최적경로알고리즘의 (K-1)번 수행이 감소된다는 사실을 입증하기 위하여 N'에 추가된 노드 및 링크표지 만으로 목적지에서 출발지까지 최적경로의 탐색이 가능함을 확인하기 위해, 〈그림 6〉의 네트워크 N 과 〈그림 7〉의 N'으로 살표 본다. 우선 N'에 추가된 노드 1',2',4',5' 와 링크 (1',2'),(2',4'),(4',5'),(5',S)는 최적경로 p'에서 추가된 것이며, (4,2'), (3,4'), (3,5')는 추가된 노드로 N 에서 유입링크로 추가된 것이다. N 를 기반으로 N'에 추가된 각 링크까지의 최적경로를 탐색하기 위해서는 우선적으로 N 의 표지정보를 활용해야 하므로 추가된 노드로의 유입링크의 표지를 확정해야 한다. 〈그림 7〉에서 링크(4'2)의 표지확정과정을 예로 들면, N 의 링크(2,4)와 (3,4)와 N'의 링크(4',2)와 링크탐색과정의 비교를 통하여 최적경로비용이 나타나는 방향으로 표지를 확정하게 된다. 즉,

$\pi^{R(4',2)} = \min(\pi^{R(2,4)} + d_{(2,4)(4,2')} + c_{(4,2')}, \pi^{R(3,4)} + d_{(3,4)(4,2')} + c_{(4,2')})$의 계산에 의한 유입링크표지의 확정이 끝나면, p'의 에서 추가된 링크의 순서에 입각하여 동일한 표지확정절차를 거치게 되면, 최종적으로 목적지와 연결된 링크 (5',S)의 표지도 확정된다. 따라서 확정된 표지를 통하여 최적경로알고리즘의 추가적인 수행 없이 최적경로의 탐색이 가능하다.

3. 알고리즘의 수행속도추정

2장에서 설명했듯이 Avezedo et al(1993)가 제안한 알고리즘은 노드집합 V에 대하여 $O(K^2|V|)$의 최

대수행시간을 갖는다. 본 알고리즘은 가감 없이 링크를 노드처럼 활용한 것으로 Avezedo et al(1993) 알고리즘을 유추하여 설명하면, 링크집합 L을 노드로 계산하여 최대수행시간은 $O(K^2|L|)$이다. 일반적으로 네트워크에는 노드 수에 보다 링크 수가 많으므로 이론상으로 노드기반의 Avezedo et al (1993) 알고리즘보다 수행속도는 느리다고 할 수 있다.

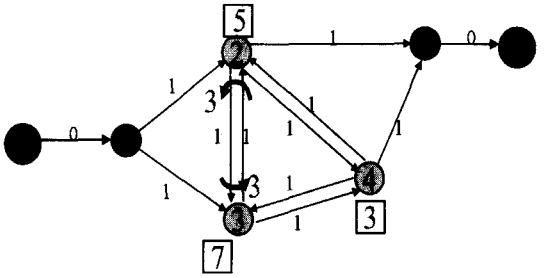
Ⅳ. 사례연구

제안된 알고리즘은 기존의 최적경로알고리즘과 달리 네트워크확장 알고리즘에 대한 세심한 고려가 필요하다. 알고리즘은 Visual Studio (6.0)의 MFC(Microsoft Foundation Class)에서 제공하는 자료구조(Data Structure)를 활용하여 구축되었다. 네트워크확장 알고리즘의 적용하기 위해 N에서 N'으로 확장 될 때 전체 네트워크의 변형 없이 추가되는 링크와 노드에 대해서만 고려가 가능하도록 CObArray 클래스를 활용한 클래스 링크리스트구조(Class Linked-List Structure)로 작성하였다. CObArray 클래스는 링크(LINK)와 노드(NODE)의 추가 클래스가 생성될 때 마다 기존 N의 LINK와 NODE 클래스에 추가된 N'의 LINK와 NODE 클래스를 리스트 형태로 연결시켜 준다. 〈표1〉

〈표 1〉 네트워크 확장을 위한 제안된 알고리즘의 자료구조

일반적인 방법	네트워크 확장이 필요 (적용된 방법)

class GeneralALG {
class LINK;
class NODE;
}
	class NewALG {
	CObArray LINK;
	CObArray NODE;
	}



〈그림 11〉 네트워크 N (5노드 10링크)

의 GeneralALG을 일반적인 방법을 적용하는 경우, LINK와 NODE 클래스를 위한 메모리가 고정되므로, 추가되는 부분을 함께 구축하는 경우 N의 메모리를 해체해야 하며, N 뿐만 아니라 N'에 추가된 LINK와 NODE 클래스를 함께 고려해야 하므로, 네트워크의 초기 구축과정부터 다시 진행해야 하는 부담이 있다. 〈표 1〉의 NewALG 클래스는 CObArray를 활용하여 적용된 클래스 구조이다.

위에서 구축된 프로그램을 통하여 〈그림 11〉의 사례

〈표 2〉 알고리즘의 수행결과

루프를 포함한 경로(Avezedo et al (1993)		노드 비루프 경로		링크 비루프 경로	
20 경로(노드순서)	비용	7 경로(노드순서)	비용	13 경로(노드순서)	비용
R→1→2→5→S	7	R→1→2→5→S	7	R→1→2→5→S	7
R→1→2→4→5→S	11	R→1→2→4→5→S	11	R→1→2→4→5→S	11
R→1→3→4→5→S	13	R→1→3→4→5→S	13	R→1→3→4→5→S	13
R→1→3→2→5→S	15	R→1→3→2→5→S	15	R→1→3→2→5→S	15
R→1→2→3→2→5→S	17	R→1→3→4→2→5→S	19	R→1→2→3→2→5→S	17
R→1→3→4→2→5→S	19	R→1→2→3→4→5→S	19	R→1→3→4→2→5→S	19
R→1→2→3→4→5→S	19	R→1→3→2→4→5→S	19	R→1→2→3→4→5→S	19
R→1→3→2→4→5→S	19			R→1→3→2→4→5→S	19
R→1→2→3→2→4→5→S	21			R→1→2→3→2→4→5→S	21
R→1→3→2→3→2→5→S	23			R→1→2→4→3→2→5→S	25
R→1→2→4→3→2→5→S	25			R→1→2→3→4→2→5→S	25
R→1→2→3→4→2→5→S	25			R→1→3→2→3→4→5→S	25
R→1→2→3→2→3→2→5→S	25			R→1→3→2→3→4→2→5→S	31
R→1→3→2→3→4→5→S	25				
R→1→2→3→2→3→4→5→S	27				
R→1→3→2→3→2→4→5→S	27				
R→1→3→4→2→3→2→5→S	29				
R→1→2→4→3→2→4→5→S	29				
R→1→2→3→2→3→2→4→5→S	29				
R→1→3→2→3→2→3→2→5→S	31				

밑줄 : 링크루프를 포함하는 경로

〈표 3〉 부분경로삭제 링크 비루프 탐색경로의 비교

K	전체경로삭제 알고리즘	비용	부분경로삭제 알고리즘	비용
1	R→1→2→5→S	7.0	R→1→2→5→S	7.0
2	R→1→2→4→5→S	11.0	R→1→2→4→5→S	11.0
3	R→1→3→4→5→S	13.0	R→1→3→4→5→S	13.0
4	R→1→3→2→5→S	15.0	R→1→3→2→5→S	15.0
5	R→1→2→3→2→5→S	17.0	R→1→2→3→2→5→S	17.0
6	R→1→3→4→2→5→S	19.0	R→1→3→4→2→5→S	19.0
7	R→1→2→3→4→5→S	19.0	R→1→3→2→4→5→S	19.0
8	R→1→3→2→4→5→S	19.0	R→1→2→3→4→5→S	19.0
9	R→1→2→3→2→4→5→S	21.0	R→1→2→3→2→4→5→S	21.0
10	R→1→2→4→3→2→5→S	25.0	R→1→2→4→3→2→5→S	25.0
11	R→1→2→3→4→2→5→S	25.0	R→1→3→2→3→4→5→S	25.0
12	R→1→3→2→3→4→5→S	25.0	R→1→2→3→4→2→5→S	25.0
13	R→1→3→2→3→4→2→5→S	31.0	R→1→3→4→2→3→2→5→S	29.0
14			R→1→3→2→3→4→2→5→S	31.0
15			R→1→3→4→2→3→2→4→5→S	33.0
16			R→1→2→4→3→2→3→4→5→S	35.0

연구를 수행하였다. 〈그림 11〉 네트워크는 12개의 링크와 7개의 노드로 구성되어 있으며, 링크 상에 거리(비용)가 나타나 있고 노드의 사각형을 중심으로 전방향 회전점수가 존재한다. U-방향회전은 방향화살표가 존재하는 방향의 2->3->2와 3->2->3 두 방향으로 가능하며, 회전점수는 동일하게 3이다. 출발지와 도착지는 각각 노드R과 S이며 출발지 R에서 도착지 S까지 최대 20개까지 경로를 탐색하여 3가지 결과- 1) 루프를 포함하는 경로, 2) 노드 비루프 경로, 3) 링크 비루프 경로-를 비교한다.

알고리즘의 3가지 수행결과는 〈표 2〉에서 보는 바와 같다. 우선 링크루프를 포함하는 경로는 20개, 노드 비루프 경로는 7개, 링크 비루프 경로는 13개를 각각 탐색하였다. 알고리즘은 기존에 Avezedo et al(1993)에서 원칙적으로 루프를 포함하는 알고리즘에 링크 및 노드 비루프 경로 탐색문제를 추가하여 해결할 수 있음을 예시하고 있다. 교차지점에서 발생하는 회전지체와 금지를 반영하여 루프경로도 제어할 수 있는 방안임이 사례연구를 통해 파악되었다.

〈표 3〉는 위의 링크 비루프 결과를 부분경로삭제 알고리즘인 Yen (1971) 알고리즘을 링크표지접근 방안을 도입-자세한 사항에 대해서는 Lee (2004) 참조-하여 비교한 것으로 최대 20개의 경로를 탐색하였다. 전체경로삭제 알고리즘은 총 13개 부분경로삭제 알고리즘은 총 16개의 합리적 통행경로를 탐색하였다. 부분경로삭제 방법에 비하여 적은 경로를 탐색하는 것은 부분경로삭제방식이 각각의 링크에 대하여 경로를 비교하기 때문에 보다 다양한 경로조합이 나타난다. 반면 전체경로삭제 방법은 전체 경로의 탐색을 미리 방지하기 때문에 다양한 경로가 나타날 확률이 적어지게 만든다.

V. 결론

도시교통망의 특징은 교차로 통행이 통행시간의 많은 비중을 차지하는데 있다. 따라서,교차로의 통행행태를 면밀하게 추정하는 것이 경로탐색에 있어서 매우 중요한 사항이다. K개의 경로탐색을 위한 전체경로삭제 알고리즘은 교차로의 통행을 고려하지 못하여 루프를 발생시키는 원천적인 문제가 존재하였다. 본 연구에서 링크표지를 활용하여 비루프 경로탐색 알고리즘을 제안하였다. 결과로 볼 때 제안된 알고리즘은 연구가정에 입각하여 매우 의미 있는 결과를 산출하는 것으로 파악되었다. 제안된 알고리즘으로 기존의 전체경로삭제에 기반을 둔 K경로탐색 알고리즘이 고려하지 못했던 (1) 교차로의 회전통행점수 및 통행규제의 고려와, (2) 링크의 반복이 허용하지 않는 도시가로망의 U-턴, P-턴 등의 합리적 통행행태의 표현과, (3) 노드의 반복도 허용하지 않아 U-턴, P-턴 등 합리적 통행행태를 전반적으로 설명하기에는 부적합하지만 일반적인 네트워크 이론의 관심사인 단순경로(Simple Paths)를 구분하여 탐색하는 것이 가능해졌다. 추후 실제 가로망에서 알고리즘의 수행시간을 계측할 수 있는 다양한 사례연구가 필요하며, 통행제약조건을 포함하는 K개의 경로탐색 알고리즘으로의 개발이 요구된다.

참고문헌

1. 임용택(2004) 일반가로망에서 교통정보제공을 위한 n-path 알고리듬의 개발, 대한교통학회지, 제22권 제4호, 대한교통학회, pp.135~145.
2. Azevedo J. A., Costa M. E. O. S., Madeira J.J.E.R.S., and Martins E.Q.V. (1993) An Algorithm from the Ranking of Shortest Paths, European Journal of Operational Research, Vol.69, pp.97~106.
3. Bellman R. and Kalaba R. (1968) On Kth Best Policies. J. SIAM 8, pp.582~588.
4. Dijkstra E. W. (1959) A Note of Two Problems in Connected with Graphs. Numerical Mathematics. I, pp.269~271.
5. Kirby R. F. and Potts R. B. (1969) The Minimum Route Problem for Networks with Turn Penalties and Prohibitions. Transportation Research 3, pp.397~408.
6. Lee M. (2004) Transportation Network Models and Algorithms Considering Directional Delay and Prohibition for Intersection Movement, Ph.D. Thesis, University of Wisconsin-Madison.
7. Martins E.Q.V. (1984) An Algorithm for Ranking Paths that May Contain Cycles, European Journal of Operational Research, Vol.18, pp.123~130.
8. Moore E. F. (1957) The Shortest Path through A Maze. Proc. Int. Conf. on the Theory of Switching. Harvard Univ., Cambridge, MA.
9. Pollack M. (1961) The Kth Best Route Through A Network, Operations Research, Vol.9, pp.578~580.
10. Potts R.B. and Oliver R.M.(1972) Flows in Transportation Networks. Academic Press.
11. Shier R. D. (1979) On Algorithms from Finding the k Shortest Paths in a Network, Networks, Vol.9, pp.195~214.
12. Yang H.H. and Chen Y.L. (2003) Finding K Shortest Looping Paths in A Traffic-Light Network, Computer & Operations Research.
13. Yen J.Y. (1971) Finding the K shortest Loopless Paths in a Network, Management Science, Vol.17, pp.711~715.

♧ 주 작 성 자 : 신성일
♧ 논문투고일 : 2004. 8. 2
논문심사일 : 2004. 10. 26 (1차)
2004. 11. 17 (2차)
2004. 11. 23 (3차)
심사판정일 : 2004. 11. 23
♧ 반론접수기한 : 2005. 4. 30