산업기술연구(강원대학교 산업기술연구소 논문집), 제21권 A호, 2001.
Journal of Industrial Technology, Kangwon Nat'l Univ., Korea, No. 21 A, 2001.

첨단교통정보시스템의 최적경로 알고리즘 개발

Development of Optimal Path Algorithm for Advanced Traveler Information System

김 성 수* 차 영 민**
Kim, Sung-Soo Cha, Young-Min

Abstract

The objective of this paper is to develop the optimal path algorithm for dynamic route guidance system in advanced traveler information system (ATIS). The travel time is forecasted in each path between network nodes. Floyd-Warshall algorithm is used to find the optimal route based on this forecasted travel time in dynamic traffic network. This algorithm is modified to apply the real traffic network that has left-turn restriction, U-turn, and P-turn. A big value is assigned to one of arcs in turn restriction and a virtual node is used to consider U-turn and P-turn for Floyd-Warshall algorithm.

키워드 : *U-턴, P-턴, 교통네트워크, Floyd-Warshall 알고리즘*
Keywords : *U-turn, P-turn, traffic network, Floyd_Warshall Algorithm*

1. 연구의 목적 및 배경

우리 나라의 교통실태를 보면 만성적인 교통혼잡으로 인하여 연 10조원 이상의 도로교통 혼잡비용을 부담하고, 매년 2조원 이상씩 증가추세에 있고, 또한 물류비 과다(국가 전체물류비 :GNP대비 15%, 기업매출액 대비 17% - 선진국의 2배 이상 과다)로 산업의 국가 경쟁력이 약화되는 요인이 되고 있다. 이런 상황하에서 첨단 정보통신 기술을 기반으로 하는 물류교통정보망을 이용한 물류교통정보시스템의 개발이 절실하다. 물류교통정보시스템이란 물류교통활동에 수반되는 정보흐름을 정보통신기술과 접목하여 전산화, 자동화하고 물류교통연결점과의 유기적인 연계운용과 물류교통업무의 일괄 처리 촉진을 통한 물류교통 비용절감과 물류교통서비스의 질적 향상을 추구는 것이다.

최근 도시 내 교통혼잡의 증가에 따라 기존 도로의 운영효율 증진 및 운전자 편의를 위해 GPS(Global Positioning System) 및 GIS(Graphical Information System)기술, 무선통신 기술 등을 결합한 첨단화물정보시스템(CVO, Commercial Vehicle Operation)과 지능형교통정보시스템(ATIS, Advanced Traveler Information System)이 지능형교통시스템(ITS, Intelligent Transport System)의 일환으로 개발되어 실용화추세에 있다. 특히, 첨단교통정보 시스템의 가장 핵심적인 기능은 최적경로 안내인데, 최적경로탐색 알고리즘은 1950년대 말에 여러 가지 알고리즘들이 완성되었다. 잘 알려진 알고리즘으로 특정시작노드에서 그 외의 모든 노드까지의 최단경로를 최적원리 즉, 최적경로의 부분경로도 또한 최적경로라는 성질을 기초로 하여 개발되어진

* 강원대학교 산업공학과 조교수, 공학박사
** 강원대학교 대학원 산업공학과 석사과정

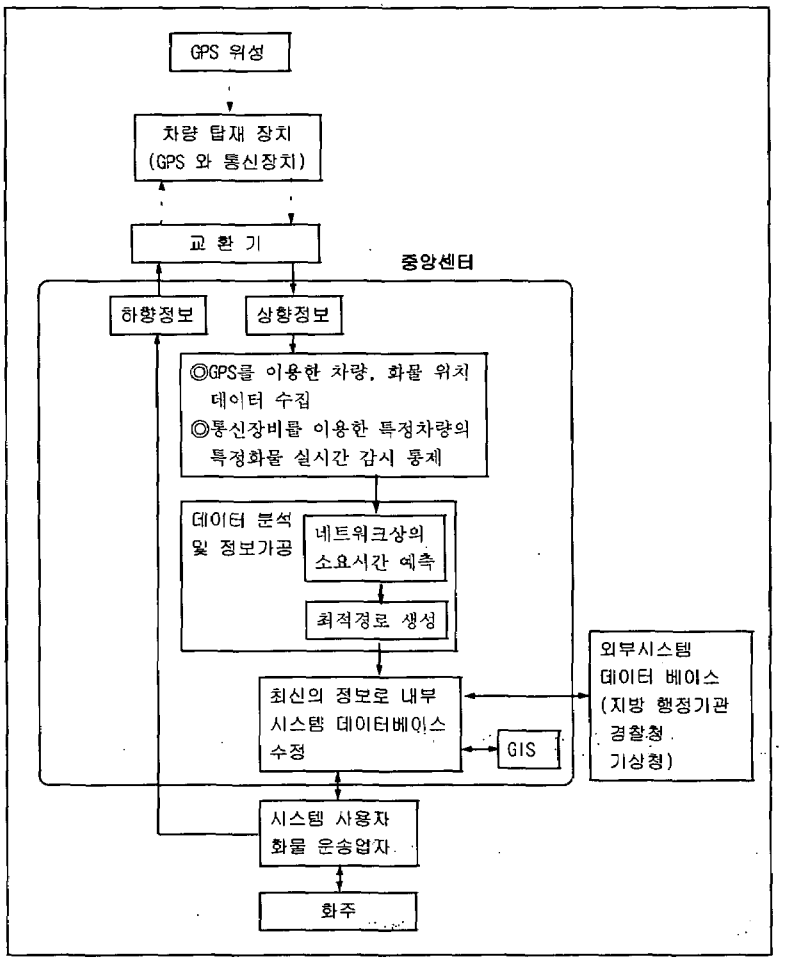


그림 1 최적경로 시스템

Dijkstra알고리즘, 음수길이의 호값이 있는 일반적인 네트워크에 적용하는 Ford-Bellman알고리즘과 Yen의 알고리즘, 다수의 최적경로를 구하는 Shier와 Dreyfus 알고리즘 등이 있다. 또한, Floyd-Warshall알고리즘은 한번에 각 교점간에 모든 최적경로를 구하는 알고리즘으로 다른 최단경로 알고리즘에 비하여 간단하고 프로그램과 실제 구현이 쉽기 때문에 본 논문에 사용하고자 한다. 그 외에도 휴리스틱에 의한 탐색방법 등이 있다. 이미 고전이 되어버린 최적경로탐색 알고리즘들은 실제 도로교통망에 적용시키기에 적합하지 않는 부분들이 있다.[1][4][5][7]

이 논문의 목적은 우리 나라의 실정에 맞는 도로 교통네트워크에서 최적경로를 탐색하기 위해 복잡한 회전 제약사항 등 다양한 도로교통 제약사항을 반영하는 탐색기법에 대하여 논하고자 한다. 본 논문에서는 Floyd-Warshall알고리즘을 바탕으로 도로교통 네트워크상에서 좌회전금지, U턴 및 P턴을 고려한 수정된 최적경로 알고리즘을 제안하고자 한다.

2. 첨단 물류교통정보시스템의 최적경로 안내서비스

최적경로시스템은 실시간이동체관리 모듈과 첨단정보제공 모듈로 구성되며, 운전자에게 가고자 하는 목적지까지 가장 짧은 소요시간 안에 도착할 수 있도록 정보를 제공하고, 교통혼잡과 안전 운행을 보장하여 주는 시스템이다. 최적경로시스템의 실시간이동 관리 모듈과 첨단정보 제공 모듈은 <

그림 1>을 참조하여 다음과 같이 설명되어질 수 있다.

실시간이동체관리모듈은 통신장비와 GPS장비를 갖춘 차량과 정보 관리 및 제공 기능을 하는 중앙센터 그리고 이 시스템의 사용자인 화물운송관리자로 구성되어지는데 이들 모두는 물류교통정보망으로 연결되어진다. 각각의 차량은 GPS장비를 탑재하여 운전자가 차량 위치를 알 수 있으며, 이들 차량 위치 정보는 통신장비를 이용하여 교환기를 통하여 중앙 센터로 전송된다(상향정보, 차량 → 중앙센터). 각 차량들로부터 수집되는 정보로 데이터 분석을 통하여 중앙센터에서 차량 운행에 필요한 유용한 정보들을 만들어낼 수 있다. 이렇게 만들어진 새로운 정보는 내부데이터 베이스를 최신정보로 수정하는데 사용된다. 내부 데이터 베이스는 경찰청, 도로공사 등 외부 데이터 베이스와 연결되어 있고, GIS와 연결되어 사용자에게 서비스할 수 있다. 화물운송업자는 이 모듈을 사용하여 효과적으로 차량 운행 및 화물운송관리를 할 수 있다. 즉, 화물과 차량의 위치, 특정차량이 싣고 있는 특정화물의 이동상태에 대한 정보를 실시간으로 감시 통제 할 수 있다. 그래서 화물 운송업자는 화주에게 실시간으로 화물의 이동현황을 서비스하게 된다. 첨단정보제공 모듈은 중앙센터, 차량 그리고 교통정보 제공처로 구성되어진다. 이들 또한 물류교통정보망으로 연결되어 실시간이동체관리모듈과 정보를 공유할 수 있다. 중앙센터에서 세 번의 단계를 거쳐 정보를 처리한다.[2][6]

첫째, 교통 물류정보를 교통정보 제공처와 실시간 이동체관리 모듈로부터 전송 받는다.
둘째, 이들 정보를 분류 등급화하고 유용한 정보로 처리 가공한다.
셋째, 통신 매체를 이용하여 적정가격에 제공한다 (하향정보, 중앙센터 → 차량).

이 모든 단계는 중앙센터에서 처리되며, 여러 정보 제공처로부터 얻어진 정보들은 교통 상황을 모델링 하는데 사용되고, 예측기법을 이용하여 몇 분 후의 교통상황을 예측하고 최적경로정보 등을 서비스 하게 된다.

3. Floyd - Warshall 알고리즘과 실제 네트워크 적용의 한계

Floyd - Warshall 알고리즘은 네트워크의 모든 두 교점간의 최적경로를 구하는 알고리즘이다. Dijkstra 알고리즘처럼 특정의 한 교점으로부터 다른 모든 (N-1)개 교점까지의 최단경로를 구하는 것이 아니라. N개의 교점 각각으로부터 다른 (N-1)개의 교점까지 N(N-1)개의 최단경로를 구한다. 이 논문에서 최적경로를 찾고자 할 때 노드와 노드사이의 소요되는 시간이 기준이 되기 때문에 도로교통 네트워크에서는 음수의 값이 필요치 않으므로 Ford-Bellman알고리즘과 Yen의 알고리즘 같은 음수길이가 존재하는 네트워크의 최적경로를 구하는 알고리즘 보다 효율적이고 쉽게 프로그램되는 Floyd-Warshall 알고리즘을 사용한다.

교점이 m개인 네트워크에서 임의의 i에서 j까지의 최적경로가 중간교점 1, 2, …, m-1만을 사용하여 구성되어 있을 때(이 때의 최적 소요시간은 $\pi_{ij}^{(m)}$임), 만약 교점 m을 통과하도록 허용한다면 새로운 최적경로는 교점 m을 통과하지 않을 수도 있고 또는 통과할 수도 있다.

(1) 산출되어진 최적경로가 교점 m을 통과하지 않게 되면 $\pi_{ij}^{(m)} = \pi_{ij}^{(m-1)}$이 될 것이며, (2) 산출되어진 최적경로가 교점 m을 통과하게 되면 $\pi_{ij}^{(m)} = \pi_{im}^{(m-1)} + \pi_{mj}^{(m-1)}$이 되어야 한다. 따라서 위의 설명을 <그림 2>에 나타나 있듯이 다음과 같이 삼각연산을 표현할 수 있다.

$$\pi_{ij}^{(m)} = \min\{\pi_{ij}^{(m-1)},\ \pi_{im}^{(m-1)} + \pi_{mj}^{(m-1)}\}$$

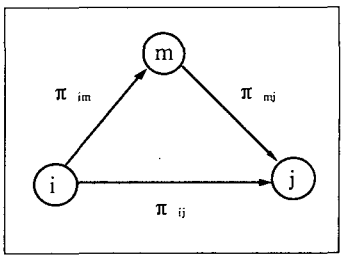


그림 2 삼각연산의 표현

$\pi_{ij}^{(0)}$는 네트워크의 교점간의 소요시간 d_{ij}로 주어진다.

(i) m = 1일 때
모든 i, j에 대해 $\pi_{ij}^{(0)}$ (즉 d_{ij}. i≠1, j≠1)을 $\pi_{i1}^{(0)} + \pi_{1j}^{(0)}$ (즉 $d_{i1} + d_{1j}$)와 비교한다.

if $\pi_{ij}^{(0)} > \pi_{i1}^{(0)} + \pi_{1j}^{(0)}$ **then**
$\pi_{ij}^{(1)} = \pi_{i1}^{(0)} + \pi_{1j}^{(0)}$
if $\pi_{ij}^{(0)} < \pi_{i1}^{(0)} + \pi_{1j}^{(0)}$ **then**
$\pi_{ij}^{(1)} = \pi_{ij}^{(0)}$

(ii) m = 2일 때
확정된 $\pi_{ij}^{(1)}$ (i≠2, j≠2)을 $\pi_{i2}^{(1)} + \pi_{2j}^{(1)}$와 비교하여 m = 1일 때와 같은 방식으로 작은 값이 새로운 최적소요시간 $\pi_{ij}^{(2)}$로 된다.
유의할 것은 m = 2일 때의 계산을 위해서는 m = 1의 결과만이 필요하다는 점이다.
즉 $\pi_{ij}^{(0)} = d_{ij}^{(0)}$에서 시작하여 이로부터 $\pi_{ij}^{(1)}$를 계

산하고, $\pi_{ij}^{(1)}$로부터 $\pi_{ij}^{(2)}$를 계산하다.

(iii) 같은 방식으로 m = N일 때의 $\pi_{ij}^{(N)}$는 두 교점 i, j간 의 최적경로의 소요시간이 된다. 즉 두 교점 i와 j에 대해 삼각연산을 수행할 때 m =1, 2, …, N의 순서로 모든 교점 m을 고려하면 최적경로를 구할 수 있다. 따라서 m은 삼각연산을 적용하는 반복순서를 나타내며 $\pi_{ij}^{(m)}$은 m번째 반복에서의 최적소요시간의 추정치를 나타낸다.

본 논문에서 두 가지 N×N행렬을 사용하였다. 첫 번째 행렬은 각 반복단계에서 계산되는 최적소요시간의 추정치를 기록한 행렬로서 최적소요시간 행렬이라고 부른다. 최적소요시간행렬의 일반적인 표현은 m번의 반복후의 행렬 $\pi^{(m)} = \pi_{ij}^{(m)}$, $\forall$i,j (i=1~N , j=1~N,i≠m, j≠m)이다. $\pi^{(m)}$은 $\pi^{(0)}$에서 시작하는데, $\pi_{ij}^{(0)} = d_{ij}$이다. $\pi^{(1)}$은 $\pi^{(0)}$의 모든 원소들에 삼각연산을 수행하여 구해지며, 같은 방법으로 m을 증가시켜가며 반복한다. 두 번째 행렬은 최적경로의 중간노드를 식별하기 위한 것으로서 최적경로행렬이라고 부른다. m번째 반복에서의 경로행렬은 $P^{(m)} = P_{ij}^{(m)}$, $\forall$i,j (i=1~N , j=1~N, i≠m, j≠m)로 정의된다. $\pi_{ij}^{(m)}$은 교점 i에서 j까지의 최적 경로의 소요시간을 $P_{ij}^{(m)}$은 교점 i에서 교점 j까지의 최적경로에서 최초 중간교점을 나타낸다.

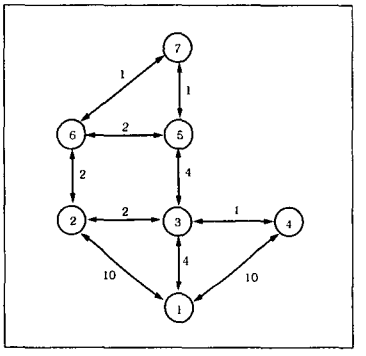


그림 3 교통 네트워크의 예

<그림 3>의 교통 네트워크의 예에서 Floyd-Warshall알고리즘을 적용하여 $\pi^{(7)}$와 $P^{(7)}$를 구하면 다음과 같다. 위 최적소요시간 행렬, $\pi^{(7)}$과 최적경로 행렬, $P^{(7)}$을 보면 노드 ①에서 노드 ⑥까지의 최적소요시간은 8이고, 최적경로는 ① → ③ → ② → ⑥이며, 또한 노드 ④에서 노드 ⑥까지의 최적소요시간은 5이고 최적경로는 ④ → ③ → ② → ⑥임을 $\pi^{(7)}$과 $P^{(7)}$행렬을 사용하여 구할 수 있다. 노드 ③에서 노드 ⑥까지의 최적경로는 언제나 변함이 없다. 이는 다음에 설명이 될 좌회전 금지 고려 시 최적경로행렬 생성과 비교하여 차이점을 설명하고자 한다.

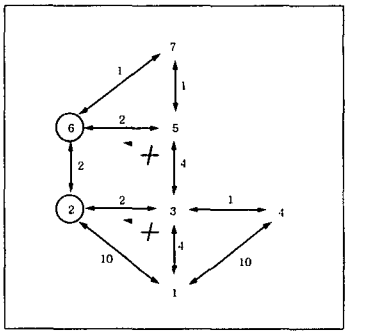
$\pi^{(7)}=$

0	6	4	5	8	8	9
6	0	2	3	4	2	3
4	2	0	1	4	4	5
5	3	1	0	5	5	6
8	4	4	5	0	2	1
8	2	4	5	2	0	1
9	3	5	6	1	1	0

$P^{(7)}=$

1	3	3	3	3	3	3
3	2	3	3	6	6	6
1	2	3	4	5	2	5
3	3	3	4	3	3	3
3	6	3	3	5	6	7
2	2	2	2	5	6	7
5	6	5	5	5	6	7

<그림 3>의 교통 네트워크에 2개의 좌회전 금지를 적용시켜 수정한 것이 <그림 4>이다. <그림 4>의 교통 네트워크는 단순한 예제로써 최적경로 알고리즘과 컴퓨터를 이용하지 않고도 간단히 최적경로를 산출할 수 있다. 특정 시작노드 ①에서 목적노드 ⑥까지의 좌회전 금지 적용 시 최적경로는 ① → ③ → ⑤ → ⑦ → ⑥ 이다. 또한 노드 ④에서 노드 ⑥까지의 최적경로는 ④ → ③ → ② → ⑥ 이다. 이들 최적경로들을 이용하여 반대로 $P^{(7)}$경로 행렬을 다음과 같이 각각 구성하여 볼 수 있다.



<그림 4> ①-③-②, ③-⑤-⑥
좌회전 금지 교통 네트워크의 예

노드	6
1	3
2	
3	5
4	
5	7
6	
7	6

$P^{(7)}$ (①에서 ⑥)

< 행렬 1 >

노드	6
1	
2	6
3	2
4	3
5	
6	
7	

$P^{(7)}$ (④에서 ⑥)

< 행렬 2 >

Floyd - Warshall 알고리즘의 $P^{(7)}$경로행렬에서는 <행렬 1>과 <행렬 2>는 같아야 한다. 하지만 위에서도 보는 바와 같이 <행렬 1>의 $P_{36}^{(7)}$ = 5이고 <행렬 2>의 $P_{36}^{(7)}$ = 2로 노드 ③에서 노드⑥으로의 중간노드가 각각 ⑤와 ②로 서로 같지 않다. 따라서, 노드가 1 ~ 7까지인 네트워크에서 $P^{(7)}$ 즉, 7회 반복에서의 경로행렬을 생성하는데 있어 좌회전금지를 적용하면 각각의 모든 노드에 최적경로의 중간노드를 식별할 수 없다. 이와같이 좌회전금지를 고려하여 최적경로를 구하고자 할 때는 본래 Floyd - Warshall 알고리즘을 사용하여 경로행렬을 생성할 수 없다.

4. 실제 물류교통 네트워크 적용을 위한 최적경로알고리즘 개발

4.1. 수정된 Floyd - Warshall 알고리즘 개발

3절에서 언급했듯이 교통 네트워크에 좌회전 금지 사항을 고려했을 때 N개의 노드로부터 N-1개 노드로의 경로행렬의 생성이 본래의 Floyd-Warshall 알고리즘을 사용하여서는 불가능하다. 하지만 특정 시작노드 S에서 목적노드 T까지의 최적소요시간과 최적경로행렬은 본 논문에서 제안하는 수정된 Floyd - Warshall알고리즘을 바탕으로 시작노드 N개와 목적노드 N-1개의 각각의 최적소요시간과 최적경로를 구할 수 있다. 시작노드 S와 목적노드 T를 알고 있을 때 좌회전 금지를 포함한 최적소요시간 및 최적경로의 구현방법에 대하여 서술하면 다음과 같다.

교통 네트워크에서 좌회전 금지 등은 사전에 알 수 있으므로, 먼저 N개의 노드를 가진 교통 네트워크에 좌회전 금지에 대한 정보를 i, L, j로써 첨가한다. 좌회전 금지요소를 포함한 새로운 교통 네트워크 <그림 5>에서 좌회전 (i→L→j)을 하지 않고, 시작노드 S에서 목적노드 T까지 우회하여 갈 수 있는 경로들을 그림으로 도시해 보면 <그림 5-1~9>과 같이 표현할 수 있다.

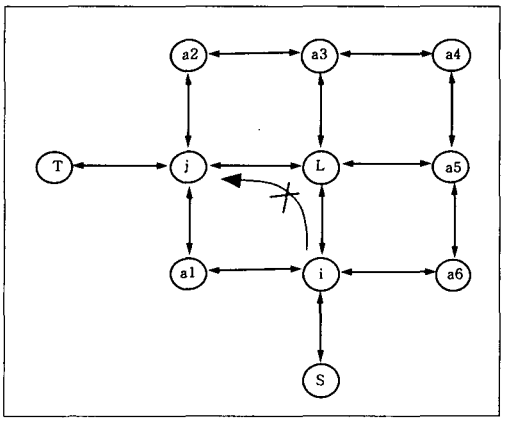
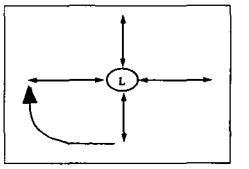
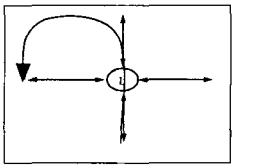


그림 5 좌회전 금지요소(i - L - j)를 포함하는 교통 네트워크

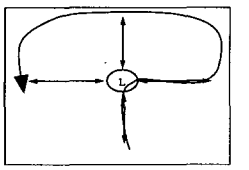
<그림5>에서 좌회전 금지 시 대체 가능한 경로들은 경유한 노드를 중복하여 경유하는 경우(P턴, U턴)를 제외하면 다음과 같다.



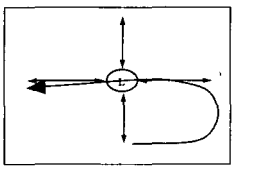
<그림 5-1> 경로 1



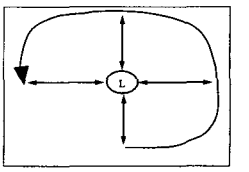
<그림 5-2> 경로 2



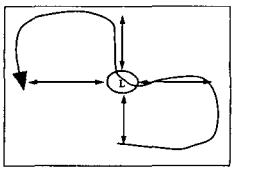
<그림 5-3> 경로 3



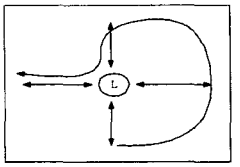
<그림 5-4> 경로 4



<그림 5-5> 경로 5



<그림 5-6> 경로 6



<그림 5-7> 경로 7

i → L = ∞ 일 때
경로 ④, ⑦
L → j = ∞ 일 때
경로 ②, ③
i → L = ∞ 이고,
L → j = ∞ 일 때
경로 ①, ⑤, ⑥

이들 단일 좌회전 금지를 고려한 경로를 포함한 교통 네트워크의 최적경로 탐색 접근 방법은 다음과 같이 설명할 수 있다. 좌회전 금지 요소 (i → L → j)가 첨가된 전체 교통 네트워크에 대하여 최적소요시간과 최적 경로를 구하고자 할 때, 먼저 사전에 알고 있는 좌회전금지요소(i → L → j)를 고려하여 Floyd - Washall 알고리즘을 사용하여 최적경로를 찾는 것을 다음과 같이 설명할 수 있다.

Floyd - Warshall 알고리즘을 이용하여 최적소요시간을 계산하기 위한 초기행렬 $\pi^{(0)}$행렬에서 좌회전금지요소 i → L → j를 고려해주기 위하여 $\pi_{iL}^{(0)}$를 ∞로 치환한 행렬을 초기행렬로 하여 첫 번째 최적소요시간 행렬 $\pi^{(N)}$를 구하고, $\pi_{Lj}^{(0)}$를 ∞로 치환한 행렬을 초기행렬로 하여 두 번째 최적소요시간 행렬 $\pi^{(N)}$를 구한다. 첫 번째 행렬은 $\pi_{iL}^{(0)}$를 ∞로 치환했으므로 좌회전금지 부분 중 노드 i, L을 지나는 호 i → L을 경유하지 않는 모든 최적경로, $P^{(N)}$를 구할 수 있고, 두 번째 행렬은 $\pi_{Lj}^{(0)}$를 ∞로 치환했으므로 좌회전금지 부분 중 노드 L, j를 지나는 호 L → j를 경유하지 않는 모든 최적경로행렬, $P^{(N)}$를 구할 수 있다. 따라서 이 두 행렬은 좌회전금지 요소 i → L → j를 경유하지 못하도록 조정하였다.

임의의 시작노드 S에서 임의의 목적노드 T까지의 최적경로는 첫 번째 최적소요시간행렬 $\pi^{(N)}$의 $\pi_{ST}^{(N)}$와 두 번째 최적소요시간행렬 $\pi^{(N)}$의 $\pi_{ST}^{(N)}$를 비교하여 작은 값을 최적소요시간으로 책정한다. 또한 최종 최적소요시간으로 선택된 최적소요시간행렬과 같이 구한 경로행렬, $P^{(N)}$에서 최적경로를 찾을 수 있다. 즉 경로행렬 $P^{(N)}$의 N×N행렬에서 임의의 시작노드 S에서 임의의 목적노드 T까지의 최적 경로를 산출한다. 예를 들어 $P_{ST}^{(N)} = q$ 이면 최적경로는 S → q → ···→ T 이고, 만약 $P_{qT}^{(N)} = r$ 이면 최적경로는 S → q → r → ··· → T 이다.

좌회전금지구간이 2개 이상일 때의 최적경로 탐색방법은 다음과 같다. 먼저 2개의 좌회전금지구간을 갖는 네트워크에서의 최적경로를 구하여보자. 좌회전금지요소 i, L, j요소 외에 좌회전금지요소 e, M, f를 추가한다. 추가된 e, M, f는 i가 e와 L이 M과 j가 f와 모두 같지 않으면 된다. 2개의 좌회전금지구간을 포함한 네트워크는 4가지의 최적소요시간행렬 $\pi^{(N)}$과 최적경로행렬 $P^{(N)}$를 생성하게된다. 단일 좌회전금지구간을 포함한 네트워크에서 좌회전금지요소 i → L → j를 고려해주기 위하여 초기행렬 $\pi^{(0)}$행렬의 $\pi_{iL}^{(0)}$를 ∞로 치환한 행렬과 $\pi_{Lj}^{(0)}$를 ∞로 치환한 행렬과 추가되어진 좌회전금지요소 e, M, f를감안하여 주기 위해 $\pi_{eM}^{(0)}$를 ∞로 치환한 행렬과 $\pi_{Mf}^{(0)}$를 ∞로 치환한 행렬을 각각 배당하여 4개의 초기행렬 $\pi^{(0)}$를 구하여 각각을 Floyd - Warshall 알고리즘을 이용하여 아래와 같이 최적소요시간행렬, $\pi^{(N)}$과 최적경로행렬 $P^{(N)}$를 구한다.

$$\pi_{iL}^{(0)} = \infty \begin{cases} \pi_{eM}^{(0)} = \infty \;\Rightarrow\; ① \;\pi^{(0)} \\ \pi_{Mf}^{(0)} = \infty \;\Rightarrow\; ② \;\pi^{(0)} \end{cases}$$

$$\pi_{Lj}^{(0)} = \infty \begin{cases} \pi_{eM}^{(0)} = \infty \;\Rightarrow\; ③ \;\pi^{(0)} \\ \pi_{Mf}^{(0)} = \infty \;\Rightarrow\; ④ \;\pi^{(0)} \end{cases}$$

즉, 1번 행렬은 $\pi_{iL}^{(0)}$과 $\pi_{eM}^{(0)}$를 ∞로 치환했으므로 좌회전금지 부분 중 호 i → L을 지나지 않고 호 e → M을 경유하지 않는 모든 최적경로, $P^{(N)}$를 구할 수 있고, 2번 행렬은 $\pi_{iL}^{(0)}$과 $\pi_{Mf}^{(0)}$를 ∞로 치환했으므로 좌회전금지 부분 중 호 i → L을 경유하지 않고 호 M → f를 경유하지 않는 모든 최적경로행렬, $P^{(N)}$를 구할 수 있고, 3번 행렬은 $\pi_{Lj}^{(0)}$과 $\pi_{eM}^{(0)}$를 ∞로 치환했으므로 좌회전금지 부분 중 호 L → j를 경유하지 않고 호 e → M을 경유하지 않는 모든 최적경로행렬, $P^{(N)}$를 구할 수 있고, 4번 행렬은 $\pi_{Lj}^{(0)}$과 $\pi_{Mf}^{(0)}$를 ∞로 치환했으므로 좌회전금지 부분 중 호 L → j를 경유하지 않고 호 M → f를 경유하지 않는 모든 최적경로행렬, $P^{(N)}$를 구할 수 있다. 따라서 이 네 행렬은 좌회전금지요소 i → L → j와 e → M → j를 경유하지 못하도록 조정하였다.

임의의 시작노드 S에서 임의의 목적노드 T까지의 최적경로는 1~4번 행렬들 중 4개의 최적소요시간행렬 $\pi^{(N)}$에서 $\pi_{ST}^{(N)}$를 비교하여 가장 적은 값을 최적소요시간으로 선택한다. 또한 최종 최적소요시간으로 선택된 최적소요시간행렬과 같이 구한 경로행렬, $P^{(N)}$에서 최적경로를 찾을 수 있다. 예를 들어 만약 1번 행렬의 $\pi^{(N)}$의 $\pi_{ST}^{(N)}$가 가장 적은 값을 갖는다면 1번의 $P^{(N)}$에서 임의의 시작노드 S에서 임의의 목적노드 T까지의 최적경로를 찾을 수 있다. 2개 이상의 좌회전금지구간을 포함

한 네트워크는 2개일 때의 최적경로를 구하는 방법처럼 늘려 가면 된다. 즉, 3개일 때는 8개, 4개일 때는 16개의 $\pi^{(N)}$과 $P^{(N)}$를 구하여 각 행렬을 비교하면 최종 최적소요시간행렬과 경로행렬을 구할 수 있다. 그러나, 이와같이 수정된 Floyd-Warshall 알고리즘을 이용하여 최적소요시간과 최적경로를 구할 때, 좌회전금지 구간 수가 많아질수록 2^n(n:좌회전금지 구간의 수)으로 계산량이 증가하고 기억용량이 필요하다는 단점이 있다. 그러므로, 이 논문에서 제안한 알고리즘은 좌회전금지 구간 수가 소수 한정되어 포함되어 있는 교통네트워크에 사용되는 것이 바람직하고, 추후 이런 단점을 개선 발전 시킬 연구가 필요하다.

이상의 설명을 정식화하면

단계 0. [초기화]

- $\pi^{(0)} \leftarrow [\pi_{ij}]$ ($\pi_{ii} = 0$, 만약 호(i,j)가 없으면 $\pi_{ij} = \infty$)
- $P^{(0)} \leftarrow [j]$
- $m \leftarrow 0$
- 전체 네트워크에서 노드(i → L → j), (e → M → f)에서 좌회전 금지이면

$$\pi_{iL}^{(0)} = \infty \begin{cases} \pi_{eM}^{(0)} = \infty \rightarrow \pi^{(N)}, \\ P^{(N)} \rightarrow \text{1번 행렬} \\ \pi_{Mf}^{(0)} = \infty \rightarrow \pi^{(N)}, \\ P^{(N)} \rightarrow \text{2번 행렬} \end{cases}$$

$$\pi_{Lj}^{(0)} = \infty \begin{cases} \pi_{eM}^{(0)} = \infty \rightarrow \pi^{(N)}, \\ P^{(N)} \rightarrow \text{3번 행렬} \\ \pi_{Mf}^{(0)} = \infty \rightarrow \pi^{(N)}, \\ P^{(N)} \rightarrow \text{4번 행렬} \end{cases}$$

← 다음과 같이 좌회전 금지 요소를 고려하여 입력

단계 1. [각각의 $\pi^{(m-1)}$에서 행 m과 열 m을 표시함]

- $m \leftarrow m + 1$
- ①, ②, ③, ④ 각각의 $\pi^{(m-1)}$에서 행 m과 열 m을 표시한다.

단계 2. [각각의 $\pi^{(m)}$을 계산함. 즉, 표시된 행과 열을 사용하여 삼각연산을 수행함]

- ①, ②, ③, ④인 경우의 $\pi^{(m)}$을 각각 구한다.
- 표시된 행 m과 열 m 이외의 모든 원소 $\pi_{vw}^{(m-1)}$ ($v \neq m$ & $w \neq m$)을 행 m과 열 m의 원소의 합인 $\pi_{vm}^{(m-1)} + \pi_{mw}^{(m-1)}$과 비교하여, 후자가 작으면 $\pi_{vw}^{(m)}$을 $\pi_{vm}^{(m-1)} + \pi_{mw}^{(m-1)}$로 수정하고 $P_{vw}^{(m)}$을 $P_{vm}^{(m-1)}$로 수정한다.
- m = N이면 단계 3으로 가고, 아니면 단계 1로 간다.

단계 3. [시작노드 S와 도착노드 T에 대한 최소소요시간 $\pi_{ST}^{(m)}$과 최적경로 $P_{ST}^{(m)}$을 구함]

- 시작노드 S와 도착노드 T에 대하여 ①, ②, ③, ④ 각각의 $\pi^{(m)}$에서 각각의 $\pi_{ST}^{(m)}$중에서 가장 작은 $\pi_{ST}^{(m)}$을 최소소요시간으로 결정하고 최소소요시간에 해당 되는 경로행렬 $P^{(m)}$에서 시작노드 S에서 목적노드 T까지의 최적경로를 구한다.

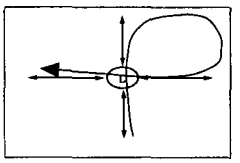
4.2. U턴과 P턴의 첨가

4.1절에서 좌회전 금지를 적용한 Floyd - Warshall 알고리즘을 살펴보았다. 사실상 도로교통 네트워크에서는 좌회전 금지뿐만 아니라 U턴도 존재하므로, 좌회전 금지 시 우회하는 경로와 U턴을 이용한 경로와 P턴을 이용한 경로를 서로 비교하여 가장 최적의 경로를 찾을 수 있다면 좀 더 실제 도로 교통 네트워크에 접근할 수 있을 것이다.[3] 첨가하고자 하는 U턴과 P턴을 이용한 경로들을 구현하기 위해서는 자기자신으로의 경로(경유한 노드를 중복하여 경유하는 경우)를 구하여야 하는데 Floyd - Warshall 알고리즘의 경우는 기본적으로 자기자신으로의 소요시간은 0 이라고 가정하기 때문에 본래의 Floyd - Warshall 알고리즘을 사용해서는 P턴과 U턴의 경로를 고려해 줄 수 없다. 이런 문제를 해결하기 위하여 i → L → j좌회전 금지 시 노드 L을 중심으로 L의 위쪽과 오른쪽에서 U턴이 가능하다고 가정하고, 또한 L을 중심으로 P턴이 가능하다고 가정한다. 이러한 U턴과 P턴은 L노드로의 복귀를 뜻하기 때문에 가상노드 L'를 첨가하여 U턴과 P턴을 이용한 경로를 구현한다.(소요시간행렬 $\pi^{(0)}$과 경로행렬 $P^{(N)}$에 L'를 첨가한다.)

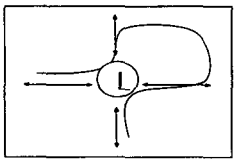
즉, 이 논문에서는 문제의 복잡도를 줄이기 위해서 U턴과 P턴은 좌회전(i→L→j) 금지의 중심이 되는 L에서만 가능하다고 가정한다.
(다른 노드에서도 가능하다면 문제가 복잡해진다.)

먼저 <그림 5-8>, <그림 5-9>, <그림 5-10>, <그림 5-11>상에서 P턴, U턴을 감안한 경로들을 그림에서 표시해 보면 다음과 같이 표시할 수 있다.

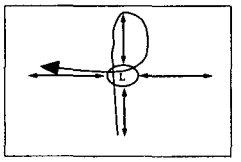
<P턴>
경로8. S → i → L → a3 → a4 → a5 → L → j → T <그림 5-8> 참조
경로9. S → i → L → a5 → a4 → a3 → L → j → T <그림 5-9> 참조
<U턴>
만약 L_node에서 a3노드로 진행 시 a3노드에서 U턴이 가능하다면
경로10. S - i - L - a3 - L - j - T <그림 5-10> 참조
또한 L_node에서 a5노드로 진행 시 a5노드에서 U턴이 가능하다면
경로11. S - i - L - a5 - L - j - T <그림 5-11> 참조



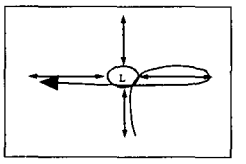
<그림 5-8>



<그림 5-9>



<그림 5-10>



<그림 5-11>

P턴, U턴을 감안한 경로 4가지를 추가한 <그림 5-1>에서 <그림 5-11> 모두 11가지의 대체 경로 중에서 가장 짧은 소요시간 즉, 최적소요시간을 구하여야 할 것이다. 이들 대체 경로 중 좌회전 금지 적용 시 최적경로 탐색에서 <그림 5-1>에서 <그림 5-7>까지의 경로(1~7)는 이미 비교 대상이 되었고 나머지 P턴을 감안한 경로 8과9, U턴을 감안한 경로 10,11를 최적경로를 구하기 위한 대안으로 사용할 수 있다. Floyd - Warshall 알고리즘에서 노드 L을 반복하여 경유하는 P턴과 U턴을 고려해 주기 위해, 즉 경로 8, 9, 10, 11를 고려해주기 위해 <그림 6>과 같이 가상 노드 L'을 생성한다. 즉, 가상노드 L'는 노드 L을 중심으로 U턴과 P턴 등 L을 반복하여 경유하는 경로를, 최적경로를 찾는데 고려해 주기 위하여 설정한 것이다. 가상 노드 L'은 좌회전 금지의 요소인 L과 연결되어 있는 노드와 노드사이의 소요시간들을 동일하게 적용한다.

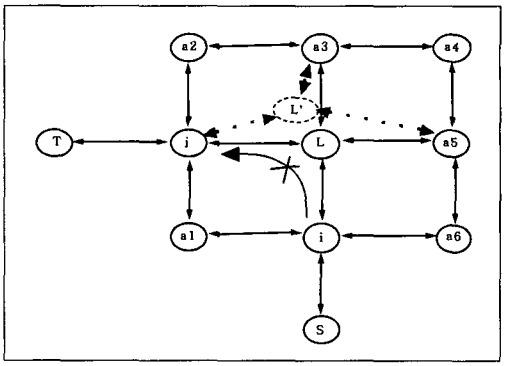


그림 6 가상노드 L'를 적용한 교통 네트워크

그러나, <그림 6>에서 보는 것과 같이 L'는 L과 연결되어 있는 모든 노드에서 좌회전 금지 시작노드인 i만 연결되지 않도록 한다. 만약, L'노드가 i노드와 연결이 된다면 i, L', j는 좌회전 금지요소로 지정하지 않았기 때문에 좌회전이 가능한 i → L' → j가 최적 경로가 될 가능성이 높다. 즉, 좌회전 금지요소 i → L → j와 어긋난다. 또한 L에서 노드 a3과 a5로 U턴이 된다면 노드 L의 분신인 노드 L'로 소요시간은 존재한다. 가상 노드 L'를 이용한 P턴과 U턴의 경로는 수정된 Floyd - Warshall 알고리즘에서 최적경로를 찾기 위해 다음과 같은 경로로 다른 경로와 비교된다.

P턴 시 경로: i → L → a3 → a4 → a5 → L' → j
i → L → a5 → a4 → a3 → L' → j

U턴 시 경로: i → L → a3 → L' → j
i → L → a5 → L' → j

즉, P턴과 U턴은 Floyd - Warshall알고리즘을 이용하여 가상노드 L'가 추가된 새로운 초기 $\pi^{(0)}$와 $P^{(0)}$을 사용하여 최적소요시간과 최적경로를 구할 수 있다.

5. 실험 및 분석

<그림 7>의 네트워크를 사용하여 본 논문의 4절에서 개발한 수정된 Floyd - Warshall알고리즘 최적경로 탐색 프로그램을 적용하여 결과를 분석하였다.

모두 14개의 노드로 구성되어 있고, 좌회전 금지구간은 2개(8 - 7 - 11과 12 - 9 -8), U턴(7-2)이 가능한 호는 한 개로 구성되어 있다. 위 실험에서 노드 12 - 9 - 8 로 이어지는 좌회전 금지를 P턴을 적용하여 최적경로를 찾아가는 것을 보여

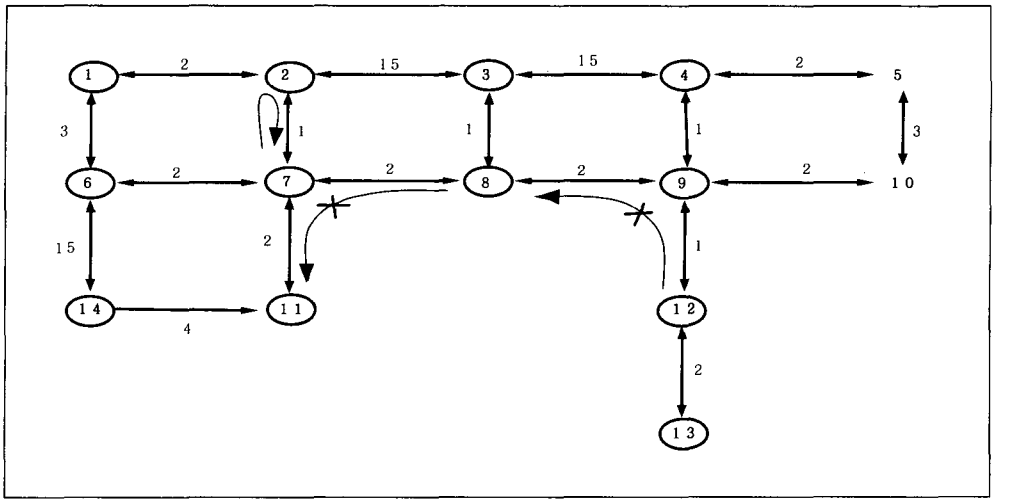


그림 7 최적경로알고리즘을 검증하기 위한 네트워크의 예

주고 있고, 노드 8 - 7 - 11에서 노드 7에서 노드 2로의 호가 U턴 가능을 적용하여 최적경로를 산출되어짐을 보여 주고 있다.

6. 결론

본 논문에서는 최적경로시스템의 실시간이동체 관리 모듈과 첨단정보제공 모듈의 구성요소와 작동원리에 대한 설명과 시스템의 가장 핵심적인 기능인 최적경로안내에 대하여 서술하였다. Floyd-Warshall알고리즘을 바탕으로 실제 교통 네트워크에 적용하기 위해 좌회전 금지, U턴 및 P턴을 고려하여 목적지까지의 최적경로를 산출해내는 수정된 Floyd - Warshall알고리즘을 개발하였다. 이 논문에서 제안한 알고리즘은 좌회전금지 구간 수가 소수 한정되어 포함되어 있는 교통네트워크에 사용되는 것이 바람직하고, 추후 좌회전금지 구간 수에 따른 계산량과 기억용량 증가를 대폭 축소할 수 있는 연구가 필요하다.

좌회전 금지와 U턴이 포함된 임의의 교통 네트워크에 적용한 바, 좌회전 금지 시 U턴과 P턴을 포함한 여러 가지 대안들의 비교 후 가장 최적의 경로를 산출 할 수 있었다. 이는 회전제약 및 교통혼잡에 유연하게 대처 할 수 있는 실질적인 최적경로의 산출임을 보이고 있다. 실시간 교통정보들을 데이터 베이스화 하고, 예측 기법을 적용, 교통네트워크를 모델링하여 이들 데이터를 제안된 최적경로 탐색 기법에 적용하여 실제 교통 네트워크에 적합한 상황에 효과적으로 사용할 수 있을 것이다.

참고 문헌

[1] 강맹규, "네트워크와 알고리듬", pp 93 ~ 105, 1995년

[2] 김성수, "지능형교통시스템을 위한 정보통신 매체와 동적경로 유도체계 확립에 관한 연구", IE Interfaces 제 11권 제 1호, pp 33-40, 1998년 3월

[3] 최기주, "U-TURN을 포함한 가로망 표현 및 최단경로의 구현", 대한교통학회지 제13권 제 3호, 1995년

[4] Ahuja, R. K, Magnamti, T. L., and Orlin, J. B., "Network Flows" , Prentice-Hall, 1993

[5] Hillier, F. S. and Lieberman, G. J, "Introduction to Operations Research" , 6th, 1995

[6] Kim, Sung-Soo, Lee, Jong-Hyun, and Kim, Hyung-Wook, "Development of Real Time Mobile Management System and Traveler Information System using Integrated Logistics Information Network", 5th World Congress on Intelligent Transport Systems, October 1998, Seoul, Korea

[7] Murty, Katta G., "Network Programming" , Prentice-Hall, 1992