

■ 論 文 ■

서비스시간 제약이 존재하는 도시부 복합교통망을 위한 링크기반의 최단경로탐색 알고리즘

A Link-Based Shortest Path Algorithm for the Urban Intermodal Transportation Network with Time-Schedule Constraints

장 인 성

(송실대학교 산업·정보시스템공학과 교수)

목 차

- | | |
|-----------------------------------|---------------|
| I. 서론 | III. 알고리즘의 개발 |
| II. 기존 알고리즘의 고찰 | IV. 적용예제 |
| 1. 단일교통망의 최단경로탐색 알고리즘 | V. 결론 |
| 2. 복합교통망의 최단경로탐색 알고리즘 | 참고문헌 |
| 3. 서비스시간 제약을 갖는 교통망의 최단 경로탐색 알고리즘 | |

Key Words : intermodal transportation network, link-based shortest path algorithm, time-schedule constraint, transfer time, multimodal path

요 약

본 연구에서 다루고자 하는 문제는 서비스시간 제약을 갖는 도시부 복합교통망에서의 기종점을 잇는 합리적인 최단경로를 탐색하고자 하는 것이다. 서비스시간 제약은 도시부 복합교통망에서의 현실성을 보다 더 사실적으로 표현하지만 기존의 알고리즘들은 이를 고려하지 않고 있다. 서비스시간 제약은 환승역에서 여행자가 환승차량을 이용해서 다른 지점으로 여행할 수 있는 출발시간이 미리 계획된 차량운행시간들에 의해 제한되어지는 것이다. 환승역에 도착한 여행자는 환승차량의 정해진 운행시간에서만 환승차량을 이용해서 다른 지점으로 여행할 수 있다. 따라서 서비스시간 제약이 고려되어지는 경우 총소요시간에는 여행시간과 환승대기시간이 포함되어지고, 환승대기시간은 여행자가 환승역에 도착한 시간과 환승차량의 출발이 허용되어지는 시간에 의존해서 변한다. 본 논문에서는 이러한 문제를 해결할 수 있는 링크기반의 최단경로탐색 알고리즘을 개발하였다. Dijkstra 알고리즘과 같은 전통적인 탐색법에서는 각 노드까지의 최단도착시간을 계산하여 각 노드에 표지로 설정하지만 제안된 알고리즘에서는 각 링크까지의 최단도착시간과 각 링크에서의 가장 빠른 출발시간을 계산하여 각 링크의 표지로 설정한다. 제안된 알고리즘의 자세한 탐색과정이 간단한 복합교통망에 대하여 예시되어진다.

1. 서론

최근에 세계적으로 활발히 진행되고 있는 지능형 교통체계(Intelligent Transportation System:ITS)의 연구와 더불어 현실성이 반영된 합리적인 경로를 효율적으로 산정할 수 있는 최단경로탐색 알고리즘의 개발에 대한 중요성이 강조되고 있다(Kaufman and Smith, 1993). 최단경로탐색 알고리즘은 ITS의 범주에 속하는 경로 안내체계(route guidance system)의 구현 및 첨단 여행자 정보체계(advanced traveler information system)의 운영을 위한 핵심기술이다. 지금까지는 최단경로탐색 알고리즘의 개발이 주로 하나의 교통수단으로 구성되는 단일교통망(unimodal transportation network)을 대상으로 이루어졌다. 예를 들면, 개인의 교통수단인 승용차를 이용하는 개인교통망(private transportation network)에서 목적지까지의 최단경로를 탐색하거나, 특정 상업용 차량(택시, 트럭 등)의 서비스를 고려해 목적지까지의 최단경로를 탐색하고자 한다. 즉, 출발지로부터 목적지까지 오직 하나의 교통수단(unimodality)이 이용되는 최단경로(unimodal path)를 탐색하고자 하는 것이다. 이에 대한 연구는 그동안 많은 연구자들에 의해 다양하고 심도있게 수행되어졌으며(Deo and Pang, 1984; Dreyfus, 1969; Romeijn and Smith, 1999; Sheffi, 1985; Thomas, 1991), 오늘날 ITS하에서 자가운전자에게 최적의 경로를 제공해주는 경로 안내 체계의 구현에 널리 활용되어지고 있다.

한편, 현대 도시의 교통망은 승용차, 대중교통(노선버스, 지하철), 직행버스, 고속버스, 기차, 항공기, 정기선박 등과 같은 다수의 교통수단(multimodalities)이 혼합된 복합교통망(intermodal transportation network)이며, 이에 대해 기존의 알고리즘을 적용하는데는 한계가 있다. 복합교통망에서는 교통수단간의 환승문제가 발생할 수 있기 때문에 교통망의 이용행태가 복잡하다. 즉, 승용차를 이용하다가 대중교통수단으로의 전환이 가능하기 때문에 환승에 소요되는 시간이 고려되어야 하지만 기존의 알고리즘은 경로 산정에 이를 반영하지 못한다. 복합교통망에서의 경로탐색은 출발지로부터 목적지까지 다수의 교통수단이 혼용 가능한 최단경로(multimodal path)를 산정하고자 하는 것이며, 이를 위한 알고리즘의 개발은 이용자에게 목적지까지의 제반 교통수단 및 다양한

노선정보를 제공함으로써 교통시설의 이용효율을 극대화하고 궁극적으로는 첨단 여행자 정보체계의 운영을 위해서도 매우 중요하다. 이에 대한 연구는 근래에 몇몇 연구자들에 의해 수행되어진 바 있으나 환승소요시간의 산정에 필요한 대기시간을 확률분포 등을 이용해서 확정적으로 추정하였으며 이를 기초로 최단 경로탐색 알고리즘을 개발하였다(김현명 외, 1999; 최기주, 장원재, 1998; Modesti and Sciomachen, 1998; Spiess and Florian, 1989). 즉, 이들 알고리즘들은 환승을 위한 차량(수단)의 어떤 승차지점에서 차량이 이용 가능하기까지의 대기시간(환승대기시간)은 이용자가 환승차량의 승차지점에 도착한 시간과 관계없이 항상 일정하다는 것을 가정으로 개발되어졌기 때문에 이들을 실제로 적용하는 경우에는 현실성이 결여된 경로를 얻게된다.

대중교통수단인 경우에는 일정한 차두간격을 유지하기 위해 미리 정해진 시간표에 따라 운행이 이루어지는 것(운행계획의 규칙성)이 일반적이며 특정 대중교통수단의 이용 가능한 서비스시간(출발시간)이 A 지점에서는 (10분, 20분, ...), B 지점에서는 (20분, 30분, ...)와 같이 각 지점에서 제한되어진다. 예를 들어서 (10분, 35분, 60분)을 A 지점에서 지하철이 이용 가능한 서비스시간이라 하자. 이용자가 타 교통수단으로부터 지하철로의 환승을 위해 A 지점에 9분에 도착하였다면 가장 빨리 이용 가능한 서비스시간은 10분이며 1분만큼의 대기시간이 소요된다. 반면에 이용자가 A 지점에 15분에 도착하였다면 가장 빨리 이용 가능한 서비스시간은 35분이며 대기시간은 20분이다. 즉, 이용자가 환승하려는 차량의 승차지점에 도착한 시간에 따라서 차량의 서비스를 받기까지의 대기시간이 달라진다. 그러므로 복합교통망 분석시 보다 정확한 자료로 제공되어지기 위해서는 복잡한 교통망의 이용행태 뿐만 아니라 서비스시간 제약(time-schedule constraints)에 의한 환승대기시간의 변동을 반영할 수 있는 최단경로탐색 알고리즘의 개발이 요구되어지지만 현재까지 보고된 연구사례는 없다.

그러므로 본 연구에서는 서비스시간 제약이 존재하는 복합교통망의 합리적인 최단경로를 탐색할 수 있는 알고리즘을 개발하고자 한다. 본 연구에서는 김현명 외(1999)가 제시한 링크기반의 최단경로탐색 알고리즘의 기본개념에 기초로해서 Chen and Tang(1998)

이 제시한 노드기반의 MTT 알고리즘을 복합교통망에 적용할 수 있도록 링크기반의 알고리즘으로 변형하였다. 링크기반의 최단경로탐색 알고리즘은 경로계산이 링크별로 단계적으로 이루어지기 때문에 계산과정에 링크 전환에 따른 환승을 반영할 수 있다. 따라서 복합교통망에서의 환승행태를 반영하기 위해 교통망을 확장하였던 기존의 연구와는 달리 교통망의 확장없이 환승행태가 반영된 합리적인 최단경로를 탐색할 수 있다. 반면에 MTT 알고리즘은 단일교통망에서 승용차가 특정노드를 통행할 수 있는 시간(서비스시간)이 사전에 제한되어지는 경우에 합리적인 경로를 탐색하기 위해 개발되어졌다. 본 연구에서는 이들 알고리즘들을 결합한 새로운 형태의 알고리즘을 개발함으로써 환승행태 뿐만 아니라 서비스시간 제약에 의한 환승대기시간의 변동이 반영된 합리적인 최단경로를 효율적으로 탐색할 수 있도록 하였다.

II. 기존 알고리즘의 고찰

1. 단일교통망의 최단경로탐색 알고리즘

대부분의 최단경로탐색 알고리즘들은 단일교통망을 대상으로 개발되어졌으며 순차적 탐색법을 채택하고 있다. 순차적 탐색법을 따르는 이들 알고리즘들은 경로계산을 위해 전노드만을 검색하는 수형망 알고리즘(tree building algorithm)과 전전노드까지를 검색 범위로 하는 덩굴망 알고리즘(vine building algorithm)의 두 가지 형태로 크게 분류할 수 있으며 Bellman의 최적원리에 의거해서 최적경로를 구축한다. 즉, 순차적 탐색법은 기점노드로부터 각 노드까지의 최단경로를 단계별로 계산하고 최종적으로 이들을 산술적으로 연결하면 기종점(origin-destination)간의 최단경로가 구축된다는 것(가정 1)과 동일한 노드가 최단경로 상에 두 번 이상 포함될 수 없다(가정 2)는 전제하에 최적경로를 탐색한다. 따라서, 상기의 가정 1이 위배되는 좌회전 금지가 발생하는 도로망이나 가정 2가 위배되는 P-turn 및 U-turn 허용 등과 같이 회전제한이 발생하는 도로망에서는 순차적 탐색법을 이용해서 합리적인 최단경로를 탐색할 수 없다. 비록 덩굴망 알고리즘이 교차로에서의 좌회전 금지를 부분적으로 반영할 수 있지만 좌회전 금지가 2개 이상 연속되는 경우에는 이를 경로 구축시 정확하게 반영하

지 못한다(김익기, 1998; 최기주, 1995). 결국, 순차적 탐색법을 따르는 기존의 알고리즘들은 Bellman의 최적원리가 적용되는 제한된 문제만을 해결할 수 있을 뿐이며 회전제한과 같이 현실성이 반영된 문제를 해결할 수가 없다.

대도시의 복잡한 도로망에서는 교차로에서의 좌회전 금지나 이로 인한 P-turn 및 U-turn이 허용되는 회전제한이 자주 발생한다. 따라서, 도시 도로망에 대한 경로 안내체계의 구현을 위해서는 교차로에서 발생하는 좌회전금지, P-turn 및 U-turn 허용 등과 같은 회전제한이 반영된 합리적인 최단경로를 신속하게 탐색할 수 있어야 한다. Caldwell(1961), Sheffi(1985) 등은 도로망을 확장 또는 변형하여 도로망에 회전제한을 표현함으로써 기존의 순차적 탐색법을 적용할 수 있도록 하였다. 그러나 이와 같은 방법은 경로탐색을 위한 노드 및 링크 수가 증가하여 경로산정에 많은 시간이 소비된다는 단점을 갖고있다. 보다 신속하고 효율적으로 최단경로를 산정하기 위해서는 도로망의 확장없이 경로 계산과정에서 회전제한을 반영할 수 있는 알고리즘이어야 하며, 이와 같은 목적으로 링크탐색 알고리즘(노정현, 남궁성, 1995; 이승환 외, 1996)과 수정형 덩굴망 알고리즘(김익기, 1998) 등이 개발되어졌다.

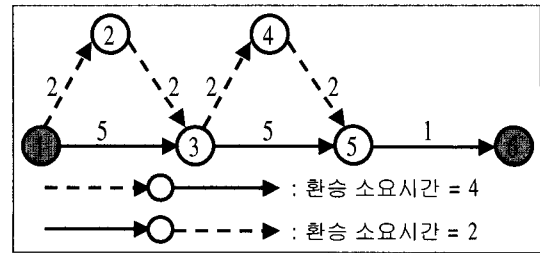
2. 복합교통망의 최단경로탐색 알고리즘

현대 도시의 교통망은 다수의 교통수단이 혼합된 복합교통망이다. 특히, 다양한 대중교통시설의 확충과 서비스의 향상으로 대중교통 이용자가 증가하면서 복합교통망을 대상으로 합리적인 최적경로를 신속하게 탐색할 수 있는 알고리즘의 개발에 대한 필요성이 대두되고 있다. 복합교통망에서는 출발지로부터 목적지까지 다수의 교통수단이 혼용 가능하기 때문에 단일교통망과는 다른 여러 특성을 가지고 있으며 이에 대해서는 최기주와 장원재(1998)가 구체적으로 설명하고 있다.

순차적 탐색법의 가정 1은 어떤 두 노드를 연결하는 링크에 선 진입한 차량이 링크에서 먼저 빠져 나오는 FIFO(First-In, First-Out)조건이 만족된다는 것을 의미하지만 FIFO조건은 좌회전금지가 발생하는 도로망이나 환승이 발생하는 복합교통망에서는 성립되지 않는다. 좌회전금지와 마찬가지로 덩굴망

알고리즘은 부분적인 환승행태를 반영하지만 환승노드가 2개 이상 연속되는 경우에는 수단의 전환을 경로의 구축과정에 반영하지 못한다. 따라서, 순차적 탐색법은 환승이 발생하는 복합교통망에는 적용할 수 없다. 복합교통망에서는 이용자가 환승을 위해 특정 지점에 도착하는 즉시 교통수단을 이용할 수 있는 것이 아니다. 승용차나 렌트카를 이용해서 출발지로부터 목적지까지 운행하는 경우에는 중간에 어떤 지점에 도착하더라도 목적지를 향해서 지체없이 운행차량을 이용해 그 지점을 벗어날 수 있지만 도중에 다른 교통수단으로 전환하는 경우에는 환승소요시간이 발생한다. 따라서 복합교통망의 환승행태가 고려된 최단경로를 산정하기 위해서는 경로계산과정에 환승소요시간을 반영할 수 있어야만 한다.

회전제한과 마찬가지로 수단간의 환승행태를 교통망에 표현하기 위해 Spiess and Florian(1989)과 Modesti and Sciomachen(1998)은 주어진 복합교통망에 환승노드나 환승링크를 추가하여 교통망을 확장하였으며, 최기주와 장원재(1998)는 수단별 교통망을 독립적으로 계층화하여 다층 네트워크를 구성하고 환승이 이루어지는 수단별 교통망의 두 노드에 환승링크를 추가함으로써 계층화된 다층 네트워크를 하나의 교통망으로 통합하였다. 이와 같이 교통망을 확장하거나 통합하는 방법은 기존의 순차적 탐색법을 그대로 적용할 수 있다는 장점을 지니지만 교통망의 확장이나 통합에 많은 시간이 소비되는 단점을 가지고 있기 때문에 교통망의 규모가 크고 교통수단이 다수인 경우에는 적용하기가 어렵다. 최근에, 김현명과 임용택(1999)은 복합교통망에서의 최단경로를 탐색하기 위해 유전자 알고리즘을 적용하였다. 이 방법은 교통망의 확장없이 합리적인 최단경로를 탐색할 수 있지만 많은 계산수행시간이 요구되어지기 때문에 비효율적이며 이를 개선하기 위해 김현명 외(1999)는 링크기반의 최단경로탐색 알고리즘을 제시하였다. 링크기반의 최단경로탐색 알고리즘은 현노드의 전노드나 전전노드를 검색해서 기점노드로부터 현노드까지의 최단경로를 구축하는 노드기반의 순차적 탐색법과는 달리 현링크의 전링크를 검색함으로써 기점노드부터 현링크의 종결노드까지의 최단경로를 구축한다. 링크기반의 최단경로탐색 알고리즘에서는 현링크의 전링크를 검색하기 때문에 링크 전환에 따른 환승을 고려할 수 있으며, 교통망의 확장이나 통합등의 추가



〈그림 1〉 환승노드가 2개 이상 연속되는 복합교통망

적인 작업없이 환승행태가 반영된 합리적인 최단경로를 산정할 수 있다.

〈그림 1〉은 환승노드가 2개이상 연속되는 복합교통망에 대하여 노드기반의 순차적탐색법과 링크기반의 최단경로탐색 알고리즘을 비교하기 위한 예이다. 노드 1과 노드 6은 각각 기점과 종점을 나타낸다. 점선링크는 버스를 실선링크는 지하철을 나타내고 있으며 링크 위에 있는 숫자는 링크를 통과하는데 소요되는 시간(분)을 나타낸다. 수단간 환승시간은 버스에서 지하철로의 환승시간이 4, 지하철에서 버스로의 환승시간은 2라고 가정하자. 수형망 알고리즘의 경우 전노드까지만 검색하므로 환승을 고려해 위의 문제를 풀 수 없다. 순차적 탐색법은 경로계산이 노드별로 단계적으로 이루어지며 기점노드 1로부터 노드 3과 4까지의 최단경로는 버스를 이용하는 경로로써 소요시간은 각각 4분과 6분이다. 이때 전전노드까지를 검색범위로 하는 덩굴망 알고리즘은 노드 3의 전 노드와 전전 노드를 노드 2와 1, 노드 4의 전 노드와 전전 노드를 노드 3과 2로 기록한다.

현재 최단경로를 계산해야할 노드가 5라할 때 덩굴망 알고리즘은 전전노드 3, 전노드 4를 경유하는 경로와 전전노드 1, 전노드 3을 경유하는 경로 그리고 전전노드 2, 전노드 3을 경유하는 경로를 비교한다. 전전노드가 3, 전노드가 4인 경우 기점노드 1로부터 노드 3까지의 최단소요시간 4분, 노드 3에는 전노드로써 노드 2가 기록되어있으므로 링크 (2,3)에서 링크 (3,4)로 전환하기 위한 환승시간 0분, 링크 (3,4)의 소요시간 2분, 링크 (3,4)에서 링크 (4,5)로 전환하기 위한 환승시간 0분, 링크 (4,5)의 소요시간 2분으로 노드 3과 노드 4를 경유해서 노드 5에 도달하는 최단소요시간은 8분이다. 마찬가지로 노드 1과 노드 3을 경유해서 노드 5에 도달하는 최단소요시간은 10분, 노드 2와 노드 3을 경유해서 노드 5에 도달하는 최단소요시간은 13분이다. 따라서 덩굴망

알고리즘은 노드 5의 전노드와 전전노드로써 노드 4와 3을 기록하고 노드 5까지의 최단경로는 전전노드 3까지의 최단경로 ①-②-③과 노드 4, 5를 산술적으로 연결한 경로인 ①-②-③-④-⑤이며 소요시간은 8분이다.

최종적으로 기점노드 1에서 종점노드 6까지의 최단경로는 전전노드 4, 전노드 5를 경유하는 경로와 전전노드 3, 전노드 5를 경유하는 경로를 비교함으로써 계산되어진다. 전전노드가 4인 경우 기점노드 1에서 노드 4까지의 최단소요시간 6분, 노드 4에는 전노드로써 노드 3이 기록되어있으므로 노드 4에서의 환승시간 0분, 링크 (4,5)의 소요시간 2분, 노드 5에서의 환승시간 4분, 링크 (5,6)의 소요시간 1분으로 종점노드 6에 도착하기 위한 최단소요시간은 13분이다. 전전노드가 3인 경우에는 기점노드 1에서 노드 3까지의 최단소요시간 4분, 노드 3의 전노드가 2로 기록되어있으므로 노드 3에서의 환승시간 4분, 링크 (3,5)의 소요시간 5분, 노드 5에서의 환승시간 0분, 링크 (5,6)의 소요시간 1분이므로 노드 6까지의 최단소요시간은 14분이 된다. 따라서, 노드 4와 노드 5가 노드 6의 전전노드와 전노드로 기록되어지므로 노드 1에서 노드 6까지의 최단소요시간은 13분, 해당 경로는 전전노드 4까지의 최단경로 ①-②-③-④와 노드 5, 6을 산술적으로 연결한 ①-②-③-④-⑤-⑥이다. 그러나 <그림 1>의 실제적인 최단경로는 ①-③-⑤-⑥이다.

링크기반의 최단경로탐색 알고리즘은 경로계산이 링크별로 단계적으로 이루어지며 기점노드 1로부터 링크 (3,4)와 링크 (4,5)의 통행이 완료되는 시점까지의 최단경로는 버스를 이용하는 경로로써 소요시간은 각각 6분과 8분이다. 이때 링크기반의 최단경로탐색 알고리즘은 링크 (3,4)의 전링크를 링크 (2,3), 링크 (4,5)의 전링크를 링크 (3,4)로 기록한다. 링크 (3,5)의 통행이 완료되는 시점까지의 최단경로를 계산하기 위해 링크 (1,3)과 링크 (2,3)이 전링크로써 검색되어진다. 전링크가 (2,3)인 경우 링크 (2,3)의 통행이 완료되는 노드 3까지의 최단소요시간 4분, 링크 (2,3)에서 링크 (3,5)로 전환하기 위한 환승시간 4분, 링크 (3,5)의 소요시간 5분으로 총 소요시간은 13분이다. 반면에 전링크가 (1,3)인 경우는 총 소용

시간이 10분이므로 링크 (3,5)의 전링크로써 링크 (1,3)이 기록된다. 최종적으로 노드 1에서 링크 (5,6)의 도착노드인 노드 6까지의 최단경로를 계산하기 위해 링크 (4,5)와 링크 (3,5)가 전링크로써 검색되어진다. 전링크가 (4,5)인 경우 링크 (4,5)의 도착노드인 노드 5까지의 최단소요시간 8분, 링크 (4,5)에서 링크 (5,6)으로의 환승시간 4분, 링크 (5,6)의 소요시간 1분으로 총 소요시간은 13분이다. 반면에 전링크가 (3,5)인 경우에는 환승이 발생하지 않으며 링크 (3,5)의 도착노드인 노드 5까지의 최단소요시간 10분, 링크 (5,6)의 소요시간 1분으로 총 소요시간이 11분이다. 따라서, 링크 (3,5)가 링크 (5,6)의 전링크로 기록되어지며 노드 1에서 노드 6까지의 최단경로는 전링크 (3,5)의 도착노드인 노드 5까지의 최단경로 ①-③-⑤와 링크 (5,6)을 산술적으로 연결한 ①-③-⑤-⑥이다.

이상과 같이 좌회전 금지문제와 동일하게 덩굴망 알고리즘은 환승노드가 2개 이상 연속되는 복합교통망에서도 최단경로를 탐색하는데 실패하므로 노드기반의 순차적탐색법을 적용하기 위해서는 교통망을 확장하거나 통합함으로써 수단간의 환승행태를 교통망에 표현해야 함을 알 수 있다. 이에반해서 링크기반의 최단경로탐색 알고리즘은 교통망의 확장이나 통합 등의 추가적인 작업이 필요없으나 순차적 탐색법이 모든 노드에 대해서 단계별로 경로계산을 수행하는데 반해 모든 링크에 대해서 단계별로 경로계산이 수행되어지므로 탐색속도가 느리다는 단점을 갖고 있음을 알 수 있다.¹⁾

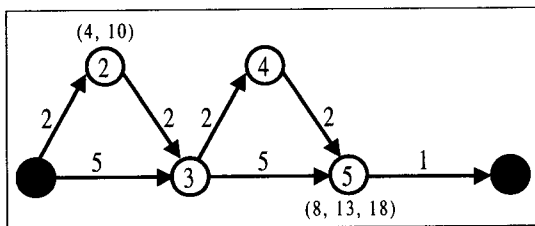
3. 서비스시간 제약을 갖는 교통망의 최단경로 탐색 알고리즘

환승소요시간은 탑승차량의 하차지점에서 환승하려는 차량의 승차지점까지의 도보소요시간에 환승을 위한 승차지점에서 환승차량이 이용 가능하기까지의 대기시간(환승대기시간)을 더한 것이다. 환승을 위한 도보소요시간은 일반인의 평균값을 이용할 수 있지만 환승대기시간을 정확하게 추정하는 것은 쉽지않다. 항공기, 기차, 정기선박, 지하철, 버스 등 승용차를 제외한 대부분의 수송용 차량은 미리정해진 시간표에

1) 하나의 노드에는 수많은 링크가 연결되어질 수 있기 때문에 교통망을 구성하는 노드의 수보다 링크의 수가 더 많다.

따라 운행이 규칙적으로 이루어지는 것이 일반적이기 때문에 이용자가 승차지점에 도착하는 즉시 교통수단을 이용할 수 있는 것이 아니다. 이와같이 서비스시간 제약에 의해 환승대기시간은 이용자가 환승차량의 승차지점에 도착한 시간에 의존해서 변한다. 그러나 고정된 환승대기시간을 이용해서 개발되어진 기존의 알고리즘들은 서비스시간 제약에 의한 환승대기시간의 변동을 반영하지 못하기 때문에 현실성이 결여되어있다.

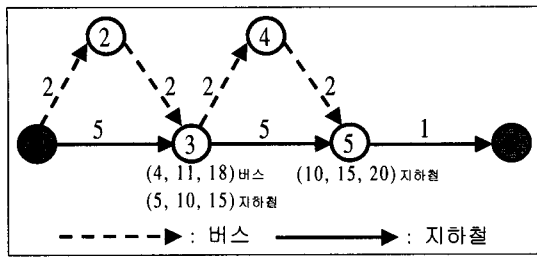
서비스시간 제약(time-schedule constraints)을 갖는 최단경로문제는 Chen and Tang(1997, 1998)에 의해서 소개되어졌다. Chen and Tang은 <그림 2>와 같이 단일교통망에서 승용차가 특정 노드를 통행할 수 있는 시간을 (8, 13, 18)로 사전에 제한함으로써 노드에 도착하는 즉시 출발할 수 없도록 하였다. <그림 2>의 교통망에서 기점노드 1로부터 노드 5까지의 경로가 ①-③-⑤인 경우 노드 5에 도착한 시간은 10분이지만 즉시 종점노드 6으로 이동할 수 없다. 노드 5에서 종점노드 6으로 이동 가능한 가장 빠른 출발시간은 13분으로 3분만큼의 대기시간이 소요된다. 서비스시간 제약은 어떤 노드에서의 통행이 특정 시간대에서만 가능함을 나타내기 위해 사용되는 시간창(time window)(Desrochers and Soumis, 1988)을 일반화 한 것으로써 복합교통망에 적용하기에 보다 실제적이다. 서비스시간 제약을 갖는 교통망에서는 링크가 임의의 두 노드에 연결되어져서 차량 통행이 가능할지라도 서비스시간 제약에 의해 실제로 통행이 불가능하게 되는 링크가 발생한다. <그림 2>에서 링크 (3,5)의 통행시간을 15분으로 수정하고 부분경로 ①-③-⑤가 이용되어진다고 가정하자. 노드 5에 도착한 시간은 20분이며 노드 5에서 노드 6으로 이동할 수 있는 가장 늦은 출발시간 18분을 초과하기 때문에 링크 (5,6)은 통행 불가능하다. 따라서 경로 ①-③-⑤-⑥은 통행 불가능한 경로이다.



<그림 2> 서비스시간 제약을 갖는 단일교통망

Chen and Tang(1998)은 서비스시간 제약이 반영된 최단경로를 탐색하기 위해 수행망 알고리즘의 대표적인 예인 Dijkstra(1959) 알고리즘을 확장한 MTT(minimum-total-time) 알고리즘을 개발하였다. Dijkstra 알고리즘은 기점으로 부터 각 노드까지의 최단도착시간(earliest reaching time)을 계산하여 각 노드에 표지로 설정하지만, MTT 알고리즘에서는 기점으로 부터 각 노드까지의 최단도착시간 뿐만 아니라 각 노드에서의 가장 빠른 출발가능시간(earliest leaving time)도 계산하여 각 노드에 표지로 설정한다. 따라서 MTT 알고리즘은 각 노드의 가장 빠른 출발 가능시간을 고려해서 경로계산을 수행하기 때문에 합리적인 최단경로를 탐색할 수 있다. 그러나, MTT 알고리즘은 단일교통망을 대상으로 개발되어졌기 때문에 복합교통망에서의 환승문제를 해결할 수 없다. 단일교통망에서의 출발제약은 공사중인 도로나 통행이 제한되는 터널, 교량 등과 같이 한시적으로 사용가능한 링크의 출발노드에서 발생할 수 있다. 그러나 복합교통망에서는 환승노드 뿐만 아니라 동일수단내의 승객의 하차와 승차에 따른 지연에 의해 모든 노드에서 출발제약이 발생한다. 특히, 환승노드에서는 출발시간이 이용가능한 수단별로 제한되어지기 때문에 최단경로를 계산하기 위해서는 환승노드에서 이용가능한 수단들의 가장 빠른 출발가능시간을 모두 비교해야만한다. 따라서, 서비스시간 제약을 갖는 복합교통망에서 합리적인 최단경로를 탐색하기 위해서는 환승소요시간 뿐만 아니라 환승노드에서의 수단별 출발시간을 고려해야하므로 매우 복잡하다.

본 연구에서는 노드기반의 MTT 알고리즘을 복합교통망에 적용할 수 있도록 링크기반의 알고리즘으로 변형함으로써 서비스시간 제약에 의한 환승대기시간의 변동이 반영된 합리적인 최단경로를 탐색할 수 있는 알고리즘을 개발하였다. 본 알고리즘은 현링크의 전링크를 검색함으로써 환승이 고려되어지며 환승노드에서의 수단별 가장 빠른 출발시간을 고려해서 경로계산을 수행하도록 개발되어졌기 때문에 서비스시간 제약을 해결할 수 있다. 본 연구에서는 동일한 수단을 이용해서 특정 노드를 경유하는 경우에는 노드에 도착하는 즉시 출발이 이루어진다고 가정한다(도착시간=출발시간). 즉, 복합교통망에서의 출발제약은 <그림 3>과 같이 환승노드에서 수단을 전환하는 경우에만 고려되어지는 것으로 가정한다.



(그림 3) 서비스시간 제약을 갖는 복합교통망

III. 알고리즘의 개발

Chen and Tang(1998)이 제시한 노드기반의 MTT 알고리즘은 단일교통망상에서 자가운전자가 승용차를 이용해 특정 노드를 출발할 수 있는 시간이 사전에 제한되는 경로를 탐색한다. MTT 알고리즘은 경로탐색과정에서 기점으로부터 각 노드까지의 최단도착시간 뿐만아니라 각 노드에서의 가장 빠른 출발가능시간도 계산하여 각 노드에 표지로 설정한다. MTT 알고리즘은 기본적으로 노드기반의 순차적 탐색법인 수형망 알고리즘이기 때문에 경로계산과정에서 환승이 반영되지 않으며 단일교통망이 대상이므로 각 노드에서의 가장 빠른 출발가능시간을 계산하기 위해 단일교통수단에 대한 서비스시간 제한만을 고려한다. 그러나 복합교통망에서는 환승이 발생하기 때문에 경로계산과정에 환승이 반영되어야 하며 환승노드에서의 가장 빠른 출발가능시간을 계산하기 위해서는 수단별로 제한되어지는 서비스시간을 모두 고려해야한다. 따라서 MTT 알고리즘을 이용해서는 복합교통망에서 발생하는 환승뿐만아니라 환승노드에서의 수단별 서비스시간 제약이 반영된 합리적인 경로를 탐색할 수 없다.

본 연구에서는 노드기반의 MTT 알고리즘을 링크기반의 알고리즘으로 변형함으로써 경로계산과정에서 환승이 반영되어질 뿐만아니라 환승노드에서 가장 빠른 시간을 계산하기 위해 수단별 서비스시간이 모두 비교될수 있도록 하였다. 제안되어지는 알고리즘은 다음과 같은 변수들을 사용해서 개발되어졌다.

- T : 탐색에 사용될 링크들의 집합
- S : 탐색된 링크들의 집합
- o : 교통망의 기점노드(origin node)
- d : 교통망의 종점노드(destination node)

- $p(d)$: 기점노드로부터 종점노드까지의 최단거리
- $tail(i)$: 링크 i 의 출발 노드
- $head(i)$: 링크 i 의 종결 노드
- D : $head(i)=d$ 인 링크 i 들의 집합
- $start(i)$: 링크 i 를 통행할 수 있는 가장 빠른 출발 시간
- $end(i)$: 링크 i 의 통행이 완료되는 시간
- $t(i)$: 링크 i 의 차량운행 소요시간
- V_1 : 출발노드가 환승노드인 링크들의 집합
- V_2 : 출발노드가 환승노드가 아닌 링크들의 집합
- $transfer(i, v)$: 링크 i 를 운행하는 차량의 하차지점에서 링크 i 의 후속링크 v 를 운행하는 차량의 승차지점까지의 도보 소요시간
- $TS(i) = (ts_1(i), ts_2(i), \dots, ts_m(i))$: 링크 i 의 차량 운행 시간표
- $reverse(i)$: 링크 i 의 전링크

알고리즘을 구성하는 구체적인 단계들은 다음과 같다.

Step 1 : 초기화

- $T =$ 교통망을 구성하는 모든 링크, $S = \emptyset$
- ① 만일 $tail(i) = o$ 이면 $start(i) = 0, end(i) = t(i)$
- ② 만일 $tail(i) \neq o$ 이면 $start(i) = \infty, end(i) = \infty$

Step 2 : $head(i) = d$ 인 링크 i 의 탐색

- ① 만일 $head(i) = d$ 인 링크 i 가 T 에 존재하지 않는다면 Step 5로 간다.
- ② 만일 $head(i) = d$ 인 링크 i 가 T 에 존재한다면 Step 3으로 간다.

Step 3 : T 와 S 의 수정

집합 T 에서 가장 작은 $start(i)$ 값을 갖는 링크 i^* 를 선택하고 집합 T 와 S 를 다음과 같이 수정한다.

$$T = T - \{i^*\}, S = S \cup \{i^*\}$$

이때 i^* 가 두 개 이상이면 그 중 하나를 임의로 선택한다.

- ① 만일 $head(i^*) = d$ 이면 Step 2로 간다.
- ② 만일 $head(i^*) \neq d$ 이고
 - ① 만일 $start(i^*) = \infty$ 이면 주어진 교통망에는 통행 가능한 경로가 존재하지 않는다.

㉞ 만일 $start(i^*) \neq \infty$ 이면 Step 4로 간다.

Step 4 : 통행완료시간 계산

(1) 링크 i^* 의 종결노드를 출발노드로 하는 링크 v 를 모두 탐색하고 각각의 v 에 대해 출발시간을 새로이 계산한다.

① $v \in V_1$ 이고

㉠ $end(i^*) + transfer(i^*, v) \leq ts_1(v)$ 이면 $start(v) = ts_1(v)$

㉡ $ts_j(v) < end(i^*) + transfer(i^*, v) \leq ts_{j+1}(v)$ 이면 $start(v) = ts_{j+1}(v)$

㉢ $end(i^*) + transfer(i^*, v) > ts_m(v)$ 이면 $start(v) = \infty$

② $v \in V_2$ 이면 $start(v) = end(i^*)$

(2) (1)의 $start(v)$ 를 이용해서 링크 v 의 통행완료시간을 다음과 같이 다시 계산한다.

$$end(v) = \min \{ end(v), start(v) + t(v) \}$$

(3) 만일 (2)에서 $end(v)$ 의 값이 갱신되었다면 $start(v)$ 와 $end(v)$ 를 새로이 계산된 값으로 수정하고, $reverse(v) = i^*$ 로 저장한다.

(4) Step 2로간다.

Step 5 : 종료

(1) 기점노드로부터 종점노드까지의 최단거리 $p(d)$ 를 다음과 같이 계산한다.

$$p(d) = end(i^*) = \min_{i \in D} end(i)$$

(2) 최단경로를 구성하는 링크들을 다음과 같이 역추적한다.

$$\{ reverse(i^*), reverse(reverse(i^*)), \dots, j \},$$

여기서 $tail(j) = o$.

(3) 역추적된 (2)의 링크들을 이용해서 최단경로를 구성하는 노드들을 다음과 같이 추적한다.

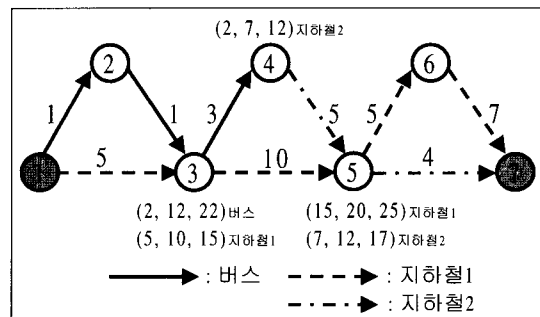
$$\{ o, head(j), \dots, head(reverse(reverse(i^*))), head(reverse(i^*), d) \}$$

Step 1은 각 링크를 운행하는 차량들의 출발시간과 통행 완료시간을 초기화하는 단계로써 기점노드 o 을 출발노드로 하는 링크들의 가장 빠른 출발시간을 0, 통행 완료시간을 차량운행 소요시간으로, 그 외의 링크들에 대해서는 가장 빠른 출발시간 및 통행 완료시간을 매우 큰 값(∞)으로 설정하고있다. 최단경로를

탐색하기 위한 알고리즘의 수행과정에서 순차적으로 링크들의 가장 빠른 출발시간과 통행 완료시간이 점차 갱신되어진다. Step 3의 ㉠㉡은 아직 탐색되지 않은 링크들의 가장 빠른 출발시간이 매우 큰 값(∞)에서 더 이상 갱신되어지지 않으면 주어진 복합교통망에는 기중점을 잇는 경로가 존재하지 않음을 의미한다. Step 4의 (1)㉠㉢은 링크 v 를 운행하는 차량으로 환승하기 위해 승차지점에 도착한 시간이 링크 v 의 운행이 가능한 가장 늦은 출발시간 $ts_m(v)$ 을 초과하는 경우를 나타내고 있다. 이 경우에 실제로 링크 v 의 통행이 불가능하기 때문에 위 알고리즘에서는 링크 v 의 출발시간을 매우 큰 값(∞)으로 설정함으로써 링크 v 가 최단경로에 포함되는 것을 방지하고 있다. Step 5는 알고리즘의 종료단계로써 기중점간의 최단거리 $p(d)$ 와 대응되는 최단경로를 계산한다.

IV. 적용예제

<그림 4>에 예시된 간단한 복합교통망을 이용해서 제안된 알고리즘의 동작과정을 살펴보고자 한다. 예시된 복합교통망은 7개의 노드와 9개의 링크, 3개의 교통수단으로 구성되어있다. 노드 1과 노드 7은 기점(o)과 종점(d)을 나타내며 링크 위에 있는 숫자는 링크를 통과하는데 소요되는 시간(분)을 나타낸다. 수단전환을 위한 도보소요시간은 <표 1>과 같으며 환승노드 3, 4, 5에서 서비스시간이 제한되어진다. MTT 알고리즘은 환승노드 3, 4, 5에서 환승이 이루어지는 경우 이를 경로계산과정에 반영하지 못한다. 또한 노드 3과 5에서는 서비스시간이 2개의 수단에 대하여 제약되어져 있으며 이러한 경우 MTT 알고리즘은 노드 3과 5에서의 가장 빠른 출발가능시간을 계산하지



<그림 4> 예제를 위한 복합교통망

9개의 모든 링크가 포함되어지도록 설정하고($S=\emptyset$), 링크 (1,2)와 링크 (1,3)을 제외한 나머지 링크들에 대해 $start(i)$ 와 $end(i)$ 를 ∞ 로 초기화 한다(반복 0). Step 2의 ②가 만족되어지므로 Step 3이 수행되어진다. Step 3에서 링크 (1,2)가 링크 i^* 로 선택되어지며, 집합 T 와 S 가 수정된다. $start(i^*=(1,2))=0 \neq \infty$ 이므로 Step 4가 수행되어지며, Step 4에서 링크 $i^*=(1,2)$ 의 후속링크 $v=(2,3)$ 의 $start(v)$ 와 $end(v)$ 의 값이 각각 1분과 2분으로 새로이 계산되어진다. $end(v)$ 의 값이 ∞ 에서 2분으로 갱신되어졌으므로 링크 $v=(2,3)$ 의 $start(v)$ 와 $end(v)$ 의 값이 수정되어지고, $reverse(v=(2,3))=(1,2)$ 로 저장된다(이상 반복1). 최종적으로 알고리즘은 반복 9에서 $T=\emptyset$ 이며 Step 2의 ①이 만족되어지기 때문에 알고리즘의 종료단계인 Step 5가 다음과 같이 수행되어지고, 최단경로로써 경로 ①-②-③-④-⑤-⑦이 추적되어진다.

Step 5 :

- (1) $D=\{(5,7), (6,7)\}$
 $i=(5,7) \rightarrow end(i)=21$
 $i=(6,7) \rightarrow end(i)=27$
 $p(d) = end(i^*=(5,7)) = \min_{i \in D} end(i) = 21$
- (2) $i^*=(5,7)$, $reverse(i^*)=(4,5)$
 $i^*=(4,5)$, $reverse(i^*)=(3,4)$
 $i^*=(3,4)$, $reverse(i^*)=(2,3)$
 $i^*=(2,3)$, $reverse(i^*)=(1,2)$
- (3) ①-②-③-④-⑤-⑦

V. 결론

복합교통망 분석시 보다 정확한 자료로 제공되어지기 위해서는 복잡한 교통망의 이용행태 뿐만 아니라 서비스시간 제약에 의한 환승대기시간의 변동을 반영할 수 있는 최단경로탐색 알고리즘의 개발이 요구된다. 복합교통망에서의 환승행태가 반영된 최단경로를 탐색하기 위해 수행되어졌던 기존의 연구들은 서비스시간 제약을 고려하지 않고 환승대기시간은 이용자가 환승차량의 승차지점에 도착한 시간과 관계없이 항상 일정하다는 것을 가정하기 때문에 현실성이 결여되어 있다. 비록 MTT 알고리즘이 특정 노드의 서비스시간 제약을 경로계산과정에 반영하지만 기본적으로 노드기반의 순차적 탐색법인 수형망 알고리즘이기 때문

에 경로계산과정에서 환승이 반영되지 않으며 단일교통망을 대상으로 하기 때문에 환승노드에서의 수단별 서비스시간 제약이 고려되지 않는다.

본 연구에서 노드기반의 MTT 알고리즘을 링크기반의 알고리즘으로 변형함으로써 복합교통망에서의 환승행태 뿐만아니라 서비스시간 제약에 의한 환승대기시간의 변동이 반영된 합리적인 최단경로를 효율적으로 탐색할 수 있는 알고리즘이 개발되었다.

본 알고리즘은 복합교통망에 대한 현실성을 잘 반영하고있기 때문에 현대의 대도시와 같이 복잡하고 대규모인 복합교통망에서 실제로 유용하게 활용될 수 있는 기법이라고 고려된다.

참고문헌

1. 김익기(1998), "ATIS를 위한 수정형 덩굴망 최단경로 탐색 알고리즘의 개발", 대한교통학회지, 제16권 제2호, pp.157~167.
2. 김현명·임용택(1999), "유전알고리즘을 이용한 전역탐색 최단경로 알고리즘개발", 대한교통학회지, 제17권 제2호, pp.163~178.
3. 김현명·임용택·이승재(1999), "통합교통망 수단선택-통행배정모형 개발에 관한 연구", 대한교통학회지, 제17권 제5호, pp.87~98.
4. 노정현·남궁성(1995), "도시가로망에 적합한 최단경로탐색 기법의 개발", 대한국토 도시계획학회지, 제30권 제5호, pp.153~168.
5. 이승환·최기주·김원길(1996), "도시부 ATIS 효율적 적용을 위한 탐색영역기법 및 양방향 링크탐색 알고리즘의 구현", 대한교통학회지, 제14권 제3호, pp.45~59.
6. 최기주(1995), "U-TURN을 포함한 가로망의 표현 및 최단경로의 구현", 대한교통학회지, 제13권 제3호, pp.35~31.
7. 최기주·장원재(1998), "복합 교통망에서의 최적 경로산정 모형개발", 대한교통학회지, 제16권 제4호, pp.167~186.
8. Caldwell, T.(1961), "On Finding Minimum Routes in a Network with Turn Penalties", Communications of the ACM, Vol. 4, pp.107~108.
9. Chen, Y. L. and Tang, K.(1997), "Shortest

- Paths in Time-Schedule Networks", International Journal of Operations and Quantitative Management, Vol. 3, pp.157~173.
10. Chen. Y. L. and Tang. K.(1998), "Minimum Time Paths in a Network with Mixed Time Constraints", Computers and Operations Research, Vol. 25, pp.793~805.
 11. Deo, N. and Pang, C.(1984), "Shortest Path Algorithms: Taxonomy and Annotation", Networks, Vol.14, pp.275~323.
 12. Desrochers, M. and Soumis, F.(1988), "A Reoptimization Algorithm for the Shortest Path Problem with Time Windows", European Journal of Operational Research, Vol. 35, pp.242~254.
 13. Dijkstra, E. W.(1959), "A Note on Two Problems in Connection with Graphs", Numerische Mathematik, Vol. 1, pp.269~271.
 14. Dreyfus. S. E.(1969), "An Appraisal of Some Shortest-Path Algorithms", Operations Research, Vol. 17, pp.395~412.
 15. Kaufman D. E. and Smith. R. L.(1993), "Faster Paths in Time-Dependent Networks for IVHS Applications", IVHS Journal, Vol. 1, pp.1~11.
 16. Modesti, P. and Sciomachen, A.(1998), "A Utility Measure for Finding Multiobjective Shortest Paths in Urban Multimodal Transportation Networks", European Journal of Operational Research, Vol. 111, pp.495~508.
 17. Romeijn, H. E. and Smith, R. L(1999), "Parallel Algorithms for Solving Aggregated Shortest-Path Problems", Computers and Operations Research, Vol. 26, pp.941~953.
 18. Sheffi, Y.(1985), "Urban Transportation Networks", Prentice-Hall.
 19. Spiess, H. and Florian, M.(1989), "A New Assignment Model for Transit Networks", Transportation Research, Vol. 23B, pp.83~102.
 20. Thomas, R.(1991), "Traffic Assignment Techniques", Avebury Technical.

✉ 주 작 성 자 : 장인성

✉ 논문투고일 : 2000. 9. 5

논문심사일 : 2000. 10. 13 (1차)

2000. 11. 28 (2차)

심사판정일 : 2000. 11. 28