經營科學
第18卷 第1號
2001年 5月

대중교통망에서의 최단경로 탐색을 위한 유전자 알고리즘

장인성* · 박승헌**

A Genetic Algorithm for Searching Shortest Path in Public Transportation Network

In-Seong Chang* · Seung-Hun Park**

■ Abstract ■

The common shortest path problem is to find the shortest route between two specified nodes in a transportation network with only one traffic mode. The public transportation network with multiple traffic mode is a more realistic representation of the transportation system in the real world, but it is difficult for the conventional shortest path algorithms to deal with. The genetic algorithm (GA) is applied to solve this problem. The objective function is to minimize the sum of total service time and total transfer time. The individual description, the coding rule and the genetic operators are proposed for this problem.

1. 서 론

최단경로문제(Shortest Path Problem)는 통신망(communication network) 및 교통망(transportation network)의 분석과 설계[3], 생산라인의 품질관리를 위한 시스템의 설계[10]등 다양한 분야에 응용될 수 있는 매우 중요한 문제이다. 특히, 교통망에서의 최단경로탐색 알고리즘은 화물수송 시에 도로를 효율적으로 이용하거나 여행자에게 편리성을 제공하기 위한 경로안내 체계의 구현을 위해 매우 중요한 핵심기술이다. 최근에 지능형 교통체계(Intelligent Transportation System : ITS)에 대한 연구가 세계적으로 활발해지면서 교통망에 존재하는 실제적인 문제들을 반영한 최단경로탐색 알고리즘의 개발에 대한 관심이 더욱 높아지고 있다.

Dijkstra[4] 알고리즘으로 대표되는 수형망 알고리즘(tree building algorithm)이 교통망의 최단경로 탐색을 위한 전통적인 기법으로 사용되어져 왔

* 숭실대학교 산업 · 정보시스템공학과
** 인하대학교 산업공학과

다[8]. 그러나 수형망 알고리즘은 교차로에서 발생하는 좌회전금지, P-turn 및 U-turn 허용 등과 같은 회전문제를 최단경로 탐색과정에서 반영하지 못한다. 이러한 단점을 극복하기 위해 덩굴망 알고리즘(vine building algorithm)[8]이 개발되어졌으며, 최근에는 좌회전금지, P-turn 및 U-turn 허용 교차로가 2개 이상 연속적으로 위치한 경우에도 현실적인 최단경로를 탐색할 수 있는 수정형 덩굴망 알고리즘[1]이 개발되어졌다. 하지만 이러한 탐색 알고리즘은 단일 교통수단으로 구성되는 단순교통망(Unimodal Transportation Network)을 대상으로 한다. 예를 들면, 개인의 교통수단인 승용차를 이용하는 개인교통망(private transportation network)에서 목적지까지의 최단경로를 탐색하거나, 특정 노선 버스의 서비스를 고려해 최적경로를 탐색하고자 한다.

그러므로 현대 도시의 교통망처럼 승용차, 대중교통(노선버스, 택시, 지하철), 직행버스, 고속버스 등과 같은 다양한 교통수단으로 구성되는 복합교통망(Intermodal Transportation Network)에 대하여 기존의 알고리즘을 적용하는데는 한계가 있다. 복합교통망에서는 교통수단간의 환승문제가 발생하기 때문에 교통망의 이용행태가 복잡하다. 즉, 승용차를 이용하다가 대중교통수단을 이용하는 등의 환승문제가 발생하기 때문에 환승에 따른 비용이나 환승 소요시간이 고려되어져야 한다.

이렇게 교통수단간의 환승을 고려해야하는 복합교통망의 경우 기존에 이용되던 알고리즘으로는 최단경로를 찾을 수 없다. 기존의 수형망 및 덩굴망 알고리즘을 적용하기 위해서는 환승이 이루어지는 지점마다 교통수단 수만큼 노드를 추가하여 교통망을 확장해서 관련된 환승비용 또는 환승소요시간을 부과하여야만 한다[2, 7]. 그러나, 교통망의 확장에 많은 시간이 소비되는 단점을 가지고 있기 때문에 교통망의 규모가 크고 교통수단이 다수인 경우에는 적용하기가 어렵다.

본 연구에서는 복합교통망의 한 축을 이루고 있는 대중교통망에서 교통수단간의 환승행태가 반영된 현실적 최단경로를 교통망의 확장없이 탐색할 수 있는 기법을 개발하기 위해 유전자 알고리즘(Genetic Algorithm : GA)을 적용하였다. 기존에 적용되던 최단경로 탐색 알고리즘의 개요와 대중교통망에서의 적용상의 문제점을 설명하고, 이를 극복할 수 있는 유전자 알고리즘의 기본개념 및 적용사례를 제시한다. 본 연구에서 개발된 알고리즘은 교차로에서의 회전, 환승뿐만아니라 ITS를 위한 다양한 문제에 유용하게 활용될 수 있는 기법이라고 고려된다.

2. 기존 알고리즘의 한계

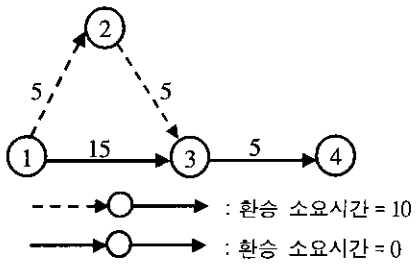
현재까지 개발된 대부분의 최단경로 탐색 알고리즘들은 기점노드로부터 각 노드까지의 최단경로를 단계별로 산출하여 최종적으로 기종점(origin-destination)간의 최단경로를 구축해가는 순차적 탐색과정을 채택하고 있다. 순차적 탐색법은 수형망 알고리즘과 덩굴망 알고리즘의 두 가지 형태로 분류되어진다.

수형망 알고리즘은 단계별로 최단경로를 구축하는 과정에서 현 노드의 전 노드만을 검색하며 대표적인 예로는 Dijkstra 알고리즘이 있다. 이에 반해 덩굴망 알고리즘은 현 노드의 전 노드와 전전노드까지를 검색범위로 가지기 때문에 수형망 알고리즘보다 적용범위가 넓다. 그러나 이러한 순차적 탐색법은 복합교통망에서의 환승행태를 정확하게 반영한 합리적인 최단경로를 찾지못한다는 단점을 갖고 있다. <그림 1>은 수형망 알고리즘과 덩굴망 알고리즘을 비교하기 위한 복합교통망의 단순한 예이다. 실선 링크는 수단 1, 점선 링크는 수단 2를 나타내고 있으며 링크위에 있는 숫자는 링크를 통과하는데 소요되는 시간(분)을 나타낸다. 다양한 교통수단들에 의해 구성되는 교통망을 복합 교통망이라 하며 복합 교통망에서는 수단전환(환승)에 따른 비용이나 시간이 추가적으로 발생된다.

기점노드 1로부터 수단 1에 의해 노드 3에 도달하는 소요시간이 15분, 기점노드 1로부터 수단 2에

의해 노드 3에 도달하는 소요시간은 10분이다. 수단 2에 의해 노드 3에 도착 한 경우 수단 1로 환승하는데 소요되는 시간이 10분이라 하자. 노드 3까지의 최단 경로는 수단 2를 이용하는 경로로써 소요시간이 10분이지만 노드 4까지의 최단경로를 고려한다면 환승 소요시간이 추가적으로 발생하므로 수단 1을 이용해서 노드 3에 도달해야 한다. 따라서 실제적으로 노드 4까지의 최단경로는 ①-③-④이며 소요시간은 20분이다.

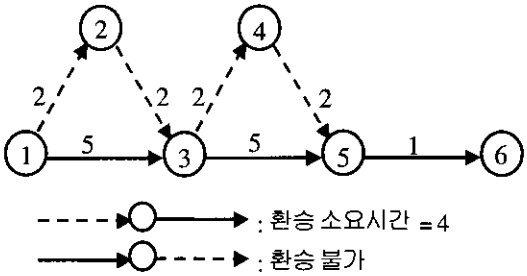
<그림 1>에 대해 수형망 알고리즘을 이용하는 경우 전 노드만을 검색하므로 기점노드 1로부터 노드 3까지의 최단경로를 계산하기 위해 전 노드 2를 경유하는 경로와 전 노드 1을 경유하는 경로를 비교한다. 전 노드가 2인 경우 노드 3까지의 소요시간은 10분, 전 노드가 1인 경우 노드 3까지의 소요시간은 15분이다. 따라서 노드 3까지의 최단경로는 노드 2를 경유하는 경로 ①-②-③이며 소요시간은 10분이다. 이때 노드 3의 전 노드로써 노드 2가 기록된다. 최종적으로 노드 4까지의 최단경로를 계산하기위해 전 노드 3만을 검색한다. 노드 3까지의 최단소요시간 10분, 노드 3의 전 노드로써 노드 2가 기록되어있으므로 노드 3에서의 환승시간 10분과 노드 3과 4를 잇는 링크의 소요시간 5분을 합한 25분을 노드 4까지의 최단소요시간으로 계산하며 노드 3까지의 최단경로와 노드 3과 4를 잇는 링크를 산술적으로 연결한 경로 ①-②-③-④를 기점노드로부터 노드 4까지의 최단경로로 탐색한다. 그러나 수형망 알고리즘에 의해 탐색된 경로는 실제적 최단경로 ①-③-④와 다른 불합리한 경로이다.



〈그림 1〉 2개의 교통수단으로 구성된 복합교통망

덩굴망 알고리즘은 전전 노드까지를 검색할 수 있으므로 <그림 1>의 환승행태를 반영할 수 있다. 덩굴망 알고리즘은 노드 4까지의 최단경로를 계산하기 위해 노드 2와 3을 경유하는 경로와 노드 1과 3을 경유하는 경로를 비교한다. 노드 2에는 전 노드로써 노드 1이 기록되어있다. 따라서 전전 노드가 2인 경우 노드 2까지의 소요시간은 5분, 노드 2에서의 환승시간 0분, 노드 2와 노드 3을 잇는 링크의 소요시간은 5분, 노드 3에서의 환승시간 10분, 노드 3과 노드 4를 잇는 링크의 소요시간 5분으로 노드 2와 3을 경유하는 소요시간은 25분이다. 전전 노드가 1인 경우 노드 1까지의 소요시간은 0분, 노드 1과 3을 잇는 링크의 소요시간 15분, 노드 3에서의 환승시간 0분, 노드 3과 노드 4를 잇는 링크의 소요시간 5분으로 노드 1과 3을 경유하는 소요시간은 20분이다. 따라서 노드 4까지의 최단경로는 ①-③-④이며 실제적인 최단경로와 일치한다.

그러나 환승노드가 2개이상 연속되는 경우에는 덩굴망 알고리즘도 수단의 전환을 경로 구축시 정확하게 반영하지 못한다. 따라서 불합리한 최단경로가 탐색되는 결과를 초래한다. <그림 2>는 덩굴망 알고리즘이 복합교통망에서 최단경로를 제대로 찾지 못하고 있음을 보여주고 있는 예이다.



〈그림 2〉 환승노드가 2개이상 연속되는 복합교통망

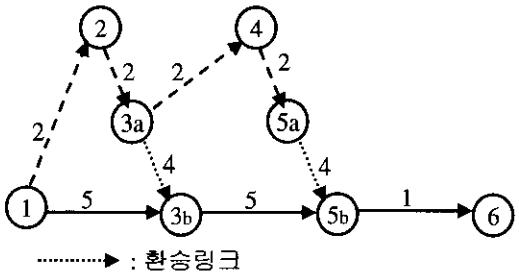
노드 1과 노드 6은 각각 기점과 종점을 나타낸다. 점선링크는 버스를 실선링크는 지하철을 나타내며, 수단간 환승시간은 버스에서 지하철로의 환

승시간이 4, 지하철에서 버스로의 환승은 불가능하다고 가정하자. 수형망 알고리즘의 경우 전 노드까지만 검색하므로 환승을 고려해 위의 문제를 풀 수 없다. 기점노드 1로부터 노드 3과 4까지의 최단경로는 버스를 이용하는 경로이며 소요시간은 각각 4분과 6분이다. 이때 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 전 노드 4, 전전 노드 3을 경유하는 경로(소요시간 8분)와 전 노드 3, 전전 노드 1을 경유하는 경로(소요시간 10분) 그리고 전 노드 3, 전전 노드 2를 경유하는 경로(소요시간 13분)를 비교한다. 따라서 덩굴망 알고리즘은 노드 5의 전 노드와 전전 노드로써 노드 4와 3을 기록하고 노드 5까지의 최단경로는 노드 3까지의 최단경로와 노드 3, 4, 5를 잇는 링크들을 산술적으로 연결한 경로 ①-②-③-④-⑤이며 소요시간은 8분이다.

노드 6까지의 최단경로는 전 노드 5, 전전 노드 4를 경유하는 경로와 전 노드 5, 전전 노드 3을 경유하는 경로를 비교함으로써 계산되어진다. 전전 노드가 4인 경우 노드 4까지의 소요시간 6분, 노드 4의 전 노드가 3이므로 노드 4에서의 환승시간 0분, 노드 4와 5를 잇는 링크의 소요시간 2분, 노드 5에서의 환승시간 4분, 노드 5와 6을 잇는 소요시간 1분으로 종점노드 6에 도착하기 위한 소요시간은 13분이다. 전전 노드가 3인 경우에는 노드 3까지의 소요시간 4분, 노드 3의 전 노드가 2이므로 노드 3에서의 환승시간 4분, 노드 3과 5를 잇는 링크의 소요시간 5분, 노드 5에서의 환승시간 0분, 노드 5와 6을 잇는 링크의 소요시간 1분이므로 노드 6까지의 소요시간은 14분이 된다. 따라서, 덩굴망 알고리즘에의해 노드 1에서 노드 6까지의 최단소요시간은 13분으로 계산되며 해당경로는 노드 4까지의 최단경로 ①-②-③-④와 노드 5, 6을 산술적으로 연결한 ①-②-③-④-⑤-⑥이다. 그러나 <그림 2>의 실제적인 최단경로는 ①-③-⑤-⑥이다.

이와같은 현상은 순차적 탐색법이 단계별로 기점노드로부터 각 노드까지의 최단경로를 계산하여 이를 산술적으로 연결하면 기종점간의 최단경로가 된다고 가정하기 때문이다. 따라서, 순차적 탐색법은 복합교통망에서와 같이 기점으로부터 특정 노드 n에 도달하는 최단경로와 기점으로부터 노드 n을 경유해서 그 다음 노드 $n+1$에 도달하는 최단경로의 부분경로인 노드 n까지의 경로가 서로 다를 경우에는 적용되지 않는다. 이러한 단점을 극복하기 위해서 Modesti and Sciomachen[7], 최기주와 장원재[2] 등은 교통망을 확장함으로써 해결할 수 있는 방법을 제시하였다. 즉, 환승이 이루어지는 지점을 교통수단 수만큼의 노드로 분할하여 교통망을 확장하고 추가된 링크에 환승소요시간을 부과함으로써 순차적 탐색법을 그대로 적용하면서도 환승행태를 반영한 최단경로를 탐색할 수 있도록 하였다. <그림 3>은 <그림 2>를 확장시킨 교통망을 나타내고 있다. 환승노드 3과 5가 2개의 노드로 분할되어 6개의 노드와 7개의 링크로 구성된 교통망이 8개의 노드와 9개의 링크로 구성된 교통망으로 확장되어졌다. 따라서 순차적 탐색법을 적용하기 위해서는 탐색절차가 2개의 노드가 추가된 8개의 노드에 대해서 단계별로 수행되어져야 한다.



〈그림 3〉 〈그림 2〉의 확장된 복합교통망

그러나 현대의 교통망은 다양한 교통수단의 상호작용을 통해 운영되어지기 때문에 수단간의 환승행태가 복잡하며 교통망의 확장에 많은 시간이 소비된다. 특히 대중교통망의 경우에는 다양한 교통수단뿐만 아니라 다수의 노선이 존재하기 때문에 수단간의 환승행태가 더욱 복잡하다. 게다가 거

의 대부분의 노드에서 환승이 가능하기 때문에 환승노드마다 교통수단과 노선을 고려한 수만큼의 노드를 추가하여 교통망을 확장시킨다는 것은 실제로 불가능하다. 또한 기존의 순차적 탐색법은 최단경로를 구축하기 위해 교통망의 모든 노드를 탐색하므로 교통망의 규모가 대규모인 경우에는 비효율적이다.

따라서 본 연구에서는 교통망의 확장없이 대중교통망에서의 환승행태가 반영된 합리적 최단경로를 탐색할 수 있는 해법을 개발하기 위해 유전자 알고리즘을 적용하였다. 개발된 알고리즘은 최단경로의 탐색과정이 단계별로 모든 노드에 대해서 이루어지는 것이 아니라 교통망의 특정경로를 구성하는 일부 노드에 대해서 이루어지며 다수의 경로들에 대하여 병렬로 수행되어지므로 신속하게 최단경로를 탐색할 수 있다는 장점을 갖고 있다.

3. 대중교통망에서의 최단경로 탐색을 위한 유전자 알고리즘

유전자 알고리즘에서는 대상이 되는 문제의 후보해들을 문자나 기호의 배열인 염색체(chromosome)로 표현하며 염색체를 구성하는 문자나 기호들을 유전자(gene)라 부른다. 이들 염색체들의 적당한 크기(population size)로 구성되는 모집단(population)을 생성하여 이들을 점차적으로 갱신함으로써 전역적인 최적해를 탐색하게 된다. 모집단의 생성과정을 초기화(initialization)라 하며 모집단의 갱신은 유전 연산자(genetic operator)에 해당되는 선택(selection), 교배(crossover), 돌연변이(mutation)의 세 가지 과정을 통해서 이루어지고 이를 1 세대(generation)라 한다. 해의 탐색과정에서 모집단을 이용한 병렬처리를 수행하기 때문에 탐색소요시간이 매우 짧다.

3.1 개체의 표현

유전자 알고리즘을 적용하는 첫 단계이며 가장 중요한 것은 대상이 되는 문제의 후보해를 염색체로 표현하기 위한 방법을 설계하는 과정이다. 본 연구에서 다루는 문제는 교통망의 기점과 종점을 잇는 최단경로를 탐색하는 것이므로 각 교통망의 모든 경로를 염색체에 대응시킬 수 있어야 한다. 또한, 유전 연산자가 수행된 뒤 초래되는 결과는 통행 가능한 경로를 나타내어야 한다. 그러나 이러한 조건을 충족시키는 유전자형의 설계는 쉽지 않다. 유전자 알고리즘이 네트워크 모델에 적용된 대표적인 예는 순회판매원문제(Traveling Salesman Problem)에서 찾을 수 있으며 순회해야 할 각 도시들의 순서를 조합으로 나열하여 염색체를 구성함으로써 후보해를 표현하고 있다[6]. Vignaux and Michalewicz[9]은 수송문제(Transportation Problem)에 유전자 알고리즘을 적용하였으며 염색체를 구성하는 유전자의 배열을 행렬의 형태나 생산지와 소비지의 순서쌍으로 나열하여 후보해를 표현하고 있다. 최근에 Zhou and Gen[11]은 최소연결나무문제(Minimal Spanning Tree Problem)에 염색체를 구성하는 유전자의 배열을 노드들의 조합인 Prüfer number[5]로 표현하여 유전자 알고리즘을 구성하였다.

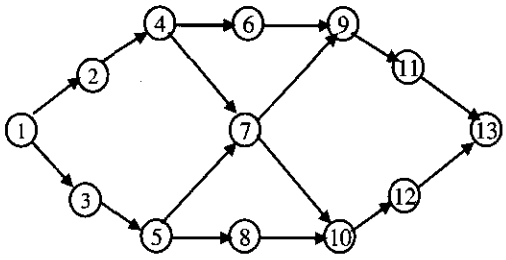
본 연구에서는 교통망상의 기종점을 잇는 경로 자체를 염색체로 이용하였다. 기점노드가 ①이고 종점노드가 ⑬인 <그림 4>의 교통망에서 기종점을 잇는 어떤 경로 ①-②-④-⑥-⑨-⑪-⑬에 대응되는 염색체를 A라하면 염색체 A는 다음과 같이 표현된다.

$$A = (g[1]\ g[2]\ g[3]\ g[4]\ g[5]\ g[6]\ g[7])$$
$$= (1\ 2\ 4\ 6\ 9\ 11\ 13)$$

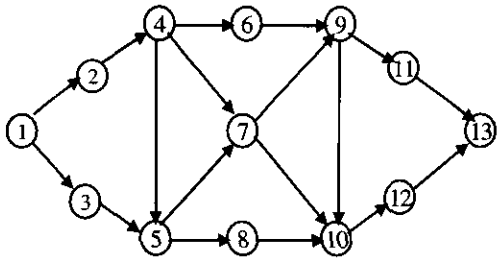
여기서 $g[i](i = 1, 2, \cdots, 7)$는 염색체 A의 i번째 유전자를 나타내며 상기의 경로를 구성하는 i번째 노드가 이에 대응되어진다. <그림 4>와 같이 각각의 경로에 대하여 경로를 구성하는 노드의 수가 7개로써 항상 일정한 경우에는 각 염색체의 유전자 수 또한 같다. 그러나 <그림 5>와 같이 각각의 경로에 대하여 경로를 구성하는 노드의 수가 일정치 않은 경우에는 각 염색체의 유전자 수는 일정치

않다. 예를 들면 <그림 5>의 교통망에서 다음과 같이 염색체의 유전자 수가 서로다른 염색체 A, B가 존재한다.

A = (g[1] g[2] g[3] g[4] g[5] g[6] g[7])
= (1 2 4 6 9 11 13)
B = (g[1] g[2] g[3] g[4] g[5] g[6] g[7] g[8])
= (1 2 4 6 9 10 12 13)



〈그림 4〉 경로를 구성하는 노드의 수가 동일한 교통망



〈그림 5〉 경로를 구성하는 노드의 수가 다른 교통망

〈표 1〉 선행관계 행렬

$$R = \begin{bmatrix} 0&1&1&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&0&0&0&0&0 \\ 0&0&0&0&1&1&1&0&0&0&0&0&0 \\ 0&0&0&0&0&0&1&1&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&1&1&0&0&0 \\ 0&0&0&0&0&0&0&0&1&1&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&0&0&0&0&1&1&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&1 \\ 0&0&0&0&0&0&0&0&0&0&0&0&1 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0 \end{bmatrix} \quad r_{58}=1 \quad r_{57}=1$$

모집단 생성을 위한 컴퓨터 코딩작업을 효율적으로 수행하기위해 노드들의 선행관계 행렬 R = $\{r_{ij}\}_{n\times n}$을 이용하였다. 선행관계 행렬 R은 n차원의 정방행렬이며 n은 종점노드, r_{ij}는 선행관계 행렬의 성분을 나타낸다. 여기서 노드 j가 노드 i의 직접적인 후속노드라면 r_{ij} = 1, 노드 j가 노드 i의 직접적인 후속노드가 아니라면 r_{ij} = 0이다. <그림 5>에 예시된 교통망에 대하여 선행관계 행렬 R을 작성하면 <표 1>과 같다. 예를 들면, 노드 5의 후속 노드는 노드 7과 노드 8이므로 r_{57} = 1, r_{58} = 1이다.

R을 이용해서 염색체 A를 나타내기 위해 먼저 기점노드(origin node)인 노드 1을 첫 번째 유전자 g[1]에 대응시킨다. i = 1로 치환하여 r_{1j} = 1인 노드 1의 후속노드 j를 찾는다. j가 2와 3인 경우에 대해 r_{12} = 1, r_{13} = 1이므로 이중 하나를 임의로 선택해서 g[2]에 대응시킨다. 선택된 j가 3이라하면 i = j = 3으로 치환하여 r_{3j} = 1인 노드 3의 후속노드 j를 찾는다. r_{35} = 1이므로 j = 5를 g[3]에 대응시키고 i = j = 5로 치환한다. r_{57} = 1, r_{58} = 1에서 임의로 j = 8을 g[4]에 대응시키고 i = j = 8로 치환한다. 이 과정을 i가 종점노드(destination node)일때까지 반복함으로써 다음과 같은 염색체 A를 얻을 수 있다.

A = (g[1] g[2] g[3] g[4] g[5] g[6] g[7])
= (1 3 5 8 10 12 13)

구체적으로는 아래와 같은 초기화 절차에 의해 모집단을 생성하였다.

```
Procedure : Initialization
begin
    l = 1 ;
    while (l ≤ population_size) do
    begin
        i = origin, n = 1, g[1] = origin ;
        while (i ≠ destination) do
        begin
            Ψi = ∅, j = 1 ;
            while (j ≤ max_node) do
            begin
                check rij ;
                if (rij = 1) then
                    Ψi = Ψi ∪{j} ;
```

```
            else
                Ψ_i = Ψ_i ;
            j = j + 1 ;
          end
          n = n + 1 ;
          set g[n] = k for ∃k∈Ψ_i ;
          i = k ;
        end
        l = l + 1 ;
    end
end
```

l은 염색체 번호, population_size는 모집단의 크기, Ψ_i는 특정 노드 i에 대해 $r_{ij}=1$을 만족시키는 노드 j들의 집합, max_node는 최대 노드번호를 나타낸다.

3.2 평가함수

다음 단계는 각 염색체에 대하여 해로써의 우수성을 평가하는 것이다. 본 연구에서는 목적함수인 기종점을 잇는 경로의 총 소요시간을 평가함수로 사용하였다. n개의 유전자로 구성되는 염색체 $A = (g[1]\ g[2]\ \cdots\ g[n])$에 대해 평가함수 $eval(A)$는 다음의 식과 같이 표현되어진다.

$$eval(A) = TST(A) + TTT(A)$$

여기서, TST는 교통수단에 의해 서비스 되어진 총 운행시간이며 TTT는 총 환승소요시간으로 다음과 같이 정식화되어진다.

$$TST(A) = \sum_{i=1}^{n-1} ST(g[i],\ g[i+1])$$

여기서, ST는 i번째 유전자에 대응되는 노드와 $i+1$번째 유전자에 대응되는 노드사이의 운행소요시간을 나타낸다.

$$TTT(A) = \sum_{i=1}^{n-2} TT(MD[g[i],\ g[i+1]],\ MD[g[i+1],\ g[i+2]])$$

여기서, MD는 두 노드를 연결하는 교통수단을 나타내며 TT는 i번째 유전자와 $i+1$번째 유전자에 대응되는 두 노드를 연결하는 교통수단으로부터 $i+1$번째 유전자와 $i+2$번째 유전자에 대응되는 두 노드를 연결하는 교통수단으로 전환하는 환승소요시간을 나타낸다.

3.3 선택

선택이란 현재의 모집단을 갱신시키기 위한 준비단계로써 지정된 전략에 따라 우수한 염색체를 다음세대로 복제(reproduction)시키거나 교배를 위한 염색체의 쌍을 만들기 위해 현재의 모집단을 재편성하는 과정이다. 대표적인 선택전략으로써 엘리트 보존전략과 roulette wheel 전략 등이 있다[6]. 본 연구에서는 두 전략을 결합한 혼합 전략을 사용하였다. 먼저 엘리트 보존전략에 의해 현 모집단 중에서 평가함수의 결과치인 적합도가 가장 우수한 염색체는 교배나 돌연변이의 대상이 되지않도록 다음세대로 복제한다. 그리고 모집단 수를 유지하기 위해 모집단의 부족한 수만큼을 현 모집단으로부터 roulette wheel 전략에 의해 적합도에 비례한 확률로 선택한다. 이 과정에서 각 세대의 가장 우수한 염색체가 다음세대에 유전되어지며 궁극적으로는 최적해가 다음세대에서 제거되어지는 것을 방지할 수 있다.

3.4 교배

교배는 재편성된 모집단으로부터 일정한 교배확률(p_c)에 따라 교배를 위한 염색체의 쌍을 만들어 각 염색체의 유전자 교환을 통해 모체보다 우수한 염색체를 생성시키기 위한 단계이다. 일반적인 교배연산자로는 일점교배, 다점교배, 일양교배가 있으며 그외에도 PMX, OX, CX등의 교배연산자가 다양한 문제에 적용되어져왔다[6]. 그러나 기존의 교배연산자를 최단경로문제에 적용하는 경우에는 교배연산자가 수행된 후 존재하지 않는 경로가 발

생한다.

본 연구에서는 상기의 교배연산자들 중에서도 최단경로문제에 적용하기에 가장 적합하다고 사려되어지는 일점교배를 본 문제에 맞게 수정하였다. 일점교배는 두 개의 염색체에 대해 교배위치를 임의로 1점 설정하고 후반의 유전자들을 교환하지만, 본 연구에서는 기점노드와 종점노드를 제외한 두 염색체의 공통유전자를 교배위치로 선정하였다. 만일 두 염색체가 공통으로 소유하고 있는 유전자가 두 개 이상인 경우에는 그 중에서 임의로 하나의 유전자를 교배위치로 선정하였다.

<그림 4>의 교통망에서 교배를 위해 선택된 한 쌍의 염색체 A, B가

$$A = (1\ 2\ 4\ \blacksquare\ 10\ 12\ 13)$$
$$B = (1\ 3\ 5\ \blacksquare\ 9\ 11\ 13)$$

인 경우, 두 염색체의 공통유전자는 7이다. 따라서 유전자 7후반의 유전자들의 교환을 통해 생성되는 새로운 염색체 A', B'는 다음과 같다.

$$A' = (1\ 2\ 4\ 7\ 9\ 11\ 13)$$
$$B' = (1\ 3\ 5\ 7\ 10\ 12\ 13)$$

<그림 5>의 교통망에서 교배를 위해 선택된 한 쌍의 염색체 C, D가

$$C = (1\ 2\ 4\ \blacksquare\ 8\ 10\ 12\ 13)$$
$$D = (1\ 3\ \blacksquare\ 7\ 9\ 11\ 13)$$

인 경우, 두 염색체의 공통유전자는 5이다. 따라서 유전자 5후반의 유전자들의 교환을 통해 생성되는 새로운 염색체 C', D'는 다음과 같다.

$$C' = (1\ 2\ 4\ 5\ 7\ 9\ 11\ 13)$$
$$D' = (1\ 3\ 5\ 8\ 10\ 12\ 13)$$

교배연산자가 수행된 후 생성된 염색체 A', B', C', D'는 교통망상에 존재하는 통행 가능한 경로들이라는 것을 <그림 4>와 <그림 5>로부터 알 수 있다.

3.5 돌연변이

돌연변이란 각 염색체에 대해 일정한 돌연변이 확률(p_m)로 모체의 성질과는 전혀 다른 새로운 염색체를 발생시키는 과정이다. 본 연구에서는 염색체를 구성하는 유전자들 중에서 임의로 하나의 유전자를 선정하고 선정된 유전자 후반의 유전자들을 <표 1>의 선행관계 행렬 R을 이용해서 재 구성하였다. <그림 4>의 교통망에서 돌연변이를 위해 선택된 염색체 A가

$$A = (1\ 2\ \blacksquare\ 6\ 9\ 11\ 13)$$

이고, 임의로 선정된 유전자가 4라면 $i=4$로 치환하여 선행관계 행렬 R을 이용해서 유전자 4 후반의 유전자들이 재구성된다. r_{4j}에서 $j=7$, r_{7j}에서 $j=10$이 선택되었다면 돌연변이가 수행된 후 생성되는 새로운 염색체 A'는 다음과 같다.

$$A' = (1\ 2\ 4\ 7\ 10\ 12\ 13)$$

3.6 GA 절차

제안된 알고리즘의 전체적인 절차는 다음과 같다.

```
Procedure : GA
begin
    t = 0 ;
    initialize the population of solutions P(t) by
    initialization procedure ;
    evaluate P(t) ;
    while (not termination condition) do
    begin
        select P(t+1) from P(t) ;
        recombine P(t+1) ;
        evaluate P(t+1) ;
    end
    t = t + 1 ;
end
```

4. 실험결과

본 연구에서는 가상 복합교통망을 임의로 생성하여 제안된 알고리즘의 성능을 평가하였다. 먼저 통행 가능한 경로를 현실적으로 모두 나열할 수 있는 소규모 교통망에 대하여 열거에 의해 계산된 최단경로를 제안된 알고리즘의 결과와 비교함으로써 성능을 검증하였다. 또한 제안된 알고리즘에서 사용되는 교배확률(p_c)과 돌연변이 확률(p_m)의 적당한 값을 설정하였다. 설정된 매개변수들의 값을 이용해서 제안된 알고리즘을 현실적으로 통행 가능한 경로를 모두 나열하기가 불가능한 중규모 이상의 교통망에 적용하였으며 우수성을 평가하기 위해 기존의 수형망 및 덩굴망 알고리즘, 노드확장 기법과 비교분석하였다. 제안된 알고리즘과 비교분석을 위해 사용된 알고리즘은 모두 Borland C++ 5.02을 사용하여 구현하였으며 Pentium 200MHz CPU, 64MB RAM를 장착한 IBM-PC에서 수행되었다.

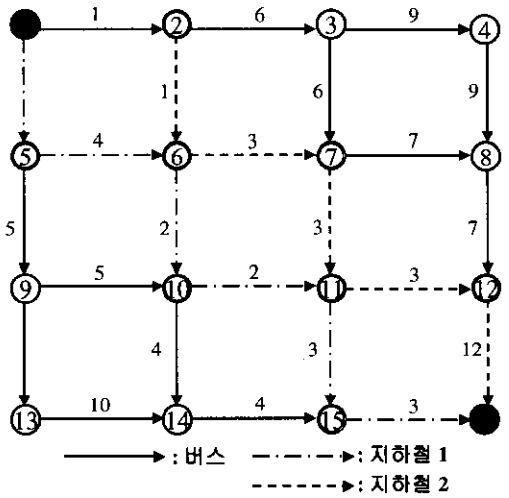
4.1 성능검증

규모가 다양한 교통망을 효율적으로 생성하기 위해 격자형 교통망(노드×노드)을 이용하였다. 성능검증을 위해 사용된 소규모 교통망은 <그림 6>에 예시된 4×4의 격자형 교통망으로써 16개의 노드와 24개의 링크, 3개의 교통수단으로 구성되어 있다. 노드 1과 노드 16은 기점과 종점노드를 나타내며, 환승노드 2, 5, 6, 7, 10, 11, 12, 15에서 수단 전환을 위한 환승소요시간은 <표 2>와 같다.

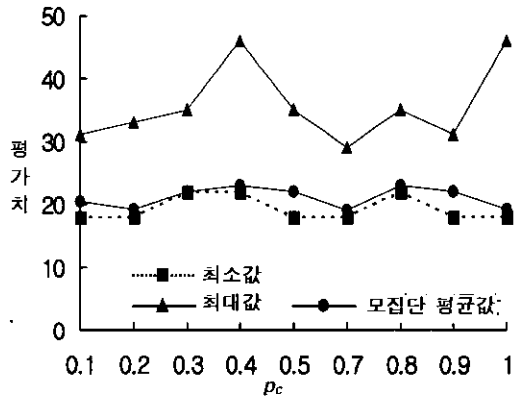
<그림 6>의 교통망에 존재하는 경로 수는 총 20가지이며 이들 경로를 모두 열거하여 비교함으로써 쉽게 최단경로 p^* = ①-⑤-⑥-⑩-⑪-⑮-⑯(총 소요시간 = 18)를 얻을 수 있다. <그림 6>의 예제에 유전자 알고리즘을 적용하기에 앞서서 모집단의 크기(population_size)와 최대세대 수(max_gen)를 population_size =10, max_gen = 20으로 고정하고, 교배확률(p_c)과 돌연변이 확률(p_m)의 적당한 값을 실험적으로 계산하였다. p_c는 0.1부터 1.0까지 0.1단위로, p_m은 0.01부터 0.05

〈표 2〉 환승소요시간

환승행태	소요시간
버 스 → 지하철 1	4
버 스 → 지하철 2	5
지하철 1 → 버 스	6
지하철 1 → 지하철 2	4
지하철 2 → 버 스	3
지하철 2 → 지하철 1	5



〈그림 6〉 소규모 복합교통망



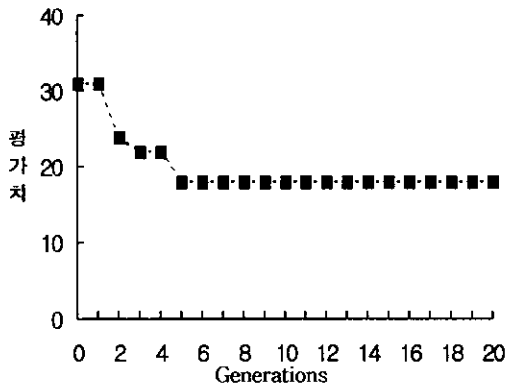
〈그림 7〉 p_c의 변화에 따른 평가치의 분포(p_m = 0.2)

단위로 1.0까지 변경시키면서 측정하였다. 실험결과, p_m은 p_c와 관계없이 0.1~0.4에서 바람직하였으며 특히 p_m=0.2일때 가장 우수한 결과를 보여주었다. <그림 7>은 p_m=0.2일때 p_c의 변화에 따른 평가치(총 소요시간)를 보여준다. 모집단중에서 총 소요시간이 최소인 가장 우수한 염색체와 총 소요시간이 최대로 가장 열등인 염색체, 모집단의 평균값을 비교해 보면 p_c=0.7이 적정 값임을 알 수 있다.

<그림 6>의 예제에 수형망 및 덩굴망 알고리즘, 노드확장 기법을 적용한 결과를 제안된 알고리즘과 비교하여 보면 <표 3>과 같다. 노드의 확장없이 순차적 탐색법인 수형망 및 덩굴망 알고리즘을 적용하는 경우에는 최단경로를 정확하게 산출하지 못한다. 이에 반해 노드를 확장하고 수형망 알고리즘의 한 형태인 Dijkatra 알고리즘을 적용하는 경우에는 최적경로를 제공하지만, 8개의 각 환승노드를 이용할 수 있는 수단 수만큼의 노드로 분할하고 링크를 추가하는 사전작업이 요구된다. 게다가 경로계산이 모든 노드에 대해서 수행되므로 많은 계산량이 요구되어져 비효율적이다. 그러나 제안된 알고리즘은 노드확장을 위한 추가적인 작업이 수반되지않으며, 특정 노드들에 대해서만 경로계산이 수행되므로 최단경로를 신속하게 탐색한다. <그림 8>은 p_c=0.7, p_m=0.2일 때 세대 반복에 따른 제안된 알고리즘의 결과를 나타내며 소규모 복합교통망에 대하여 해의 수렴속도가 우수함을 알 수 있다. 특히 3세대와 4세대에서 총 소요시간이 22인 경로 p = ①-②-⑥-⑩-⑪-⑮-⑯가 탐색되어지며 이것은 최적 해는 아니지만 수형망 및 덩굴망 알고리즘에서 탐색된 결과보다 우수하다. 따라서 제안된 알고리즘은 소규모 복합교통망에 대하여 빠른 시간안에 현실적인 최단경로를 탐색하는 것을 알수 있다.

〈표 3〉 알고리즘의 비교(4×4 격자형)

알고리즘	최 단 경 로	소요시간
수 형 망	①-②-⑥-⑦-⑪-⑮-⑯	24
덩 굴 망	①-②-⑥-⑦-⑪-⑮-⑯	24
노드확장 기법	①-⑤-⑥-⑩-⑪-⑮-⑯	18
GA	①-⑤-⑥-⑩-⑪-⑮-⑯	18

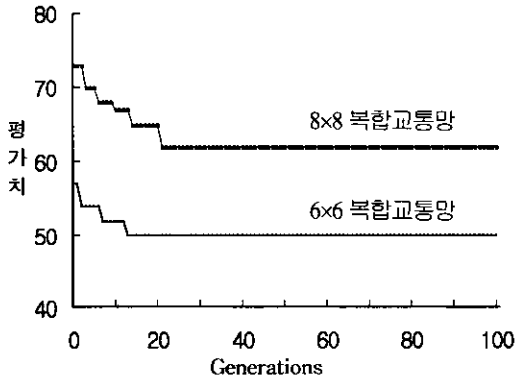


〈그림 8〉 세대별 진화과정(4×4 격자형)

4.2 비교분석

제안된 알고리즘의 성능을 중규모 이상의 교통망에서 평가하기 위해 36개의 노드와 60개의 링크, 14개의 환승노드로 구성된 6 × 6의 격자형 복합교통망과 64개의 노드와 112개의 링크, 19개의 환승노드로 구성된 8 × 8의 격자형 복합교통망을 이용하였다. 각 링크의 소요시간은 1과 10사이의 정수를 임의로 발생하여 설정하였으며, 사용 가능한 교통수단은 3개, 환승노드에서 수단전환을 위한 환승소요시간은 <표 2>와 동일하다. 6 × 6의 격자형 복합교통망에는 126가지의 가능한 경로가 존재하고 8 × 8의 격자형 복합교통망에는 3432가지의 통행 가능한 경로가 존재한다. 따라서 모든 경로를 나열하여 비교하는 것은 현실적으로 불가능하다. 각각의 교통망에 대하여 p_c = 0.7, p_m = 0.2, population_size = 30, max_gen = 100으로 설정하여 수행한 결과를 세대별로 나타내면 <그림 9>와 같다. <그림 9>로부터 본 알고리즘은 중규모 이상의 복합교통망에 대하여도 해의 수렴속도가 우수함을 알 수 있다. <표 4>는 노드확장 기법과 제안된 알고리즘을 적용하여 얻어진 최단경로의 소요시간과 알고리즘의 수행시간을 나타낸다. 각각의 예제 교통망에 대

하여 노드확장 기법을 적용하는 경우에는 노드와 링크를 추가하는 사전작업을 포함해서 최단경로를 탐색하는데 각각 20분, 32분이 소요되었다. 그러나, 제안된 알고리즘은 노드확장을 위한 추가적인 작업없이 10초이내의 매우 빠른 시간안에 최단경로를 산출하는 것을 알 수 있다. 따라서 본 알고리즘은 현실성이 반영된 대규모의 복합교통망에서도 합리적인 최단경로를 탐색하기에 유용하다고 할 수 있다.



〈그림 9〉 세대별 진화과정(6×6, 8×8 격자형)

〈표 4〉 알고리즘의 비교(6×6, 8×8 격자형)

예 제	노드확장 기법		GA	
	소요시간	계산시간 (min)	소요시간	계산시간 (sec)
6 × 6	50	20	50	4.25
8 × 8	62	32	62	8.79

5. 결 론

본 연구에서는 다양한 교통수단의 상호작용을 통해 운영되어지는 대중교통망에서 수단간의 환승행태를 반영한 최단경로를 탐색하는 경우에 대해 기존 알고리즘의 문제점을 제시하였으며 또한 이를 극복할수 있는 유전자 알고리즘을 제시하였다. 본 연구에서 개발된 알고리즘의 성능을 통행가능한 경로를 모두 나열할 수 있는 소규모 복합교통망에 대하여 적용한 결과 최단경로를 탐색하는 것으로 나타났다. 또한 중규모 복합교통망에 대하여 본 알고리즘과 기존의 알고리즘을 비교분석한 결과 본 알고리즘의 효율성 및 우수성이 입증되었다. 추후의 연구과제로는 대중교통망 분석시 보다 정확한 자료로 제공되어지기 위해 수단 전환에 의한 환승 소요시간뿐만 아니라 역내 대기시간도 고려할 수 있는 알고리즘의 개발이 요구되어진다.

참 고 문 헌

[1] 김익기, "ATIS를 위한 수정형 덩굴망 최단경로 탐색 알고리즘의 개발", 「대한교통학회지」, 제16권 제2호 (1998), pp.157-167.

[2] 최기주, 장원재, "복합 교통망에서의 최적경로 산정 모형개발", 「대한교통학회지」, 제16권 제4호 (1998), pp.167-186.

[3] Deo, N. and C. Pang, "Shortest Path Algorithms : Taxonomy and Annotation," *Networks*, Vol.14 (1984), pp.275-323.

[4] Dijkstra, E.W., "A Note on Two Problems in Connection with Graphs," *Numerische Mathematik*, Vol.1 (1959), pp.269-271.

[5] Dossey, J., A. Otto, L. Spence and C. Eynden, *Discrete Mathematics*, Harper Collins. 1993.

[6] Michalewicz, Z., *Genetic Algorithm + Data Structure = Evolution Programs*, Springer-Verlag, New York, 1994.

[7] Modesti, P. and A. Sciomachen, "A Utility Measure for Finding Multiobjective Shortest Paths in Urban Multimodal Transportation Networks," *Europen Journal of Operational Research*, Vol.111 (1998), pp.495-508.

[8] Thomas, R., *Traffic Assignment Techniques*, Avebury Technical, 1991.

[9] Vignaux, G.A., and Z. Michalewicz, "A Genetic Algorithm for the Linear Transportation Problem," *IEEE Transaction on Systems, Man, and Cybernetics* Vol.21 (1991), pp.445-

452.

[10] White, L.S., "Shortest Route Models for the Allocation of Inspection Effort on a Production Line," *Management Science*, Vol.15 (1969), pp.249-259.

[11] Zhou, G. and M. Gen, "Genetic Algorithm Approach on Multi-Criteria Minimum Spanning Tree Problem," *European Journal of Operation Research*, Vol.114 (1999), pp. 141-152.