Jcurnal of the Korean Institute
of Industrial Engineers
Vol.23, N0.4, December, 1997

Rural Postman 문제에서 헤밀토니안 그래프 변환에 의한 유전자 알고리즘 해법

A Genetic Algorithm Using Hamiltonian Graph for Rural Postman Problem

강명주* · 한치근**

Kang, Myung-Ju* · Han, Chi-Geun**

Abstract

For an undirected graph G=(V, E), the Rural Postman Problem (RPP) is a problem that finds a minimum cost tour that must pass edges in E'(⊆ E) at least once. RPP, such as Traveling Salesman Problem (TSP), is known as an NP-Complete problem. In the previous study of RPP, the structure of the chromosome is constructed by E' and the direction of the edge. Hence, the larger the size of |E'| is, the larger the size of the chromosome and the size of the solution space are. In this paper, we transform the RPP into a Hamiltonian graph and use a genetic algorithm to solve the transformed problem using restructured chromosomes. In the simulations, we analyze our method and the previous study. From the simulation results, it is found that the results of the proposed method is better than those of the previous method and the proposed method also obtains the near optimal solution in earlier generations than the previous study.

1. 서 론

Rural Postman Problem (RPP)은 노드 V와 에지 E로 구성된 임의의 그래프 G=(V, E)에서 특정 에지 E'(⊆ E)을 반드시 한 번 이상 경유하는 최소 경비 경로를 구하는 문제이다[5]. 이러한 최소 경비 경로를 구하는 문제로는 Traveling Salesman Problem (TSP)[5,7] 과 Chinese Postman Problem(CPP)[5] 등이 있다. TSP는 N개의 도시를 모두 연결하는 최소 비용 경로를 구하는 것으로서 RPP와 함께 NP-Complete 문제로 알려져 있다[1,5,10]. 그리고 CPP는 모든 에지를 한 번 이상 경유하는 최소 비용 경로를 구하는 문제로서 무방향 그래프에서는 다항 시간 내에 구할 수 있지만, 에지(edge)와 아크(arc)가 혼재된 그래프에서는 NP-Complete 문제로 알려져 있다[5].

RPP인 경우 많은 국소 최소점(Local Optimum Point)이 존재하기 때문에 전역 최소점(Global Optimum Point)을 찾기 위해서는 해 공간(Solution Space) 전체를 검색하여야 하는 어려움이 있다. 따라서, 현실적으로 근사 최적해를 구하게 되는데, 근사 최적해를 구하는 알고리즘에는 담금질(Simulated Annealing) 방법[8,14,16], 신경회로망 (Neural Network) 방법[3,4,8,14,16] 그리고 유전자 알고리즘 (Genetic Algorithms)[6,9,11,13,15]등이 있다.

* 경희대학교 대학원 전자계산공학과 박사과정 수료

** 경희대학교 전자계산공학과 부교수

담금질 방법은 최적해를 구할 수 있지만 무한회의 잠정적인 반복이 이루어져야 하기 때문에 문제 크기가 크면 그만큼 시간이 오래 걸린다는 단점이 있다[14]. 신경회로망 방법을 최적화 문제에 적용한 것은 Hopfield에 의해 시도되었다[14,16]. 그러나 TSP인 경우 노드의 수가 많아질수록 성능이 떨어진다는 단점이 있다[14]. 또한, RPP인 경우, 입력층(Input Layer)은 E'을 이루는 노드들로 구성되고 출력층(Output Layer)은 전체 라우팅이 이루어지는 노드들로 구성되어야 하기 때문에 TSP에 비해 네트워크 구조가 더 복잡하며, |E'|이 커질수록 성능이 떨어진다. 반면, 유전자 알고리즘은 1975년 John Holland에 의해 처음 도입된 것으로 여러 개의 개체를 동시에 발생시켜 그 개체들에 대해 적자 생존(The Survival of the Fittest) 방법으로 진화하여 최적해를 구하는 알고리즘[6,9,11]으로서, TSP[6,7,11]와 Vehicle Routing Problem (VRP)[2]등과 같은 여러 가지 최적화 문제에 응용되어 좋은 결과를 주고 있다. 따라서, 본 논문에서는 유전자 알고리즘을 RPP의 최적화 문제에 적용하였다.유전자 알고리즘에 관한 자세한 사항은 참고문헌 [2,6,9,11,12,13]에 잘 설명되어 있다.

본 논문에서는 RPP의 근사 최적해를 구하기 위해 유전자 알고리즘을 적용하였다. RPP에서 유전자는 그래프 G=(V, E)에서 E'(⊆ E)을 반드시 한번 이상 경유해야 하기 때문에 E'의 순서와 E' 내의 각 에지들의 방향성이 모두 포함되어 있어야 한다. 기존의 [15]에서 사용한 염색체 코딩 방법은 E'과 그 E' 내의 각 에지의 라우팅 방향 정보를 이용하였다. 이렇게 할 경우, 한 염색체의 크기가 2|E'|이 되고, 또한, 해 공간의 크기는 $2^{|E'|} \times |E'|!$이 된다. 그리고 두가지의 정보로 염색체를 구성하였기 때문에 유전자 연산을 어떻게 적용하느냐에 따라 근사 최적해의 값과 해의 수렴 속도가 달라질 수 있다는 단점이 있다. 본 논문에서는 이러한 단점을 해결하기 위해 염색체의 코딩이 이루어지기 전에 RPP를 헤밀토니안 그래프(Hamiltonian Graph)로 변환한 후 염색체를 코딩하였다.

RPP를 헤밀토니안 그래프로 변환하는 방법은 E'에 속하는 에지들을 노드(V^H)로 변환시키고, 이 노드들을 완전 연결(Completely connected)시킴으로써 가능해진다. 그리고 변환 과정에서 헤밀토니안 그래프의 노드들 사이의 실제 경유되는 노드 정보를 테이블에 저장하여 디코딩(Decoding) 단계에서 참조할 수 있도록 하였다. 변환과정을 거쳐 RPP 문제는 헤밀토니안 문제가 된다. 이때 염색체의 구성은 헤밀토니안 그래프의 노드들로만 구성함으로써 염색체의 크기는 $|V^H|$이 되고 해 공간의 크기는 $|V^H|!$이 된다. 또한, 유전자 연산을 일정하게 적용할 수 있기 때문에 [15]에서의 단점을 해결할 수 있다. 본 논문에서 사용한 연산자들은 TSP 등에 적용되는 교환 연산자인 Partially Matched Crossover(PMX) 방법[6,9,11]과 돌연변이 연산자인 교환(Exchange) 및 반전(Inversion) 방법[9,11]을 적용하였다.

2. 헤밀토니안 그래프와 Rural Postman Problem

2.1 헤밀토니안 그래프

헤밀토니안 그래프는 그래프의 모든 노드들을 연결하는 사이클이 존재하는 그래프를 말한다[1,10]. 즉, 헤밀토니안 사이클(Hamiltonian Cycle)이 존재하는 그래프는 헤밀토니안 그래프가 된다. 또한, 어떤 그래프가 헤밀토니안 사이클을 가지고 있는지를 결정하는 문제를 헤밀토니안 사이클 문제라 하며 NP-Complete 이다[1,10]. 그리고, 헤밀토니안 그래프의 정의에 의해 모든 완전 연결 그래프는 헤밀토니안 사이클임을 알 수 있다.

본 논문에서는 RPP를 헤밀토니안 그래프로 변환한 후, 헤밀토니안 사이클을 나타낼 수 있도록 염색체를 구성하여 유전자 알고리즘에 적용하였다.

2.2 Rural Postman Problem

Rural Postman Problem (RPP)은 노드 V와 에지 E로 구성된 무방향 그래프 G = (V, E)에서 반드시 지나야 하는 에지 E'(⊆ E)을 한번 이상 모두 거치는 최소 비용의 경로를 구하는 문제이다[5]. 그림 1은 RPP의 하나의 해를 나타내고 있다. a-a', b-b', c-c', d-d', 그리고 e-e'은 E'에 속하는 에지들이다. 그리고 a'~b, b'~c, c'~d, d'~e, 그리고 e'~a는 아직 알려지지 않은 경로의 일부로서, 반드시 지나야 할 필요는 없지만 E'의 에지를 통과하기 위한 경로에 사용되는 중간 에지들을 나타낸다.

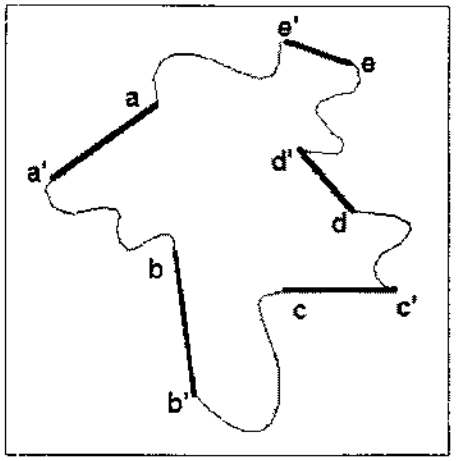

그림 1. RPP 그래프의 해

3. Rural Postman Problem을 위한 유전자 알고리즘

3.1 염색체 코딩(Coding)

3.1.1 기존의 코딩 방법

기존의 코딩 방법은 반드시 지나야 하는 에지($\in E'$)의 정보와 그 에지에 대한 방향 정보를 이용하여 염색체를 구성하였다[15]. 예를 들어, A, B, C, D($\in E'$)인 에지 정보에서, A(a, a '), B(b, b'), C(c, c'), 그리고 D(d, d')일 때, E'에 속한 에지에서의 방향을 0과 1로 표현하면, 하나의 염색체 구성은 다음과 같다.

$$P = \begin{bmatrix} A & C & B & D \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

즉, 방향 정보가 1이면 원래의 방향으로 라우팅이 이루어지고, 0이면 역방향으로 라우팅을 하게 된다. 위의 예에서 A의 방향 정보가 0이므로, 라우팅 방향은 a'에서 a로 라우팅이 이루어지고, C의 방향 정보는 1이므로, 라우팅 방향은 원래의 방향인 c에서 c'으로 라우팅이 이루어진다. 그리고 염색체 구성요소인 E' 원소들 간의 라우팅은 Dijkstra 알고리즘을 이용하여 최단 경로로 라우팅을 하게 된다. 이 방법에서는 염색체 구성이 E'의 정보와 방향 정보, 2가지의 서로 다른 형태로 구성되기 때문에 유전자 연산자들을 어떻게 적용하느냐에 따라 해의 수렴속도와 최적해의 값이 달라질 수 있다는 단점이 있다.

3.1.2 헤밀토니안 그래프 변환에 의한 코딩 방법

본 논문에서는 염색체를 구성하기 전에 E'($\subseteq$ E)을 갖는 RPP 그래프 G=(V, E)를 헤밀토니안 그래프 $G^H = (V^H, E^H)$로 변환하는 그래프 초기화 단계가 필요하다. 다음은 그 단계를 설명하고 있다.

1. If $e_i \in E'$, make $v_i^H \in V^H$.
2. For all $v_i^H, v_j^H \in V^H$ $(i \neq j)$, make edges$(v_i^H, v_j^H) \in E^H$.

위와같은 방법을 이용함으로써 그래프는 $|V^H|$개의 노드를 갖는 완전 연결 그래프가 되며, V^H내의 노드들 사이의 라우팅 비용은 RPP 그래프에서의 최소 라우팅 비용을 할당함으로써 헤밀토니안 그래프가 된다. 이때 변환된 그래프는 반드시 헤밀토니안 사이클을 가져야 한다. 그림 3은 그림 2를 헤밀토니안 그래프로 변환시켰을 때의 예를 나타내고 있다. 여기서, (a, a'), (b, b'), (c, c'), (d, d'), (d', e') 그리고 (e, e')은 E'에 속하는 에지들을 나타낸다. 그리고, 헤밀토니안 그래프에서의 노드 1, 2, 3, 4, 5, 6은 각각 에지(a, a'), (b, b '), (c, c'), (d, d'), (d', e'), (e, e')상의 임의의 점을 의미하며, 노드 1, 2, 3, 4, 5, 그리고 6 사이에는 가상 에지가 설정된다. 또한, 각 가상 에지 상의 라우팅 비용은 RPP 그래프에서 Dijkstra 알고리즘을 이용한 최단 경로의 비용을 할당하였다. 이러한 헤밀토니안 그래프에서의 노드와 가상 에지들은 디코딩 단계에서 원래의 RPP 그래프로 다시 변환하여 평가가 이루어져야 하기 때문에, 헤밀토니안 그래프를 원래의 RPP 그래프로 매핑하기 위한 정보를 가지고 있어야 한다. 즉, 헤밀토니안 그래프에서의 노드들 사이의 최단 경로, 헤밀토니안 그래프 노드와 RPP 그래프 노드 사이의 경로, 그리고 RPP 그래프 노드간의 경로들을 Dijkstra 알고리즘을 이용하여 라우팅 경로를 구하고, 이를 임시 테이블에 등록하여 디코딩시 이용하게 된다. V'을 E'에 사용되는 노드들의 집합이라고 하면, 표 1 ~ 표 3은 그림 2를 그림 3으로 변환할 때 생성된 테이블로서, 디코딩할 때 참조되는 테이블들이다.

이와 같은 방법으로 RPP 그래프를 헤밀토니안 그래프

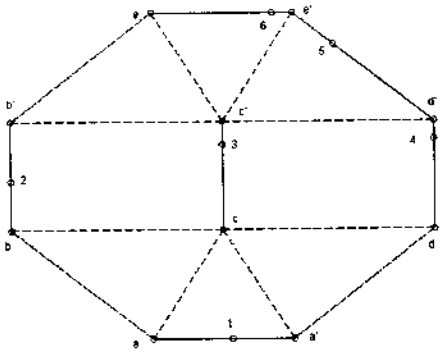

그림 2. RPP 그래프

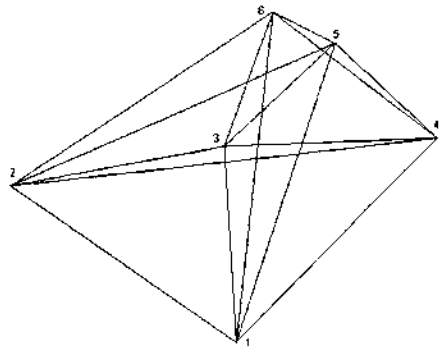

그림 3. RPP 그래프를 변환한 헤밀토니안 그래프

표 1. V^H에 포함된 노드간의 최단 경로 테이블

$i \in V^H$ \ $j \in V^H$	2	3	…	6
1				
2	↓			
…	$i, j \in V^H$ 사이의 최단 경로를 일련의 $v \in V$로 표현			
6				

표 2. V^H에 포함된 노드와 V'에 포함된 노드간의 최단 경로 테이블

$i \in V^H$ \ $j \in V'$	a	a′	b	b′	…	e
1						
2	↓					
…	$i \in V^H$, $j \in V'$ 사이의 최단 경로를 일련의 $v \in V$로 표현					
6						

로 변환시킨 후, 유전자 알고리즘의 염색체는 V^H에 속한 모든 노드들로 구성하고, 이 때 노드의 순서는 라우팅 순서를 나타내도록 한다. 따라서, 염색체의 크기는 $|V^H|$이 되고, 해 공간의 크기는$|V^H|!$이 되어 기존 염색체 구성보다 훨씬 작아진다. 또한, 기존 염색체 구성 방법이 염색체 자체에 에지의 방향 정보가 있기 때문에 디코딩 이전에 이미 라우팅이 결정되는 반면, 본 논문에서 제시한 염색체 구성 방법은 디코딩 단계에서 테이블들을 참조함으로써 비로소 라우팅 방향이 결정되도록 하여 염색체의 크기를 줄였기 때문에 해 공간의 크기를 줄이는 것이 가능해진다.

표 3. V에 포함된 노드간의 최단 경로 테이블

$i \in V$ \ $j \in V$	a	a′	b	b′	…	e′
a						
a′	↓					
…	$i, j \in V$ 사이의 최단 경로를 일련의 $v \in V$로 표현					
e′						

3.2 디코딩 (Decoding)

디코딩 (Decoding)은 염색체를 해석하여 해를 얻는 과정이다. 본 논문에서의 디코딩 과정에서는 헤밀토니안 그래프에서의 노드 정보를 에지 정보로 변환해야 한다. 이 과정에서는 그래프 초기화와 코딩 단계에서 생성된 테이블을 참조하게 된다. 그림 4는 노드 정보를 에지 정보로 변환하여 라우팅 경로로 디코딩하는 과정을 나타내고 있다.

예를 들어, 그림 2의 문제에서 생성된 염색체가 (1 2 6 4 3 5)이라고 가정하면, 1에서 2까지의 최단 경로를 테이블1을 참조하여(1과 2 사이의 최단경로가 a→b라 가정) RoutSet에 포함시키면, RoutSet={1-a-b-2}가 되고, RoutSet의 원소는 4개가 되기 때문에 count는 4가 된다.

```
Procedure Routing
    RoutSet ⇦ ∅
    s ⇦ CHROM(0)
    d ⇦ CHROM(1)
    RoutSet ⇦ (RoutSet → s → Table1(s,d) → d)
    td ⇦ RoutSet[1]
    for i from 2 to n-1
        count = |RoutSet|
    /* a : CHROM(i-1)을 포함하는 에지((a, a') ∈ E ')에 속하는 노드 */
        if RoutSet[count-1] is equal to a
            RoutSet ⇦ (RoutSet → a')
        else
            RoutSet ⇦ (RoutSet → a)
        count = |RoutSet|
        s ⇦ RoutSet[count]
        d ⇦ CHROM(i)
        RoutSet ⇦ (RoutSet → s → Table2(s, d) → d)
    endfor
    count ⇦ |RoutSet|
  /* a : CHROM(n)을 포함하는 에지((a, a') ∈ E ')에 속하는 노드 */
    if RoutSet[count-1] is equal to a
        RoutSet ⇦ (RoutSet → a')
    else
        RoutSet ⇦ (RoutSet → a)
    count = |RoutSet|
    s ⇦ RoutSet[count]
    d ⇦ td
    RoutSet ⇦ (RoutSet → s → Table3(s,d) → d → CHROM(0))
end procedure
```

그림 4. 디코딩 과정

RoutSet[count-1]이 b이므로 RoutSet에 b'을 포함시킨다. 즉, RoutSet={1-a-b-2-b'}, count=5가 된다. 다음은 b'과 6 사이의 최단 경로를 테이블2를 참조하여 RoutSet에 포함시키면(b'과 6 사이의 최단경로가 e를 경유한다고 가정), RoutSet={1-a-b-2-b'-e-6}, count=7이 되고, RoutSet[count-1]이 e이므로 RoutSet에 e'을 포함시켜서 RoutSet={1-a-b-2-b'-e-6-e'}, count=8이 된다. 이와 같은 방법으로 하여 염색체의 마지막 원소인 5까지 디코딩하면 RoutSet={1-a-b-2-b'-e-6-e'-d'-4-d-c-3-c'-e'-5-d'}, count=17이 된다. 마지막으로 처음 출발점인 1로 라우팅을 하기 위해, 염색체의 첫 번째 원소에 속하는 노드들(a, a') 중에 지나지 않은 노드(a')와 RoutSet[count]인 d'사이의 최단 경로를 테이블3을 참조하여(d'과 a'사이의 최단 경로가 d를 경유한다고 가정) RoutSet에 포함시키고 최종적으로 첫 번째 염색체를 포함시키면 RoutSet={1-a-b-2-b'-e-6-e'-d'-4-d-c-3-c'-e'-5-d'-d-a'-1}이 되어 RPP의 해를 구할 수 있다. 여기

서 1, 2, 3, 4, 5, 그리고 6은 E '의 에지 번호에 해당되며, a~e'은 RPP 그래프에서의 실제 노드가 된다. 그리고 E'을 구성하는 에지 번호의 앞과 뒤의 노드들은 그 E'을 통과하기 위한 시작 노드와 끝노드가 된다.

3.3 목적함수

디코딩이 이루어진 후에는 그 Population에 대한 평가가 이루어진다. 다음은 디코딩 후에 즉, 헤밀토니안 그래프를 실제 RPP 그래프로 변환한 후에 평가 함수의 값을 구하기 위한 RPP 문제의 파라미터와 목적함수를 나타내고 있다.

파라미터 :

$a_{e_i^2,e_{i+1}^1}$ $(e_i^2,e_{i+1}^1 \in V,\ e_i = (e_i^1,e_i^2) \in E',\ e_{i+1}=(e_{i+1}^1,e_{i+1}^2) \in E')$: i번째 에지($\in E'$)의 끝 노드로부터 $(i+1)$번째 에지($\in E'$)의 시작 노드까지의 최소 비용, 즉, 디코딩에서 생성된 라우팅에서 i번째 에지의 끝 노드와 $(i+1)$번째 에지의 시작 노드 사이에 경유되는 중간 경로의 비용.

c_{e_i} $(e_i \in E')$: i번째 에지의 비용, 즉, 디코딩에서 생성된 라우팅에서 E'에 속하는 i번째 에지의 시작노드와 끝 노드 사이의 비용.

C : 총 라우팅 비용.

목적함수 :

$$\min\ C = \sum_{i=1}^{n} (c_{e_i} + d_{e_i^2,e_{i+1}^1}),$$

여기서, n=|E'|이고 만일 i=n이면 i+1 = 1로 한다. 즉, 시작 노드에서 출발하여 시작 노드에서 종료한다.

Fitness 함수 :

$$F(C) = \frac{1}{C^m}.$$

여기서, m은 스케일링 인자(Scaling Factor)이며 자연수이다. 스케일링 인자는 염색체의 우성과 열성의 차이를 크게 함으로써 선택(Selection) 과정에서 우성인 염색체를 선택할 확률을 높게 한다.

3.4 유전자 연산자 (Genetic Operator)

3.4.1 PMX (Partially Matched Crossover)

교차 연산자(Crossover)는 실제로 선택된 스트링 한 쌍에 대한 실질적인 진화 조작이다. 교차 연산을 수행할 때 전형적인 교차 연산 방법을 본 문제에 그대로 사용하는 것은 의미가 없다. 왜냐하면 성질이 다른 염색체 간에 교차 연산을 허용할 경우 염색체 구조가 깨지기 때문이다. 따라서 본 문제에서는 이러한 점을 해결하기 위해서 PMX (Partially Matched Crossover) 방법을 사용하였다. PMX 방법에 대한 자세한 내용은 [6, 9, 11]에 설명되어 있다.

3.4.2 교환 (Exchange) 과 반전 (Inversion)

PMX 방식은 이전 세대의 선택된 두 개의 염색체에 대해 새로운 두 개의 염색체를 생성하는 반면, 교환과 반전은 선택된 하나의 염색체에 대해서만 적용된다. 본 논문에서는 교환과 반전 방법[6,9,11]을 돌연변이 연산자(Mutation)로 적용하였다. 즉, 새로운 세대의 염색체에 대해 적용함으로써 지나칠 수 있는 더 좋은 해를 찾을 수 있도록 해준다.

교환 방법은 선택된 한 염색체에 대해 두 노드를 택하여 교환하는 방식이다. 그리고, 반전 방법은 선택된 한 염색체에 대해 반전을 수행하기 위한 스트링을 선택하여, 그 스트링의 순서를 반전시키는 방법이다. 교환과 반전에 대한 자세한 내용은 [6, 9, 11]에 설명되어 있다.

4. 실험 결과

본 논문에서는 각 노드와 에지들을 임의로 구성한 서로 다른 8가지 문제에 대해 헤밀토니안 그래프 변환에 의해 염색체를 구성하여 유전자 알고리즘을 적용하였다. 시뮬레이션 환경은 IBM PC 펜티엄 75와 Microsoft Visual C^{++} 컴파일러를 이용하였다. 시뮬레이션에 사용된 문제들의 구성은 표 4와 같다.

유전자 알고리즘에 사용된 한 세대의 염색체 수는 100으로 하였으며, 최대 1000세대를 수행하였다. PMX를 위한 교차 연산률은 0.6으로 하고, 교환과 반전을 위한 돌연변이 연산률은 각각 0.03과 0.04로 하였으며, 현 세대에서 우성 염색체를 선택하는 방식은 Roulette Wheel 방

표 4. 사용된 문제들의 구성

문제	|V|	|E|	|E'|	Cost Matrix
1	10	18	10	임의의 값
2	15	23	7	유클리디안 거리
3	17	36	10	임의의 값
4	40	82	19	유클리디안 거리
5	50	63	17	유클리디안 거리
6	30	49	20	유클리디안 거리
7	79	128	20	유클리디안 거리
8	45	76	12	유클리디안 거리

식을 사용하였다.

표 5는 8가지의 RPP 실험 문제들에 대해 E'의 에지정보와 방향정보에 의한 염색체 구성과 헤밀토니안 그래프 변환에 의한 염색체 구성에 대해 PMX를 스케일링 인자(m) 값에 따라 각각 적용한 유전자 알고리즘의 근사 최적해의 결과를 비교한 것이다. 여기서 진하게 표시된 부분은 각 문제에 대해 가장 좋은 해를 표시한 것으로, 각 문제에 대해 헤밀토니안 그래프로 변환하여 염색체를 구성한 경우의 해가 전반적으로 좋은 결과를 얻고 있음을 알 수 있다.

표 6은 8가지의 RPP 실험 문제들에 대해 E'의 에지정보와 방향정보에 의한 염색체 구성과 헤밀토니안 그래프 변환에 의한 염색체 구성에 대해 PMX를 적용하였을 때, m을 1, 2, 3으로 각각 수행하여 그 중 가장 좋은 근사 최적해로 가장 빨리 진화한 결과를 비교한 것이다. 표에서 진하게 표시된 부분은 가장 좋은 해를 표시한 것이며, [문제 5]를 제외한 모든 문제에 대해 헤밀토니안 그래프로 변환하여 염색체를 구성한 경우가 E'의 에지정보와 방향정보에 의해 염색체를 구성한 경우보다 빨리 수렴하고 있음을 알 수 있다.

그림 5와 그림 6은 스케일링 인자(m)에 대해 모두 동일한 해를 갖는 [문제 1], [문제 2], [문제 3], 그리고 [문제 6] 중에, [문제 1]과 [문제 6]에 대해 스케일링 인자 m의 값에 따라 진화해 가는 과정을 나타내고 있다. 이 결과로부터 E'의 에지정보와 방향정보로 구성된 염색체를

표 5. PMX를 적용한 경우의 근사 최적해

문제	염색체 구성					
	E' + 에지의 방향정보			헤밀토니안 사이클의 노드		
	m = 1	m = 2	m = 3	m = 1	m = 2	m = 3
1	22	22	22	22	22	22
2	955.342	955.342	955.342	955.342	955.342	955.342
3	31	31	31	31	31	31
4	1635.005	1213.906	1198.598	1410.275	1176.672	1192.613
5	1200	920	1060	1000	1000	920
6	36	36	36	36	36	36
7	913.076	912.030	912.030	909.215	912.030	913.885
8	2174.264	2074.264	2024.264	2024.264	2024.264	2024.264

표 6. PMX를 적용한 근사최적해와 진화횟수 비교

문제	염색체 구성			
	E' + 에지의 방향정보		헤밀토니안 사이클의 노드	
	근사최적해	진화횟수	근사최적해	진화횟수
1	22	21	22	5
2	955.342	76	955.342	6
3	31	13	31	5
4	1198.598	768	1176.672	692
5	920	140	920	242
6	36	96	36	12
7	912.030	12	909.215	780
8	2024.264	368	2024.264	26

이용하여 유전자 알고리즘을 수행하는 경우에는 m의 값에 따라 근사 최적해로의 수렴에 큰 영향을 미친다. 즉, 스케일링 인자(m)가 적당하게 적용되면 수렴 속도가 빠르고 그렇지 않으면 수렴 속도가 느려진다. 따라서 스케일링 인자를 어떻게 설정하느냐가 해의 수렴에 중요한 요인이 된다. 반면, 헤밀토니안 그래프 변환에 의한 염색체를 이용하여 유전자 알고리즘을 수행한 경우에는 m의 값

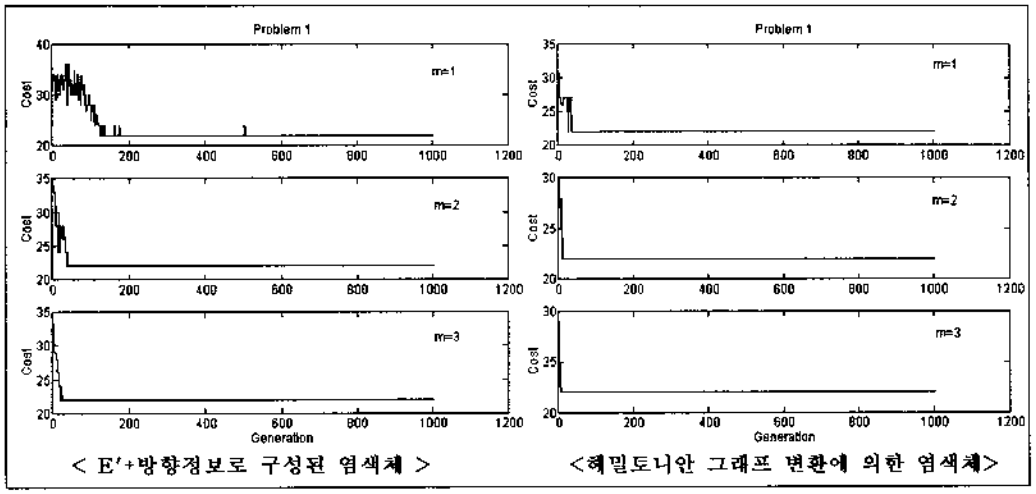

그림 5. 문제 1에서 m의 값에 따른 진화 과정 비교

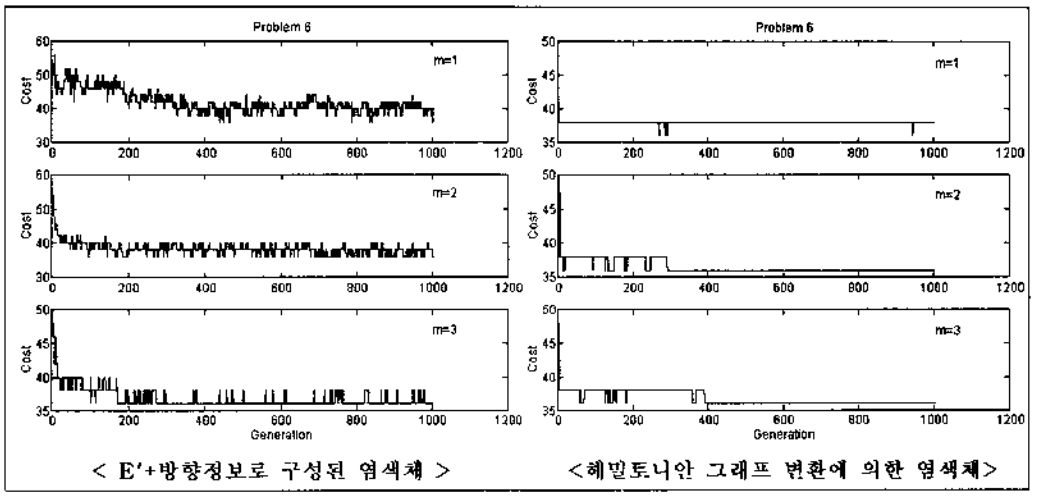

그림 6. 문제 6에서 m의 값에 따른 진화 과정 비교

에 따라 근사 최적해로의 수렴에 큰 영향이 없다는 것을 보여 주고 있다. 즉, 스케일링 인자(m)가 크고 작음에 관계없이 수렴 속도가 비슷하다는 것을 알 수 있다. 그리고 스케일링 인자가 같은 경우, 헤밀토니안 그래프 변환에 의한 염색체 구성 방법이 기존 방법에 비해 해의 수렴 속도가 빠르다는 것을 알 수 있다. 예를 들어, 그림 5와 그림 6에서 m=1인 경우, 혹은, m=2, m=3인 경우를 각각 헤밀토니안 그래프 변환에 의한 방법과 기존 방법과의 해의 수렴 속도를 비교해 보면 헤밀토니안 그래프 변환에 의한 방법이 기존 방법 보다 해의 수렴 속도가 빠르다는 것을 알 수 있다. 이는 E'의 에지정보와 방향정보를 그대로 염색체에 반영하는 경우에는 에지 정보와 에지의 방향성을 이용하기 때문에, 염색체의 크기가 2|E'|이 되어 해 공간의 크기는 $2^{|E'|} \times |E'|!$이 되는 반면, 헤밀토니안 그래프 변환에 의한 염색체 구성을 이용하여 유전자 알고리즘을 수행하는 경우에는 단지 헤밀토니안 그래프를 구성하는 노드 정보만을 이용하기 때문에, 염색체 크기가 |V|이 되어 해 공간의 크기는 $|V^H|!$이 된다. 따라서, 본

논문에서 적용한 염색체 구성 방법의 해 공간이 훨씬 작기 때문에 스케일링 인자(m)에 큰 영향이 없고, 또한 빨리 수렴한다는 것을 알 수 있다.

5. 결 론

본 논문에서는 유전자 알고리즘을 이용하여 NP-Complete 문제로 알려진 Rural Postman Problem (RPP)의 해법에 대해 살펴 보았다. 본 논문에서는 RPP의 그래프에서 특정 에지($E' \subseteq E$)를 헤밀토니안 그래프의 노드로 변환하여 염색체를 구성함으로써 기존의 염색체 구성 방법보다 염색체의 크기와 해 공간을 줄일 수 있었다. 또한, 해 공간이 작기 때문에 기존 방법보다 해의 수렴속도가 빠르고, 진화 과정에서 스케일링 인자(m)에 영향을 적게 받는다는 것을 알 수 있었다.

기존의 염색체 구성 방법은 E' 정보와 에지의 방향 정보를 이용하기 때문에 유전자 연산자를 어떻게 적용하느냐에 따라 근사 최적해 수렴에 영향을 주지만, 본 논문에서 제안한 염색체 구성은 단일 정보로만 구성되기 때문에 유전자 연산자를 일괄적으로 적용할 수 있다는 장점이 있다. 그러나, RPP 그래프를 헤밀토니안 그래프로 변환하는 과정에서 디코딩에 사용하기 위해 생성되는 추가적인 테이블들을 저장하기 위한 메모리가 필요하다는 단점이 있고, 또한, 디코딩 단계에서 헤밀토니안 그래프를 원래의 RPP 그래프로 변환하여 에지의 방향과 라우팅 순서를 결정하기 위한 과정이 필요하다는 단점이 있다.

향후 본 논문에서 제안한 방법을 응용하여 실제 문제에 적용이 가능하며, 또한 유향 그래프나 에지(edge)와 아크(arc)가 혼합된 그래프에 대해서도 적용될 수 있다.

참고문헌

[1] Bondy, J. A. and Murty, U. S. R., *Graph Theory with Applications*, Macmillan Press Ltd, 1977.

[2] Chambers, L. *Practical Handbook of Genetic Algorithms*, CRC Press, 1995.

[3] Fausett, L., *Fundamentals of Neural Networks Architectures, Algorithms, and Applications*, Prentice Hall, 1994.

[4] Freeman, J. A. and Skapura, D. M., *Neural Networks Algorithms, Applications, and Programming Techniques*, Addison-Wesley, 1991.

[5] Garey, M. R. and Johnson, D. S., *Computers and Intractability - A Guide to the Theory of NP-Completeness*, p. 213, FREEMAN, 1979.

[6] Goldberg, D. E., *Genetic Algorithm in Search, Optimization & Machine Learning*, Addison-Wesley, 1989.

[7] Goldberg, D. E. and Lingle, R., "Alleles, loci, and the Traveling Salesman Problem.", *Proceedings of an International Conference on Genetic Algorithms and Their Applications*, pp. 154-159, 1985.

[8] Haykin, S., *Neural Networks - A Comprehensive Foundation*, Macmillan Publishing Company, 1994.

[9] Ladd, S. R., *Genetic Algorithms in C++*, M&T Books, 1995.

[10] Mchugh, J. A., *Algorithmic Graph Theory*, Prentice Hall, 1990.

[11] Michalewicz Z., *Genetic Algorithms + Data Structures = Evolution Programs Second*, Extended Edition, Springer-Verlag, 1994.

[12] Pal, S. K. and Wang, P. P., *Genetic Algorithms for Pattern Recognition*, CRC Press, 1996.

[13] Srinivas, M. and Patnaik, L. M., "Genetic Algorithms: A Survey", *Computer*, pp. 17-26, June 1994.

[14] 김대수, *신경망 이론과 응용 (I)*, 하이테크정보, 1992.

[15] 이영훈, 강명주, 한치근, "Rural Postman 문제에서 근사해를 구하는 유전자 알고리즘", *정보과학회 논문지*, 23권, 11호, pp. 1118-1125, 1996.

[16] 최형진, 정영준, 양해술, *뉴로컴퓨팅*, 홍릉과학출판사, 1994.

97년 5월 최초 접수, 97년 9월 최종 수정