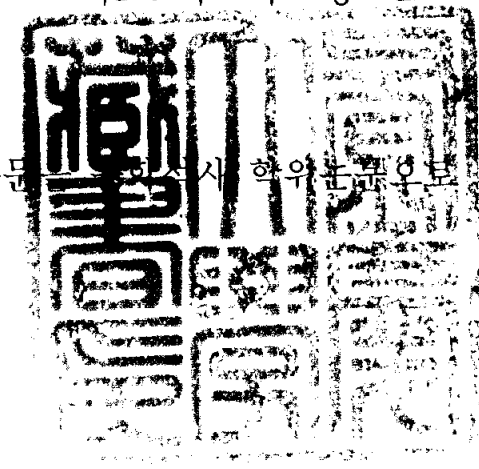
공학석사 학위논문

스타이너 트리를 구하기 위한 부동소수점 표현을 이용한 유전자 알고리즘

지도교수 우 종 호



이 논문을 공학석사 학위논문으로 제출함.

2004년 2월

부경대학교 대학원

컴퓨터공학과

김 채 주

김채주의 공학석사 학위논문을 인준함

2003년 12월 26일

주 심 공학박사 서 경 룡 

위 원 공학박사 권 오 흠 

위 원 공학박사 우 종 호 

목 차

그 림 목 차

표 목 차

Genetic Algorithm Using Floating Point Representation for Steiner Tree

Chae-Joo Kim

Dept. of Computer Eng., Graduate School,

Pukyong National University

Abstract

The problem for getting the optimal steiner tree from a given network is NP-hard, so the genetic algorithms occasionally have been used to take a near optimal solution. In this thesis, the chromosomes in genetic algorithm are represented as the floating point notation instead of the binary string as usual for solving this problem. First of all, a spanning tree is got from a given network using Prim's algorithm. Next, the new steiner point is computed using genetic algorithm with the chromosomes in the floating point notation and it is added to the tree for approaching the result. Repeating this step(evolving), finally the near optimal steiner tree is obtained. Using this method the tree is more quickly and exactly approached to a near optimal steiner one compared with the existing genetic algorithms using binary string.

Ⅰ. 서 론

네트워크의 최단경로 설정에 대한 문제는 오랫동안 연구되고 있는 분야이다. 이에 대한 연구는 최소비용트리(minimum spanning tree) 방식과 최소비용 스타이너 트리(minimal steiner tree) 방식을 중심으로 이루어졌다. Pawel Winter 등은 스타이너 트리 문제의 기하학적 특성을 이용하여 직선거리 문제의 최적해를 구하는 해법들을 제시하였다. 그러나 이 연구들에서는 주어진 계산 시간내에 무작위로 발생되는 해를 풀지 못하여 노드의 최대 크기를 30개 이하로 제한한다[1-4]. Chang[5]은 최소비용트리에 스타이너 점을 추가하여 거리를 줄이는 방법을 제안하였으나, 고정된 점의 수가 100개 이상이 될 경우 스타이너 트리는 SI(Steiner Insertions) 알고리즘에 의해 끊어진 최소트리를 적용하기 어렵다. David[6]는 유전자 알고리즘과 수학적 프로그래밍을 이용해서 스타이너 점을 찾는 방법을 제안하였으나, 스타이너 점이 증가할수록 교차연산이 길어진다. Joseph[7]은 주어진 그래프의 각 노드를 특정 비트로 사상시켜 이진염색체 형태로 나타내었다. 이진염색체는 이론적 분석을 용이하게 하고 특정 유전자의 성장과 쇠퇴를 예측할 수 있는 장점이 있으나, 정밀도를 높이고 해에 대한 탐색구간을 확대하면 염색체의 길이가 길어질 수 있다[8].

본 논문에서는 최소경로에 대한 거리의 정밀도를 높이고 Mark Johnson[9] 등의 연구에 적용된 거리기준 응용에 주로 사용되는 실수표현을 적용할 수 있는 유전자 알고리즘으로 구성하였다. 이 알고리즘은 최단거리 문제에 널리 이용되는 Prim 알고리즘을 사용하여 최소비용트리를 구성하고 스타이너 트리의 형태를 유지하기 위한 염색체를 실수형태로 표현한다. 유전연산자는 토너먼트 선택연산자와 노드좌표를 더 정확히 결정하기 위해 부동소수점을 사용하는 산

순교차연산자, 불필요한 점에 대한 보정작업을 위한 돌연변이연산자를 적용하였다. 그 결과 요구시간 내에 최적해 또는 최적해에 근접한 근사해를 구할 수 있었으며, 유전자 알고리즘을 노드간 거리에 따라 최소비용트리와 비교하여 최소 거리값을 평가하였다.

본 논문의 구성은 다음과 같다. 먼저 Ⅱ장에서 최단경로 기법과 유전자 알고리즘에 대해 기술하고, Ⅲ장에서는 이용된 유전자 알고리즘을 소개한다. Ⅳ장에서는 결과 및 고찰을 살펴보고, Ⅴ장에서 결론을 내린다.

II. 최단경로 문제와 유전자 알고리즘

1. 최단경로 문제

1.1 최소신장트리

신장트리란 주어진 네트워크의 모든 교점을 연결하는 트리로서, 사이클이 존재할 수 없으며 모든 노드는 하나의 간선으로 연결되어야 한다. 네트워크에 N개의 교점이 있으면 신장트리는 $N-1$개의 호로 구성된다.

최소신장트리란 무방향 네트워크의 각 호(i, j)에 비용 또는 길이를 나타내는 C_{ij}가 주어졌을 때, 신장트리에 포함된 모든 호의 비용의 합이 최소인 트리이다. 연결된 무방향 그래프 $G=(V, E)$의 간선 (u, v)마다 가중치 $w(u, v)$가 부여되어 있는 가중 그래프에 대하여

$$E' \subseteq E$$

$$w(E') = \sum_{(u,v)\in E'} w(u, v)$$

에서 최소인 트리 $G'=(V, E')$는 최소신장트리 또는 최소비용트리를 이루게 된다.

1.2 Prim 알고리즘

Prim 알고리즘은 임의의 한 교점에서 인접한 호 중 최소비용인 호를 선정한다. 이에 의해 두 개의 교점으로 구성된 트리가 형성되고, 이 두 교점에 인접한 비용의 호를 선정하여 모든 교점이 연결될 때까지 계속 반복하게 된다. 예를 들어, 가중치가 부여된 그래프 $G=(V, E)$에서 $V=\{1, 2, \cdots, n\}$으로 구성되어 있을 때 한 노드를 임의로 선택하고, 선택된 노드에서 비용이 가장 적은

```
begin
    T ← 0
    W ← 0
    E로부터 최소 비용인 간선 (v, w)를 선택
    while (T는 n-1개 이하의 간선을 포함 && E는 공집합이 아님) do
    begin
        E에서 간선 (v, w)를 제거
        if ((v, w)가 T내에서 사이클을 생성 안함) then
        begin
            T ← T ∪ {v, w}
            W ← W∪ {v, w}
        end
        else 간선 (v, w)를 버림
        E로부터 W내의 정점과 최소비용으로 연결된 간선 (v, w)를 선택
    end
end
```

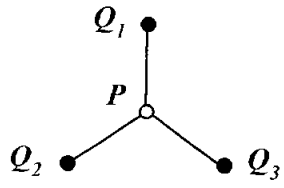
그림 2.1 Prim 알고리즘

노드 중 연결이 되지 않은 노드를 찾아서 이 두 노드를 연결시키게 된다. 그림 2.1은 Prim 알고리즘을 나타낸다.

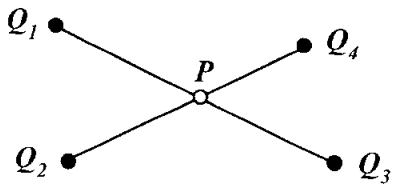
1.3 스타이너 트리 문제

스타이너 트리란 삼각형에서 거리의 합이 최소가 되게 하는 단일점을 찾는 문제이다. 이 단일점을 스타이너 점(steiner point)이라고 부른다. 그림 2.2(a)에서와 같이 스타이너 점은 세 개의 변이 정확하게 단일점 P에서 만나며, 그 각들은 각각 120°를 이룬다. 이 때 거리는 세 점 $Q_1Q_2Q_3$에 이르는 최소거리의 합이 된다. 그러나 세 점이 아닌 경우의 예로 그림 2.2(b)와 같이 나열되어 있

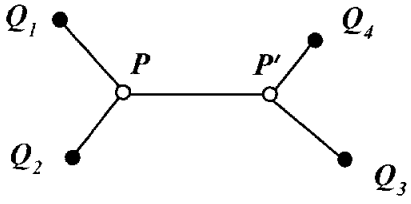
는 네 점에 대해서, 점 P는 사각형 $Q_1Q_2Q_3Q_4$의 대각선들의 교점이다. 단일점 P는 두 대각선의 교차점이지만 거리의 합을 최소로 하는 변의 길이를 구하는 데 불필요하다. 따라서 네 점 이상의 스타이너 트리 문제에서 거리를 최소로 하기 위해 단일점 P를 찾는 것은 포기하고, 대신에 그림 2.2(c)와 그림 2.2(d)와 같이 길이의 총합이 짧아질 수 있는 새로운 점을 추가하는 문제가 제안되었다[10].



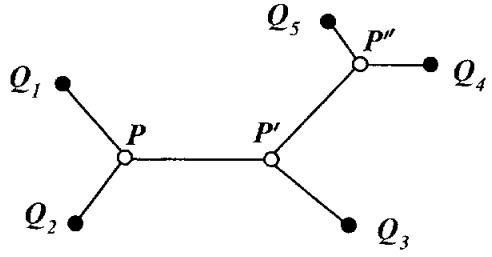
(a) 거리를 최소로 하는 단일점 P



(b) 네 점의 연결



(c) 새로운 점 P'의 추가



(d) 거리를 최소로 하는 점 P', P''의 추가

그림 2.2 네 점 이상에서 거리의 최소합

Ronald Graham 등은 스타이너 트리의 길이를 계산하는 문제가 NP-hard 임을 증명하였다[10]. 따라서 이 문제의 해를 구하기 위한 최적화 알고리즘을 찾아내는 것이 어렵고, 최적화 알고리즘이 존재해도 문제의 크기가 증가함에 따라 전체 최적점을 찾는 것은 높은 계산복잡도와 공간복잡도를 요구하여 실용성이 없게 된다.

2. 유전자 알고리즘

2.1 유전자 알고리즘의 개념

유전자 알고리즘(Genetic Algorithm : GA)이란 "적자생존"의 생물학 원리에 바탕을 둔 최적화 기법중의 하나이다. 이 알고리즘은 자연계의 생명체 중 환경에 잘 적응한 개체가 좀더 많은 자손을 남길 수 있다는 자연선택 과정과 생명체의 설계도와 같은 유전자의 변화를 통해서 좋은 방향으로 발전해 나간다는 자연진화의 과정을 모방하여 컴퓨터로 모의 수행을 하는 확률적 탐색 알고리즘의 하나이다. 그림 2.3은 유전자 알고리즘의 구조를 나타낸다.

```
begin
    t ← 0
    P(t)의 초기화(초기 모집단 생성)
    P(t)의 적응도 평가
while(종료조건이 만족되지 않으면) do
    begin
        t ← t+1
        P(t-1)로부터 P(t)를 선택
        P(t)의 유전연산(교차와 돌연변이)
        P(t)의 적응도 평가
    end
end
```

그림 2.3 유전자 알고리즘의 구조

여기서 $P(t)$는 세대 t에서의 모집단을 나타낸다. 모집단은 매 세대마다 일정수의 개체를 유지하고 각 개체의 적응도(fitness)를 평가하여 다음 세대에 생존할 개체들을 확률적으로 선택한다. 선택된 개체들 중 일부의 개체들이 임의

로 짝을 지어 교차함으로써 자손을 생성한다. 이 때 교차에 의해 부모의 유전자가 자손에게 상속되고 돌연변이가 발생할 수 있다. 자손은 부모로부터 좋은 유전형질을 상속받는다고 가정하여 다음 세대의 잠재해들이 평균적으로 전 세대 보다 더 좋아진다고 본다. 이러한 진화과정은 종료조건을 만족할 때까지 반복한다.

2.2 유전자 알고리즘의 연산자

유전자 알고리즘은 3개의 기본 연산자인 선택연산자(selection operator), 교차연산자(crossover operator), 돌연변이연산자(mutation operator)에 의해 수행된다.

(1) 선택연산자

선택연산은 환경에 대한 적응도에 의해 현 세대의 모집단으로부터 다음 세대에 생존할 개체를 선택하는 과정이다.

(2) 교차연산자

교차연산은 염색체로 표현된 해를 전역 최적점으로 수렴하도록 하기 위해 두 부모의 유전자 조합으로 자손을 생산하는 과정이다. 따라서 개체군에 존재하는 개체들이 정보를 서로 교환함으로써 해 공간내의 새로운 영역으로 탐색을 시도하게 된다.

(3) 돌연변이연산자

돌연변이연산은 교차연산과는 달리 두 개의 자손을 만드는 것이 아니라 하나의 자손을 부분적으로 조작하여 새로운 자손을 만들어내는 과정으로서 한 개체에서 아주 작은 수의 유전자를 임의로 변화시키는 과정이다.

2.3 관련 연구

Li, Bouchebaba[11]는 유전자 알고리즘에서 염색체와 세대의 개체를 트리로 표현하였다. 이 염색체를 트리에서 다루기 위해 경로교환에 사용되는 경로교차(path crossover), 서브트리 교환에 사용되는 트리교차(tree crossover)의 두 교차연산을 사용하였고, 완전그래프로부터 임의의 경로를 선택하기 위해 경로변이(path mutation)를 사용하였다. 이 알고리즘의 사용은 OCST(Optimal Communication Spanning Tree) 문제에 효과적인 최적의 통신비용트리를 구성하였으나, 목적해의 표현에 정수를 사용함으로써 보다 정밀한 목적해를 구하지 못했다.

Jorge Barreiros[12]는 스타이너 트리의 계산 최적화를 위해 계층구조 유전자 알고리즘을 제안하였다. 하위단계는 스타이너 트리 문제를 해결하기 위해 각 해를 스타이너 점들에 대응되도록 표현하였으며, 상위단계는 노드집합을 분할하여 작은 집합으로 표현하였다. 이 분할은 알고리즘이 수행되는 동안 동적으로 만들어진다. 이러한 방법은 좋은 결과와 효율적인 해를 구하는 대신에 극단적으로 민감한 작은 변화에도 해의 값이 바뀔 수 있다.

Z.Michalewicz[8]는 이진염색체를 실수 표현에 적용할 경우 발생할 수 있는 문제점에 대하여 기술하였다. 일반적으로 이진염색체의 표현은 실수값을 이진벡터로 변환하여 염색체에 표현한다. 예를 들어, 이진염색체를 실수값으로 변환하기 위해 소수점 이하 2자리의 정밀도와 변수의 범위가 [-5.12, 5.12]라고 하면, 100개의 변수에 대하여 실수구간은 10.24×100이 되어, 이 값에 대한 이진 해 벡터의 길이는 10이 된다. 만약, 소수점 이하 6자리의 정밀도가 요구되는 [-500, 500] 범위의 영역에서 100개의 변수에 대한 실수구간은 1000×10^6이며, 이 값에 대한 이진 해 벡터의 길이는 3000이 된다. 따라서 고정밀도를 요

구하는 수치문제에서 이진염색체를 사용하여 실수를 표현할 경우 이진염색체의 길이가 길어져 유전자 알고리즘이 잘 동작하지 않을 수 있다. 따라서 수치최적화를 위한 유전자 알고리즘에서는 부동소수점의 벡터로 염색체를 표현한다.

Jianhua Jiang[14]은 수치최적화 문제와 관련하여 세대수를 5개 사용한 micro-GA(μGA)에서 이진염색체와 실수염색체를 비교·실험하였다. 그림 2.4에서의 실험 조건은 변수가 100개이고, 변수의 범위는 [-5.12, 5.12]이다. 결과에서와 같이 이 문제에서 실수염색체가 이진염색체보다 훨씬 효과적임을 나타내고 있다.

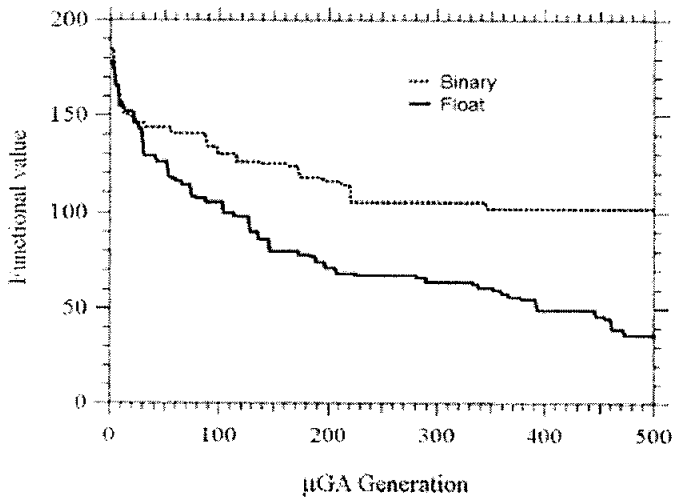


그림 2.4 100개의 변수에서 이진염색체와 실수염색체의 비교

그림 2.5는 실수염색체에 사용되어진 세 가지 형태의 교차연산에 대한 비교·실험을 나타내었다. 비교대상은 산술교차연산, 휴리스틱 연산, 그리고 둘을 조합한 연산이다. 결과에서 산술교차연산의 목적값이 적은 세대에서 빨리 수렴하여 가장 좋은 해를 구하는 것으로 나타난다.

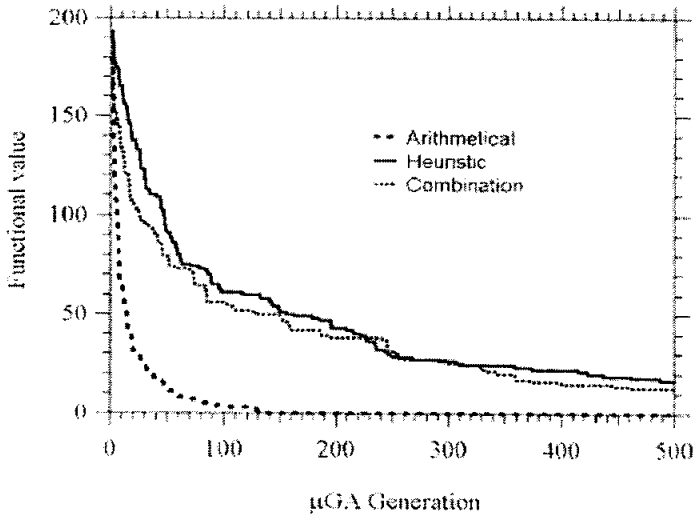


그림 2.5 실수염색체를 위한 교차연산의 비교

III. 유전자 알고리즘에서 부동소수점 표현

본 장에서는 유전자 알고리즘을 이용하여 노드상의(즉, 통신망의 연결이나 설비 배치 문제등) 최단거리를 구하기 위한 과정과 방법에 대하여 기술한다. 알고리즘에 입력된 자료를 대상으로 공간상의 좌표값을 직접 실수로 표현하여 염색체로 나타낸다. 이 염색체는 Prim 알고리즘을 사용하여 트리구조로 표현하고 부동소수점 표현의 산술교차연산을 사용한 유전자 알고리즘을 이용하여 최단거리를 구한다.

1. 유전자의 부동소수점 표현

지금까지의 유전자 알고리즘은 비트열을 중심으로 한 이진표현에 의존하고 있으며 이와 같은 일차원 표현은 해석하기 쉬운 이점이 있다. 실수로 표현된 노드를 유전자 알고리즘에서 다루기 위해 이진스트링 표현을 사용할 수 있으나, 해의 탐색공간이 커짐에 따라 실수 표현을 위해 변환되는 과정에서 이진 스트링의 길이가 길어질 수 있다.

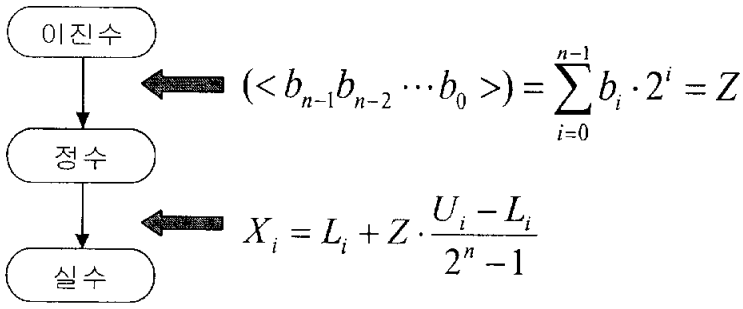


그림 3.1 이진수에서 실수로의 변환

또한, 그림 3.1과 같이 이진수를 정수로 변환하고 정수를 다시 실수로 변환해야 하는 코드화 및 디코드화의 기술을 필요로 한다[8].

가장 간단한 실수 표현 방법은 실수 하나를 하나의 유전자로 택하는 것이다. 따라서 인자들의 속성이 실수일 경우 그림 3.2에서 인자들과 유전자들이 일대일로 대응되어진다. 이러한 경우에 동일한 위치에 있는 두 부모의 유전자 값을 평균내어 자손의 동일한 위치 유전자 값으로 결정하는 산술교차연산을 사용한다.

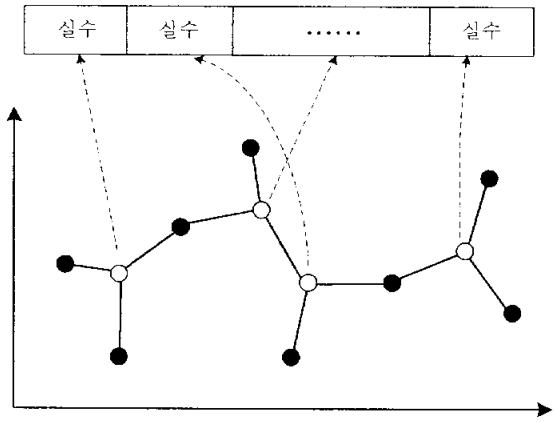


그림 3.2 실수와 일대일로 대응

본 논문에서는 노드 사이에 존재하는 선분의 거리를 보다 정확하게 나타내기 위하여 수치최적화에 이용되는 부동소수점 표현을 사용하였다.

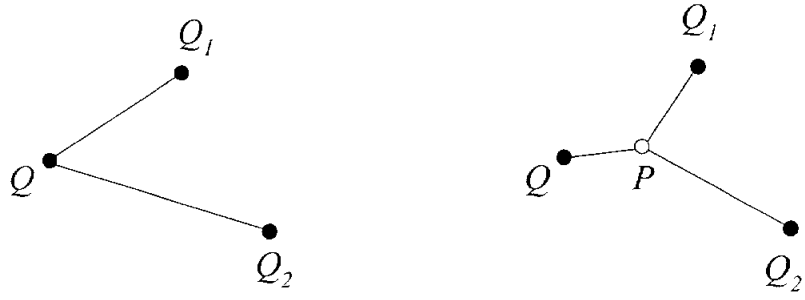
2. 스타이너 트리의 구성방법

스타이너 트리 문제는 스타이너 점이 증가함에 따라 다항시간에 계산하기 어려운 NP-hard 문제가 있다. 이에 대한 방안으로 Prim 알고리즘을 이용하여 스타이너 트리와 유사해지도록 트리를 구성한다.

먼저 주어진 점 $Q_1 \cdots Q_m$에 적당한 개수의 점 $P_1 \cdots P_n$을 추가하여 최소비용트리를 만든다. $P_1 \cdots P_n$을 조금씩 움직여 최소비용트리의 길이를 줄여가는 방법으로 스타이너 트리 형태를 만들어 간다.

점 P에서 만나고 있는 선분이 1.3절의 그림 2.2(b)와 같이 4개 이상이라면 점 P 부근에 단일점 P'를 새롭게 추가하여 그림 2.2(c)와 같이 P, P'에 각각 3개의 선분이 만나고 있는 것처럼 함으로써 선분의 길이를 줄여준다.

또 하나는 점 Q에서 2개 이상의 선분이 만나고 있고, 그 중 2개의 선분의 각이 120°이하라면 그림 3.3(a)에서 그 각의 안쪽에 그림 3.3(b)와 같이 새로운 점 P를 추가한다.



(a) Q에서 만나는 2개의 선분　　(b) 거리를 최소로 하는 점 P의 추가

그림 3.3　120°이하의 각에서 점 P를 추가

이와 같은 조작을 통해 추가할 점이나 삭제할 점이 없어질 때까지 반복한다. 이렇게 구성된 스타이너 트리를 개체로 표현하고 유전자 알고리즘을 이용하여 개체를 진화시킨다. 즉, 각 세대의 개체들은 스타이너 트리가 되고, 유전자 알고리즘을 적용하여 세대를 반복할 수록 이전 세대의 것보다 우수한 개체를 형성하여 나가는 방법을 이용한다. 이 과정을 효율적으로 수행하기 위해 스타이너 트리의 표현을 트리구조로 프로그래밍함으로써 알고리즘의 구현을

효율적으로 수행할 수 있도록 한다.

스타이너 트리 표현을 통해 Prim 알고리즘으로 구해진 최소비용트리 상에 초기 개체군을 구성하고 유전자 알고리즘의 선택연산자, 교차연산자, 돌연변이 연산자를 적용하여 후속세대를 반복적으로 생성한다. 주어진 노드에서 초기 개체군은 그 크기만큼 무작위로 만들어 낸다. 구현 시에는 실험의 용이성을 고려하여 사용자가 임의로 노드를 입력하여 트리구조를 생성할 수 있도록 설계하여 수행하였다.

3. 유전자 알고리즘의 적용

유전자 알고리즘의 적용은 노드 정보를 위한 기본 데이터를 무작위로 입력하고, 이 데이터를 기준으로 초기 개체군에 대한 하나의 실수를 인코딩함으로써 새로운 스타이너 점을 생성하게 된다. 새로운 스타이너 점의 생성은 Prim 알고리즘을 이용하여 최소비용트리를 순환하도록 하며, 이렇게 생성된 초기 개체군은 각 트리에서 선분길이에 대한 가중치로 계산하도록 한다. 선택연산자는 가중치에 의해 구해진 최적함수값을 이용함으로써 새로운 모집단을 만든다. 다음으로 선택연산에 의해 생성된 모집단을 부동소수점 표현을 하는 산술 교차연산자를 사용하여 새로운 자손들로 이루어진 새로운 모집단을 생성하게 된다. 마지막으로 돌연변이연산자를 수행한 후 적합도를 평가하고 새로운 개체군을 생성하도록 한다. 주어진 세대의 연산자 적용이 종료조건에 도달할 때까지 반복하면서 염색체를 진화시켜 최적해를 구한다.

본 논문에서 사용된 유전자 알고리즘의 적용 예를 그림 3.4의 수행과정을 통하여 나타내고, 각 절차를 제시하였다.

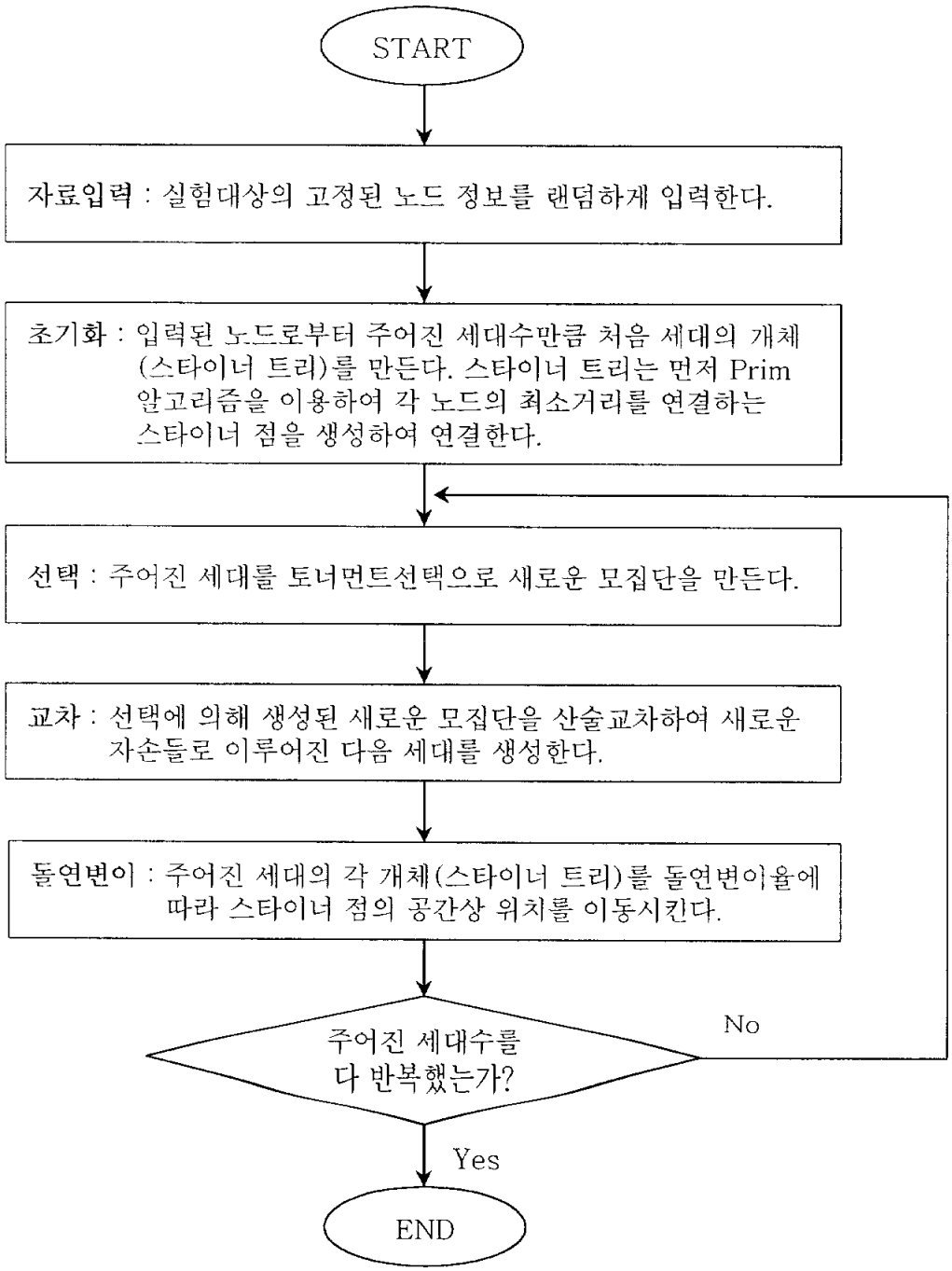


그림 3.4 구현된 프로그램의 수행과정

(1) 인코딩

입력된 데이터를 대상으로 실수로 표현된 공간상의 좌표값을 염색체로 구성하였다. 실수를 직접 염색체에 표현하기 위하여 유전자 하나에 하나의 실수값을 직접 표현하는 방법을 이용하여 염색체를 선택한다. 이 방법은 실수를 이진수로 표현하거나 이진수를 실수로 바꾸는 과정이 필요 없기 때문에 값을 변환하기 위한 오버헤드가 없는 장점이 있다.

염색체의 유전자와 해 벡터의 요소는 일대일로 일치하게 하였으며 염색체의 구성은 그림 3.5와 같이 스타이너 점의 좌표를 하나씩 지정하도록 하였다.

(x_1, y_1)	(x_2, y_2)	· · · ·	(x_n, y_n)

그림 3.5 염색체의 구성

이러한 방법으로 염색체를 표현함으로써 두 염색체의 일차조합으로 다음 세대 염색체를 생성하도록 하는 교차연산과 변이연산을 이용할 수 있다. 따라서 노드를 실수로 표현하는 응용에 적합하도록 하였다.

(2) 초기화

본 실험은 임의로 데이터를 입력하여 랜덤하게 만들어진 최소비용트리를 순환하게 함으로써 스타이너 트리 형태를 만들어 간다. 한 개체에서 스타이너 트리 형태를 만드는 방법은 먼저, 입력된 노드 정보로부터 주어진 개체군의 크기(population size)만큼 모든 노드에 이르는 최소비용트리를 랜덤하게 만든다. 스타이너 점들의 위치를 유전자 알고리즘의 개체로 표현하고, 최소비용트리에 의해 적합도를 판정하도록 한다. 모집단의 개체수이자 초기해의 수 P는

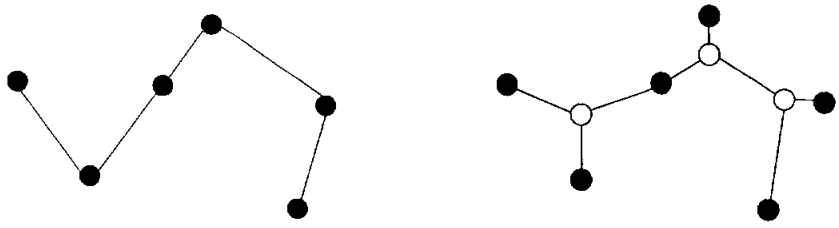
100개를 사용하였다. k 세대에서의 모집단 $P(k)$의 염색체는 다음과 같다.

$$P(k) = \{ S_1(k), S_2(k), \cdots, S_{100}(k) \}$$

여기서 $S(k)$는 탐색공간 상의 한 점 (x, y) 좌표값을 나타낸다.

다음으로 최소비용트리 상에 존재하는 인접한 두 쌍의 기본점들의 선분들에 새로운 점을 추가하여 얻어진 최소비용트리를 순환함으로써 스타이너 트리 형태를 유지하게 한다.

예를 들어, 입력된 노드에 대해 Prim 알고리즘을 이용하여 최소비용트리를 만들고, 최소비용트리를 순환하게 함으로써 스타이너 트리 형태를 유지하게 한다. 그림 3.6은 입력된 노드에 의해 만들어진 스타이너 트리 형태를 나타내고 있다.

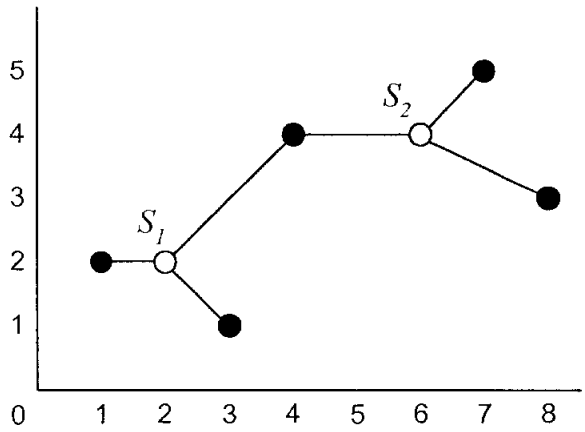


(a) 최소비용트리 (b) 최소비용트리의 순환

그림 3.6 랜덤하게 만들어진 스타이너 점

(3) 토너먼트선택

주어진 한 염색체의 가중치는 스타이너 점에 연결되어 있는 선분길이의 총합으로 계산된다. 선택되는 가중치는 스타이너 점과 각 노드들을 연결하기 위한 최적함수값을 구하기 위해 사용된다. 그림 3.7은 트리에서 계산된 가중치의 간단한 예를 보여준다.



S_1의 가중치 = 5.24, S_2의 가중치 = 5.65

그림 3.7 트리의 가중치

최단경로는 최소거리를 찾는 문제이므로 가중치가 작은 염색체일수록 적합함수(fitness function)값은 커야 한다. 이 때 적합함수값이 큰 염색체가 작은 염색체보다 선택될 확률이 높아지게 되며, 이렇게 구해진 적합함수값을 이용하여 선택연산을 수행한다.

본 논문의 유전자 알고리즘에서는 선택연산의 수행시간을 고려하여 수행시간이 짧고 선택압(selection pressure)으로 교차연산의 질을 조절할 수 있는 토너먼트선택을 이용한다. 이 때 수행시간의 단축을 우선시하여 토너먼트의 크기는 2로 제한하여 사용하였고, 다음의 절차를 따른다.

단계 1 : 토너먼트 크기 k(=2)를 결정한다.

단계 2 : 현재의 모집단에 있는 모든 개체를 임의의 순서로 나열한다.

단계 3 : 나열된 개체들 중에서 처음 k개의 개체와 비교하여 가장 좋은 개체를 생성시킨다.

단계 4 : 나열된 개체가 모두 비교되었으면 현재 모집단의 개체들을 새

로이 임의의 순서로 나열한다.

단계 5 : 모든 개체가 구해질 때까지 단계 3~4를 반복한다.

(4) 산술교차

교차연산은 염색체가 실수표현으로 인코딩되어 있고 자손염색체 값의 범위를 두 부모염색체가 표현하는 점으로 구축되는 공간상에 존재하게 제한할 필요가 있다. 따라서 일반적으로 실수표현에 사용되고 교차율로 범위를 한정지을 수 있는 교차연산이 필요하다. 본 논문에서는 두 부모(parent) P_1, P_2의 선형조합(linear combination)에 의해 두 자손(offspring) O_1과 O_2를 생산하는데 널리 사용되는 산술교차법을 이용한다. 그 과정은 다음과 같다.

단계 1 : 교차할 두 부모를 임의로 선택한다.

단계 2 : 각 부모에서 유전자를 교환할 트리를 임의로 선택한다.

단계 3 : 선택된 각각의 트리에서 선분의 위치를 근거로 교환하여 자손들을 구성한다.

단계 4 : 단계 1~3 과정을 교차율×세대수 만큼 반복 수행한다.

즉, 두 부모가 P_1, P_2 이고, 교차율을 α라고 할 때,

$$P_1 = (P_{11}, P_{12}, \ldots\ldots\ldots, P_{1m})$$

$$P_2 = (P_{21}, P_{22}, \ldots\ldots\ldots, P_{2n})$$

이 때 생산되는 자손은 다음과 같다.

$$O_1 = \alpha P_1 + (1 - \alpha) P_2$$

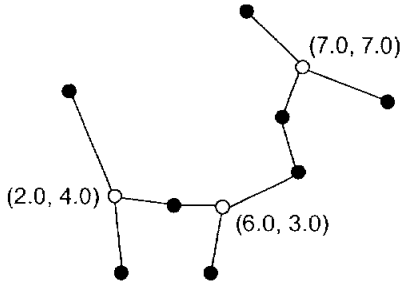
$$O_2 = (1 - \alpha) P_1 + \alpha P_2$$

여기서, 해들이 탐색영역에 계속 머물기 위해서 $\boldsymbol{\alpha} \in [0, 1]$을 만족하는 변수가 사용된다.

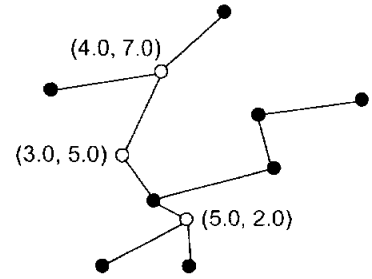
자손 모집단의 모든 개체에 대하여 *0*에서 *1*사이의 랜덤함수 값을 생성시켜 그 값이 산술교차 파라메터 값보다 작은 개체들을 선택하고 선택된 개체들에 대하여 산술교차를 수행한다.

예를 들어, 선택연산에 의해 생성된 새로운 모집단을 새로운 자손들로 이루어진 다음 세대로 만들기 위해 다음과 같이 염색체가 세 점의 (x, y) 좌표로 구성되고, 교차율 $\alpha = 0.3$ 일 때 두 부모 P_1, P_2에서 생산된 자손 O_1, O_2는 다음과 같다.

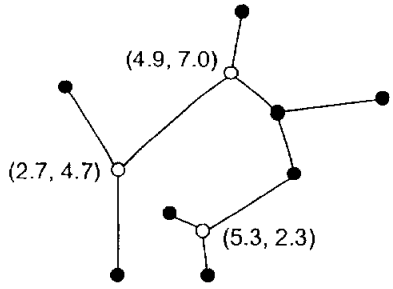
$$P_1 = ((2.0, 4.0), (6.0, 3.0), (7.0, 7.0))$$
$$\updownarrow \quad \updownarrow \quad \updownarrow$$
$$P_2 = ((3.0, 5.0), (5.0, 2.0), (4.0, 7.0))$$



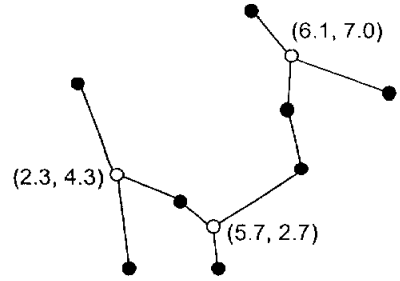
(a) 교차연산 전 부모 1



(b) 교차연산 전 부모 2



(c) 교차연산 후 자손 1



(d) 교차연산 후 자손 2

그림 3.8 산술교차연산

$O_1 = 0.3P_1 + 0.7P_2 = ((2.7, 4.7), (5.3, 2.3), (4.9, 7.0))$

$O_2 = 0.7P_1 + 0.3P_2 = ((2.3, 4.3), (5.7, 2.7), (6.1, 7.0))$

그림 3.8은 산술교차연산의 수행으로 두 부모 P_1, P_2에서 생산된 자손 O_1, O_2를 보여준다.

여기서 한 가지 고려되어야 할 점은 그림 3.8의 예와는 달리 교차하려는 두 부모의 스타이너 점 수가 다를 경우를 생각해야 한다. 다음과 같이 두 부모 P_1, P_2가 주어졌을 때,

$P_1 = ((2.0, 4.0), (6.0, 3.0), (7.0, 7.0))$

$P_2 = ((3.0, 5.0), (5.0, 2.0))$

P_1은 P_2보다 스타이너 점을 많이 가지고 있다. 이 경우에는 모든 스타이너 점들이 선택될 수 있는 기회를 가질 수 있도록 스타이너 점을 많이 포함한 부모를 우선 랜덤하게 재배치한다.

$P_1 = ((2.0,4.0), (6.0,3.0), (7.0,7.0)) \Rightarrow P_1 = ((6.0,3.0), (7.0,7.0), (2.0,4.0))$

다음으로 P_1과 P_2를 산술교차한다.

$$P_1 = ((6.0,3.0), \quad (7.0,7.0), \quad (2.0,4.0))$$
$$\updownarrow \qquad\qquad \updownarrow$$
$$P_2 = ((3.0, 5.0), (5.0, 2.0))$$

산술교차연산에 의하여 생산된 자손 O_1, O_2는 다음과 같다.

$O_1 = 0.3P_1 + 0.7P_2 = ((3.9, 4.4), (5.6, 3.5), (2.0, 4.0))$

$O_2 = 0.7P_1 + 0.3P_2 = ((5.1, 3.6), (6.4, 5.5))$

이러한 방법으로 교차하려는 두 부모의 스타이너 점 수에 관계없이 교차연산을 수행하여 새로운 자손을 생산하게 된다. 그리고 본문에서 제시한 예와는 달리 실제 프로그램 상에서 교차점의 좌표는 거리의 오차를 줄이기 위하여 공간상에 부동소수점으로 표현하였다.

(5) 돌연변이연산 및 보정

돌연변이연산은 스타이너 점의 공간상 위치를 이동시키기 위해 사용된다. 선택된 개체의 스타이너 점의 좌표를 자신의 좌표값에서 돌연변이율 10% 이하의 변화를 준 좌표와 대체하도록 한다. 즉, x_i, y_i 좌표를 변화시키면 $x_i+\alpha$, $y_i+\beta$의 범위에서 랜덤함수에 의해 구해진다. 여기서, α는 $-0.1\times x_i \le \alpha \le 0.1\times x_i$ 이고, β는 $-0.1\times y_i \le \beta \le 0.1\times y_i$ 이다.

돌연변이연산을 적용함으로써 보다 다양한 해를 발생시켜 좋은 해의 값은 상속시키고 나쁜 해를 제거하여 지역 최적해에서 벗어날 수 있다. 돌연변이는 해 벡터의 한 좌표값만을 변화시킨다. 만약 해 벡터가 $x=(x_1, \cdots, x_k, \cdots, x_n)$이라면, 돌연변이 후 $x=(x_1, \cdots, x_k', \cdots, x_n)$가 된다. 여기서 x_k'는 x_k에서 임의로 선택되어진 값이 된다.

그림 3.9에서와 같이 연결점 중 노드사이의 연결거리를 줄이기 위해 돌연변이연산이 임의로 S_1을 선택하게 되면, S_1에 해당하는 스타이너 점의 위치는 돌연변이율과 곱해져서 보다 가까운 스타이너 점인 S_2로 이동하게 된다. 이렇게 생성된 자손염색체의 적합도가 부모염색체보다 나은 경우 그 값을 상속시켜서 진화하게 된다.

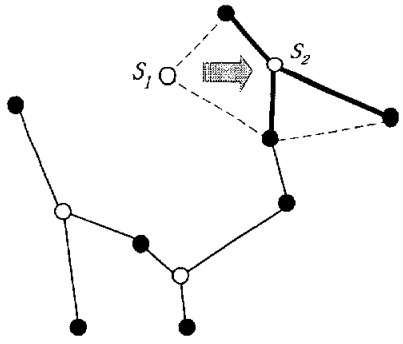


그림 3.9 돌연변이연산 후의 위치이동

Ⅳ. 결과 및 고찰

유전자 알고리즘을 수행하기 위한 교차율과 돌연변이율의 적정한 파라메터를 선택하도록 하였다. 노드간의 길이에 대해 교차율과 돌연변이율을 각각 변경하면서 모의실험을 수행하였다. 교차율의 변화는 0.1~0.5, 돌연변이율의 변화는 0.05~0.2 사이의 범위에 대하여 표 4.1과 같이 10개의 노드에 대해 최소비용트리의 거리값과 비교하여 유전자 알고리즘의 적정 파라메터를 구하도록 하였다. 실험결과 교차율은 0.3의 범위에서, 돌연변이율은 0.1의 범위에서 좋은 결과가 나왔다.

표 4.1 10개 노드의 거리값에 대한 적정 파라메타 비교·분석

교차율	돌연변이율	유전자 알고리즘	최소비용트리	평균절약값(%)
0.1	0.05	895.23	920.63	2.76
0.3	0.05	891.71		3.14
0.5	0.05	890.27		3.30
0.1	0.1	893.83		2.91
0.3	0.1	888.22		3.52
0.5	0.1	891.82		3.13
0.1	0.2	892.48		3.06
0.3	0.2	890.54		3.30
0.5	0.2	894.26		2.86

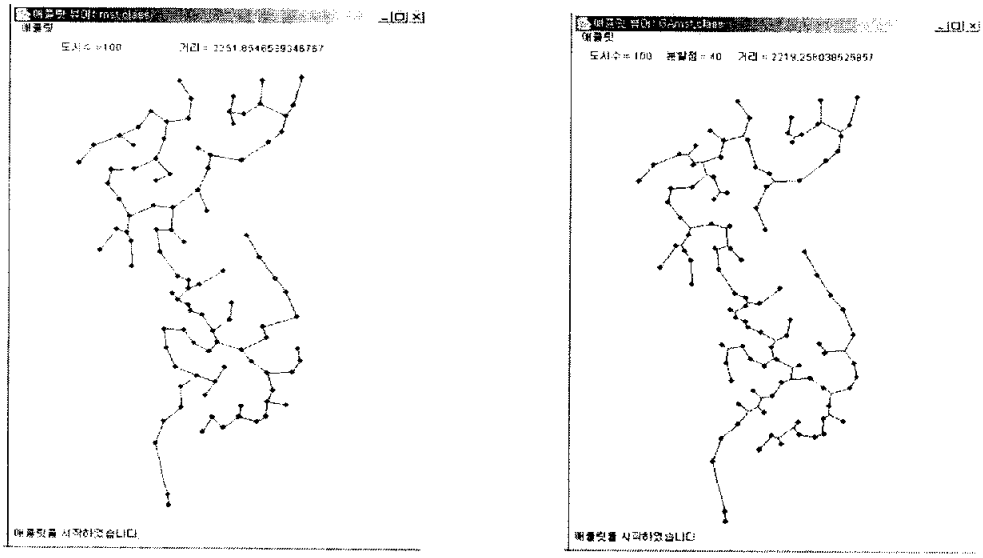
최소비용트리와 본 논문의 유전자 알고리즘의 성능을 평가하기 위해 자바 애플릿을 이용하여 모의실험을 수행하였다. 구현된 프로그램의 좌표상에 무작위로 점들을 생성하는 방법을 이용하였다. 알고리즘의 성능을 측정한 환경은

표 4.2와 같다.

표 4.2 성능 측정 환경

인자	적용
세대수	100 세대
교차율 (P_c)	*0.3*
돌연변이율 (P_m)	*0.1*
시스템 (CPU, Memory)	AMD 1600+, 256MB
운영체제 (OS)	Windows 2000
프로그래밍 언어	JDK 1.4.1

그림 4.1은 우리나라 100개의 도시를 Prim 알고리즘과 본 논문에서 사용된 유전자 알고리즘을 이용하여 최소거리로 연결한 결과이다.



(a) 최소비용트리 (b) 유전자 알고리즘

그림 4.1 100개의 도시를 연결한 결과

그림 4.1(a)의 최소비용트리는 Prim 알고리즘에 의해 구성되었다. 그림

4.1(b)는 100개의 도시를 Prim 알고리즘을 이용해 구한 후, 스타이너 트리와 유사해지도록 변형시키고 유전자 알고리즘을 적용한 결과를 나타낸다. 총 거리값은 부동소수점을 사용하여 소수점 12자리까지 표현함으로써 실생활에 사용될 경우 보다 정확한 거리를 표시할 수 있도록 하였다.

표 4.3에서는 그림 4.1에서와 같은 방법으로 최소비용트리와 본 논문의 유전자 알고리즘을 100개의 노드에서 수렴된 결과를 나타내었다. 프로그램의 특성상 입력의 정확성이 떨어지는 경우가 나타날 수도 있으므로 각각 10회 실험하여 거리값을 비교하였다. 최소비용트리로 연결된 노드의 선 분들보다 유전자 알고리즘으로 연결된 선분들의 거리값이 평균 2~3%정도 절약되었다. 그림 4.2는 100개의 노드를 각 실험횟수에 따라 비교한 그래프로 모두 거리가 줄어든 것을 알 수 있다.

표 4.3 100개 노드에서 거리에 따른 수렴 결과

횟수	최소비용트리	본 논문의 유전자 알고리즘
1	2253.80	2206.73
2	2254.16	2199.88
3	2256.17	2197.11
4	2247.65	2204.31
5	2262.77	2207.64
6	2252.90	2201.10
7	2248.68	2185.71
8	2251.38	2201.97
9	2246.15	2198.40
10	2260.23	2205.39

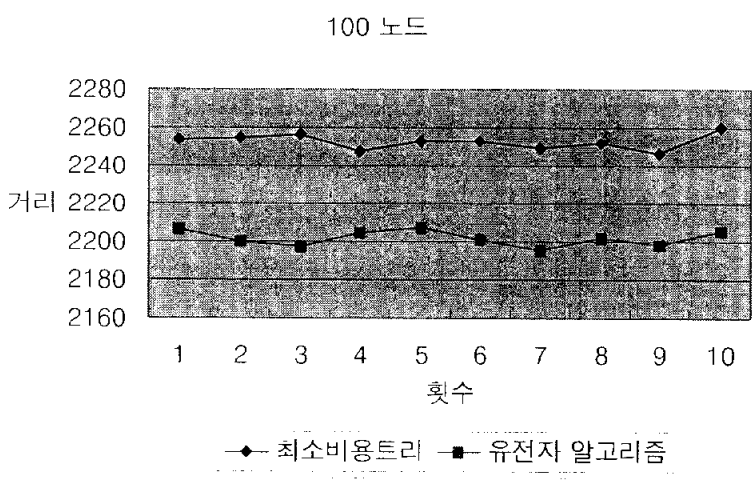


그림 4.2 유전자 알고리즘 구현 결과

표 4.4와 그림 4.3은 각 노드 수에 따른 최소거리를 본 논문의 유전자 알고리즘과 최소비용트리를 비교·실험하여, 연결된 각 노드 수에 대한 거리비(%)

표 4.4 연결 노드 수에 따른 결과

노드수	최소비용트리	본 논문의 유전자 알고리즘	거리절감율 (%)
10	920.63	888.22	3.52
20	1117.89	1095.10	2.04
30	1325.91	1280.84	3.40
40	1513.59	1448.80	4.29
50	1708.47	1640.75	3.97
60	1805.29	1744.16	3.39
70	1958.97	1890.03	3.52
80	2085.60	2021.35	3.09
90	2170.36	2106.95	2.93
100	2252.39	2201.82	2.32

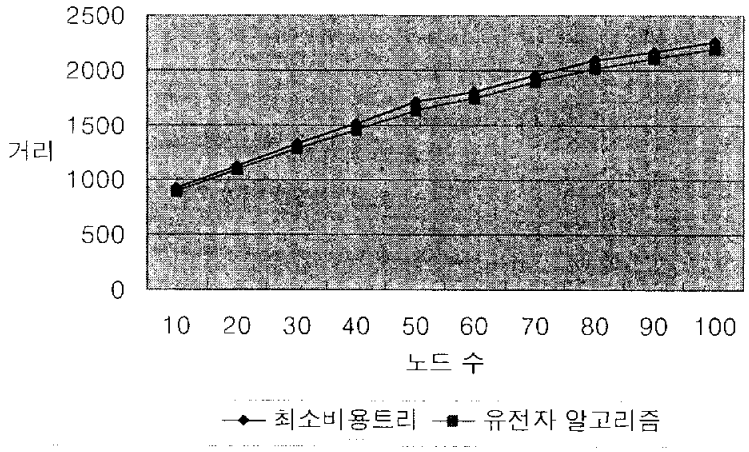


그림 4.3 각 노드별 수행결과

를 나타내었다. 결과를 통해서 본 논문의 유전자 알고리즘을 이용하여 각 노드를 연결하면 최소비용트리를 이용한 것보다 2~4.3%의 거리단축 효과를 기대할 수 있다.

표 4.5는 Joseph에 의해 제안된 스타이너 최소트리 문제를 위한 유전자 알고리즘과 본 논문의 유전자 알고리즘을 비교한 것이다. 실험방법은 최소비용트리의 거리를 100%로 두고 비교·분석하였다.

표 4.5 최소거리의 비교

	16개의 Grid 노드	100개의 랜덤노드
스타이너 최소트리[7]	91 %	97%
Joseph의 유전자 알고리즘[7]	95 %	98.8 %
본 논문의 유전자 알고리즘	94.3 %	97.7 %

Joseph의 유전자 알고리즘에서는 트리의 각 노드를 이진스트링 형태로 표현하고, 룰렛 휠 선택방법과 이점교차를 사용하였고, 본 논문에서는 트리의 좌표값을 그대로 이용할 수 있도록 부동소수점 표현방법을 사용하는 산술교차를 이용하였다. 성능비교는 컴퓨터 환경 등이 서로 다른 상태에서 시행되었고, Joseph의 논문에서 제시된 결과를 나타내었다. 본 논문의 유전자 알고리즘은 Joseph의 유전자 알고리즘보다 0.7~1.1% 정도 거리를 줄이는 것으로 나타났다. 스타이너 최소트리와의 비교에서는 0.7~3.3% 정도 거리가 길게 나타났다. 그러나 스타이너 최소트리를 구하기 위해서는 일반적으로 시간복잡도가 커지는 단점이 있어 실용성에 대한 문제를 가지고 있다.

V. 결 론

네트워크망에서 최단거리를 구하기 위한 문제는 스타이너 트리를 이용할 수 있다. 그러나 스타이너 트리는 NP-hard로 알려져 있어 그 해를 구하기 위한 실용성에 대한 문제가 제기되고 있다. 따라서 유전자 알고리즘을 이용하여 요구시간 내에 최적해 또는 최적해에 근접한 해를 구할 수 있도록 하였다. 유전자 알고리즘을 적용하여 해를 구하는 경우, 고전적인 유전자 알고리즘에서 사용되는 이진스트링 형태의 염색체와 교차연산은 탐색공간이 커짐에 따라 실수 표현을 위한 이진스트링의 길이가 길어질 수 있고, 코드화 및 디코드화의 기술이 요구된다.

본 논문에서는 Prim 알고리즘으로 최소비용트리의 선분을 분할하고 트리 형태의 구조를 유지하기 위해 부동소수점 표현에 의한 유전자 알고리즘을 사용하였다. 공간상의 좌표를 트리형식으로 나타내고, 직접 염색체에 표현하기 위하여 부동소수점 표현법을 선택하여 스타이너 트리에 근접하도록 하였다.

성능비교를 위해 자바 애플릿을 이용하여 모의실험을 수행하였다. 실험은 이진스트링 염색체를 사용한 Joseph의 유전자 알고리즘과 최소거리를 비교하였다. 그 결과 이진스트링으로 표현된 염색체보다 부동소수점 표현의 유전자 알고리즘이 0.7~1.1%의 거리를 단축시킬 수 있었고, 최소비용트리와 비교하여 노드 사이의 총 거리가 2~4.3% 절감되는 것을 확인하였다. 또한 트리형식의 염색체 표현과 부동소수점 표현으로 교차하는 산술교차법을 사용한 접근이 최적해에 더 근접함을 알 수 있었다. 따라서 부동소수점 표현을 이용하면 보다 정확한 실제 네트워크망의 연결거리를 계산하는 데 효율적이다.

참 고 문 헌

[1] G. R. Raidl and B. A. Julstrom, "Edge-Sets: An Effective Evolutionary Coding of Spanning Trees", Technishe Univ., Technical Report, 2002.

[2] Pawel Winter and Martin Zachariasen, "Euclidean Steiner Minimum Trees: An Improved Exact Algorithm", Networks, Vol.30, pp.149-166, 1997.

[3] Pawel Winter, "An algorithm for the Steiner problem in the Euclidean plan", Networks, Vol.15, pp.323-345, 1985.

[4] Cockayne, and D. E. Hewgill, "Improved Computation of Plane Steiner Minimal Tree", Algorithmica, Vol.7, pp.219-229, 1992.

[5] Shi-Kuo Chang, "The Generation of Minimal Trees with a Steiner Topology", Journal of the ACM, Vol.19, pp.699-711, 1972.

[6] David J. Thuente and Pulin Sampat, "Mathematical Programming in A Hybrid Genetic Algorithm for Steiner point Problems", Proc. of the ACM symposium, 1995.

[7] Joseph Jones, Frederick C and Harris. Jr, "A Genetic Algorithm for the Steiner Minimal Tree Problem", ISCA's Int. Conf. on Intelligent Systems, 1996.

[8] Zbigniew Michalewicz, *Genetic Algorithms+Data Structures=Evolution Programs*, Third Extended Edition, Springer-Verlag, 1995.

[9] Mark Johnson, David Herold and Josko Catipovic, "The Design and Performance of a Compact Underwater Acoustic Network Node", IEEE

Proceeding, Vol.3, pp.13-16, 1994.

[10] Richard Courant and Herbert Robbins, *What is Mathematics?*, Second Edition, Oxford University Press, 1996.

[11] Yu Li and Youcef Bouchebaba, "A New Genetic Algorithm for the Optimal Communication Spanning Tree Problem", Proc. LNCS. Vol.1829, pp.162-173, 1999.

[12] Jorge Barreiros, "An Hierarchic Genetic Algorithm for Computing (near)Optimal Euclidean Steiner Trees", GECCO2003, pp.56-65, 2003.

[13] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, 1989.

[14] Jianhua Jiang, "Rigorous Analysis and Design of Diffractive Optical Elements", A Dissertation, Univ. Alabama, pp.74-89, 2000.

감사의 글

본 논문이 이루어지기까지 세심한 배려와 지도로 이끌어주시며 학문의 길을 가르쳐주신 우종호 교수님께 한없는 감사를 드립니다. 그리고 본 논문을 심사해 주시고 부족한 부분을 보완해주신 서경룡 교수님, 권오흠 교수님께 진심으로 감사드립니다. 또한 학위과정 동안 많은 가르침을 주셨던 학과의 여러 교수님들께도 감사드립니다.

힘겨운 대학원 생활동안 연구해야 할 방향을 제시하고 힘이 되어준 수진씨, 봉기씨에게 감사드립니다. 힘든 시간동안 웃음으로 생활할 수 있게 해준 기호, 항상 듬직하던 형상이, 힘들어도 웃음으로 대해주던 석주, 지친 어깨를 기댈 수 있도록 힘이 되어준 창수형, 희숙누나에게도 감사의 마음을 전합니다. 2년 동안 어려움을 함께 했던 동기들 환조, 성주, 윤희에게도 감사드립니다. 그리고 모자란 저를 잘 따라준 1년차 후배들 종우, 정철, 경헌, 성록, 병길, 정식, 남진이와 다정다감하던 이향씨, 그리고 연구실에서 궂은 일 마다하지 않던 동생들 철범, 경화, 락기, 승호, 성욱, 보랑이에게도 고마운 마음 전합니다.

오늘 이 시간이 있기까지 항상 가슴졸이며 걱정하시던 어머님, 부족한 사위 믿음으로 지켜봐 주시던 장인·장모님께 오늘의 영광을 돌리고 싶습니다. 보이지 않는 곳에서 힘이 되어 준 형, 누나들과 동서, 처제, 처남에게도 감사의 마음을 전합니다. 끝으로 지친 몸을 희망과 용기로 북돋워 주던 이제 태어날 아기와 힘들고 어려운 일이 있어도 얼굴 한 번 찌푸리지 않고 못난 남편 묵묵히 지켜온 사랑하는 아내 성민에게 이 지면을 빌어 감사와 사랑한다는 말 전하며 이 논문의 결실을 함께 나누고 싶습니다.

2004년 1월 김 채 주 드림.