IE Interfaces
Vol. 15, No. 3, pp. 241-246, September 2002.

Minimum Spanning Tree 응용 문제에 대한 유전연산의 개선

고시근† · 김병남

부경대학교 산업공학과

Improvement of Genetic Operations for Minimum Spanning Tree Application Problems

Shie-Gheun Koh · Byung-Nam Kim

Department of Industrial Engineering, Pukyong National University, Busan, 608-739

Some extensions of minimum spanning tree problem are NP-hard problems in which polynomial-time solutions for them do not exist. Because of their complexity, recently some researchers have used the genetic algorithms to solve them. In genetic algorithm approach the Prufer number is usually used to represent a tree. In this paper we discuss the problem of the Prufer number encoding method and propose an improved genetic operation. Using a numerical comparison we demonstrate the excellence of the proposed method.

Keywords: minimum spanning tree, genetic algorithm, prufer number

1. 서 론

Minimum Spanning Tree(이하 MST) 문제는 대표적인 combinatorial 문제로서 1926년 Boruvka가 전력선 네트워크의 구성을 위해 처음으로 정식화하였다. 그 후 MST 문제는 수송 및 배송문제, 통신 네트워크 설계문제 등과 같은 여러 분야에서 응용되어져 왔으며 몇 개의 간단한 해법도 개발되어졌다(Graham and Hell, 1985).

나아가 기존의 MST 문제에 대한 여러 가지 변형된 형태도 연구되었는데, Narula and Ho(1980)는 한 node에 연결된 arc의 수가 제한된 경우의 MST 문제를 연구하였고, Bertsimas(1990) and Shiode *et al.*(1981)은 확률적 MST 문제를, Gilbert(1967)는 Minimum Steiner Tree 문제를, 그리고 Myung *et al.*(1995)은 Generalized Minimum Spanning Tree 문제를 연구하였다. 이러한 변형 MST 문제들은 일반적으로 NP-hard의 성질을 가지고 있어 최적해를 구하는 것은 문제의 크기가 커질 경우 거의 불가능해진다. 따라서 이러한 문제를 해결하기 위해서는 최적해의 보장은 없지만 비교적 좋은 해를 빠른 시간에 구할 수 있는 여러 가지 휴리스틱 방법들, 그 중에서도 특히 유전 알고리듬과 같은 메타 휴리스틱 기법이 많이 사용되고 있다.

인공지능의 한 기법인 유전알고리듬은 생태계의 적자생존 및 유전법칙에 그 원리를 둔, 최적해에 대한 개선탐색 알고리듬으로서 일반적인 휴리스틱 기법에 비해 다음과 같은 여러 가지의 장점을 가진다고 알려져 있다(Goldberg, 1989). 첫째, 휴리스틱 기법은 문제의 유형에 따른 고유한 해법을 일일이 개발해야 하나 유전알고리듬은 거의 모든 유형의 문제에 대하여 일관성 있게 적용할 수 있다. 둘째, 휴리스틱 기법에서는 주어진 문제의 상황이나 가정이 조금만 달라져도 종전의 해법이 더 이상 적용불가능한 경우가 허다하나 유전알고리듬에서는 변경내용에 따라 목적함수 및 제약조건의 코딩만 수정하면 계속적인 적용이 가능하다. 셋째, unimodal 문제로부터 multimodal 문제, 그리고 combinatorial 문제에 이르기까지 광범위한 문제영역에 걸쳐, 비록 최적해의 여부는 알 수 없어도 좋은 해를 제공하는 robustness를 지니고 있다.

본 연구에서는 유전알고리듬을 사용해 MST 관련문제를 해결하는 데 있어서의 문제점을 지적하고 이 문제점을 해결하기

† 연락저자 : 고시근 교수, 608-739 부산광역시 남구 용당동 산100 부경대학교 산업공학과, Fax : 051-620-1546, e-mail : sgkoh@pknu.ac.kr
2002년 1월 접수, 3회 수정 후 2002년 5월 게재 확정.

위한 방안을 제시하고자 한다. 2장에서는 MST 문제를 유전알고리듬으로 해결하기 위한 해의 표현방식 중에서 가장 널리 사용되는 Prufer 수에 대해 설명하고 이 방식의 문제점을 지적한다. 3장에서는 새로운 유전연산 방식에 대해 설명한 뒤 예제를 통해 기존의 연산방식과 비교한다. 마지막으로 4장에서는 실험을 통해 제안된 방식의 우수성을 입증한다.

2. Prufer 수

유전알고리듬을 사용해 MST 관련 문제를 해결하기 위해서는 우선 tree를 염색체(chromosome)로 표현하는 방법에 대해 논의하여야 한다. 이 작업을 Encoding이라고 하는데 Gen and Cheng (1997)에 따르면 Edge encoding, Vertex encoding, Edge and vertex encoding 등의 3가지 유형으로 나눌 수 있다고 하였다. 그러나 가장 일반적으로 통용되는 방법은 Vertex encoding 방법의 하나인 Prufer 수를 이용하는 것이다.

Prufer 수는 n개의 vertex(혹은 node)로 이루어진 tree를 $(n-2)$개의 숫자로 된 수열로 표시하는 방법으로서 다음과 같은 절차를 거쳐 tree를 표현할 수 있다(Gen and Cheng, 1997).

[Encoding 절차]

단계 1. tree의 잎새 node 중 최소 번호를 i라고 하자(단, 여기서 잎새 node란 하나의 arc에만 연결된 node를 말한다. 즉, <그림 1>에서 node 2, 4, 5, 6이 해당됨).

단계 2. node i에 연결된 node의 번호가 j라면 j가 Prufer 수에 추가된다.

단계 3. node i와 arc (i, j)를 tree에서 제거한다.

단계 4. (총 node 수 − 2) 개의 숫자로 이루어진 Prufer 수가 완성될 때까지 반복한다.

예를 들어 <그림 1>과 같은 Tree가 있다면 위의 Encoding 절차를 거쳐 (3-3-1-1)이라는 수열을 만들어낼 수 있다. 즉, 잎새 node인 2, 4, 5, 6 중에서 최소값인 2가 선택되고 이 node에 연결된 node 3이 Prufer 수의 첫번째 수가 된다. 이제 node 2와 arc (2, 3)을 제거하면 잎새 node가 4, 5, 6이 되고 이 중에서 최소인 node 4에 연결된 node 3이 Prufer 수에 추가된다. node 4와 arc (3, 4)를 제거하면 잎새 node가 3, 5, 6이 되고, 이 중 최소는 3이다. 3에 연결된 node는 1이므로 이제 Prufer 수는 (3-3-1)이 되고 node 3과 arc (1, 3)은 제거된다. 이제 잎새 node는 5, 6이므로 최소값 5에 연결된 node 1이 Prufer 수에 추가되어 총 node 수 6보다 2개 적은 4개의 숫자로 된 Prufer 수 (3-3-1-1)이 완성되는 것이다.

이제 반대로 Prufer 수로 표현된 염색체로부터 실제 tree를 구하는 절차를 설명하면 다음과 같다.

[Decoding 절차]

단계 1. P를 Prufer 수라고 하자. 또한 Q는 P에 속하지 않는 node들의 집합이라고 하자.

단계 2. Q의 원소 중에서 최소값을 i라고 하자. j는 P의 가장 왼쪽 숫자이다. 그러면 arc (i, j)를 tree에 추가하고 P에서는 j를 Q에서는 i를 제거한다. 그런데 j가 P에 더 이상 나타나지 않게 되면 j를 Q에 추가한다. P에 숫자가 없어질 때까지 반복한다.

단계 3. P에 숫자가 없어지면 Q에는 두 개의 숫자가 남는다. 이 숫자를 r, s라고 하면 arc (r, s)를 tree에 추가하여 tree를 완성한다.

예를 들어 (1-4-2-6) 이라는 Prufer 수를 이용해 tree를 만들어 보자. P는 (1-4-2-6)이고 Q는 {3, 5}이므로 Q에서 3과 P에서 1이 선택되어 arc (1, 3)이 만들어진다. 그리고 Q에서는 3이 P에서는 가장 왼쪽의 1이 제거된다. 그리고 P에서 제거된 1은 더 이상 P에 나타나지 않으므로 Q에 추가되어 Q={1, 5}이다. 다음은 Q에서 1, P에서 4가 선택되어 arc (1, 4)가 만들어진 후 Q에서 1을, P에서는 4를 제거한다. 그리고 4는 Q에 추가된다. 그러면 다음에는 Q에서 4, P에서 2가 선택되어 arc (2, 4)를 생성한 뒤 제거된다. 이제 Q에는 2가 추가되어 {2, 5}가 P에는 6 만 남아있으므로 arc (2, 6)이 생성되고 P에서 제거된 6이 Q에 들어가 마지막 arc인 (5, 6)을 만들고 절차가 종료된다. 그 결과 <그림 2>와 같은 tree가 만들어지게 된다.

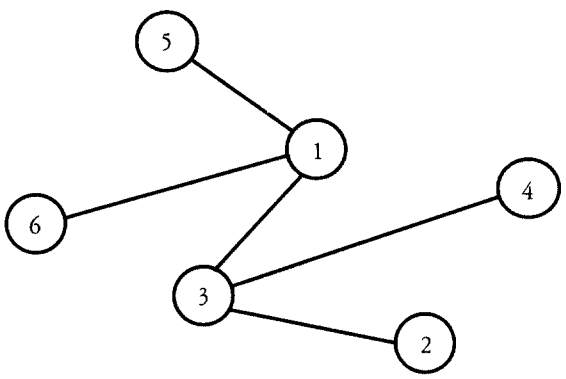

그림 1. Tree 및 Encoding의 예.

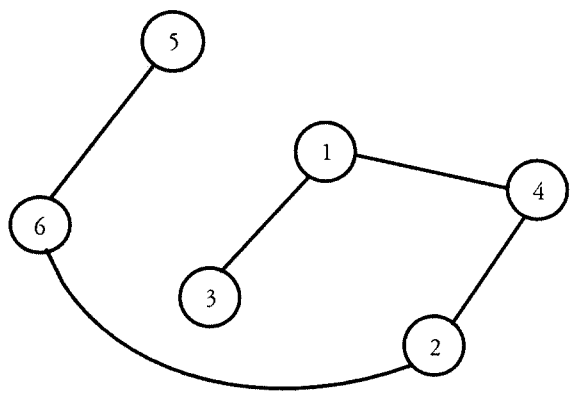

그림 2. Decoding의 예.

이상과 같은 절차를 사용하면 Prufer 수를 사용해 tree를 표현하는 것이 매우 간단하므로 유전알고리듬을 사용해 MST 관련 문제를 해결할 때는 Prufer 수를 사용해 알고리듬의 염색체를 표현하는 것이 일반적이다. 그런데 Palmer and Kershenbaum (1995)이 지적하였듯이 Prufer 수에서의 매우 작은 변화가 그 결과 생성되는 tree에 매우 큰 변화를 주는 것이 문제이다.

유전알고리듬의 원리는 교배연산을 통해 부모 세대로부터 우수한 유전자는 물려받고, 돌연변이 연산을 통해 우연한 변화를 시도하는 것이다. 그런데 Prufer 수를 사용해 단순한 유전자교환 방식의 교배 연산을 행하면 부모가 갖고 있던 특성이 자손에게 전해지지 않는다. 또한 돌연변이 연산을 통해 Prufer 수의 한 숫자만 변해도 전혀 다른 모양의 tree가 만들어지므로 일반적으로 생각하는 돌연변이가 될 수 없다. 즉, Prufer 수를 사용한 단순 교배 및 돌연변이 연산으로 유전알고리듬을 시행하는 것은 우연탐색(Random Search)과 크게 다르지 않다는 것이다. 따라서 본 연구에서는 Prufer 수를 사용해 tree를 표현하는 것은 좋으나 교배 및 돌연변이 연산을 수행할 때는 tree 형태로 Decoding하여 원래 형질을 사용한 연산을 시행할 것을 제안한다. 자세한 절차 및 기존 방식과 비교한 예를 3장에서 설명한다.

3. 유전 연산

3.1 교배(crossover)

교배 연산을 거치면 두 개의 tree로부터 각각의 특성을 고루 물려받은 새로운 tree 두 개가 생성된다. 그 절차는 다음과 같다.

[교배연산 절차]

단계 0. Prufer 수로 표시된 두 개의 염색체를 Decoding하여 두 개의 tree A, B를 만든다. 생성될 tree를 C라 하고 node 집합 Q={1}라고 하자. 또한 Q'는 Q에 속해 있지 않은 node들의 집합이라고 하자.

단계 1. tree A에서 Q와 Q'를 연결하는 arc 중에서 하나를 임의로 선택한다. Q에서 i, Q'에서 j, 즉 arc (i, j)가 선택되었다면 j를 Q'에서 제거하여 Q에 넣고 arc (i, j)를 C에 넣는다.

단계 2. tree B에서 Q와 Q'를 연결하는 arc 중에서 하나를 임의로 선택한다. Q에서 i, Q'에서 j, 즉 arc (i, j)가 선택되었다면 j를 Q'에서 제거하여 Q에 넣고 arc (i, j)를 C에 넣는다.

단계 3. 단계 1과 단계 2를 번갈아 시행하여 Q'가 공집합이 되면 멈춘다.

이 절차를 거치면 한 개의 tree만 생성되므로 두 번째 tree는 순서를 바꾸어, 즉 위의 절차에서는 tree A부터 시작하였는데 두 번째는 tree B에서 시작하여 단계 1과 2를 번갈아 시행한다.

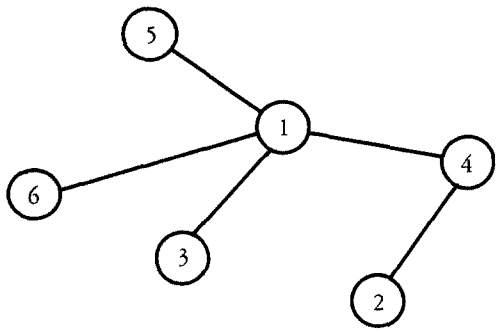

그림 3. 첫 번째 자손 Tree.

예를 들어 <그림 1>과 <그림 2>를 사용해 교배 연산을 수행한다. <그림 1>의 tree를 A, <그림 2>의 tree를 B라고 하자. 시작시의 Q={1}, Q' ={2, 3, 4, 5, 6} 이다. A에서 Q와 Q'를 연결하는 arc는 (1, 3), (1, 5), (1, 6)의 세 개가 있고 이 중에서 (1, 5)가 선택되었다고 하자. 그러면 Q={1, 5}, Q'={2, 3, 4, 6}이다. 이번에는 B에서 Q와 Q'를 연결하는 arc를 보면 (1, 3), (1, 4), (5, 6)의 세 개가 있고 이 중에서 (1, 4)가 선택되었다면 Q={1, 4, 5}, Q' ={2, 3, 6}이 된다. A에서 Q와 Q'를 연결하는 arc는 (1, 3), (1, 6), (3, 4)이고 여기서 (1, 3)이 선택되면 Q={1, 3, 4, 5}, Q' ={2, 6}이다. B에서 Q와 Q'를 연결하는 arc는 (2, 4), (5, 6)이고, 이 중에서 (2, 4)를 선택하면 Q={1, 2, 3, 4, 5}, Q'={6}이 된다. 마지막으로 A에서 (1, 6)이 선택되면 Q'= ϕ 이 되어 절차가 종료된다. 이 결과는 <그림 3>에 주어져 있다.

마찬가지 방법으로 이번에는 B부터 시작해보자. 이 경우에도 Q={1}, Q'={2, 3, 4, 5, 6} 이다. B에서 Q와 Q'를 연결하는 arc는 (1, 3), (1, 4)의 두 개가 있고 이 중에서 (1, 3)이 선택되면 Q ={1, 3}, Q'={2, 4, 5, 6}이다. A에서 Q와 Q'를 연결하는 arc는 (1, 5), (1, 6), (2, 3), (3, 4)의 네 개가 있고, 이 중에서 (1, 6)이 선택되면 Q={1, 3, 6}, Q'={2, 4, 5}가 된다. B에서 Q와 Q'를 연결하는 arc는 (1, 4), (5, 6)이고 여기서 (5, 6)이 선택되면 Q={1, 3, 5, 6}, Q'={2, 4}이다. A에서 Q와 Q'를 연결하는 arc는 (2, 3), (3, 4)이고 이중에서 (2, 3)을 선택하면 Q={1, 2, 3, 5, 6}, Q'={4}가 된다. 마지막으로 B에서 Q와 Q'를 연결하는 arc는 (1, 4), (2, 4)이고 이중에서 (2, 4)를 선택하여 <그림 4>와 같은 결과를 얻을 수 있다.

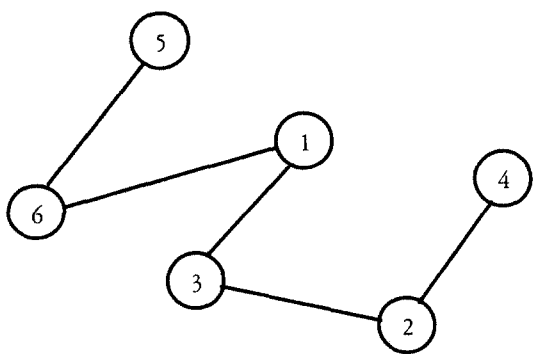

그림 4. 두 번째 자손 Tree.

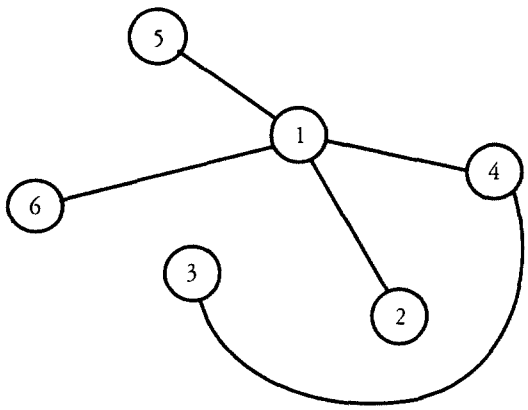

그림 5. 단순 유전자 교환 방식에 의한 결과.

생성된 두 개의 tree는 원래 tree로부터 arc를 물려받았으므로 부모의 특성을 골고루 물려받았다고 생각할 수 있다. 이렇게 함으로써 유전알고리듬의 취지인 "부모의 좋은 특성을 물려받는 것"이 가능해지는 것이다.

반면 위의 두 개 염색체 (3-3-1-1)과 (1-4-2-6)을 사용해 유전자교환 방식의 교배연산을 수행해보면 부모가 갖고 있지 않던 새로운 유전자가 발생하는 등 부모의 형질이 제대로 전달되지 않음을 알 수 있다. <그림 5>는 두 염색체의 가운데를 잘라 교환한 염색체 중에서 (1-4-1-1)에 대응되는 tree이다. 실제로 이 결과에서 arc (1, 2)와 (3, 4)는 부모 중 어느 염색체에서도 갖고 있지 않던 arc이다.

3.2 돌연변이(mutation)

돌연변이 연산을 위해서도 Prufer 수의 형태로 된 염색체를 arc의 집합으로 된 tree 형태로 Decode하여야 한다. 일단 Decode된 tree에 대한 돌연변이 연산은 비교적 간단하다. 즉, tree를 구성하는 arc들 중에서 임의로 하나를 선택하여 삭제해 버린다. 그러면 원래의 tree는 두 개의 tree로 분할되는데 양쪽 tree로부터 node를 하나씩 임의로 선택하여 연결해버리면 새로운 tree가 만들어진다. 물론 이렇게 만들어진 tree는 기존의 tree에서 하나의 arc만 변하게 되므로 대부분의 특성은 유지된다. Prufer 수를 사용한 돌연변이에서와 같이 전혀 다른 형태의 tree가 생성되는 문제를 해결할 수 있게 되는 것이다.

예를 들어 <그림 1>에서 임의로 하나의 arc를 선택하였더니 그 결과 (1, 3)이 선택되었다고 하자. 그러면 원래의 tree는 <그림 6>과 같이 {1, 5, 6}과 {2, 3, 4}의 두 개 tree가 되고 이때 양쪽 tree에서 하나의 node를 임의로 선택하여 연결하면 되는 것이다. 즉, 모든 arc를 그대로 두고 하나의 arc만 변화시키는 것이다.

그러나 기존의 방식에서는 Prufer 수에서 하나의 숫자를 선택하여 변화시키므로 예를 들어 (3-3-1-1)에서 마지막의 1을 2로 변화시켰다고 해보자. 그 결과는 <그림 7>에서 보는 것처럼 기존의 tree와는, 같은 arc를 (1, 3)과 (3, 4)의 두 개밖에 가지

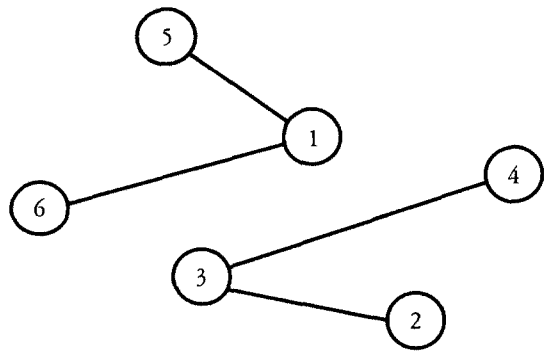

그림 6. 분리된 Tree의 예.

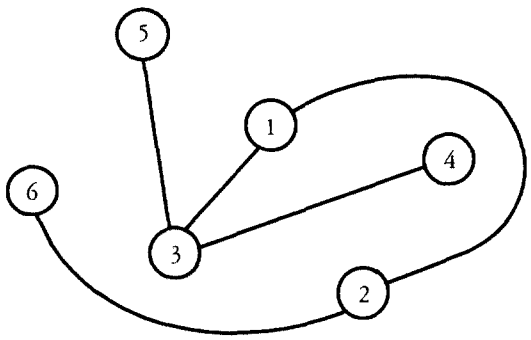

그림 7. (3-3-1-2)에 의한 Tree.

지 않는, 전혀 닮지 않은 tree로서 랜덤하게 다시 생성한 것과 크게 다르지 않음을 알 수 있다.

기존의 방식이 우연탐색에 가까워 탐색효율이 떨어짐은 Palmer and Kershenbaum(1995)이 이미 지적한 바 있지만, 위의 예를 통해 그 사실을 예상할 수도 있을 것이다. 또한 본 연구에서 제안한 방법으로 얻어진 자손 개체의 형질은 부모의 특성을 잘 보존하고 있다는 사실도 위의 예를 통해서 볼 수 있었다. 다만 제안된 방법의 문제점이라면 각각의 연산이 수행될 때마다 Encoding과 Decoding이 매우 자주 이루어져야 한다는 것이다. 따라서 본 연구에서 제안한 방법의 우수성을 주장하기 위해서는 소요시간을 기준으로 평가가 이루어져야 한다.

4. 비 교

본 연구에서 제안된 방법을 평가하기 위해 <표 1>과 같이 node의 개수가 20개인 문제를 만들어서 실험하였다. C++ 언어를 사용하여 MS Visual C++에서 구현된 테스트 프로그램에서는 Population의 크기를 50으로 하였고 초기해는 랜덤하게 생성하였다. 즉, node의 개수가 n이라면 1 부터 n까지의 정수를 순서대로 $n-2$개 랜덤하게 생성하여 한 개의 Prufer 수(다시 말해 염색체)를 생성하는 것이다.

교배확률 및 돌연변이 확률은 여러 번의 실험을 거쳐 각 문제에 가장 잘 맞는 값으로 정하였다. 정하는 방법으로는 교배확률과 돌연변이 확률을 변화시키면서 각각의 경우에 대해 10

표 1. node 사이의 거리 data

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1	0	22	22	36	67	30	53	80	94	45	39	54	28	35	44	35	66	63	29	74
2	22	0	20	20	44	28	40	72	76	39	55	18	56	62	88	74	92	48	62	68
3	22	20	0	40	56	44	60	92	94	94	28	58	47	32	18	59	57	83	74	96
4	36	20	40	0	40	20	20	53	58	68	93	88	73	55	49	83	66	64	76	85
5	67	44	56	40	0	60	44	78	51	77	99	73	44	58	63	18	68	48	75	55
6	30	28	44	20	60	0	28	50	70	33	39	77	28	53	49	38	48	62	55	33
7	53	40	60	20	44	28	0	36	42	84	59	99	85	76	54	66	49	40	37	50
8	80	72	92	53	78	50	36	0	50	44	70	66	30	71	84	68	90	48	47	55
9	94	76	94	58	51	70	42	50	0	50	69	38	57	68	49	63	59	60	48	50
10	45	39	94	68	77	33	84	44	50	0	88	74	30	58	66	20	38	54	44	30
11	39	55	28	93	99	39	59	70	69	88	0	40	58	29	38	53	44	50	37	66
12	54	18	58	88	73	77	99	66	38	74	40	0	60	30	58	37	66	44	39	68
13	28	56	47	73	44	28	85	30	57	30	58	60	0	77	81	29	55	30	62	73
14	35	62	32	55	58	53	76	71	68	58	29	30	77	0	60	94	88	72	49	56
15	44	88	18	49	63	49	54	84	49	66	38	58	81	60	0	91	33	59	84	75
16	35	74	59	83	18	38	66	68	63	20	53	37	29	94	91	0	66	49	46	51
17	66	92	57	66	68	48	49	90	59	38	44	66	55	88	33	66	0	80	49	66
18	63	48	83	64	48	62	40	48	60	54	50	44	30	72	59	49	80	0	55	60
19	29	62	74	76	75	55	37	47	48	44	37	39	62	49	84	46	49	55	0	58
20	74	68	96	85	55	33	50	55	50	30	66	68	73	56	75	51	66	60	58	0

번씩 실험을 행하여 최선 및 최악의 해를 제거한 8회의 실험결과를 평균한 결과를 최소화하는 확률값들을 사용하는 것이다. 그 결과 Prufer 수를 사용한 직접적인 교배 및 돌연변이, 즉 기존의 방법에는 교배확률이 0.1, 돌연변이 확률이 0.04일 때 가장 좋은 결과가 얻어졌고, 본 연구에서 제안된 새로운 방법일 때는 교배확률이 0.1, 돌연변이 확률이 0.03일 때 가장 좋은 결과가 도출되었다. 여기서 교배확률의 의미는 이전 세대를 평가하여 평가된 점수에 따라 랜덤하게 선택된 두 개의 염색체가 교배될 확률이다. 즉, 두 개의 염색체가 선택되면 90%는 그대로 뒷 세대로 복사되고, 10%는 교배하여 새로운 염색체 두 개를 뒷 세대로 전달하는 것이다. 돌연변이는 Population 내의 전체 tree의 모든 arc에 대해 돌연변이 확률에 의해 돌연변이를 시행한다.

앞에서도 예상하였지만 제안된 방법은 기존의 방법에 비해 컴퓨터 연산시간이 길어짐을 확인할 수 있었다. 본 실험에서 측정한 수행시간 비교에 따르면 제안된 방법의 시간이 기존의 방법에 비해 같은 세대수일 경우 대략 7배 정도 더 소요되는 것으로 관측되었다. 따라서 세대수에 기준한 해의 우수성 비교는 그 의미가 적을 것이므로 소요시간을 기준으로 두 방법을 비교하기로 하였다. 우선 각 방법을 사용해 같은 조건에서 10회씩 반복하여 해를 구하고 그 결과를 요약하여 <표 2>에 제시하였다. 본 연구에서 제안된 방법의 세대수 1,000세대를 기준으로 하고, 시간을 맞추기 위해 기존 방법의 세대수는 7,000세대로 하였다. 그 결과 각 방법의 10회 수행을 통해 구한 평균 소요시간은 모두 2.96초로 같게 되었다. 하지만 그 결과 얻어진 최고해들의 평균을 구해보면 제안된 방법이 기존의 방법에 비해 10% 이상 우수한 해를 보장해준다는 사실을 알 수 있다. 앞서 언급하였듯이 이 결과는 각 문제에 대해 가장 우수한 교배 및 돌연변이 확률을 사용한 결과이다. 그러나 이러한 확률값에 변화가 있어도 정도의 차이만 있을 뿐 제안된 방법의 해가 일관되게 우수하였다.

표 2. 실험결과

	총 세대수	평균 소요시간	평균 최고해
기존 방법	7,000	2.96 초	546
제안된 방법	1,000	2.96 초	486

이러한 결과가 나온 과정에 대한 관찰결과도 제안된 방법의

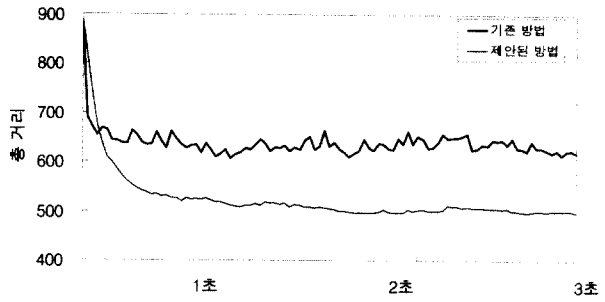

그림 8. 세대별 해의 발전과정 비교.

우수성을 잘 보여주고 있다. <그림 8>은 위의 실험에서 세대의 진행에 따른 해의 변화과정을 보여준다. 유의할 점은 수평축이 세대수가 아닌 시간을 표시한다는 것이다. 기존방법은 총 7,000세대, 제안된 방법은 총 1,000세대이므로 같은 관점에서 비교하기 위해 시간단위로 본 것이다. 또한 그래프로 표시된 점들은 10회의 반복시행결과를 평균한 값이다. 예를 들어 설명하면 수평축의 1초에 대응하는 각 그래프의 값은 기존 방법의 경우 대략 2,333세대 째, 제안된 방법의 경우 대략 333세대 째의 결과를 10회 반복시행에 대해 평균한 값을 나타낸다. 그 결과는 그래프에서 볼 수 있듯이 매우 현격한 차이를 가진다. 즉, 새로운 방법이 기존의 방법에 비해 훨씬 좋은 결과를 얻어줄 뿐 아니라 매우 안정적인 결과를 얻어준다는 것을 쉽게 알 수 있다.

또한 이 그래프에서 기존방법이 제안된 방법에 비해 빨리 수렴하는 것처럼 보이는 것도 이 그래프의 수평축이 세대수가 아니라 시간이라는 사실에 기인한다. 즉, 기존의 방법이 시간축을 기준으로는 빨리 수렴하는 것처럼 보여도 세대수를 기준으로 보면 그렇지 않다는 것이다. 그리고 기존의 방법에 의한 결과가 충분한 세대수가 경과된 후에도 안정되어 있지 못하다는 사실은 그 방법이 우연탐색에 가깝다는 것을 보여주는 현상이다. 이 결과를 안정시키기 위해서는 돌연변이 확률을 줄여주면 되는데 그렇게 되면 금방 하나의 지역해(Local Optimum)에 빠져버리게 되므로 해의 질이 떨어지게 된다.

5. 결 언

본 연구에서는 MST 관련문제를 유전 알고리듬으로 해결할 때 발생하는 문제점에 대한 새로운 해결방안을 제안하였다. 즉, 가장 일반적인 염색체 표현방식인 Prufer 수가 갖는 취약점을 해결하기 위한 새로운 유전 연산방식을 제안한 것이다. 기존의 방식은 Prufer 수를 직접 교배 및 돌연변이 연산에 적용하여, 그 연산결과 생성된 자손 염색체가 부모 염색체의 좋은 특성을 완전히 잃어버리게 됨으로써 유전알고리듬이 아니라 우연탐색에 가까워지는 문제점을 가지고 있었다.

본 연구에서는 이러한 문제를 개선하기 위해 교배 및 돌연변이 연산시 Prufer 수를 그대로 사용하지 않고 Tree 형태로 Decoding 한 다음에 연산을 수행하여 부모 염색체의 특성을 물려받을 수 있도록 하였다. 제안된 방법에 의하면 연산시간이 많이 소요되는 문제가 발생하나 그 약점을 감안하더라도 훨씬 좋은 결과를 얻을 수 있음을 실험을 통해 제시하였다.

참고문헌

Bertsimas, D. (1990), The probabilistic minimum spanning tree problem, *Networks*, **20**, 245-275.

Gen, M. and Cheng, R. (1997), *Genetic Algorithms and Engineering Design*, John Wiley and Sons, Inc.

Gilbert, E. (1967), Minimum cost communication networks, *Journal of Bell Systems Technology*, 9, 2209-2227.

Goldberg, D. E. (1989), *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley.

Graham, R. and Hell, P. (1985), On the history of the minimum spanning tree problem, *Annals of the History of the Computing*, 7, 43-57.

Myung, Y. S., Lee, C. H. and Tcha, D. W. (1995), On the generalized minimum spanning tree problem, *Networks*, **26**, 231-241.

Narula, S. and Ho, C. (1980), Degree-constrained minimum spanning tree, *Computers and Operations Research*, 7, 239-249.

Palmer, C. and Kershenbaum, A. (1995), An approach to a problem in network design using genetic algorithms, *Networks*, **26**, 151-163.

Shiode, I., Nishida, T. and Namasuya, Y. (1981), Stochastic spanning tree problem, *Discrete Applied Mathematics*, 3, 263-273.

고 시 근
고려대학교 산업공학과 학사
KAIST 산업공학과 석사
KAIST 산업공학과 박사
현재: 부경대학교 산업공학과 교수
관심분야: 생산/물류관리, TOC

김 병 남
서울대학교 산업공학과 학사
KAIST 산업공학과 석사
현대중공업 과장
현재: 부경대학교 산업공학과 교수
관심분야: CIMS, 경제성공학