최적 통신 걸침 나무 문제를 해결하기 위한 진화 알고리즘

(Evolutionary Algorithm for solving Optimum Communication Spanning Tree Problem)

석 상 문 † 장 석 철 † 변 성 철 †† 안 병 하 †††
(Sang-Moon Soak) (Seok-Cheol Chang) (Sung-Cheal Byun) (Byung-Ha Ahn)

요 약 본 논문은 최적 통신 걸침 나무 문제(Optimum Communication Spanning Tree Problem : OCST)를 다룬다. 일반적으로, OCST문제는 NP-hard 문제로 알려져 있으며 최근에 Papadimitriou 와 Yannakakis에 의해서 MAX SNP-hard로 밝혀졌다. 그럼에도 불구하고 OCST 문제를 해결하기 위한 기존의 주된 접근법은 polynomial time 알고리즘들 이었다. 본 논문에서는 OCST 문제를 해결하기 위한 진화 알고리즘을 소개한다. 특히, 진화 알고리즘을 어떤 문제에 적용할 때 가장 우선적으로 고려되어야 하는 사항은 해를 어떻게 표현할 것인가 하는 표현법(representation)에 관한 것이다. 따라서 본 논문에서는 기존에 차수 제약 걸침 나무 문제를 해결하기 위해 제안한 표현법의 단점을 개선하는 새로운 표현법을 제안하고 이 표현법을 이용해서 트리(tree)를 만들어 내는 decoding 방법 또한 소개한다. 그리고 제안하는 해 표현법에 맞는 유전 연산자를 찾기 위해 네트워크의 정보 및 부모세대가 지닌 유전 정보를 이용하는 3가지 방법을 실험하였다. 결론적으로, 다양한 실험을 통해서 제안하는 방법이 기존의 방법에 비해 우수한 결과를 보여 준다는 것을 확인할 수 있었다.

키워드 : 최적 통신 걸침 나무 문제, 진화 알고리즘, 유전 알고리즘

Abstract This paper deals with optimum communication spanning tree(OCST) problem. Generally, OCST problem is known as NP-hard problem and recently, it is reveled as MAX SNP hard by Papadimitriou and Yannakakis. Nevertheless, many researchers have used polynomial approximation algorithm for solving this problem. This paper uses evolutionary algorithm. Especially, when an evolutionary algorithm is applied to tree network problem such as the OCST problem, representation and genetic operator should be considered simultaneously because they affect greatly the performance of algorithm. So, we introduce a new representation method to improve the weakness of previcus representation which is proposed for solving the degree constrained minimum spanning tree problem. And we also propose a new decoding method to generate a reliable tree using the proposed representation. And then, for finding a suitable genetic operator which works well on the proposed representation, we tested three kinds of genetic operators using the information of network or the genetic information of parents. Consequently, we could confirm that the proposed method gives better results than the previous methods.

Key words : Optimum communication spanning tree problem, Evolutionary algorithm, Genetic algorithm

1. 서 론

최적 통신 걸침 나무 문제(optimum communication spanning tree : OCST)는 최소걸침나무(minimum spanning tree : MST)를 확장한 문제로써 Hu[1]에 의해서 처음 소개되었다. OCST 문제는 모든 주어진 노드(node)들을 연결하고 최소 비용으로 통신요구(communication requirements) 들을 만족시키는 트리(tree) 형

† 비 회 원 : 광주과학기술원 기전공학과
soakbong@gist.ac.kr
stniron@gist.ac.kr
†† 정 회 원 : 광주과학기술원 기전공학과
scbyun@gist.ac.kr
††† 비 회 원 : 광주과학기술원 기전공학과 교수
bayhay@gist.ac.kr
논문접수 : 2004년 7월 16일
심사완료 : 2005년 2월 2일

태의 통신 네트워크를 찾는 문제이다. 일반적으로 어떤 그래프의 OCST를 찾는 것은 Garry와 Johnson에[2] 의해서 NP-hard 문제로 알려졌으며, 그 이후 Papadimitriou와 Yannakakis에[3] 의해 MAX SNP-hard로 밝혀졌다. 그리고 최근에 이 문제는 Arora 등에[4] 의해, 만약 NP=P가 아니면 MAX SNP-hard 문제를 polynomial time으로 근사화해서 해결하는 접근법(polynomial time approximation scheme) 들을 이용해서는 해결할 수 없음이 증명되었다. 그럼에도 불구하고, 그 동안 많은 연구자들이 이 문제를 해결하기 위한 근사화 알고리즘(approximation algorithms) 들을 제안해 왔다[5-7].

최근 들어 OCST 문제를 해결하기 위해 새로운 접근법들이 시도 되고 있는데 그 중 가장 두드러진 방법은 진화 알고리즘(Evolutionary Algorithm : EA)을 이용하는 방법이다[8-13]. 이러한 문제를 해결하기 위해서 EA를 사용하는 것은 그 동안 유사한 복잡한 문제들을 해결하기 위한 EA의 시도들이 성공적인 결과들을 보여 준 것에 비춰보면 당연한 일 일 지도 모른다[14].

특히 OCST 문제는 현실 문제와 중요한 관련성을 지니고 있기 때문에 몇몇 연구자들은 이러한 점에 초점을 맞춰 해결하려고 시도하였다[9-11]. Palmer와 Kershenbaum[9,10] 등은 미국에 있는 실제 도시와 일치하는 노드(node) 들을 사용하는 현실 문제를 다루었는데, 이 문제에서 각 도시 사이의 거리는 실제 두 지점 사이의 거리를 나타내는 tariff에 기반하여 산출하였고 수요량(demands)은 도시들 사이의 거리의 역수를 이용하였다. 또한, Rothlauf 등은[11] 전송 라인의 이산적인 용량(discrete capacity)과 각 노드들 사이의 수요량(demands)을 충족시키기 위한 비용을 사용하는 현실 문제를 다루었다. 그들의 실험에서, 각 도시 사이의 비용은 Palmer 등과 유사하게 독일 통신 회사(German telecommunication company)의 tariff와 어떤 회사가 월별 특정 길이와 용량(capacity)의 통신 라인을 임대하는데 소요되는 비용을 근거로 해서 계산되었다. 이들 두 연구 모두 현실 문제를 해결하기 위한 접근법으로 진화 알고리즘을 사용하였다.

일반적으로 진화 알고리즘을 어떤 문제에 적용할 때 가장 우선적으로 고려하여야 하는 사항은 해를 어떻게 표현할 것인가 하는 해 표현법(representation)과 표현법에 맞는 연산자를 개발하는 것이다. 왜냐하면 알고리즘의 성능이 이러한 해 표현법과 이를 이용해서 해를 개선시켜 나가는 연산자들에 많은 영향을 받기 때문이다. 따라서 본 논문에서는 기존에 저자들이 차수 제약 걸침 나무 문제(degree constrained minimum spanning tree : DCMST)를 해결하기 위해 제안한 표현법을 개선하는 새로운 표현법을 제안한다. 그리고 제안하는 방법에 알맞은 새로운 유전 연산자 또한 제안한다. 제안하는 연산자는 부모가 지닌 유전 정보 뿐만 아니라 네트워크가 지니고 있는 정보 또한 이용하면서 새로운 자손을 만들어 낸다. 그리고 제안하는 방법과 기존에 현실세계의 OCST 문제를 해결하기 위해 적용된 두 방법과 비교를 통해 제안하는 방법의 성능을 보인다.

본 논문은 다음과 같이 구성된다. 우선 2장에서는 OCST 문제에 대해서 설명하고, 3장에서는 기존의 트리 표현법들에 대해 간단하게 소개한다. 그리고 4장에서는 새로운 트리 표현법과 decoding 방법을 소개하고 5장에서는 사용된 연산자들에 관해 설명하고, 6장은 제안하는 방법과 기존 방법들과의 비교 실험을 수행한다. 그리고 7장에서 결론을 맺는다.

2. OCST 문제

우선 방향성이 없는 완전한 그래프 $G=(V, E)$를 고려해 보자. 여기서, $V=\{1, 2, \ldots, N\}$는 N개의 노드들의 집합이고 $E=\{1, 2, \ldots, M\}$는 M개의 에지 (edge) 또는 링크 (link) 들의 집합이다. 일반적으로, MST 문제는 최소거리를 가지는 걸침나무를 찾는 것이지만, OCST문제의 경우에는, 이에 덧붙여서 각각의 노드들의 쌍과 관련이 있는 통신 요구량(communication requirement) 들을 가진다. 노드 i와 j 사이의 통신 요구량을 $R(i, j)$로 나타내는데, $R(i, j)$는 두 도시사이의 전화량을 나타낼 수도 있고 라인이 가지는 용량을 나타낼 수도 있다. 그래프 G의 걸침나무 T에 대해서 두 노드 사이의 통신비용(communication cost)은 걸침나무 T상의 (두 도시 사이의 거리x통신 요구량)으로 정의될 수 있다. 따라서 OCST 문제를 위한 목적함수는 최소 통신 비용(minimum communication cost)을 가지는 걸침나무를 구성하는 것이다. 즉, 이를 수식화하면 (1)과 같다.

$$Min \quad Cost(T) = [\sum_{i,j \in V} R(i, j) d_T(i, j)] \tag{1}$$

여기서, $d_T(i, j)$는 트리 T 상에 있는 노드 i와 j사이의 에지들의 거리 합이다. 그러므로 i와 j의 쌍은 하나의 에지를 나타낼 수도 있고 i에서 j까지의 어떤 경로(path)를 나타낼 수도 있다.

예를 들어 5개의 노드를 가지는 문제를 고려해보자. 비용 테이블은 D, 요구량 테이블은 R그리고 만들어진 트리는 그림 1과 같다.

이를 이용해서 목적함수 값을 계산해 보면 다음과 같다.

$$\begin{aligned} Cost(T) &= R(1,2)\,d(1,5) + d(5,2) + R(1,3)d(1,3) + \ldots \\ &= 12(9+2) + 5(6) + \ldots + 9(2+9+6) + \ldots \\ &\quad + 8(1+9) = 639 \end{aligned}$$

$$D=\begin{pmatrix} 0 & 3 & 6 & 1 & 9 \\ & 0 & 4 & 3 & 2 \\ & & 0 & 8 & 6 \\ & & & 0 & 5 \\ & & & & 0 \end{pmatrix} \quad R=\begin{pmatrix} 0 & 12 & 5 & 10 & 2 \\ & 0 & 9 & 6 & 9 \\ & & 0 & 3 & 7 \\ & & & 0 & 8 \\ & & & & 0 \end{pmatrix}$$

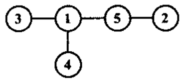


그림 1 OCST 문제의 예

3. 다양한 트리 표현법

3.1 Link and Node Biased 표현법

Link and Node Biased (LNB) 표현법은 Palmaer와 Kershenbaum[9,10]이 OCST문제를 해결하기 위해 제안한 방법이다. 이 방법은 가중치가 부여된 벡터(weighted vector)를 사용해서 트리 네트워크의 구조를 표현하고 진화 알고리즘이 네트워크에 있는 노드(node)와 링크(link) 들 중에서 우수한 것들을 가려 낼 수 있도록 만든다. 이 표현법에서 각각의 chromosome은 각 노드의 bias 값을 이용해서 나타낸다. 각 노드 및 링크의 bias는 [0, 255]사이의 정수 값으로 표현되고, LNB의 코드와 일치하는 걸침나무는 수정된 비용값(C')을 가지고 Prim 알고리즘을 수행하여 만들어 낸다. 비용값을 수정하는 계산식은 식 (2)와 같다.

$$C'_{(i,j)} = C_{(i,j)} + P_1 b_{(i,j)} C_{\max} + P_2(b_i + b_j) C_{\max} \qquad (2)$$

여기서, $C_{\max}$는 그래프에서의 최대 링크 비용(maximum link cost)을, $b_{(i,j)}$는 노드 i, j로 나타내지는 링크의 bias 값을, b_i는 노드 i의 bias 값을 나타낸다.

노드와 링크의 bias 값들은 우선 [0, 1]사이의 값으로 평준화 (normalize) 시킨 다음에 (2)식에 적용한다. 그리고 P_1과 P_2는 제어 파라메터(control parameter)로 Palmer와 Kershenbaum은 실험에서 $P_1=0$, $P_2=1$의 값을 사용하였다.

이 표현법은 각 평가(evaluation) 이후에 비용 테이블이 수정되고, 수정된 비용 테이블을 이용해서 Prim 알고리즘을 수행하여 트리를 만들어 내기 때문에 많은 계산시간을 필요로 한다. 하지만, Abuali 등은[15] 확률적 최소 걸침 나무(probabilistic minimum spanning tree) 문제에 이 표현법을 적용하여 다양한 표현법과 비교 실험을 수행하였는데 몇몇 경우에서 LNB를 이용하는 방법이 더 좋은 결과를 찾음을 보여주었다.

3.2 Network Random Key 표현법

Random Key를 이용하는 방법은 기존의 스케쥴링 문제와 같은 조합문제를 해결하기위해 Bean[16]에 의해 처음 제안된 방법이다. 그 이후 Rothlauf 등은[11] OCST 문제를 해결하기 위해 이 표현법을 확장하였다. 걸침나무문제에서 사용되는 Network Random Key(NetKey) 표현법은 기존의 MST를 해결하기 위한 알고리즘인 Kruskal의 알고리즘과 아주 유사하다. Kruskal의 알고리즘은 네트워크 상의 에지들을 정렬하기 위해 에지의 비용 값을 이용하지만 NetKey에서는 임으로 부여된 벡터 값을 이용한다. 하지만 NetKey 표현법 역시 Kruskal의 알고리즘과 똑같은 결점을 지니고 있는데, 만약 네트워크가 완벽하게 연결되어 있고 많은 노드를 포함하는 경우, 많은 계산 시간과 메모리 량을 필요로 하게 된다. 예를 들어, 100개의 노드를 가지고 완벽하게 연결된 네트워크의 경우 하나의 해를 표현하는데 100(100-1)/2 = 4950개의 bit를 필요로 한다. 왜냐하면, 완벽하게 연결된 네트워크에 존재하는 에지(edge) 수는 $L = N(N-1)/2$와 같기 때문이다. 또한 각 에지에 부여된 NetKey 값을 정렬하기 위해서 $O(L \log L)$의 계산시간을 필요로 한다. 그리고 유전 연산자를 설계할 때, NetKey 표현법은 부모세대가 지니고 있는 정보라든지 네트워크가 내포하고 있는 정보들을 이용할 수가 없기 때문에 효율적인 유전 연산자를 설계하기가 매우 어렵다. 또한 전체 에지 수 $N(N-1)/2$ 중에서 $N-1$개의 벡터(vector) 만을 이용하기 때문에, 저자들이[17,18] 보여 준 것처럼 네트워크의 크기가 커질수록 성능은 떨어진다.

3.3 d-Based 표현법

저자들은 기존의 순회판매원문제(Traveling Salesman Problem : TSP)를 진화 알고리즘을 이용해서 해결할 경우의 해 표현법을 확장한 표현법을 이용해서 차수 제약 걸침 나무(degree constrained minimum spanning tree : DCMST) 문제에 적용하여 표현법의 우수성을 보여 준 바 있다[17,18]. 이 표현법을 저자들은 d-Based 표현법이라고 불렀는데 그 이유는 차수(degree) 제약에 따라 해의 길이가 달라지기 때문이다. 하지만 OCST 문제의 경우 DCMST 문제와는 달리 각 노드가 가지는 차수 제약이 없다. 즉, 모든 노드는 최대 N-1개의 에지를 가질 수 있다. 따라서 이 표현법을 OCST 문제에 적용할 경우 decoding 하는 과정에 많은 시간을 필요로 할 뿐만 아니라 많은 메모리 량도 필요로 하게 된다. 따라서 본 논문에서는 계산 시간 및 메모리 량을 줄 일수 있는 효율적인 새로운 트리 표현법을 제안한다.

3.4 새로운 트리 표현법 : Edge 기반(e-based) 표현법

모든 걸침나무(spanning tree)는 정확하게 $N-1$개의 에지를 이용해서 구성된다. 따라서 걸침나무를 나타내기 위해서 필요한 최소 bit수는 $N-1$개이면 충분하다. 예를 들어 그림1에 있는 걸침나무를 고려해 보자. 이 걸침나무는 4개의 에지 즉, (1, 3), (1, 4), (1, 5), (5, 2)로 구성되어 있다. 이를 차례로 연결하는 string을 만들어 보면 (1, 3, 1, 4, 1, 5, 5, 2)가 된다. 따라서 본 논문에서 제안하는 새로운 해 표현법은 N개의 노드를 지닌

트리 네트워크 문제의 경우 $(N-1)\times 2$의 길이를 가지는 string을 이용해서 해를 표현한다. 단, 모든 노드는 최소한 1번씩은 나타나야만 한다. 간단하게 모든 노드가 한번씩 나타날 때 에지를 형성한다고 하면 제안하는 해 표현으로 $N-1$개의 에지를 충분히 형성할 수 있다. 문제는 이렇게 만들어진 string을 이용해서 어떻게 트리를 구성할 것인가 하는 것인데, 여기서 2가지 트리 구성 룰(tree construction rule : TCR)을 소개한다.

하나는 Cycle-Free TCR (CF-TCR) 로 이미 저자들이 d-Based 표현법을 소개할 때 제안한 방법이다[17, 18]. 이 방법은 string의 가장 왼쪽에 있는 두 노드를 우선 고려한다. 그리고 그 다음 한 bit씩 오른쪽으로 이동하면서 트리를 구성해 나간다. 위의 예에서 가장 처음에 나타나는 두 노드는 1과 3인데 이들은 이전에 선택된 적이 없기 때문에 에지 집합 $ES\cdot$에 추가하고 선택되었음을 확인하기 위해 선택된 노드 집합 $NS\cdot$에 추가한다. 그리고 한 bit 오른쪽으로 이동하고 다시 두 노드를 체크하는데 이번에 선택된 노드들은 3과 1이다. 이 두 노드는 이전에 선택되었기 때문에 다음으로 넘어 간다. 다음은 1과 4인데 4는 이전에 선택된 적이 없기 때문에 에지(1, 4)를 에지 집합 $ES\cdot$에 추가하고 이전에 선택되지 않은 4를 선택된 노드 집합 $NS\cdot$에 추가한다. 여기서 고려하는 두 노드 중 처음에 나타나는 노드는 string의 첫 번째 노드를 제외하면 항상 이전에 나타난 노드들이고 두 번째 노드는 이전에 나타난 노드일 경우에는 cycle을 형성하기 때문에 그냥 넘어가고 그렇지 않을 경우에는 에지를 만든다. 이러한 방식으로 에지를 추가해 나가고 만약 $N-1$개의 에지가 에지 집합에 모여 지면 종료한다. 만들어진 해에 CF-TCR를 적용하여 decoding을 하면 그림 1의 같은 트리를 만들어 낼 수 있다.

하지만 CF-TCR의 단점은 항상 왼쪽에 나타나는 에지들을 우선적으로 고려를 하기 때문에 뒤쪽에 유용한 정보를 지닌 유전인자(gene)가 있을 경우 고려에서 배제되게 된다는 것인데 이는 이 방법이 cycle을 허용하지 않는 방법으로 해를 decoding하기 때문이다. 따라서 본 논문에서는 새로운 decoding 방법을 제안한다.

제안하는 방법은 CF-TCR을 확장한 방식인데 우선 CF-TCR과 똑 같은 방법으로 에지를 추가해 나가다가 고려하는 두 노드가 연결될 경우 cycle을 형성하게 되면 이를 형성하는 에지들을 찾은 다음 가장 비용 값이 큰 에지를 제거하는 방식으로 트리를 구성한다. 따라서 이 방법을 적용하면 CF-TCR이 지니고 있는 단점을 극복할 수 있다. 단, 이 방법의 경우 cycle을 형성하는 에지들을 찾는 과정의 추가로 계산시간이 오래 걸린다는 단점을 지니고 있긴 하지만 CF-TCR 보다 우수한 결과를 도출해 냄은 물론 기존의 방법들보다도 우수한 결과를 도출함을 확인 할 수 있었다. 이 방법을 Cycle-Breaking TCR(CB-TCR)이라 부른다. 다음은 CB-TCR의 절차이다.

Step 0. (초기화 단계) $NS\{\cdot\}\leftarrow\phi$, $ES\{\cdot\}\leftarrow\phi$, $Cn\leftarrow 0$ 그리고 i and $k\leftarrow 1$로 초기화 한다. 여기서 $n\in\{1, 2, \ldots, N\}$이다.

Step 1. $gene_k$ 를 $NS\{\cdot\}$에 추가하고, $C(gene_k)$을 1 증가 시킨다.

Step 2. $gene_{k+1}\notin NS\{\cdot\}$이면,

Step 2.1 $gene_{k+1}$를 $NS\{\cdot\}$에 추가하고, 에지 $E_i(gene_k, gene_{k+1})$를 $ES\{\cdot\}$에 추가한다. 그리고 $C(gene_k)$와 $C(gene_{k+1})$을 1씩 증가 시키고 i도 1 증가시킨다.

Step 3. $gene_{k+1}\in NS\{\cdot\}$ 이면,

Step 3.1. Cycle을 형성하는 모든 에지를 찾고 그 에지들 간에 거리 비교를 해서 가장 긴 에지를 제거하여 Cycle을 깨뜨린다. 단, 여기서 선택된 에지 (a, b)의 각 노드 a 와 b의 $C(a)$, $C(b)>1$ 이어야만 한다. 만약 이 조건을 만족시키지 못할 경우 그 다음 긴 에지를 같은 방법으로 고려한다.

Step 3.2. $gene+k+1$를 $NS\{\cdot\}$에 추가하고, $E(a, b)$를 $E(gene_i, gene_{k+1})$로 대체한다. 그리고 $C(gene_a)$와 $C(gene_b)$는 1씩 감소시키고 $C(gene_k)$와 $C(gene_{k+1})$은 1씩 증가시킨다. i도 1 증가 시킨다.

Step 4. $i\equiv N$이면 Step 5.로 가고 아니면 Step 2.로 가서 반복 수행한다.

Step 5. 종료.

4. 유전 연산자

4.1 선택 연산자(selection operator)

본 논문에서는 저자들이[15] 제안한 Real World 토너먼트 선택전략을 이용한다. 이 방법의 경우 비록 강한 선택압(selection pressure)을 지니고 있지만 실험 결과 기존의 토너먼트 전략을 사용하는 경우보다 우수한 결과를 준다는 것을 보여 주었다. 또한 이 선택 전략은 기존의 토너먼트 전략과는 달리 해 집단(population)에 있는 모든 해들을 경쟁에서 고려하기 때문에 기존의 엘리트 선택(elitism) 전략을 사용하지 않고도 항상 가장 우수한 해를 보존할 수 있다는 특징을 지니고 있다.

4.2 교차 연산자(crossover operator)

여기서는 두 가지 교차 연산자를 소개한다. 하나는 DCMST 문제를 해결하기 위해 적용되어 좋은 결과를 보여주었던 연산자로, Edge Distance 교차 연산자

(EDX)이고 다른 하나는 새롭게 개발된 연산자 이다.

일반적으로 부모 세대가 지닌 유전 정보를 이용하는 연산자들이 우수한 결과를 도출해 내는 것으로 알려져 있다. 하지만 현재까지 OCST 문제의 경우 효율적인 연산자가 알려져 있지 않을 뿐만 아니라 형성된 트리의 구조에 따라 목적함수 값이 결정되기 때문에 사전에 문제에 대한 어떤 지식을 알고리즘에 반영하는 것이 아주 어렵다. 즉, 만약 다루는 문제가 어떤 형태의 트리를 선호 한다면 그러한 형태를 가진 트리를 만들어 내도록 알고리즘에 반영을 하면 되지만 이 경우 다루는 문제를 위한 올바른 형태의 트리가 알려져 있지 않다. 따라서 본 논문에서는 3가지 측면에서 연산자를 실험하였다.

1) 네트워크의 에지(edge) 거리 정보를 이용하는 연산자 - Edge Distance 교차 연산자(EDX).
2) 부모세대가 지니고 있는 공통 정보를 선호하는 연산자 - Uniform 교차 연산자.
3) 부모세대가 지니고 있는 공통 정보뿐만 아니라 에지 거리 정보 또한 이용하는 연산자 - Common Gene Preservation 교차 연산자 (CGPX).

앞의 두 연산자의 경우는 이미 기존의 연구들에서 사용되어진 것들이기 때문에 여기서는 새롭게 제안하는 CGPX에 대해 설명한다. CGPX의 경우 부모 세대가 지니고 있는 공통 정보 즉, 동일한 유전자 좌(allele)에 존재하는 gene은 보존하고 그렇지 않은 gene들은 바로 전에 선택된 gene과의 에지 거리를 비교하여 짧은 쪽을 자손을 위한 다음 gene으로 선택한다. CGPX의 절차는 다음과 같다.

Step 0. (초기화 단계) $i \leftarrow 1$을 넣고, O_i^1에 P_i^1를 넣는다. 여기서 O_i^1와 P_i^1는 각각 자손1과 부모 1의 i 번째 gene을 의미한다.

Step 1. $i \leftarrow i+1$, 만약 P_i^1과 P_i^2에 같은 노드가 존재하면 O_i^1에 P_i^1를 넣는다.

Step 2. 그렇지 않으면, 바로 이전에 자손에 선택된 gene과의 거리 값을 비교해서 짧은 값을 형성하는 gene을 다음 gene으로 선택한다. 즉, 에지 (O_{i-1}^1, P_i^1)과 에지 (O_{i-1}^1, P_i^2)의 거리 비교를 해서 작은 값을 형성하는 P_i^1과 P_i^2 둘 중에 하나의 노드를 선택한다.

Step 3. 만약 $i \equiv L$이면 종료하고 그렇지 않으면 Step 1로 간다.

위의 절차는 자손 1만을 고려한 경우인데 자손 2도 똑같은 방법으로 형성한다.

4.3 변이 연산자(mutation operator)

변이 연산자로는 선행 실험들에서 우수한 결과를 보여준 상호 교환 변이 연산자(reciprocal exchange mutation operator)를 사용하였다[14]. 상호 교환 변이 연산자의 경우 우선 string에서 임의의 두 지점을 선택하고 두 지점에 있는 노드들을 상호 교환하는 방식으로 해에 변화를 주는 방법이다.

4.4 수정 과정

본 논문에서 사용되어진 교차 연산자들을 제안하는 해 표현법에 적용된 이후 생성된 자손들이 표현법이 요구하는 모든 노드가 적어도 한번씩은 나와야 한다는 제약조건을 충족시키지 못한다. 따라서 해를 수정하는 과정이 필요하다. 이는 아주 간단한 방법으로 해결이 가능한데 우선 교차 연산자들이 적용될 때 선택되는 노드들의 선택횟수($C(n)$)를 저장한다. 그래서 모든 연산자가 적용된 이후 한번도 선택되지 않은 노드를 확인해서 그러한 노드가 존재할 경우, 임으로 하나의 노드 a를 선택하고 이 노드를 한번도 선택되지 않은 노드로 대체한다. 단, 선택된 노드 a의 선택횟수는 $C(a)>1$이어야만 한다. 만약, 선택된 노드가 해당 해 표현에서 한번만 나타나는 노드일 경우 트리에 포함되지 않은 채로 남게 되기 때문이다.

이러한 간단한 수정 방법을 적용하여 만들어진 해를 4.2에서 소개한 TCR방법을 이용해서 decoding하면 항상 유효한 트리를 만들어 낼 수 있다.

5. 실험 결과

본 논문에서는 제안하는 방법의 성능을 보이기 위해 두 가지 실험 데이타를 이용하였다. 하나는 기존에 Palmer등이 사용한 실험 데이타이고 다른 하나는 임으로 만들어낸 실험 데이타이다. 임의로 만든 실험 데이타의 경우 거리 값은 500×500 격자망에 임의로 선택한 노드들의 Euclidean 거리를 이용하였고, 요구량(Requirement)은 [0, 100]사이에서 uniform 하게 임의로 선택한 값을 이용하였다. 이러한 방법을 이용해서 임의 실험 데이타를 만든 이유는, Rothlauf 등의 논문[19]을 보면 기존의 실험 데이타들은 OCST 문제를 위한 트리 구조와 일반적인 MST를 위한 트리 구조가 거의 차이가 없는 것으로 나타났다. 즉, 이러한 문제들의 경우 기존의 MST를 해결하기 위한 알고리즘(Prim 알고리즘이나 Kruskal 알고리즘)을 조금만 수정하면 쉽게 최적에 가까운 해를 찾을 수가 있다는 것을 의미한다. 그리고 그들의 실험에서 이러한 방법을 이용해서 임의로 만들어 낸 실험 데이타가 가장 OCST를 위한 트리 구조와 MST를 위한 트리 구조가 차이가 많이 남을 보여 주었다. 따라서 문제의 복잡성을 높이기 위해 이 방법을 이용하였다. 만들어진 데이타의 크기는 N=10에서 N=30개까지 다양한 규모의 네트워크가 만들어졌다.

그리고 모든 알고리즘은 Visual C++를 이용해서 작성하였고 Pentium IV 2.2 Ghz 컴퓨터상에서 실험을 하였다.

각 알고리즘에 사용된 파라메타 및 연산자들은 표 1과 같다. 여기서 LNB의 경우 Palmer 등[9,10]의 논문을 따라 적용을 하였고, NetKey의 경우는 Rothlauf 등[11]은 돌연변이 연산자를 사용하지 않았고, Schindler 등[8]은 진화 전략의 파라메트를 추정하기 위해 적용한 One-Max 트리 문제에서 Unifrom 교차 연산자와 해에 있는 하나의 gene을 선택해서 임의의 값으로 바꿔주는 random perturbation 변이 연산자를 이용하였다. 하지만 저자들이 OCST문제에서 수행한 초기실험에서 NetKey의 경우 Uniform 교차 연산자와 상호 교환 변이 연산자를 이용할 경우가 가장 우수한 결과를 보여 주었다. 따라서 NetKey의 경우 이 방법을 이용하였다.

그리고 모든 실험 데이타 상에서 각 알고리즘을 10번씩 수행하였고 실험 결과는 이를 평균한 값이다. 실험결과에서 보여 지는 결과 값들은 기존에 알려진 가장 좋은 해 또는 실험을 통해서 찾은 가장 좋은 해 (C_{best})와 각 실험 결과의 평균값 (C)과의 차이(Gap)를 백분율로 나타낸 것이다. 이의 계산은 다음 식을 이용하였다.

$$(C - C_{best})/C_{best} \times 100\ (\%) \qquad (3)$$

표 2는 Palmer 등의 실험 데이타를 이용한 결과이다. 이 실험은 기존에 저자들이 제안한 방법(d-based representation)과 본 논문에서 제안하는 방법(e-based representation)과의 비교 및 제안하는 방법을 위한 가장 적합한 교차 연산자를 찾기 위해 수행하였다.

결과를 보면 P12의 경우 e-based 표현법이 P24의 경우 d-based 표현법이 전반적으로 좋은 결과를 보여 줌을 확인할 수 있다. 그리고 각 표현법에서 사용된 TCR 간의 비교 결과를 보면 P24의 EDX 교차 연산자를 이용한 경우를 제외하고 모든 경우에서 CB-TCR이 좋은 해를 산출하였다. 그리고 각 교차 연산자들 간의 결과를 비교해 보면 Uniform 교차 연산자와 제안하는 CGPX를 이용하는 경우가 더 우수한 결과를 보여 주었다.

이 실험결과를 보면 실질적으로 Uniform 교차 연산자와 제안하는 CGPX 교차 연산자 중 어느 연산자가 더 우수한 결과를 보여 주는지 우열을 가리기가 힘들다. 따라서 이 결과를 바탕으로 임으로 만들어낸 실험 데이타에서는 두 방법 모두 Uniform 교차 연산자와 CGPX를 이용한 방법을 가지고 기존의 방법들(LNB와 NetKey)과 비교 실험을 수행하였다.

표 3은 임으로 만든 실험데이타 상에서의 비교 실험 결과를 보여준다. 모든 실험 데이타에서 d-based 표현법과 e-based 표현법은 best solution을 1번 이상 찾은 반면, NetKey는 N=30에서, 그리고 LNB는 N=15, 25 그리고 30에서 한번도 찾지를 못했다. 그리고 N=20의 경우를 제외 하고 모든 실험 데이타에서 e-based 표현법과 CGPX 교차 연산자를 이용한 알고리즘이 가장 우수한 결과를 보여 주었다. 그리고 모든 경우에서 찾은 가장 우수한 해와의 차이(Gap)가 0.5%를 넘지 않았고 모든 실험 데이타에서의 평균 차이(Avg. Gap)은 0.12%였다.

그리고 계산 시간을 비교해 보면, d-based 표현법에 비해 e-based 표현법을 사용할 경우 상당부분 계산 시간을 줄일 수 있음을 확인 할 수 있다. 하지만 e-based 표현법의 경우도 전반적으로 다른 알고리즘에 비해 더

표 1 각 알고리즘에 사용된 파라메타 및 연산자들

	d & e-based	NetKey	LNB
Population 크기	100	100	100
교차 확률	0.6	0.6	0.6
돌연변이 확률	0.6	0.6	0.01
교차 연산자	EDX, Uniform, CGPX	Uniform	One-Point
변이 연산자	Reciprocal Exchange	Reciprocal Exchange	Random Perturbation
종료 조건	Generation = 10,000	Generation = 10,000	Generation = 10,000

표 2 Palmer의 실험 데이타를 이용한 실험결과

Instances		d-based representation						e-based representation					
		EDX		Uniform		CGPX		EDX		Uniform		CGPX	
		CF-TCR	CB-TCR	CF-TCR	CB-TCR	CF-TCR	CB-TCR	CF-TCR	CB-TCR	CF-TCR	CB-TCR	CF-TCR	CB-TCR
P12 (3428509)	Gap	0.011	0.002	0.012	0.002	0.006	0	0.04	0.026	0	0	0.008	0
	Num.	3	9	1	9	4	10	1	2	10	10	4	10
P24 (1086656)	Gap	0.121	0.576	0.119	0.182	1.102	0.14	17.187	5.34	0.063	0.007	9.833	1.419
	Num.	4	0	3	0	0	0	0	0	6	9	0	0

( ) 안의 값은 해당 instance에서 알려진 가장 우수한 해의 값을 나타냄. "Num."는 가장 우수한 해를 찾은 횟수를 의미함.

표 3 기존 방법들과의 실험결과 비교

Instances		d-based Uniform	d-based CGPX	e-based Uniform	e-based CGPX	NetKey	LNB
N=10 (792177)	Gap	0	0	0	0	0	0
	CPU(초)	179.9	148.8	121	118.9	59.7	61.3
	Num.	10	10	10	10	10	10
N=15 (1724448)	Gap	0	0.041	0.052	0	0.163	0.073
	CPU(초)	460.3	353.5	268.7	267.4	181.9	178.1
	Num.	10	9	8	10	6	0
N=20 (3232359)	Gap	0.877	0.857	0.376	0.501	1.509	0.304
	CPU(초)	979.1	718.2	517.8	514.6	394.1	431.7
	Num.	3	4	7	6	1	1
N=25 (5118285)	Gap	0.275	0.611	0.327	0.041	1.42	0.792
	CPU(초)	1751.5	1270.3	878.4	881.6	749.2	851.9
	Num.	5	5	7	8	4	0
N=30 (6363993)	Gap	0.302	0.343	0.195	0.06	0.921	1.797
	CPU(초)	2588.9	2162	1440.5	1449.1	1330.1	1476.6
	Num.	1	1	2	5	0	0
Avg. Gap		0.291	0.37	0.19	0.12	0.803	0.593

( ) 안에 있는 값은 각 실험 데이타에서 찾아낸 가장 우수한 해의 값을 나타냄. "Num."은 가장 우수한 해를 찾은 횟수를 의미함

많은 계산시간을 필요로 하였는데 이는 앞에서 언급한 것처럼 decoding 과정에 사용되는 CB-TCR이 많은 계산시간을 요하기 때문이다. 하지만 그 편차가 네트워크의 크기가 커질수록 작아져 N=30에서는 오히려 LNB 보다 더 적은 계산 시간이 소요 되었다. 그리고 OCST 문제의 특성상 트리로부터 목적함수 값을 계산하는데 소요되는 시간 때문에 현실적으로 아주 큰 규모의 문제를 다룬다는 것은 어렵다. 하지만 상대적으로 큰 규모의 문제를 다룰 수 있는 차수제약 걸침나무 (DCMST) 와 같은 다른 트리 네트워크 문제에 적용될 경우 LNB 뿐만 아니라 NetKey 표현법 또한 제안하는 e-based 표현법 보다 더 많은 계산 시간을 요하게 될 것이다. 이는 그림 2를 통해서 확인 할 수 있다.

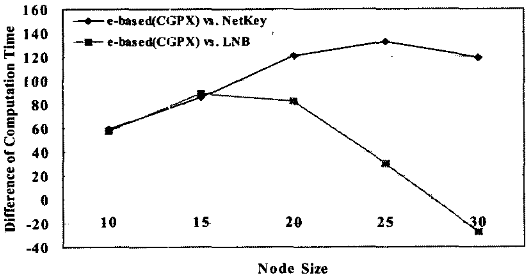


그림 2 알고리즘들 간의 계산시간의 편차

그림 2는 각 실험 데이타에서의 e-based+CGPX를 사용하는 경우와 NetKey 및 LNB를 사용하는 경우의 계산 시간의 편차를 보여 주는 그래프이다. 이를 보면 LNB와 e-based+CGPX를 비교할 경우 N=15에서부터 편차가 줄기 시작해서 N=30에서는 오히려 더 많은 계산시간을 필요로 하였고, NetKey와 e-based+CGPX의 비교에서도 N=25를 기점으로 계산시간의 편차가 줄고 있음을 확인 할 수 있다.

표 3은 알고리즘의 실험 결과들을 통계적으로 분석하기 위해 N=25와 30에서 e-based+CGPX와 NetKey, LNB간에 수행한 t-Test 결과를 보여 준다. 결과를 보면 e-based+CGPX와 NetKey간의 t-Test 결과는 유의수준 95%에서 유의한 결과를 보여 주고, e-based+CGPX와 LNB간의 t-Test 결과는 유의수준 99%에서 유의한 결과를 보여준다는 것을 확인할 수 있다. 따라서 모든 경우 유의수준 95%에서 각 알고리즘의 실험 결과들 간의 평균차이는 유의한 결과를 가진다는 것을 확인할 수 있다.

6. 결 론

본 논문에서는 기존의 트리 표현 방법(d-based representation)을 개선하는 새로운 표현 방법(e-based representation)을 제안하였다. 그리고 제안하는 트리 표현방법을 이용해서 OCST 문제를 해결하기 위해 기존에 사용된 NetKey 및 LNB 방법과 비교 실험을 수행하였다. 또한 제안하는 트리 표현법에 맞는 유전 연산자를 찾기 위해서 3가지 각각 다른 정보를 이용하는 교차 연산자를 실험하였다. 실험 결과 부모세대가 지니고 있는 공통 정보 및 네트워크가 지니고 있는 정보를 이용하는 CGPX 교차 연산자를 이용하는 알고리즘이 더 우수한

표 4 t-Test 분석 결과

		e based+CGPX vs. NetKey	e based+CGPX vs. LNB
N=25	t value	2.0939	10.0547
	P	0.025	< 0.001
N=30	t value	2.8785	15.208
	P	0.004	< 0.001

결과를 보여 주었다. 특히, e-based 표현법과 CGPX 교차 연산자를 이용하는 알고리즘이 다른 기존 알고리즘에 비해서도 더 우수한 결과를 보여준다는 것을 통계적인 방법을 이용해서 확인할 수 있었다.

비록 제안하는 방법이 다른 알고리즘에 비해 더 많은 계산 시간을 필요로 하였지만 실험결과에서 보여준 바와 같이 네트워크의 크기가 커질수록 편차가 줄고 있음을 확인할 수 있었다. 즉, 상대적으로 큰 규모의 트리 네트워크를 다룰 수 있는 문제들에 적용할 경우 다른 알고리즘에 비해 적은 계산시간을 요할 것이다.

제안하는 방법의 경우 모든 가능한 트리 네트워크를 형성할 수 있기 때문에, 앞으로 다양한 트리 네트워크 문제들에 제안하는 방법을 적용하여 알고리즘의 성능을 확인해 볼 계획이다.

참 고 문 헌

[1] Hu, T.C., "Optimum communication spanning trees," SIAM J. Computing, Vol.3, No.3, pp. 188-195, 1974.

[2] Garey, M.R. and Johnson, D.S., "Computers and Intractability: A Guide to the Theory of NP-Completeness," San Francisco, Freeman, 1979.

[3] Papadimitriou, C. H and Yannakakis, M., "Optimization, approximation, and complexity classes, Journal of Computer and System Science," Vol.43, pp. 425-440, 1991.

[4] Arora, S. Lund, C. Motwani, R. Sundan, M. and Szegedy, M., "ProofVarification and the Hardness of Approximation Problems," Colloquium on Computational Complexity Report TR98-008, University of Trier.

[5] Peleg, D. and Reshef, E., "Deterministic Polylog Approximation for Minimum Communication Spanning Trees," ICALP'98, LNCS Vol.1443, pp. 670-681, 1998.

[6] Wu,, B.Y. Chao, K.M. and Tang, C.Y., "Approximation algorithms for Some Optimum Communication Spanning Tree Problem," ISAAC'98, LNCS Vol.1533, pp. 407-416, 1998.

[7] Wu,, B.Y. Chao, K.M. and Tang, C.Y., "A Polynomial Time Approximation Scheme for Optimal Product-Requirement Communication Spanning Trees," Journal of Algorithms, Vol.36, pp. 182-204, 2000.

[8] Gaube, T. and Rothlauf, F. "The Link and Node Biased Encoding Revisited: Bias and Adjustment of Parameters," EvoWorkshop 2001, LNCS Vol. 2037, pp.1-10, 2001.

[11] Rothlauf, F. Goldberg, D.E. and Heinzl, A., "Network Random Keys - A Tree Network Representation Scheme for Genetic and Evolutionary Algorithms," Evolutionary Computation, Vol.10 (1), pp. 75-97, 2002.

[12] Yu Li, "An Effective Implementation of a Direct Spanning Tree Representation in GAs," EvoWorkshop 2001, LNCS Vol.2037, pp. 11-19, 2001.

[13] Yu Li and Bouchebaba, Y., "A New Genetic Algorithm for the Optimal Communication Spanning Tree Problem," Proceedings of Artificial Evolution: Fifth European Conference, pp. 162-173, 1999.

[14] Gen, M. Cheng, R., Genetic Algorithms and Engineering Design, John Wiley & Sons, 1997.

[15] Abuali, F.N. Wainwright, R.L. and Schoenefeld, D.A., "Determinant Factorization: A New Encoding Scheme for Spanning Trees Applied to the Probabilistic Minimum Spanning Tree Problem," in Proceedings of the Sixth International Conference on Genetic Algorithms, Larry J. Eshelman, Ed, pp. 470-477, 1995.

[16] Bean, J. C., "Genetic algorithms and random keys for sequencing and optimization," ORSA Journal on Computing, Vol.6, No.2, pp. 154-160, 1994.

[17] Soak, S.M. Crone, D. and Ahn, B.H., "A New Encoding for the Degree Constrained Minimum Spanning Tree Problem," KES2004, LNAI Vol. 3213, pp. 952-958, 2004.

[18] 석상문, 안병하, "차수 제약 걸침 나무 문제를 해결하기 위한 트리 표현법", 한국정보과학회 추계 학술 대회, 2003.

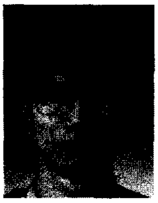
[19] Rothlauf, F. Gerstacker, J. and Heinzl, A., "On the Optimal Communication Spanning Tree Problem," Working Papers in Information Systems, University of Mannheim, 2003.

석 상 문

1998년 한국해양대학교 물류시스템공학과 졸업(공학사). 2000년 한국해양대학교 물류시스템공학과 졸업(공학석사). 2001년~현재 광주과학기술원 기전공학과 박사과정. 관심분야는 Evolutionary Computation, Network Topology Design, Optimization Algorithm 개발

장 석 철

1997년 인하대학교 산업공학과 졸업(공학사). 2000년 광주과기원 기전공학과 졸업(공업석사). 2000년~현재 광주과기원 기전공학과 박사과정. 관심분야는 Optimization problem, Dynamic game theory, Artificial Intelligence & Life



변 성 철

1987년 금오공과대학교 전자공학과 졸업(공학사). 2000년 광주과학기술원 기전공학과 졸업(공학석사). 2004년 광주과학기술원 기전공학과 졸업(공학박사). 관심분야는 Digital Watermarking, Image Signal Processing, 비디오 편집 및 색인

안 병 하

1965년 공군사관학교 졸업. 1977년 한국과학기술원 산업공학과 졸업(공학석사) 1980년 한국과학기술원 산업공학과 졸업(공학박사). 1995년~현재 광주과학기술원 기전공학과 교수. 관심분야는 Evolutionary Computation, Digital Watermarking, Optimization