GOSST 문제 해결을 위한 지그재그 스타이너 포인트 배치 방법을 이용한 휴리스틱의 제안

김 인 범[†] · 김 재 각[†]

요 약

본 논문에서 GOSST(Grade of Services Steiner Minimum Tree) 문제에 대한 개선된 휴리스틱을 제안한다. GOSST 문제는 스타이너 포인트 문제의 한 변형으로 G-Condition을 만족하는 최소비용의 네트워크 구성을 찾는 문제이며, NP-Hard 혹은 NP-Complete 문제로 알려져 있다. 이 문제에 대한 이전의 연구에서 우리는 거리 우선 최소 신장 트리 생성방법과 직접 스타이너 포인트 배치 방법을 결합한 휴리스틱을 제안했었다. 본 논문에서는 스타이너 포인트 배치 방법으로 지그재그 스타이너 포인트 배치방법을 새롭게 제안한다. 이 방법과 거리우선 최소 신장 트리 생성 방법을 결합한 거리 지그재그 GOSST 휴리스틱은 컨트롤인 G-MST에 비해 31.5%의 네트워크 구축 비용의 절감을 얻었고 이전의 가장 좋은 GOSST 휴리스틱인 거리 직접 GOSST 휴리스틱에 비해 2.2%의 비용 개선을 보였다.

키워드 : GOSST, 스타이너 포인트, 최소 신장 트리, 스타이너 포인트 배치방법, 최소 신장 트리 구축 방법, 지그재그 배치, 거리 우선, G-Condition, G-MST

A Proposal of Heuristic Using Zigzag Steiner Point Locating Strategy for GOSST Problem

Kim, Inbum[†] · Kim, Chae-kak[†]

ABSTRACT

We propose more enhanced heuristic for the GOSST (Grade of Services Steiner Minimum Tree) problem in this paper. GOSST problem is a variation of Steiner Tree problem and to find a network topology satisfying the G-Condition with minimum network construction cost. GOSST problem is known as one of NP-Hard or NP-Complete problems. In previous our research, we proposed a heuristic employing Direct Steiner point Locating strategy with Distance Preferring MST building strategy. In this paper, we propose new Steiner point locating strategy, Zigzag Steiner point Locating strategy. Through the results of out experiments, we can assert this strategy is better than our previous works. The Distance Zigzag GOSST method which hires the Distance Preferring MST building strategy and Zigzag Steiner point Locating strategy defrays the least network construction cost and brings 31.5% cost saving by comparison to G-MST, the experimental control and 2.2% enhancement by comparison to the Distance Direct GOSST method, the best GOSST method in our previous research.

Keyword : GOSST(Grade of Services Steiner Minimum Tree), Steiner Point, Minimum Spanning Tree, MST, Steiner Point Locating Strategy, MST Building Strategy, Zigzag Locating, Distance Preferring, G-Condition, G-MST

1. 서 론

네트워크의 물리적인 구축을 위한 설계에서 기본적이고 중요한 문제 중의 하나는 충분한 전송능력과 구축 비용의 최소화이다. 즉, 구축비용의 절약과 동시에 네트워크에 존재하는 임의의 두 노드의 전송 또는 처리능력의 손실이 없어야 한다. GOSST(Grade of Services Steiner Minimum Tree) 문제에 대한 해는 이러한 문제에 대한 좋은 해답을 제공한다. 이러한 GOSST 문제는 또한 도로건설, CAD 설계등에 적용될 수 있을 것이다.

GOSST에 관한 대부분의 연구자들은 최적화에 집중하여 왔다. 그 대표적인 연구자들이 Du와 그의 동료들이다 [1,2,3,4]. 그들은 스타이너 비율에 대한 Gilbert-Pollak Conjecture[1]와 같은 스타이너 트리 문제와 관련된 어려운 문제들을 해결하였고 더 나아가 스타이너 트리와 관련된 많은 문제들을 개발했다. 그것들 중의 하나가 바로 GOSST 문제다. 이 문제는 최적 해의 이론적 한계에 도달한 것으로 간주되고 있다[4]. GOSST와 관련된 대부분의 최적화 알고

† 정 회 원 : 김포대학 인터넷정보과 부교수
논문접수 : 2007년 5월 28일, 심사완료 : 2007년 8월 6일

리즘은 GOSST 문제가 PTAS(Polynomial Time Approximation Scheme) 문제 [4,5]에 속하는 것을 증명하기 위해 개발되어 왔다. NP-Hard 문제에서, PTAS 문제는 비록 매우 많은 다항식적 시간이 소요되더라도 그 문제는 (1+ε)비율의 최적 해를 얻을 수 있다는 것으로 정의할 수 있다. 어떤 문제가 PTAS에 속한다는 것이 증명이 되면 계산 이론의 중요한 이슈중의 하나인 문제의 분류가 완성되는 것이다. 그러나 비록 실행 시간이 다항식적이라 할 지라도 그것은 엄청나게 긴 시간이 될 것이다. 최적화 알고리즘을 개발하는 목적은 이론적 관점에서 주어진 문제가 다항식적 시간 안에 해결할 수 있는 알고리즘이 존재함을 보이기 위함이지 실제적으로 사용하기 위함은 아니다. PTAS 문제에서, 근사 알고리즘의 실행시간을 현실적인 수준으로 떨어뜨린다면 근사도는 의미 없는 수준으로 나빠질 것이다.

많은 연구자들은 GOSST 문제를 스타이너 트리 문제와 관련된 이론적인 문제로 간주했기 때문에 이를 실제적으로 이용할 수 있는 연구에 대한 시도가 충분하지 않다. 그러나 이 문제에 대한 실제적인 해법은 네트워크나 회로 설계등에 이용가능 하므로 이 문제에 대한 적절한 시간 복잡도를 가진 휴리스틱의 개발은 필요하다고 생각된다. 이에 따라 우리는 GOSST 휴리스틱에 대한 연구를 해왔고, 본 연구에서는 이전의 GOSST 문제에 대한 휴리스틱 보다 좀 더 개선된 휴리스틱을 제안한다. 이 결과를 이용해서 현실세계의 다양한 실제적이고 유용한 응용에 적용할 수 있을 것이다.

2. 배 경

본 논문에서 제안하는 GOSST 휴리스틱의 이해를 위해서는 최소 신장 트리(Minimum Spanning Tree), 스타이너 최소 트리(Steiner Minimum Tree) 및 GOSST에 대한 기본적인 설명이 필요하다.

2.1 문제정의

GOSST 문제는 터미널 포인트 집합 P와 어떤 스타이너 포인트를 연결하는 최소비용의 네트워크를 구하는 것이다. GOSST 문제의 정의는 다음과 같다[2]; $P=\{p_1, p_2, \cdots, p_n\}$는 유클리드 평면에 위치하는 n개의 터미널 포인트들의 집합이라 하자. 각 포인트 p_i 는 서비스 요청 등급 $grade(p_i) \in \{1, 2, \cdots, r\}$를 갖는다. $0 < c(1) < c(2) < \cdots < c(r)$를 실수라고 할 때, $c(i)$는 서비스 i를 제공하기 위한 비용이다. 네트워크에서 각 에지(Edge)는 $\{1, 2, \cdots, r\}$ 내의 어떤 한 숫자로 표현되는 등급의 서비스가 지정된다. $grade(e)$를 에지 e의 서비스 등급으로 표현한다. 어떤 네트워크의 터미널 포인트 p_i와 p_j 사이에는 최소 서비스 등급이 $\min(grade(p_i), grade(p_j))$인 어떤 경로가 반드시 존재하고 그러한 네트워크를 구성하는 비용은 최소이어야 한다. 서비스 등급이 g인 에지의 구축비용은 에지의 유클리드 길이와 $c(g)$의 곱이다. 이 GOSST 문제는 이러한 조건을 만족하는 네트워크를 찾는 것이며, 모든 터미널 포인트들이 같은 서비스 요청 등급을 갖는 ESMT (Euclidean Steiner Minimum Tree) 문제의 일반화라고 볼 수 있다.

2.2 최소 신장 트리

n개의 노드를 연결하기 위해서는 최소 $n-1$개의 에지가 필요하다. 노드 집합 V와 어떤 노드 쌍들의 가능한 모든 연결들의 집합 E에 대해, 연결 및 비방향성 그래프(Connected, Undirected Graph) G는 G=(V, E)로 표현된다. 각 에지 $(u, v) \in E$ 에 대해 가중치 $w(u, v)$는 노드 u와 노드 v를 연결을 위한 비용으로 정의한다. 최소 신장 트리 문제는 모든 노드들은 최소의 가중치를 이용해서 비순환 부분집합 $T \subseteq E$를 찾는 문제이다. 이 부분집합은 비순환적이고 모든 노드들을 연결하기 때문에 트리를 형성한다. 이 트리를 형성하는 과정에서 모든 노드들을 연결할 때까지 계속 확장(Spanning)하기 때문에 이를 신장 트리(Spanning Tree)라고 한다. 최소한의 가중치의 합을 가지는 네트워크를 찾아내는 것이 최소 신장 트리 문제이다. Kruskal과 Prim의 알고리즘이 이 문제에 대한 대표적인 알고리즘이다[6]. 본 연구에서 제안하는 GOSST 휴리스틱에서 사용할 최소 신장 트리는 Prim의 알고리즘을 사용하여 구축하였다.

2.3 스타이너 최소 트리

네트워크의 길이의 최소화는 중요한 최적화 문제중의 하나이다. 이 스타이너 최소 트리 문제도 이러한 문제에 속한다. 포인트 S로부터 주어진 다른 세 포인트까지의 거리의 합이 최소가 되는 포인트 S를 찾거나 혹은 어떤 삼각형의 세 점까지의 거리의 합이 최소가 되게 하는 포인트 S를 찾는 문제가 있다. 이 문제는 평면상의 많은 포인트로 구성된 집합을 대상으로 하는 문제로 확장 시킬 수 있다. S가 다른 포인트들과 연결될 때 별모양을 형성할 때, 이를 스타이너 스타(Steiner Star)라고 불린다. 또한 이 문제는 평면에서 모든 노드들을 연결하는 가장 거리가 짧은 네트워크를 구성하기 위해 임의의 개수의 S들을 추가 시키는 문제로 확장할 수 있다. 스타이너 포인트라고 불리는 포인트들의 추가와 최소 네트워크를 생성하기 위해 트리를 생성하는 것이 바로 최소 스타이너 트리(Minimum Steiner Tree) 문제이다.

이 문제를 이산적 포인트들을 대상으로 할 때 NP-Hard 혹은 NP-Complete 문제가 된다[3]. 즉 이 문제는 다항식적 시간 내에 해를 얻을 수 없으므로 이를 해결하기 위해서는 적절한 휴리스틱이 필요로 한다. 본 연구에서는 스타이너 포인트와 최소 스타이너 트리에 대한 휴리스틱을 GOSST 문제를 위해 사용하였다.

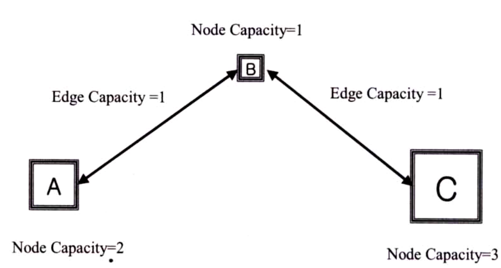
2.4 Grade Of Services Steiner minimum Tree 문제

GOSST(Grade Of Services Steiner minimum Tree) 문

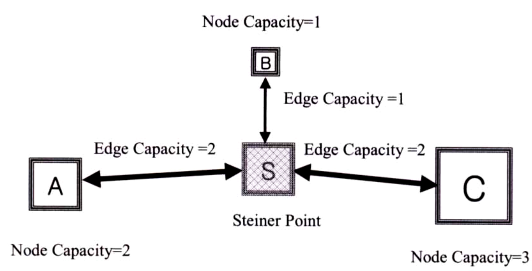
제는 유클리드 평면에서 주어진 포인트들을 연결하는 최소 비용 네트워크를 구하는 ESMT 문제의 한 변형이다 [1,7,8,9,10,11,12]. 이 GOSST 문제는 NP-Complete 문제로 알려져 있다. 이것은 작은 규모의 문제를 해결하기 위해서도 많은 계산량과 막대한 메모리를 필요로 하는 것을 의미한다. GOSST에 대한 많은 연구들은 근사 알고리즘에 대한 기하학적 분석 및 구현에 집중되어 왔다. 그러나 GOSST 문제에 대한 휴리스틱에 대한 연구는 그리 활발히 발표되지 않아왔다.

GOSST 문제는 터미널 포인트 집합의 포인트들과 스타이너 포인트들을 연결하는 네크워크를 찾는 것이다. GOSST에서 터미널 포인트 쌍 p_i와 p_j사이에는 서비스 등급이 *grade*(p_i)와 *grade*(p_j)의 최소값 이상인 경로가 존재해야 한다. 우리는 이것을 *G-Condition*이라 정의한다[13]. 동시에 이 조건을 만족하는 네트워크 중에서 구축 비용이 최소야 한다. 여기서 u 등급 서비스를 제공하는 어떤 에지의 비용은 유클리드 에지 길이와 서비스 등급 u와의 곱으로 계산된다.

(그림 1)의 (A)에서는 서비스 요청 등급으로 간주할 수 있는 처리 용량을 갖고 있는 세 개의 터미널 노드 A, B, C가 보인다. 이 네트워크에서 비록 A의 처리 용량은 2이지만, B의 처리 용량이 1이기 때문에 A와 B의 연결 에지의 처리 용량은 1을 초과할 수 없다. B를 경유하는 A와 C 연결 경로의 용량도 1을 초과할 수 없다. 연결 에지의 용량을 밴드위드로 간주 한다면 이 네트워크에서의 밴드위드의 손실은 불가피하다. 이 손실을 피하기 위해서 혹은 위에서 언급한 *G-Condition*을 만족시키기 위해서, (그림 1)의 (B)와 같이 본 연구에서는 새로운 스타이너 포인트를 생성하여 이를 노드



(A) 세 개의 터미널 노드 A, B, C로 구성된 네트워크



(B) 새로운 스타이너 포인트 생성.

(그림 1) 각각 다른 처리 용량을 가진 세 개의 터미널 노드 A, B, C로 구성된 네트워크와 노드들 사이의 *G-Condition*을 만족시키기 위해 새로운 스타이너 포인트 생성.

A, B, C에 연결시킨다. 이 경우 A와 C사이의 경로의 처리 용량은 2로 증가한다. 스타이너 포인트 위치의 적절한 결정이 네트워크의 구축비용을 줄이는 중요한 변수가 된다. 네트워크의 모든 터미널 노드사이의 경로에 대해서 *G-Condition*을 만족하면서 동시에 네트워크 구축비용을 최소화 하는 것이 GOSST 문제의 최종 목표이다.

3. GOSST 문제를 위한 이전의 휴리스틱

이전 연구에서 제안했던 휴리스틱은 적용된 스타이너 포인트 배치 전략과 최소 신장 트리 생성 전략에 따라 분류된다. 스타이너 포인트 배치 전략에는 전역적(Global), 지역적(Local), 직접적(Direct) 방법이 있고 최소 신장 트리 생성 전략에는 순수(Naïve)와 거리우선(Distance Preferring) 방법이 있다[13,14].

3.1 최소 신장 트리 생성 전략

3.1.1 순수 최소 신장 트리 생성 전략

최소 신장 트리를 순수 구축 방법(Naïve Building Strategy)으로 생성하는 방법은 다음과 같다;

Step 1: 각 연결의 길이와 처리용량의 곱으로 각 연결의 가중치를 구한다.
Step 2: 처음 노드와 연결을 만드는 선택되지 않은 노드들 중에서 연결 가중치가 가장 작은 노드를 선택하고 그 연결을 최종 트리의 한 에지로 결정한다.
Step 3: 선택된 노드와 연결을 만드는 선택되지 않은 노드들 중에서 가장 작은 연결 가중치를 만드는 노드를 선택하고 그 연결을 최종 트리의 에지 중의 하나로 결정한다.
Step 4: 모든 터미널 노드가 선택된 에지들에 의해 연결될 때까지 Step 3을 반복한다. 순수 최소 신장 트리는 이러한 방식으로 선택된 에지들의 모임이다.

3.1.2 거리우선 최소 신장 트리 생성 전략

최소 신장 트리를 거리우선 구축 방법(Distance Preferring Building Strategy)으로 다음과 같이 생성한다;

Step 1: 처음 노드에 연결된 노드들 중에서 가장 거리가 짧은 노드를 선택하고 그 연결을 최종 트리의 한 에지로 결정한다.
Step 2: 선택된 노드와 연결된 노드들 중에서 가장 거리가 짧은, 처리되지 않은 노드를 선택하고 그 연결을 최종 트리의 에지 중의 하나로 결정한다.
Step 3: 2단계에서 선택된 노드가 2개 이상이면 처리용량이 큰 노드를 선택하여 연결을 만들고 이것을 최종 트리의 한 에지로 결정 한다.
Step 4: 모든 터미널 노드가 결정된 에지들에 의해 모두 선택되어 연결될 때까지 Step 2와 Step 3을 반복한

다. 거리우선 최소 신장 트리는 이러한 에지들의 모임이다.

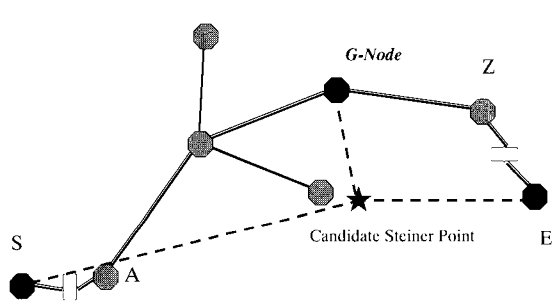
3.2 스타이너 포인트 배치 전략

3.2.1 스타이너 포인트의 전역적 배치 전략

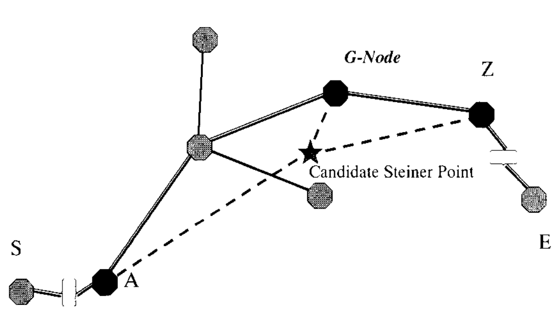
(그림 2)의 (A)는 전역적 배치에 대한 방법을 보인다. 노드 S와 E 사이의 경로상의 어느 에지에서 *G-Condition*이 만족되지 않으면 후보 스타이너 포인트를 다음과 같이 생성된다. 세 개의 노드 G-노드, 노드 S, 노드 E를 이용해 후보 스타이너 포인트를 생성하고 이것을 세 노드와 각각 연결한다. 여기서 G-노드란 S와 E사이에 존재하는 노드로 Steiner 길이를 최소화 시키는 노드이다. 네트워크 구축비용을 절약 및 트리 구조를 유지하기 위해 불필요한 기존의 연결들은 삭제한다.

3.2.2 스타이너 포인트의 지역적 배치 전략

(그림 2)의 (B)에는 지역적 배치에 관한 방법이 표현된다. 노드 S와 E 사이의 어떤 경로상에서 *G-Condition*이 위배가 되면 실제 위반되는 서브경로의 시작 노드 A와 마지막 노드 Z를 찾는다. 그 후에 노드 A와 Z, 그리고 G-Node를 이용해서 후보 스타이너 포인트를 생성하는데 G-노드는 경로 A와 Z사이에 존재하는 네트워크 구성 비용을 최소화 시키는 노드이다. 새로운 연결로 인해 생기는 불필요한 기존의 연결들은 네트워크 구축 비용의 증가를 방지하기 위해 제거한다.



(A) 전역적

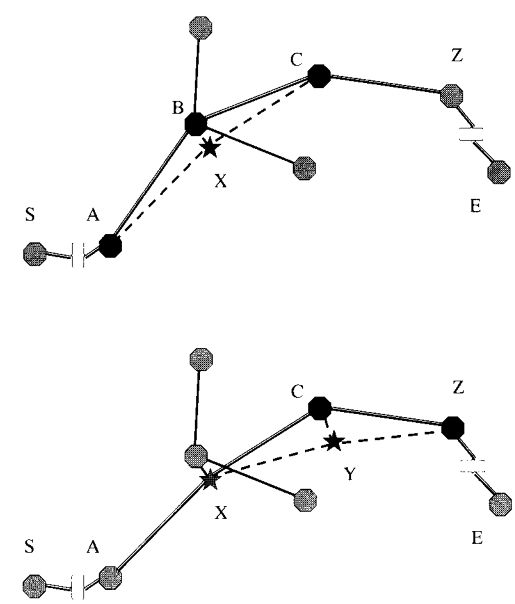


(B) 지역적

(그림 2) 생성되는 스타이너 포인트의 배치 및 연결을 위한 두 가지 전략

3.2.3 스타이너 포인트의 직접적 배치 전략

(그림 3)은 직접적 스타이너 포인트 배치 방법을 보이고 있다. 즉, 경로 S와 E사이의 서브경로 A와 Z에서 *G-Condition*의 위반이 발견되면 해당 경로의 연속적인 세 개의 노드 A, B, C를 이용해서 후보 스타이너 포인트를 생성한다. 이 후보 스타이너 포인트를 이용한 비용이 중간 노드 B의 위치에 처리용량을 높여 계산한 신장 트리 비용보다 작다면 이 후보 스타이너 포인트는 최종 스타이너 포인트로 결정된다. 그렇지 않다면 노드 B의 위치가 새로 생성되는 스타이너 포인트의 위치가 된다. 새로 생성된 스타이너 포인트 X와 다음 연속 노드인 C와 Z를 이용해서 또 다른 스타이너 포인트를 생성하게 된다. 이러한 작업은 *G-Condition*을 위반한 경로가 네트워크에서 모두 사라질 때까지 반복하게 된다.



(그림 3) 직접적 스타이너 포인트 배치 전략

4. GOSST 문제를 위한 개선된 휴리스틱

본 논문에서 제안하는 GOSST 방법은 스타이너 포인트의 배치에 관한 새로운 방법인 지그재그 배치 방법(Zigzag Steiner Point Locating Strategy)을 기반으로 한다.

4.1 스타이너 포인트의 지그재그 배치 전략

지그재그 배치 방법은 다음과 같은 단계로 실행한다.

Step 1: 어떤 터미널 노드 start와 end 사이의 경로에 위치하는 G-Condition을 위반하는 노드 fparent 와 bparent 사이의 서브 경로를 찾는다.

Step 2: 연속적인 2개의 노드와 두 번째 노드와 인접한 노드들을 이용해서 후보 G-Node들을 생성한다. 두 연속 노드란 fparent 와 bparent 사이의 경로

상의 연속적으로 위치한 두 개의 노드이다. 이 후보 G-Node들 각각과 연속적인 두 노드를 이용해서 모든 후보 스타이너 포인트들을 생성하고 이들 중에서 가장 비용 이익이 큰 것을 선택하여 G-Node와 최종 후보 스타이너 포인트를 결정한다.

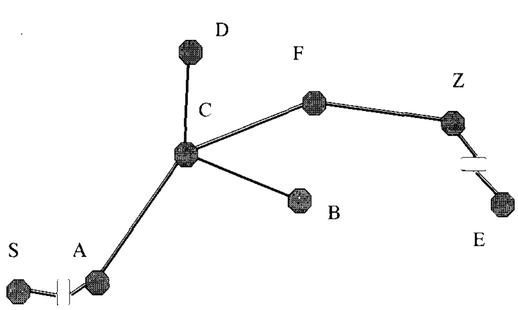
Step 3-a: 이전 단계에서 결정된 후보 스타이너 포인트를 이용한 Steiner 비용이 Spanning 비용보다 적을 경우에는 후보 스타이너 포인트를 최종 스타이너 포인트로 결정하고 불필요한 기존의 연결들은 비용을 감소하기 위해 삭제한다.

Step 3-b: 이전단계에서 결정된 후보 스타이너 포인트를 이용한 Steiner 비용이 Spanning 비용보다 큰 경우에는 후보 스타이너 포인트를 버리고, fparent와 bparent 사이의 경로에 연속적으로 위치하는 두 번째 노드 위에 최종 스타이너 포인트를 생성하고 적절한 연결을 만들고 불필요한 기존의 연결들은 비용을 감소하기 위해 삭제한다.

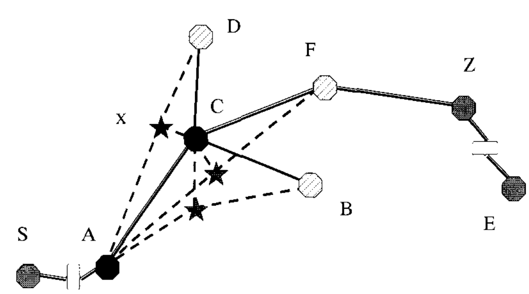
Step 4: 터미널 노드 start와 end 사이의 어떠한 서브경로에서도 G-Condition을 위반하지 않도록 Step 1부터 Step 3 까지를 반복한다.

Step 5: 현재 구성된 네트워크의 모든 터미널 노드로 생성되는 모든 경로에 대해 G-Condition을 위반하지 않도록 Step 1부터 Step 4 까지를 반복한다.

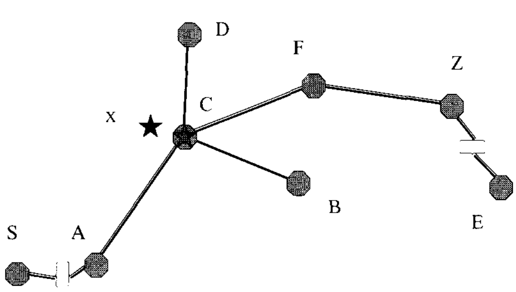
(그림 4)의 (A)는 지그재그 배치 방법을 설명하기 위한 네트워크의 예이다. 검증 경로인 노드 S와 노드 E 사이에 위치하는 서브 경로 A와 Z 사이에 *G-Condition*의 위반이 발생한다고 가정한다. (그림 4)의 (B)에는 후보 스타이너 포



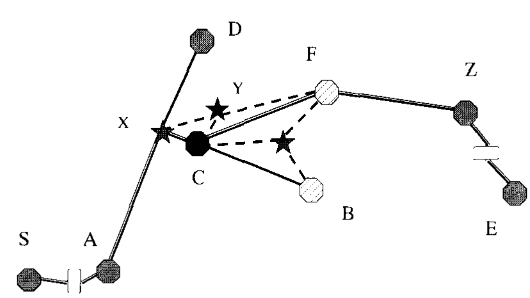
(A) 네트워크 예



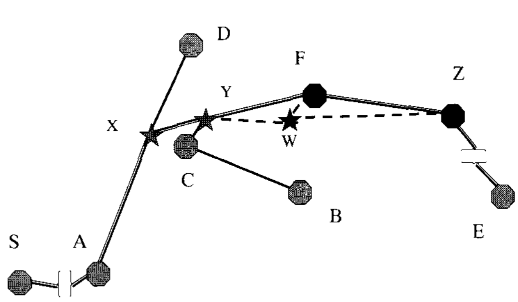
(B) 후보 스타이너 포인트들의 생성 및 최종 후보 스타이너 포인트 X의 선정



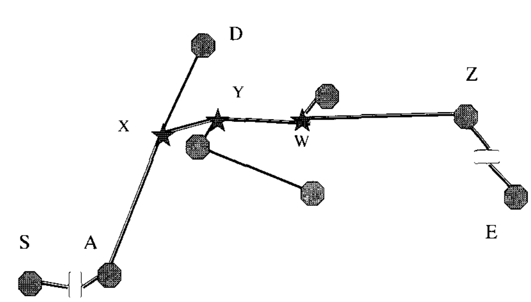
(C) 최종 후보 Steiner의 폐기 및 스타이너 포인트의 C위치에 생성



(D) 최종 스타이너 포인트 X의 생성 및 다음 최종 후보 스타이너 포인트 Y의 결정



(E) 다음 최종 후보 스타이너 포인트 W의 결정



(F) 최종 네트워크

(그림 4) 지그재그 스타이너 포인트 배치 전략 과정

인트 X를 찾는 방법이 표시되어 있다. *G-Condition*의 위반이 발생한 경로의 첫번째 노드인 A와 두 번째 노드인 C 그리고 C에 인접한 D, F, B로 3개의 삼각형 ACD, ACF, ACB를 각각 생성한다. 두 번째 노드 C에 인접한 노드들을 후보 G-Node라 한다. 각 삼각형을 이용해 생성되는 3개의 후보 스타이너 포인트 중에서 Spanning 비용에 비해 Steiner 비용이 가장 절약되는 후보 스타이너 포인트를 최종 후보 스타이너 포인트로 결정한다. 그러나 (그림 4)의 (C)와 같이, 선택된 최종 스타이너 포인트로 계산되는 Steiner 비용이 Spanning 비용보다 크다면 선정된 최종 후보 스타이너 포인트는 폐기되고 두 번째 노드인 C의 위치에 최종 스타이너 포인트가 생성된다. 여기서 Spanning 비용이란 세 개의 노드로 생성되는 삼각형의 가장 작은 두 변의 합이고 Steiner 비용이란 후보 스타이너 포인트로부터 해당 삼각형의 세 점까지의 거리의 합이다. 생성되는 스타이너 포인트의 처리 용량은 노드 S와 노드 E의 처리 용량의 최소값으로 결정한다. Steiner 비용이 Spanning 비용보다 적다면 (그림 4)의 (D)와 같이 최종 후보 스타이너 포인트 X가 최종 스타이너 포인트로 선택되고 적절한 연결을 생성하고 불필요한 기존의 연결들을 제거한다. 계속해서 경로 S와 E의 다음 서브 경로에 대해 *G-Condition*을 검증한다. (그림 4)의 (D)와 같이 서브경로 X와 Z사이에 *G-Condition*이 위반되면 두 노드 X, C와 노드 C에 인접한 노드 F와 B를 이용해서 두개의 삼각형 XCF와 XCB를 만들고 이를 이용해 두개의 후보 스타이너 포인트들을 생성한다. 후보 스타이너 포인트 중에서 최종 후보 스타이너 포인트를 W로 결정하고 이를 Spanning 거리와 비교하여 최종 스타이너 포인트로 선정할 지를 결정하게 된다. (그림 4)의 (E)와 (F)는 서브 경로 A와 Z 사이의 *G-Condition*의 위반을 스타이너 포인트를 생성해서 해결해가는 나머지 과정을 보이고 있다.

4.2 순수 지그재그 GOSST 휴리스틱

순수 지그재그(Naïve Zigzag) GOSST 방법은 순수 최소 신장 트리 생성 전략과 지그재그 스타이너 포인트 배치 전략의 결합이다. 단계별 수행 내용은 다음과 같다.

Step 1: 주어진 터미널 노드들과 에지들에 대해 거리와 처리용량의 곱인 가중치를 이용해서 초기 순수 최소 신장 트리를 생성한다.

Step 2: 생성된 트리(네트워크)의 모든 터미널 노드의 쌍으로 생성되는 경로에 대해 G-Condition을 검사한다.

Step 3: 어떤 터미널 노드의 start와 end의 경로에서 G-Condition 이 위반되었다면, 최종 스타이너 포인트를 앞에서 기술한 지그재그 배치 전략을 이용해서 생성하여 위치시킨다.

Step 4: 변경된 노드들과 에지를 이용해서 순수 최소 신장 트리를 재구성한다.

Step 5: 주어진 네트워크의 모든 터미널 노드의 쌍의 경로에 대해 G-Condition 을 만족할 때 까지 Step 2부터 Step 4를 반복한다.

4.3 거리우선 지그재그 GOSST 휴리스틱

거리 지그재그(Distance Zigzag) GOSST 방법은 거리우선 최소 신장 트리 생성 전략과 지그재그 스타이너 포인트 배치 전략을 사용한다. 단계 별 수행 내용은 다음과 같다.

Step 1: 주어진 터미널 노드들과 에지들의 거리와 처리용량을 이용해서 초기 거리우선 최소 신장 트리를 생성한다.

Step 2: 생성된 트리(네트워크)의 모든 터미널 노드의 쌍의 경로에 대해 G-Condition을 검사한다.

Step 3: 어떤 터미널 노드 start와 end의 경로에서 G-Condition이 위반되었다면, 최종 스타이너 포인트를 지그재그 배치 전략을 이용해서 생성하고 위치시킨다.

Step 4: 변경된 노드들과 에지를 적용해서 거리우선 최소 신장 트리를 재구성한다.

Step 5: 주어진 네트워크의 모든 터미널 노드의 쌍의 경로에 대해 G-Condition을 만족할 때 까지 Step 2부터 Step 4를 반복한다.

5. 실험 및 결과분석

본래의 GOSST 문제는 입력으로 에지와 관련된 정보를 다루지 않는다. 즉 처리 용량을 가지는 터미널 노드 정보만 주어진다. 그러나 본 연구에서는 GOSST 문제의 일반화를 위해서 어떤 에지 정보를 문제의 입력으로 추가하였다.

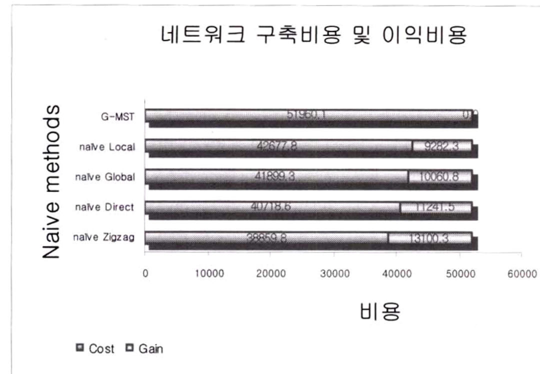
본 연구의 실험을 위한 입력에는 노드의 개수, 한 노드당 최대 연결 수, 노드의 처리 용량의 종류 수, 노드의 이름 및 그 노드의 유클리드 평면에서의 x, y 좌표 그리고 연결 정보 등이다. 최소 신장 트리는 Prim의 알고리즘을 기반으로 생성하게 하였다. 세 개의 점으로 생성되는 스타이너 포인트의 생성 및 그것과 연관된 연결을 위해 이와 관련된 휴리스틱이 사용되었다. *G-Condition*이 만족하지 않을 때마다 스타이너 포인트가 생성되었고, 네트워크 구축 비용 및 트리를 생성하기 위해 불필요한 연결들은 삭제되었다. 새로운 스타이너 포인트 및 연결의 생성 후에는 이를 바탕으로 한 최소 신장 트리를 재구성하였고, 이러한 작업은 네트워크의 모든 터미널 노드로 구성되는 경로에서 *G-Condition*을 만족할 때까지 반복하게 된다. 본 연구에서 사용된 실험 인자들은 다음과 같다. 즉, 터미널 노드 수의 수는 25, 50, 75, 100 이고 이 노드들의 위치는 무작위로 생성된다. 각 노드

에서 연결 가능한 최대 연결의 수는 3, 5, 7, 9 이고 이 연결 또한 무작위로 생성된다. 각 노드의 처리 용량의 종류는 3, 5, 7, 9 이고 이 범위 내에서 각 노드들은 자신의 처리용량을 무작위로 배정 받게 된다. 분석하게 될 GOSST 방법은 본 논문에서 제안하는 지그재그 배치 방법을 적용한 Naïve Zigzag GOSST 와 Distance Zigzag GOSST 이고 이것들의 결과는 이전 우리의 논문에서 제안했던 6가지 방법[13,14]과 컨트롤인 G-MST와 비교하게 된다. 실험은 위의 실험 인자들의 조합에 의해 512 (4×4×4×8) 경우를 행하게 된다. 즉 4개 그룹의 터미널 노드의 수와 4개의 각 노드 당 최대 연결의 수와 각 노드의 4 종류의 처리 용량의 종류와 8가지 GOSST 방법의 조합에 대한 실행이다. 각 8개의 GOSST 방법은 64번(4×4×4) 실험의 결과로 네트워크 구축 비용, 생성되는 스타이너 포인트의 수, 실행시간을 얻는다. 또한 비용 절약 비율은 이 64번의 실험 결과의 평균으로 얻는다. 본 연구의 컨트롤로 G-MST (Grade of service Minimum Spanning Tree)를 사용했는데, 이것은 최초의 최소 신장 트리의 변형으로 트리 내의 각 노드의 처리능력을 *G-Condition*을 만족시키도록 최소 증가시킨 것이다. 본 실험은 Intel Pentium4 (M) 1.83 GH, 1 Giga Memory의 Laptop에서 실행시켰으며, 각 GOSST 휴리스틱은 Microsoft Visual C++로 구현하였다.

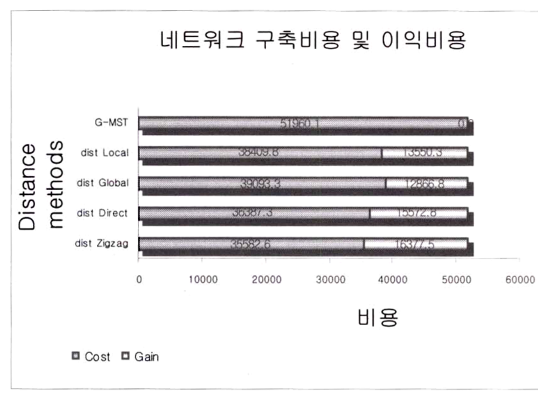
5.1 제안된 지그재그 배치 전략에 대한 실험

(그림 5)에는 지그재그 배치 방법을 사용한 네트워크 구축 비용을 비교한 것이다. (그림 5)의 (A)는 순수 최소 신장 트리 생성 방법을 적용한 결과이고 (B)는 거리 우선 최소 신장 트리 생성방법을 적용한 결과이다. 본 연구에서 제안한 지그재그 배치 방법이 두 개의 최소 신장 트리 생성방법 모두에 대해서, 다른 GOSST 휴리스틱 방법과 G-MST보다 네트워크 구축비용이 적음을 확인할 수 있다. 두 최소 신장 트리 구축 방식 모두에 대해 새로운 방식을 적용한 지그재그 방법이 가장 적은 네트워크 구축 비용과 가장 많은 이익비용을 보였다. 또한 모든 배치 방법에 대해 거리우선 최소 신장 트리 생성 방식이 순수 지역 최소 신장 트리 생성 방식 보다 네트워크 구축비용이 적다. 순수 지그재그 방식은 순수 직접방식보다 4.6%, 거리우선 지그재그 방식은 거리우선 직접 방식보다 2.2%의 비용 절감을 얻었다.

(그림 6)은 본 연구에서 제안한 지그재그 배치 방식에 대한 성능을 확인하기 위해 4개의 배치 방식에 대한 결과를 보인다. 4개의 배치 방식에 대해, 거리우선 최소 신장 트리 생성 방식이 순수 최소 신장 트리 생성방식보다 비용이 적게 소요된다. 사용한 최소 신장 트리 생성 방식에 따라 가장 비용 차이가 많은 배치 방식은 직접 배치 방식으로 거리우선 직접 배치 방식이 순수 직접 배치 방식보다 평균 10.6 %의 비용 절감효과를 얻는다. 반면에 가장 적은 것은 전역 배치 방식으로 6.7%의 비용절감을 얻는다

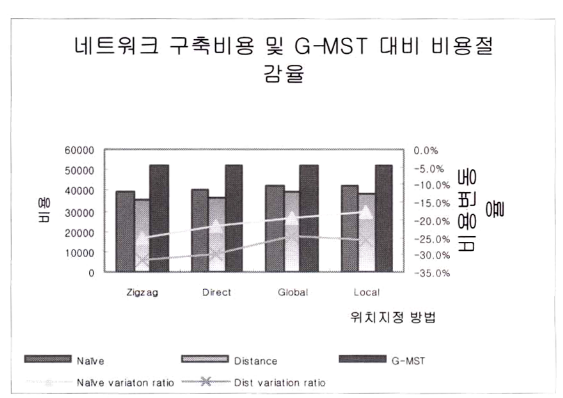


(A) 순수 최소 신장 트리 생성 방식을 이용한 GOSST 방법



(B) 거리우선 최소 신장 트리 생성 방식을 이용한 GOSST 방법

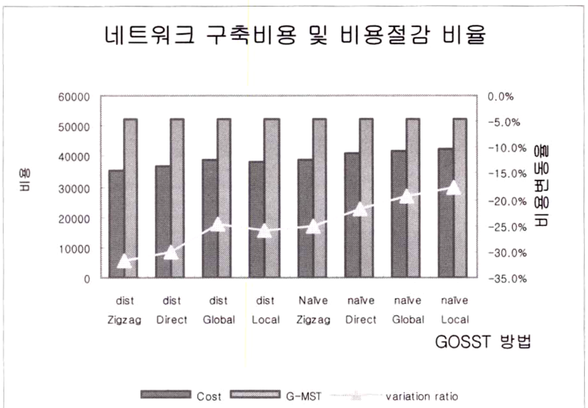
(그림 5) 최소 신장 트리 생성 전략에 따른 GOSST 방법들의 네트워크 구축 비용 및 G-MST 대비 절감 비용



(그림 6) 4개의 배치 방법에 대한 네트워크 구축비용 및 G-MST 대비 비용 절감 비율

5.2 GOSST 문제를 위한 8개 GOSST 휴리스틱의 네트워크 구축비용 비교

(그림 7)은 64 경우의 실험에 대한 누적 결과로 8개의 GOSST 방법의 네트워크 구축 비용 및 컨트롤인 G-MST와의 비용 차이 비율을 보이고 있다. 본 논문에서 제안하는 거리우선 지그재그 GOSST 방법이 가장 큰 비용 절감

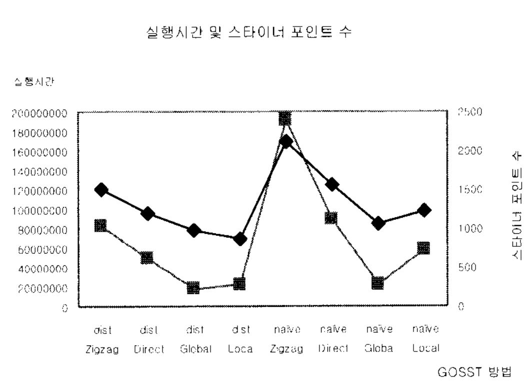


(그림 7) 누적 네트워크 구축 비용 및 G-MST 대비 비용 절감비율

(31.5%)을 보이고 반면에 순수 지역 GOSST 방법은 가장 작은 비용절감 (17.9%)을 나타낸다. 또한 거리 우선 지그재그 GOSST 방식은 순수 지그재그 GOSST 방식보다 8.4%의 비용 절감을 보이고 우리의 이전 연구에서 가장 좋은 성능을 보였던 거리 우선 직접 방식보다 2.2%의 비용절감을 보인다. 배치 방식에 있어, 지그재그 방식이, 직접 방식에 비해 3.5%, 전역방식에 비해 8.1%, 지역 방식에 비해 8.2%의 비용 절감을 얻을 수 있다.

5.3 실행시간과 생성된 스타이너 포인트의 수에 대한 실험

그림 8에는 8개의 GOSST 방식의 실행시간과 생성되는 스타이너 포인트의 수가 표현되어 있다. 순수 지그재그 방법이 가장 많은 실행시간이 필요로 하고 거리 우선 전역 방법이 가장 적은 시간이 소요된다. 순수 지그재그 방법이 가장 많은 스타이너 포인트를 생성하며 반면에 거리 우선 지역 방법이 가장 적은 스타이너 포인트를 생성한다. 순수 최소 신장 트리 생성 전략이 거리 우선 최소 신장 트리 생성 전략보다 실행시간과 생성되는 스타이너 포인트의 수가 많다. 지그재그 배치 방식이 이전 연구에서 제안하였던 직접, 지역, 전역 방식보다 많은 실행시간과 스타이너 포인트들을 요구한다. 본 연구에서 제안한 거리 지그재그 방식은 이전 연구의 가장 좋은 성능의 거리 직접방식에 비해 실행시간은 66.9%, 스타이너 포인트의 수는 27.2%의 오버헤드가 발생하였다.



(그림 8) 실행 시간 및 생성되는 스타이너 포인트 수

6. 결 론

본 연구는 이전의 우리의 연구결과[13,14] 보다 성능이 개선된 GOSST 휴리스틱에 관한 것이다. GOSST 문제는 네트워크 내의 모든 노드들 사이의 경로에서 *G-Condition*을 만족하는 최소구축 비용의 네트워크 토폴로지를 구하는 것이다. 이 문제에 대한 연구는 최적화에 집중되어 왔고 연구자들은 이 문제가 스타이너 트리 문제와 연관된 문제로 간주하여 GOSST에 관한 휴리스틱은 큰 관심을 보이지 않았기 때문에 실제 응용에 관한 연구는 많이 발표되지 않은 것 같다. 그러나 이 문제는 실제적으로 적용 가능한 분야가 많기 때문에 실용적인 시간 복잡도를 갖는 휴리스틱의 구현은 필요하다고 생각한다.

본 논문에서는 개선된 GOSST 휴리스틱을 위해 새로운 스타이너 포인트 배치전략으로 지그재그 방법을 제안한다. 이전 연구의 직접 배치 방법과의 차이점은 후보 스타이너 포인트를 얻기 위한 G-노드의 고려 대상을 대상 서브 경로 상에 위치하는 두 번째 노드의 인접 노드까지 확장한 것이다. 이 방식을 이용한 배치전략은 이전의 지역 방식 [13]에 비교해 8.2%, 전역방식에 비교해 8.1% 그리고 직접방식[14]에 비교해 3.5%의 비용절감을 얻을 수 있었다. 스타이너 포인트 배치전략과 최소 신장 트리 생성전략을 조합하여 우리는 이전 연구에서 제안했던 6개의 휴리스틱 방법 [13,14] 이외에 2개의 새로운 GOSST 휴리스틱 방법을 제안하였다. GOSST 문제에 대한 8개의 방법 중에서 본 논문에서 새로 제안한 거리 우선 지그재그 GOSST 방법이 G-MST에 대해 가장 큰 비용절약(31.5%)을 보였고, 순수 지역 GOSST 방법이 가장 적은 비용절약(17.9%)을 보였다. 거리 우선 지그재그 GOSST 방법은 이전 연구에서 가장 큰 비용 절약방법 이었던 거리 우선 직접 GOSST 방법에 비해서는 약 2.2%의 비용개선을 얻을 수 있었다. 실행시간과 생성되는 스타이너 포인트의 수에 대해서, 순수 지그재그 GOSST 방법은 가장 많은 실행시간을 필요로 했으며, 반면에 거리 전역 GOSST 방법은 가장 적은 시간이 소요되었다. 순수 지그재그 GOSST 방법은 가장 많은 스타이너 포인트를 생성해야 했으며 반면에 거리 지역 GOSST 방법은 가장 적은 스타이너 포인트를 요구하였다. 분석결과 순수 최소 신장 트리 생성 전략은 거리 우선 최소 신장 트리 생성 전략에 비해 많은 실행시간과 생성되는 스타이너 포인트의 수를 필요로 하였다. 지그재그 배치 전략은 이전 연구의 배치 전략에 비해 많은 스타이너 포인트의 수와 실행시간을 필요로

하였다. 그러나 이 오버헤드는 GOSST 문제의 목적과 NP 문제의 특성 상 감당할 수 있는 정도라고 판단된다.

앞으로 우리의 실험결과에 대한 좀 더 세밀한 분석 및 평가가 필요하다. 또한 제안된 방법에 대한 좀 더 많은 연구와 개선이 수행될 것이다. 성능 개선, 실행시간 및 생성되는 스타이너 포인트의 수, 비용 및 비용절약 비율, 네트워크의 용량 이득 및 네트워크 길이 오버헤드에 대한 연구가 요구된다. GOSST 휴리스틱을 실제로 적용할 수 있는 유용한 어플리케이션의 개발도 필요하다.

참 고 문 헌

[1] D.Z. Du and F.K. Hwang, "An Approach for Providing Lower Bounds; Solution of Gilbert-Pollak Conjecture on Steiner Ratio", Proceedings of IEEE 31sy FOCS, pp.76-85, 1990.

[2] G.L. Xue, G.H. Lin and D.Z. Du, "Grade of Service Steiner Minimum Trees in Euclidean Plane", Algorithmica, Vol.31, pp.479-500, 2001.

[3] J. Kim and I. Kim, "Approximation Ratio 2 for the Minimum Number of Steiner Points", Journal of KISS, pp.387-396, 2003

[4] J. Kim, M. Cardei, I. Cardei and X. Jia, "A Polynomial Time Approximation Scheme for the Grade of Service Steiner Minimum Tree Problem", Algorithmica, Vol.42, pp.109-120, 2005.

[5] S. Arora, "Polynomial-Time Approximation Schemes for Euclidean TSP and Other Geometric Problems", Proceeding of 37th IEEE Symposium on Foundations of Computer Science, pp.2-12, 1996.

[6] T.H. Cormen, C.E. Leiserson, R.L. Rivest and C. Stein, Introduction to Algorithms, 2nd Ed., MIT Press, 2001.

[7] E.J. Cockayne and D.E. Hewgrill, "Exact Computation of Steiner Minimal Trees in the Plane", Information Processing Letters, Vol.22, pp.151-156, 1986.

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

[9] F.K. Hwang, "A Primer of the Euclidean Steiner problem", Annals of Operations Research, Vol.33, pp.73-84, 1991.

[10] F.K. Hwang, D.S. Richards and P. Winter, "The Steiner Tree Problem", Annals of Discrete Mathematics, Vol.53, North-Holland, 1992.

[11] M.J. Smith and B. Toppur, "Euclidean Steiner Minimal Trees, Minimum Energy Configurations and the Embedding Problem of Weighted Graph in E3", Discrete Applied Mathematics, Vol.71, pp.187-215, 1997.

[12] P. Winter, "An Algorithm for the Steiner Problem in the Euclidean Plane", Networks, Vol.15, pp.323-345, 1985.

[13] I. Kim, C. Kim and S.H. Hosseini, "A Heuristic Using GOSST with 2 Connecting Strategies for Minimum Construction Cost of Network", International Journal of Computer Science and Network Security, Vol.6, No.12, pp.60-72, 2006.

[14] I. Kim and C. Kim, "An Enhanced Heuristic Using Direct Steiner Point Locating and Distance Preferring MST Building Strategy for GOSST Problem", International Journal of Computer Science and Network Security, Vol.7, No.2, pp.164-175, 2007.

[15] A. Balakrishnan, T.L. Nagnanti and P. Mirchandani, "Modeling and Heuristic Worst Case Performance Analysis of the Two Level Network Design Problem", Management Science, Vol.40, pp.846-867, 1994.

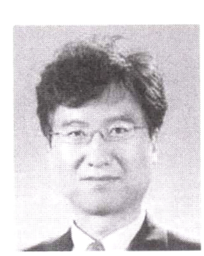
[16] C. Duin and A. Volgenant, "The Multi-weighted Steiner Tree Problem", Annals of Operations Research, Vol.33, pp.451-469, 1991.

[17] G.H. Lin and G.L. Xue , "Steiner Tree Problem with Minimum Number of Steiner Points and Bounded Edge-length", Information Processing Letters, pp.53-57, 1999.

[18] J.R. Current, C.S. Revelle and J.L. Cohon, "The Hierarchical Network Design Problem", European Journal of Operational Research, Vol.27, pp.57-66, 1986.

[19] P. Mirchandani, "The Multi-tier Tree Problem", INFORMS Journal on Computing, Vol.8, pp.202-218, 1996.

[20] P. Winter and M. Zachariasen, "Large Euclidean Steiner Minimum Trees: An Improved Exact Algorithm", Networks, Vol.30, pp.149-166, 1998.



김 인 범

e-mail : ibkim@kimpo.ac.kr

1989년 서울대학교 컴퓨터공학과(학사)

1991년 서울대학교 공과대학원 컴퓨터공학과 (공학석사)

현재 University of Wisconsin-Milwaukee, EE&CS (박사과정)

1991년~1995년 대우통신, 한국 오라클 근무

1996년~현재 김포대학 인터넷정보과 부교수

관심분야 : 컴퓨터이론, 네트워크, 데이터베이스 등

김 재 각

e-mail : ckkim@kimpo.ac.kr

1981년 숭실대학교 전자계산학과(학사)

1985년 연세대학교 산업대학원 전자계산 전공(공학석사)

2002년 숭실대학교 대학원 컴퓨터공학과 (공학박사)

1985년~1994년 LG전자, 삼보컴퓨터 근무

1996년~현재 김포대학 인터넷정보과 부교수

관심분야 : 암호이론, 시스템보안, 인터넷응용 등