대한산업공학회/한국경영과학회 2002 춘계공동학술대회
한국과학기술원(KAIST) 2002년 5월 3일-4일

추가제약이 있는 최소 신장나무 문제에 대한 유사다항시간 알고리듬 및 근사 해법

A pseudo-polynomial algorithm and approximation algorithm for the constrained minimum spanning tree problem

홍 성 필*, 정 성 진**, 박 범 환**
(* : 중앙대학교 상경학부, ** : 서울대학교 산업공학과)

Abstract

본 연구는 추가제약이 있는 최소 신장나무 문제(Constrained Minimum Spanning Tree : *CMST*문제)에 대한 유사다항시간 알고리듬 및 근사 해법 개발에 관한 것이다. *CMST*문제는 *NP-hard*문제임이 이미 증명 되었으며, 이후 이 문제에 대해서는 근사해법 개발이 주된 관심이 되어왔다. [Ravi and Goemans 96]는 다항시간 근사해법(*PTAS*)을 이미 개발하였고, [Marathe et al 98]은 가능해(feasible solution)는 아니지만, 앞으로 서술할 $(1+1/\varepsilon, 1+\varepsilon)$근사해를 구하는 완전다항시간 근사해법(*FPTAS*)을 제시하였다.
이와는 달리 [Papa. and Yan, 00]는 파레토 근사 최적해를 구하는 *FPTAS*를 제시하였는데, 본 연구는 이들의 연구에서 주로 의존하고 있는 행렬-나무 정리(Tree-Matrix Theorem)를 보다 일반화하여, *CMST*문제에 대한 유사다항시간 알고리듬과 $(1+\varepsilon, 1+\varepsilon)$근사해를 구하는 *FPTAS*를 제시할 것이다.

1. 서론

추가제약이 있는 최소 신장나무 문제(Constrained Minimum Spanning Tree Problem, 이하 *CMST*문제)는, 호 (i,j)에 대해 두 개의 비용구조 c_{ij}(길이,length)와 d_{ij}(비중,weight)를 가지고 있는 그래프 $G=(V,E)$상에서, 비중에 대한 제약이 주어졌을 때, 최소 길이를 갖는 신장나무를 구하는 문제이다.

*CMST*문제

$$\min\ c(T)$$
$$s.t\ \ d(T)\le D$$
T : spanning tree

[Aggarwal 82]은 이 문제가 *NP-hard*문제임을 증명하였고, 라그랑지안 완화법(Lagrangean Relaxation)을 이용하여 상한(upper bound)에 해당하는 해를 구한 후, 분지한계법(Branch and Bound)을 이용하여 최적해를 구하는 방법을 제시하였다. 이 해법은 비중에 대한 제약식을 완화한 완화문제는 다항시간 안에 풀 수 있다는([chandrasekaran 77]) 사실을 이용한 것이며, 라그랑지안 완화를 이용하는 대부분의 알고리듬은 이 사실에 기반하고 있다.

이 후 [Ravi and Goemans 96]는 *CMST*문제의 제약식을 다소 완화하여 '(α,β)-근사해' 개념을 제시하였다. (α,β)-근사해란, 비중에 대한 제약식이 보다 완화되어, $d(T)\le(1+\alpha)D$를 만족하면서, 원래 *CMST*문제의 최적해의 목적함수값을 OPT라 했을 때, $c(T)\le(1+\beta)OPT$를 만족하는 해 T를 말한다. 그들은 *CMST*문제의 라그랑지안 완화 문제의 최적해들이 항상 연결(connected)되어 있음을 이용하여 간단한 (2,1)-근사해를 구하는 해법을 제시하였으며, 이것에 기초한 $(1+\varepsilon,1)$근사해를 다항시간안에 계산하는 다항시간 근사해법(Polynomial Time Approximation Sche mes : *PTAS*)을 제안하였다. 이것은 간단한 이진탐색(Binary Search)를 이용하

여 $(1, 1+\varepsilon)$-근사해로 전환할 수 있으며 이 것은 곧 *CMST*문제에 대한 *PTAS*가 된다.

[Marathe et al 98]은 Parametric Search를 이용하여, $(1+1/\varepsilon, 1+\varepsilon)$-근사해를 구하는 완전다항시간 근사해법(Fully Polynomial Time Approximation Schemes : *FPTAS*)을 제시하였다. 물론 이 문제는 제약식을 만족시키지 못하므로, *CMST*문제에 대한 정확한 *FPTAS*라고는 할 수 없다.

비중에 대한 제약식을 없애고, 길이와 비중을 모두 목적함수로 본, 두가지 척도 신장나무 문제(Bicriterion Minimum Spann ing Tree Problem : *BMST*문제)에 대한 연구도 활발하다.([Ehrgott and Klamroth, 96], [Andersen et al 96],[Ramos et al, 98])

*BMST*문제
min $c(T)$
min $d(T)$
s.t T : spanning tree

두가지 비용을 모두 목적함수로 본다는 것은, 파레토 최적해(Pareto Optimal Solution)를 구함을 의미하는데, 파레토 최적해란, 서로를 '지배(dominate)'하지 않는, 즉 길이와 비중 모두에서 우월한 해를 포함하지 않는 해들의 집합을 말한다. 일반적인 다목적 선형계획법에서 모든 파레토 최적해는 목적함수들의 일차결합(linear combination)을 목적함수로 하는 단일 선형계획법의 최적해로 구할 수 있지만 [Andersen et al 96], 최단경로나 최소 신장나무문제와 같은 조합적 문제(Combinatorial Problems)에 대해서는, 문제의 이산적인(discrete) 성질 때문에, 일부의 파레토 최적해만을 구할 수 있다.([Ehrgott and Klamroth, 96] 이러한 문제들로 인해, 위 논문들은 모두, 두 개의 목적함수의 볼록조합으로 표현가능한 해들을 구한 다음, 휴리스틱을 이용하여 나머지 해들을 구하는 방법을 제시하였다.

이러한 파레토 최적해에 대한 가장 최근 연구인 [Papa. and Yann. 00]의 연구에서는, 파레토 최적해가 아닌, ε-근사 파레토 최적해 개념을 도입하여, ε-근사 파레토해들을 완전 다항시간안에 계산하는 *FPTAS*를 제시하였다. 이들은 기존의 조합적인 방법들과는 달리, 길이와 비중을 비례축소한 후, 행렬-나무 정리(Tree-Matrix Theorem)을 이용하여 *FPT AS*를 유도하였다.

위의 논문들 중 가장 좋은 계산 복잡도를 갖는 알고리듬을 정리하면, 먼저 *CMST*문제에 대해서는 *PTAS*([Ravi and Goemans, 96])와 $(1+1/\varepsilon, 1+\varepsilon)$-근사 *FPTAS*([Mara the et al 98])가 있으며, *BMST*문제에 대해서는 ε-근사 파레토 최적해를 구하기 위한 *FPTAS* [Papa. and Yann. 00]가 있다.

본 논문은 [Papa. and Yann 00]에서 주로 의존하고 있는 행렬-나무 정리를 보다 일반화하여, *CMST*에 대한 간단한 유사다항시간 알고리듬(Pseudo-Polynomial Time Algorithm)과 이것에 기반한 $(1+\varepsilon, 1+\varepsilon)$-근사 *FPTAS*를 제시할 것이다.

2장에서는 행렬-나무정리에 대해 서술하고, 이것을 기초로, *CMST*문제에 대한 유사다항시간 알고리듬을 제시하며, 3장에서는 $(1+\varepsilon, 1+\varepsilon)$-근사 *FPTAS*을 구성한다.

2. 행렬-나무 정리와 유사다항시간 해법

2.1 행렬-나무 정리

먼저 주어진 그래프 $G=(V, E)$상에 호의 비용 c_{ij}가 주어져 있을 때, 다음과 같은 n×n행렬 $L=(l_{ij})$를 정의 하자. (이 행렬을 라플라시안 행렬(Laplacian Matrix)라 한다.)

$$l_{ij}=\begin{cases}\sum_{k:(i,k)\in E} c_{ik} & \text{if } i=j\\ -c_{ij}, & \text{if } i\neq j, (i,j)\in E\\ 0, & otherwise\end{cases}$$

행렬 L상에서 n 번째 행과 열을 삭제한 행렬을 $L[\{\overline{n}\},\{\overline{n}\}]$라 하면 다음이 성립한다.

[정리1] [Even 79]

$$det\, L[\{\overline{n}\},\{\overline{n}\}] = \sum_{T\in S} w(T)$$
$$(\text{단}\quad w(T) = \Pi_{(i,j)\in T} c_{ij})$$

[Barahona and Pulleyblank 87],[broder and Mayr 97]은 이 정리를 변수 x를 도입하여 전체 나무 길이에 대해 세분화된 정보를 얻을 수 있는 나무-행렬 정리를 제시하였다. 앞의 행렬 L과 구별하기 위해 $L^x = (l^x_{ij})$로 두고, l^x_{ij}를 다음과 같이 설정하면,

$$l^x_{ij} = \begin{cases} \sum_{k:(i,k)\in E} x^{c_{ik}} & \text{if } i=j \\ -x^{c_{ij}}, & \text{if } i\neq j,(i,j)\in E \\ 0, & otherwise \end{cases}$$

정리 2가 성립한다.

[정리 2] [Barahona and Pulleyblank 87][broder and Mayr 97]

$$det\, L^x[\{\overline{n}\},\{\overline{n}\}] = \sum_k a_k x^k$$ 이고, a_k는
나무 T의 전체 길이가 k인 나무의 개수.
(단, $k = \sum_{(i,j)\in T} c_{ij} = c(T)$)

[Barahona and Pulleyblank 87]는 위의 정리를 이용하여, "나무의 전체 길이 $c(T)$가 C인 나무가 존재하는가?" 라는 문제에 대한 유사 다항시간 알고리듬을 제시하였다.
이들은 위 정리의 좌변에 있는 $det\, L[\{\overline{n}\},\{\overline{n}\}]$를 계산하기 위하여, 다음과 같은 방법을 이용하였다.
우선, 우변항의 최대 항의 개수를 $p = (n-1)\cdot \max\{c_{ij}\}+1$로 설정할 수 있다.그러면 우변항의 항의 개수는 많아야 p개가 되고, x대신에 p개의 값을 대입한 후, 각각의 행렬식을 계산하면, $p+1$개의 연립 선형 방정식을 얻을 수 있는데, 이것을 풀면, 발생할 수 있는 모든 k와 a_k를 계산 할 수 있다. 이와 같은 방법을 이용했을 경우 전체 계산 시간은 $O((n^3+p^2)p^2\log p)$가 된다.[Barahona and Pulleyblank 87]

[Papa. and Yann. 00]는 목적함수가 여러 개 있을 때, 나무-행렬 이론을 적용하는 방법을 제시하였다. 실제로, 그들은 ε-근사 파레토해를 구하는 $FPTAS$를 유도하기 위해, 목적함수를 비례축소한 후, 다음의 방법을 적용했지만, 그 방법을 비례축소하기 전의 원래 문제로 유추하여 적용해 보면 아래와 같다. 예를 들어 목적함수가 나무의 전체 길이($c(T)$)와 비중($d(T)$)이라고 했을 때, l_{ij}를

$$l_{ij} = \begin{cases} \sum_{k:(i,k)\in E} x^{(p_3+1)c_{ik}+d_{ij}} & \text{if } i=j \\ -x^{(p_3+1)c_{ij}+d_{ij}}, & \text{if } i\neq j,(i,j)\in E \\ 0, & otherwise \end{cases}$$

($p_3 = \max\{p_1,p_2\}$ where
$p_1 = (n-1)\cdot \max\{c_{ij}\}+1$,
$p_2 = (n-1)\cdot \max\{d_{ij}\}+1$)

라 두고, 정리 2의 우변의 식을, 0과 $p_1(p_2)$ 사이에 있는 모든 값을 $\alpha(\beta)$에 대입하고, x대신에 $p_1\cdot p_2$개의 값을 대입하여 행렬식을 구하면, $p_1\cdot p_2$ 연립 선형 방정식을 얻을 수 있다. 이때, $a_{p\alpha+\beta}\neq 0$이라면, $c(T)=\alpha$ $d(T)=\beta$인 나무가 존재함을 의미한다.
그러나, 앞의 계산시간 분석에서, p가 $p_1\cdot p_2$가 됨으로, 계산시간이 비록 유사다항시간이지만, 상당히 많은 계산 시간을 요구한다. 다음 장에서, 행렬식을 보다 안정적이고 빠른 시간 안에 구할 수 있는 조합적인(combinatorial) 알고리듬을 소개할 것이다.
그러나, 계산의 복잡함에도 불구하고, 행렬식의 계산 결과에서, $d(T)\leq D$인 나무들 중에서 최소의 $c(T)$를 갖는 항을 찾으면, 이것이 곧 $CMST$문제에 최적해가 된다. 따라서, $CMST$문제에 대한 유사 다항시간 알고리듬이 존재하게 되어 [정리 3]이 성립한다.

[정리 3] $CMST$문제는 유사다항시간 알고리듬이 존재한다.

두 개의 목적함수를 고려하기 위해 [Papa.

and Yann 00]는 x의 지수에 충분히 큰 수(p_3)를 고려했는데, 그렇게 하지 않고, 아래와 같이, 비중 d_{ij}에 대한 새로운 변수 y를 도입하여, l_{ij}^{xy}를 다음과 같이 정의하면,

$$l_{ij}^{xy}=\begin{cases}\sum_{k:(i,k)\in E} x^{c_{ik}}y^{d_{ik}} & \text{if } i=j\\ -x^{c_{ij}}y^{d_{ij}}, & \text{if } i\neq j,(i,j)\in E\\ 0, & \text{otherwise}\end{cases}$$

[정리 4]가 성립한다.

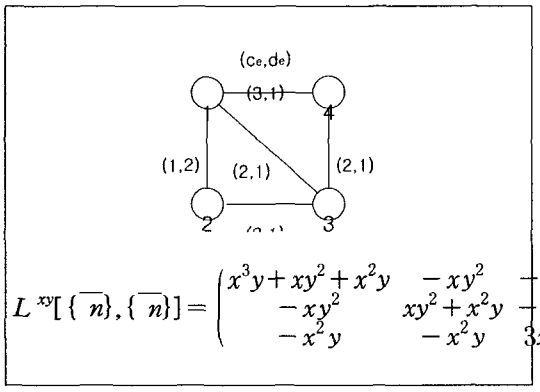
[정리 4]

$$det\,L^{xy}[\{\overline{n}\},\{\overline{n}\}]=\sum_k a_k x^{k_c}y^{k_d}$$ 이고,

a_k는 $c(T)=k_c$, $d(T)=k_d$인 나무의 개수.

(단, $k=\sum_{(i,j)\in T} c_{ij}=c(T)$)

예를 들어, [그림 1]은 주어진 그래프에서 구한 $L^{xy}[\{\overline{n}\},\{\overline{n}\}]$를 나타낸다.



$$L^{xy}[\{\overline{n}\},\{\overline{n}\}]=\begin{pmatrix}x^3y+xy^2+x^2y & -xy^2 & -x^2y\\ -xy^2 & xy^2+x^2y & -x^2y\\ -x^2y & -x^2y & 3x^2y\end{pmatrix}$$

[그림 1] 행렬-나무 정리

위 행렬의 행렬식을 구해보면,

$$det\,L[\{\overline{n}\},\{\overline{n}\}]=3x^6y^4+2x^5x^4+2x^7y^3+x^6y^3$$

가 된다. 여기서 $3x^6y^4$는 나무의 길이가 6이고 비중이 4인 나무의 개수는 3개임을 나타낸다. 나머지 항들도 동일한 해석을 할 수 있다.

다음 절 에서는 $detL[\{\overline{n}\},\{\overline{n}\}]$를 계산하는 보다 효율적인 해법에 대해 서술한다.

2.2. 행렬식 계산을 위한 조합적(combinatorial) 알고리듬

$n\times n$행렬 $A=(a_{ij})$에 대한 행렬식 $det(A)$는 다음과 같이 정의된다.

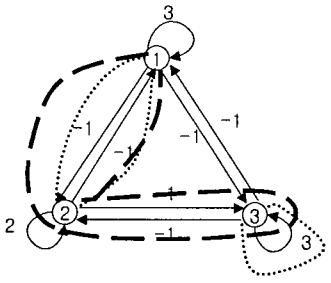
$$det(A)=\sum_{\sigma\in S_n} sgn(\sigma)\Pi_i a_{i\sigma(i)}$$

여기서 S_n은 모든 permutation의 집합이고 $sgn(\sigma)$는 permutation σ이 짝수 치환이냐 홀수 치환이냐에 따라, 각각 1과 -1의 값을 갖는다. 행렬 A의 permutation을 그래프 관점에서 해석하면, 노드 개수가 n이고, 노드 i와 노드 j간의 호의 비용을 a_{ij}로 둔 그래프 $G=(V,E)$상에서의 cycle cover라 볼 수 있다.
따라서, 행렬 A를 표현한 그래프 $G=(V,E)$관점에서 $det(A)$는 다음과 같이 정의할 수 있다.

$$det(A)=\sum_C sgn(C)\Pi_{(i,j)\in C} a_{ij}$$

이 때, $sgn(C)$는 $(-1)^{n+c}$가 된다.(c는 cycle cover C의 크기)

예를 들어, 3×3 행렬 $A=\begin{pmatrix}3 & -1 & -1\\ -1 & 2 & -1\\ -1 & -1 & 3\end{pmatrix}$ 주어졌을 경우, 이것을 그래프로 표현하면 [그림 2]와 같이 되고, [그림 2]에서 가는 점선으로 표시된 cycle cover를 노드 순서로 표현하면 1-2-1, 3-3이 되는데 이것은 permutation (1,2)(3)에 대응된다. 이때 cycle cover의 크기는 2가 된다.



[그림 2] cycle cover and clow

[mahajan and Vinay 99]은 closed walk(clow)의 개념을 도입하여, 보다 간단한 행렬식 계산 방법을 고안하였다. clow란, 노드의 중복을 허용하며, 모든 노드를 포함하지 않아도 되는 환이라 볼 수 있다. [그림 2]에서 굵은 점선으로 표시된 것을 노드 순서에 따라 표현하면 (1,2,3,2)가 된다. 이것은 1-2-3-2-1을 순서대로 거치는 clow를 의미한다. 또한 (1,2,2)는 1-2-2-1을 순서대로 거치는 clow이다.
이러한 clow(C_i)을 여러 개 나열한 clow sequence $W=\langle C_1, \cdots, C_k\rangle$란, 다음과 같은 성질을 모두 만족하는 sequence를 말한다.

(i) 각 clow C_i의 맨 처음 노드는 C_i에 속해 있는 모든 노드의 최소 index가 되어야 한다. 이러한 노드를 $head(C_i)$라 하자.
(ii) $head(C_1)<head(C_2)<\cdots<head(C_k)$
(iii) $\sum_i |C_i| = n$

예를 들어, [그림 2]에서 $\langle(1,2)(2)\rangle$,$\langle(1,3)(2)\rangle$,$\langle(1,2,2)\rangle$등이 clow sequence가 되며, $\langle(2,1)(3)\rangle$은 (i)에, $\langle(2,3)(1)\rangle$은 (ii)에, 그리고, $\langle(1,2)(2,3)\rangle$은 (iii)에 위배되므로 clow sequence가 아니다.

[mahajan and Vinay 99]은 $det(A)$를 clow sequence관점에서 새롭게 계산할 수 있음을 증명하였다.

[정리 5] [Mahajan et al, 99]

$$det(A) = \sum_{W:\ clow\ sequences} sgn(W)\Pi_{(i,j)\in W} a_{ij}$$

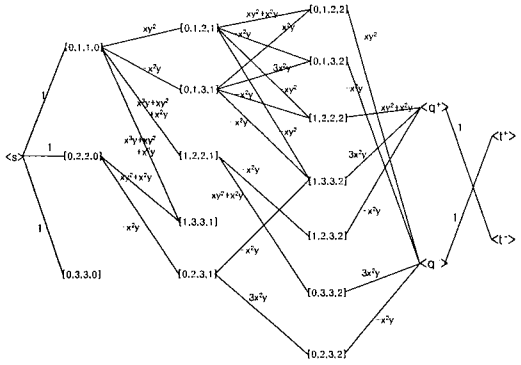
여기서, $sgn(W) = (-1)^{n+k}$(k는 W의 크기)

따라서, 모든 clow sequences를 효율적으로 계산할 수 있으면, $det(A)$를 계산할 수 있다. 이것을 위해 다음과 같은 그래프 H를 구성하자.
먼저 H의 각 노드는 $[p,h,u,i]$로 구성된다. 먼저 p는 현재까지 구성된 clow sequence의 개수가 홀수인지 짝수인지를 나타내는 0/1 값이다. 그 다음 h는 현재 구성중인 clow의 맨 처음 노드를 나타내고, u는 현재 고려중인 노드를 말하며, i는 현재 clow sequence까지 몇 개의 노드가 포함됐는지를 나타낸다.
그리고, 노드간의 호는 다음과 같이 구성된다.

(i) $([p,h,u,i],[p,h,v,i+1])$ if $v>h$, $i+1<n$
호의 비용은 a_{uv}.
(ii) $([p,h,u,i],[\overline{p},h',h',i+1])$ if $h'>h$, $i+1<n$
호의 비용은 a_{uh}.
(iii) $([p,h,u,n-1],q^+)$ if $p=1$
호의 비용은 a_{uh}.
(iv) $([p,h,u,n-1],q^-)$ if $p=0$,
호의 비용은 a_{uh}
(6) $(q^+,t^+),(q^-,t^-)$ if n is even,
호의 비용은 1.
(7) $(q^+,t^-),(q^-,t^+)$ if n is odd,
호의 비용은 1.

이렇게 구성된 그래프 H는 $O(n^3)$의 노드 개수와 $O(n^4)$의 호를 갖는다.

[그림 1]에 제시된 예제 $L[\{\overline{n}\},\{\overline{n}\}]$를 가지고 H를 구성해 보면 [그림 3]과 같다.



[그림 3] 행렬식 계산

여기서, <s>-[0,1,1,0]-[0,1,2,1]-[1,3,3,2]로 가는 경로를 고려하자. 이 경로는 clow sequence <(1,2)(3)>을 의미하고, <s>-[0,1,1,0]-[0,1,3,1]-[0,1,3,2]는 clow sequence <(1,3,3)>을 의미한다. 결국, <s>로부터 $<t+>$까지의 경로는 sgn가 1인 clow sequence를 의미하고, $<t->$까지의 경로는 sgn가 -1인 clow sequence를 의미한다.

그래프 H에서 임의의 경로 ρ에 대해 $w(\rho)$를 그 경로에 속해있는 모든 호의 비용의 곱이라 하면 다음이 성립한다.

[따름정리 1] [mahajan and Vinay 99]

$$det(A) = \sum_{\rho: s \to t^+} w(\rho) - \sum_{\eta: s \to t^-} w(\eta)$$

위 정리의 우변항은 다음과 같은 알고리듬에 의해 쉽게 계산할 수 있다.

Determinant[mahajan and Vinay 99]

step1. Assign value 1 to s and to all vertices.

step2. For $i=1$ to n,

For all vertices y in H at layer k assign the value

$$V(j) = \sum_{i:(i,j)\in H} V(i) \cdot a_{ij}$$

step3. Return $V(t^+) - V(t^-)$

알고리듬 Determinant는 결국 각 호에 대해 더하기와 곱하기를 한번씩 하는 것이 되어 전체 계산시간은 $O(n^4)$이 된다.

이 해법의 장점은 가우스 소거법처럼 나누기 연산이 없어서, 본 연구에서 중점적으로 다루는 다항식을 원소로 갖는 행렬의 경우, 계산이 매우 용이하다는 점이다. 즉, 나누기 연산이 적용되지 않는 링(Ring)상의 원소들로 구성된 행렬에 대해서도 쉽게 적용 가능하다는 점이다.

이 해법을 이용하여, [정리 4]에서 제시한 $L[\{\overline{n}\},\{\overline{n}\}]$의 행렬식을 구하면, 이전의 [Papa. and Yann 00]의 해법보다 개선된 계산복잡도를 얻을 수 있다.
우선, 위 알고리듬 *Determinant*에서 $V(i)$에 오는 다항식의 최대 길이는 앞 절에서 정의된 $p_1 \cdot p_2$ 이므로, [정리 4]에 제시된 $det\, L[\{\overline{n}\},\{\overline{n}\}]$를 계산하려면 $O(n^4 \cdot p_1 \cdot p_2)$ 의 계산시간이 걸린다.
그런데, *CMST*문제에 대한 상한값이 UB라고 하면, $V(i)$에 오는 다항식에서 실제로 계산이 필요한 부분은 x의 지수에 UB보다 작은 지수를 가지고 있는 항들뿐이다. 또한 가능해만 유지하면 되므로, y의 지수도 D이하만을 유지하면 된다. 따라서, $p_1 = UB$, $p_2 = D$로 두면, 전체 계산 복잡도는 $O(n^4 \cdot UB \cdot D)$가 된다.

[정리 6] *CMST*문제는 $O(n^4 \cdot UB \cdot D)$ 계산 시간 안에 최적해를 구할 수 있다.

3. 근사해법

이 장에서는 앞의 유사다항시간 알고리듬을 이용하여 $(1+\varepsilon, 1+\varepsilon)$근사해를 구하기 위한 *FPTAS*를 제시한다.
[Papa. and Yanna. 00]는 앞에서 서술한대로, 행렬-나무정리를 이용하여 ε-근사 파레토 해를 구하기 위한 *FPTAS*를 제시하였다.
본 연구는 이것을 약간 수정하여, $(1+\varepsilon, 1+\varepsilon)$근사해를 구하는 *FPTAS*로 변환하였다.

먼저, 양의 실수 C에 대해, 두 개의 비용 c_{ij}, d_{ij}를 비례축소한 $\hat{c}_{ij}, \hat{d}_{ij}$를 다음과 같이 두자.

$$\widehat{c_{ij}} = \min\left\{ \lfloor \frac{c_{ij} \cdot r}{C} \rfloor , r \right\}$$

$$\widehat{d_{ij}} = \min\left\{ \lfloor \frac{d_{ij} \cdot r}{D} \rfloor , r \right\} \quad r = \lceil \frac{n-1}{\varepsilon} \rceil$$

[정리 7] [Papa. and Yann. 00]

$$\widehat{c}(T) > r \rightarrow c(T) > C,$$
$$\widehat{c}(T) \le r \rightarrow c(T) \le (1+\varepsilon)C$$
($d, \widehat{d}$에 대해서도 마찬가지)

이렇게 비례축소된 두 개의 비용 $\widehat{c_{ij}}, \widehat{d_{ij}}$에 대해, [정리 4]의 행렬-나무 정리를 적용하면 다음과 같은 행렬식을 얻을 수 있다.

$$det\, L^{xy}[\{\overline{n}\},\{\overline{n}\}] = \sum_k a_k x^{k_c} y^{k_d}$$

여기서 a_k는 $\widehat{c}(T) = k_c$, $\widehat{d}(T) = k_d$ 인 나무의 개수가 된다.

위 식에서, 우변항을 k_d에 대해 오름차순으로 정리하고, $\widehat{d}(T) = k_d \le r$인 항에 대응되는 나무의 집합을 S라 하자. 즉, $S = \{T \mid \widehat{d}(T) \le r\}$가 된다. 이 때, S에 속하는 모든 트리 T는 반드시 $d(T) \le (1+\varepsilon)D$를 만족하게 된다. 그리고 S에는 $d(T) \le D$를 만족하는 모든 가능해를 포함하고 있다. 만약 이러한 나무 중 $\widehat{c}(T) \le r$인 나무가 존재하면, $CMST$문제의 최적해는 $c(T) \le (1+\varepsilon)C$를 만족해야 하므로, $(1+\varepsilon)C$를 상한(UB)으로 두고, 그렇지 않으면, S에 있는 모든 나무는 $c(T) > C$를 만족하므로, C를 하한(LB)으로 둘 수 있다. 이것을 알고리듬으로 정리하면 다음과 같다.

$(1+\varepsilon, 1+\varepsilon)$**FPTAS**

step1. $UB \leftarrow (n-1)\max c_{ij}$ $LB \leftarrow 1$

step2. If $UB/LB \le (1+\varepsilon)$ stop.
o/w, let $C \leftarrow (UB \cdot LB)^{1/2}$ and
calculate $det\, L^{xy}[\{\overline{n}\},\{\overline{n}\}]$

step3. Define $S = \{T \mid \widehat{d}(T) \le r\}$
If $\exists$ tree in S such that $\widehat{c}(T) \le r$, let $UB \leftarrow (1+\varepsilon)C$. go to step2.
otherwise, let $LB \leftarrow C$, go to step2.

step2에서 UB와 LB의 산술평균 대신에 기하평균을 사용함으로써, 위 알고리듬의 전체 iteration수는 $\log\log(UB/LB)$가 된다.[Hassin 92]
그리고, [정리 6]에서 $\widehat{UB}$, $\widehat{D}$는 모두 nr보다 작게 되므로, 전체 계산시간은
$O(n^4 \cdot \widehat{UB} \cdot \widehat{D} \cdot \log\log(UB))$
$= O(\frac{n^8}{\varepsilon^2} \log\log(UB))$가 된다.

[정리 8] $CMST$문제에 대해, $(1+\varepsilon, 1+\varepsilon)$ 근사해를 위한 $O(\frac{n^8}{\varepsilon^2} \log\log(UB/LB))$시간의 $FPTAS$가 존재한다.

4. 결론

지금까지, $CMST$문제에 대한 유사다항시간 해법과 $(1+\varepsilon, 1+\varepsilon)$근사해를 구하는 $FPTAS$를 제시하였다.

이 연구는 대부분 [Papa. and Yann. 00]의 결과를 보다 확장하고 분명하게 하는 과정이다. 그들은 $CMST$문제에 대한 고려없이, 오로지 ε-근사 파레토해만을 다룸으로서, 위 문제에 대한 명시적인 해법을 제시하지 않았다. 본 연구는 이러한 그들의 연구에 바탕하여, $CMST$문제에 대한 보다 나은 해법들을 제시하였다. 특히, 기존의 행렬식 계산방법 대신 조합적인 계산 방법을 택함으로써 보다 나은 계산복잡도를 얻을 수 있었다.

그러나, $(1+\varepsilon, 1+\varepsilon)$근사해가 아닌 $(1, 1+\varepsilon)$근사해, 즉 가중치에 대한 제약 $d(T) \le D$를 만족하는 진정한 의미의 $FPTAS$는 아직 알려져 있지 않다.

앞에서 서술했듯이, 이 문제는 NP-$hard$이지만, 유사다항시간 알고리듬이 존재하고, 또한 $PTAS$가 존재하므로, $FPTAS$가 존재할 가능성은 열려 있다.

[참고 문헌]

[Aggarwal et al 82] V. Aggarwal and Y. P. Aneja and K. P. Nair, Minimal Spanning Tree Subject to a Side Constraint, *Comput. and Ops Res.*, Vol. 9(4) , 287-296 , 1982.

[Andersen et al 96] K. A. Andersen and K. Jornsten and M. Lind, On Bicriterion Minimal Spanning Trees : An Approximation, *Comput. and Ops Res,* Vol. 23(12) 1171-1182, 1996.

[Barahona and Pulleyblank 87] F. Barahona and W. R. Pulleyblank, Exact Arborescences, Matchings and Cycles, *Discrete Applied Mathematics,* Vol 16, 91-99, 1987

[Broder and Mayr 97] A. Z. Broder and E. W. Mayr, Counting Minimum Weight Spanning Trees, *Journal of Algorithms*, Vol. 24 , 171-176, 1997.

[Chandrasekaran 77] R. Chandrasekaran, Minimal Ratio Spanning Trees, *Networks*, Vol 7 , 335-342, 1977

[Ehrgott and Klamroth 96] M. Ehrgott and K. Klamroth, Connectedness of Efficient Solutions in Multiple Criteria Combinatorial Optimization, *European Journal of Operations Research*, Vol. 97 , 159-166, 1996

[even 79] S. Even, *Graph Algorithms,* Comp. Sci. Press, New York, 1979.

[Hassin 92] R. Hassin, Approximation Schemes for The Restricted Shortest Path Problem, Mathematics of Operations Research, 17(1), 36-42, 1992

[mahajan and Vinay 99] M. Mahajan and V. Vinay, Determinant : Old Algorithms, New Insights, *SIAM J. Discrete Mathematics*, 1999, Vol. 12(4), 474-490.

[Marathe et al 98] M. V. Marathe and R. Ravi and R. Sundaram and S. S. Ravi and D. J. Rosenkrantz and H. B. Hunt III, Bicriteria Network Design Problems, *Journal of Algorithms*, Vol. 28, 142-171, 1998.

[Papa. and Yann. 00] C. H. Papadimitriou and M. Yannakakis, On the Approximability of Trade-offs and Optimal Access of Web Sources, *Symposium on Foundations of Computer Science (FOCS)*, 2000

[Ramos et al 98] R. M. Ramos and S. Alonso and J. Sicilia and C. Gonzalez, The Problem of the optimal biobjective spanning tree, *European Journal of Operations Research*, 1998, Vol. 111, 617-628.

[Ravi and Goemans 96] R. Ravi and M. X. Goemans, The Constrained Minimum Spanning Tree Problem , *Proceedings of the Scandinavian Workshop on Algorithmic Theory (SWAT)*, 66-75, 1996