Journal of The Korea Society of Computer and Information
Vol. 20, No. 5, May 2015
www.ksci.re.kr
http://dx.doi.org/10.9708/jksci.2015.20.5.031

k-opt를 적용한 차수 제약 최소신장트리 알고리즘

이 상 운*

A Degree-Constrained Minimum Spanning Tree Algorithm Using k-opt

Sang-Un Lee *

요 약

방향 가중 그래프의 차수제약 최소신장트리 (degree-constrained minimum spanning tree, d-MST) 문제는 정확한 해를 구하는 다항시간 알고리즘이 존재하지 않아 NP-완전 문제로 알려져 왔다. 따라서 휴리스틱한 근사 알고리즘을 적용하여 최적 해를 구하고 있다. 본 논문은 차수와 사이클을 검증하는 Kruskal 알고리즘으로 d-MST의 초기 해를 구하고, d-MST의 초기 해에 대해 k-opt를 수행하여 최적 해를 구하는 다항시간 알고리즘을 제안하였다. 제안된 알고리즘을 4개의 그래프에 적용한 결과 2-MST까지 최적 해를 구할 수 있었다.

▸ Keywords : 최소신장트리, 차수제약, 해밀턴 경로, 간선 교환 (k-opt)

Abstract

The degree-constrained minimum spanning tree (d-MST) problem is considered NP-complete for no exact solution-yielding polynomial algorithm has been proposed to. One thus has to resort to an heuristic approximate algorithm to obtain an optimal solution to this problem. This paper therefore presents a polynomial time algorithm which obtains an intial solution to the d-MST with the help of Kruskal's algorithm and performs k-opt on the initial solution obtained so as to derive the final optimal solution. When tested on 4 graphs, the algorithm has successfully obtained the optimal solutions.

▸ Keywords : Minimum spanning tree, Degree constrained, Hamiltonian path, k-opt edge swap

•제1저자 : 이상운
•투고일 : 2015. 02. 11, 심사일 : 2015. 04. 08, 게재확정일 : 2015. 04. 10.
* 강릉원주대학교 멀티미디어공학과 (Dept. of Multimedia Eng., Gangneung-Wonju National University)

I. 서 론

가중치를 갖는 간선들 (weighted edges, E)로 정점들 (vertices, V)이 연결된 무방향 그래프 $G=(V,E)$에 대해, 최소신장트리 (minimum spanning tree, MST)는 간선들의 가중치 합 $\Sigma w(e)$가 최소가 되면서 사이클이 발생하지 않도록 모든 정점들을 연결하는 간선의 개수는 $|e|=|v|-1$이 되어야 한다[1-5]. 일반적으로 MST는 하나의 정점 v에 부속된 간선의 수인 차수 $\deg(v)$에 제약을 가하지 않는다. 따라서 이를 무제약 최소신장트리 (unconstrained MST) 또는 단순히 MST라 부른다. 차수 제약 (degree-constrained) 최소신장트리 (DCMST 또는 d-MST)는 모든 정점의 차수가 d개 이하인 MST이다. 특히, $d=2$인 2-MST를 해밀턴경로 (Hamiltonian path)라 한다[6-17].

지금까지 $d \leq 2$인 2-MST나 $d \leq 3$인 3-MST를 다항시간 (polynomial time)으로 구하는 알고리즘이 존재하지 않아 NP-완전 (NP-complete) 문제로 알려져 있다[6-17]. 따라서 정확한 알고리즘 대신 근사 알고리즘으로 유전자, 진화, 신경망 등의 방법들이 적용되고 있는 실정이다.

차수 무제약 MST의 대표적인 알고리즘으로는 Borůvka[1,2], Prim[3], Kruskal[4]과 역-삭제 (reverse-delete)가 있다. 그러나 차수 무제약 MST 알고리즘을 적용하여 d-MST의 해를 구할 수 없다. 따라서, 본 논문에서는 사이클 검증 기능만을 가진 Kruskal 알고리즘을 사이클과 차수제약 검증기능을 갖는 알고리즘으로 변형시켜 d-MST의 초기 해를 구한다. 다음으로, k-opt의 간선교환 개념을 변형시켜 최적화 과정을 수행하여 d-MST의 최적 해를 다항시간으로 구하는 알고리즘을 제안한다.

2장에서는 DCMST의 다항시간 알고리즘이 존재하지 않음을 예로 제시한 Savelsberg와 Volgenant[17] 그래프를 대상으로 Kruskal 알고리즘으로 차수 제약을 갖지 않은 일반적인 MST를 구한다. 3장에서는 사이클과 차수제약 검증 기능을 갖는 변형된 Kruskal 알고리즘을 접목시켜 d-MST의 초기 해를 구하고, 변형된 k-opt를 적용하여 최적 해를 구하는 방법을 제안한다. 4장에서는 실험 대상 그래프들을 대상으로 제안된 알고리즘의 적용성을 평가해 본다.

II. 관련연구와 연구배경

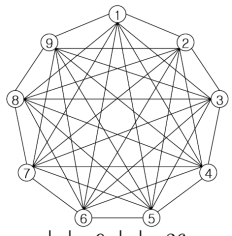
일반적인 차수 무제약 MST를 구하는 Borůvka[1,2], Prim[3], Kruskal[4]과 역-삭제 알고리즘들 중 본 장에서는 Kruskal 알고리즘에 한정해 고찰한다.

Kruskal 알고리즘은 그래프의 모든 간선들을 대상으로 가중치 오름차순으로 정렬시키고, 첫 번째 최소 가중치 간선 (minimum weighted edge, MWE)부터 시작하여 사이클이 발생하지 않는 간선들을 $|V|-1$개가 될 때까지 선택하는 방법이다. 즉, 차수 무제약 MST 알고리즘은 단지 사이클이 발생하는지 여부를 검증하여, 사이클이 발생하면 MST에서 배제하고, 그렇지 않으면 MST에 포함시키는 포함과 배제원칙 (include and exclude principle)을 적용한다.

차수 제약 MST인 d-MST에 대해서는 정확한 다항시간 알고리즘이 제안되지 않고 있으며, 근사 알고리즘들이 연구되고 있는 실정이다[6-17].

Kruskal 알고리즘을 Savelsberg와 Volgenant[17]로부터 인용된 그림 1의 G_1 그래프에 적용하여 차수 무제약 MST를 구하여 보자. G_1은 6개 정점이 36개 간선 ($|v|=9, |e|=36$)으로 연결된 K_6-완전 그래프에 대한 각 정점들 간의 거리를 나타내고 있다.

G_1 그래프를 대상으로 Gen[9], Zhou와 Gen[11,16], Knowles et al.[12]과 Duan et al.[13] 등 근사 알고리즘에 대한 다양한 연구가 수행되었다. Savelsberg와 Volgenant[17]는 간선 교환 휴리스틱 방법을 적용하여 3-MST의 $\Sigma w(e)=2,256$을 얻었다. Zhou와 Gen[11,16]은 $d \leq 3$인 MST를 구하기 위해 유전자 알고리즘 (genetic algorithm, GA)을 25,000회 수행하여 3-MST의 $\Sigma w(e)$이 2,256 (66.7%), 2,292(23.3%), 2,332(3.3%)과 2,378(3.3%)을 얻어 가장 많은 확률로 얻은 2,256을 최적해로 결정하였다.



$|v|=9, |e|=36$

	1	2	3	4	5	6	7	8	9
1		224	224	361	671	300	539	800	943
2	224		200	200	447	283	400	728	762
3	224	200		400	566	447	600	922	949
4	361	200	400		400	200	200	539	583
5	671	447	566	400		600	447	781	510
6	300	283	447	200	600		283	500	707
7	539	400	600	200	447	283		361	424
8	800	728	922	539	781	500	361		500
9	943	762	949	583	510	707	424	500	

그림 1. G_1 그래프
Fig. 1. G_1 Graph

G_1 그래프에 대해 Kruskal 알고리즘을 적용하여 MST를 구하는 과정은 그림 2에 제시되어 있다. 원래의 Kruskal 알고리즘은 간선 수 $|e|=36$회를 수행한다. 본 논문에서는 보다 효율적으로 수행하기 위해 $|e|=|v|-1$조건을 만족하는 $\{7,9\}=424$까지 15회 수행하는 방법을 적용하여 동일한 결과를 얻었다. 이 경우 MST의 간선 가중치 합 $\Sigma w(e)=2,209$을 얻었다. 이와 같이 Kruskal 알고리즘으로 얻은 MST의 최대 차수는 정점 ④로 $\deg(4)=4$이다. 즉, G_1 그래프의 차수 무제약 MST는 $d\le 4$로 4-MST이다.

E 오름 차순 정렬	V	부분신장트리	MST	Cycle
–	{1,2,3,4,5,6,7,8,9}	$\{\phi\}$	–	–
{2,3}=200	{1,4,5,6,7,8,9}	{**2**,3}	{2,3}=200	–
{2,4}=200	{1,5,6,7,8,9}	{2,3,**4**}	{2,4}=200	–
{4,6}=200	{1,5,7,8,9}	{2,3,**4**,6}	{4,6}=200	–
{4,7}=200	{1,5,8,9}	{**2**,3,4,6,7}	{4,7}=200	–
{1,2}=224	{5,8,9}	{**1**,2,**3**,4,6,7}	{1,2}=224	–
{1,3}=224	{5,8,9}	{1,**2**,3,4,**6**,7}	–	{1,3}=224
{2,6}=283	{5,8,9}	{1,2,3,4,**6**,**7**}	–	{2,6}=283
{6,7}=283	{5,8,9}	{**1**,2,3,4,**6**,**7**}	–	{6,7}=283
{1,6}=300	{5,8,9}	{**1**,2,3,**4**,6,7}	–	{1,6}=300
{1,4}=361	{5,8,9}	{1,2,3,4,6,**7**}	–	{1,4}=361
{7,8}=361	{5,9}	{1,**2**,3,4,6,**7**,8}	{7,8}=361	–
{2,7}=400	{5,9}	{1,2,**3**,**4**,6,7,8}	–	{2,7}=400
{3,4}=400	{5,9}	{1,2,3,**4**,6,7,8}	–	{3,4}=400
{4,5}=400	{9}	{1,2,3,4,5,6,**7**,8}	{4,5}=400	–
{7,9}=424	{}	{1,**2**,3,4,**5**,6,7,8,9}	{7,9}=424	–

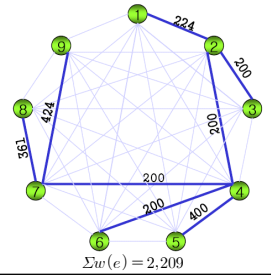


그림 2. G_1 그래프의 Kruskal MST 알고리즘 적용
Fig. 2. Kruskal's MST Algorithm for G_1 Graph

만약, $d\le 3$이나 $d\le 2$의 3-MST나 2-MST를 얻고자 한다면 Kruskal 알고리즘을 적용할 수 없고 새로운 방법이 요구된다.

III. k-opt d-MST 알고리즘

Kruskal 알고리즘은 모든 간선들을 오름차순으로 정렬하는데 $O(E\log E)$, 사이클 발생 최대 가중치 간선 제거에 $O(E)$이 수행되어 수행 복잡도는 $O(E\log E)$으로 다항시간 알고리즘이다. 본 장에서는 사이클 검증 기능만을 고려한 Kruskal 알고리즘에 차수 검증 기능을 추가하여 d-MST의 초기 해를 구하는 방법을 제안한다. 즉, 제안된 방법은 사이클이나 차수 초과시 MST에서 배제시키고, 사이클 미발생과 차수를 초과하지 않을 경우 MST에 포함시키는 방법으로 확장하였다.

다음으로 d-MST의 초기 해에 대해 비용을 감소시킬 수 있는 k-opt 최적화 과정을 수행 복잡도 $O(E)$로 수행하는 방법을 제안한다. 따라서 제안된 k-opt d-MST 알고리즘의 수행 복잡도는 $O(E\log E)$이다. 제안된 k-opt d-MST 알고리즘은 그림 3에 제시되어 있다.

V: 정점들, E: 간선들, C: 사이클 발생과 차수초과 간선들
M: MST 간선들, d: 차수 제약 상수

/* d-MST 초기 해
E에 대해 간선 가중치 오름차순으로 정렬 /* $O(n\log n)$
for $|V|=0\cap|S|=1$ or $E=\{\phi\}$
E에서 최소가중치 간선 $\min w\{u,v\}$ 선택, $E=E\setminus w\{u,v\}$.
if $d(u)+1\in M\le d$ and $d(v)+1\in M\le d$ then
if $u,v\in V$ then $S_i\leftarrow(u,v)$, $V=V\setminus(u,v)$, $M\leftarrow w\{u,v\}$, $d(u)\leftarrow d(u)+1$, $d(v)\leftarrow d(v)+1$
else if $u\notin V$ and $v\in V$ then $u\in S_i\leftarrow v$, $V=V\setminus v$ $M\leftarrow w\{u,v\}$, $d(u)\leftarrow d(u)+1$, $d(v)\leftarrow d(v)+1$
else if $u,v\notin V$, $u\in S_i$ and $v\in S_j$, $i=j$ then $C\leftarrow w\{u,v\}$
else if $u,v\notin V$, $u\in S_i$ and $v\in S_j$, $i\neq j$ then $S_i\leftarrow S_i+S_j$, $M\leftarrow w\{u,v\}$, $d(u)\leftarrow d(u)+1$, $d(v)\leftarrow d(v)+1$
endif
else if $d(u)+1\in M>d$ or $d(v)+1\in M>d$ then $C\leftarrow w\{u,v\}$
endif
end

/* d-MST 최적화
if $\max w\{u,v\}\in M$, $\deg(v)=1$**에 대해** $w\{v,i\}\notin C$ **then**
최적화 과정 종료
endif
if $|M|=1$ then /* Spanning Tree (ST)가 생성된 상태.
/* 2-opt 수행
$\max w\{u,v\}\in M$ 삭제, $\min w\{i,j\}\in C$ 추가.
$\min w\{i,j\}\in C$ 추가로 인해 발생한 사이클에서 $w\{i,k\}$ $(w\{i,k\}>w\{j,k\})$ 삭제.
if $\deg(v)=0$인 정점 v의 $w\{v,l\}\in C$ 중 $\deg(l)<d, l=i,$ j,or k then $w\{v,l\}$추가
/* 3-opt 수행
else $\deg(v)=0$, 정점 v의 $w\{v,l\}\in C$ $(l\neq i,j)$ 추가.
if $\deg(i)>d, \deg(j)>d$ 발생 then $w\{i,j\}$ 삭제.
if Spanning Tree가 분리 then PST1과 PST2 간의 Inter-PST인 최소 가중치 간선 $\min w\{k,l\}\in C$, $(\deg(k)=1\in PST_1, \deg(l)=1\in PST_2)$ 연결.
endif
if $\Sigma w(e)^- > \Sigma w(e)^+$ then k-opt로 간선 교환
else d-MST 초기 해를 최적 해로 결정
endif
else if $|M|\ge 2$ then
/* Partial Spanning Tree (PST)가 생성된 상태로 $|M|=1$인 Spanning Tree로 연결 후 k-opt 수행
while $|M|=1$ do
E에서 PST간 간선 (Inter-PST Edges)만 남기고, 각 PST 내부 간선 (Intra-PST Edges)은 삭제.
if $\exists,\deg(u)=1\in PST_1, \exists,\deg(v)=1\in PST_2$ then 최소 가중치 간선 $\min w\{u,v\}$를 M에 추가
else $\deg(u)=1\in PST_1$, $\deg(v)=2\in PST_2$중에서 최소 가중치 간선 $\min w\{u,v\}$선택.
$\deg(i)=2\in PST_1$, $\deg(j)=1\in PST_2$중에서 최소 가중치 간선 $\min w\{i,j\}$ 선택.
$\min(\min w\{u,v\}, \min w\{i,j\})$를 M에 추가.
$\deg(u)>d$ 정점 부속 간선 중 $\max w\{u,i\}$ 삭제
if $|M|\ge 2$ then loop
endif
end
endif

그림 3. k-opt d-MST 알고리즘
Fig. 3. k-opt d-MST Algorithm

d-MST의 초기 해를 구하면 1개의 신장트리 (spanning tree, ST)가 생성되는 경우와 다수의 분리된 부분신장트리 (partial spanning tree, PST)가 생성되기도 한다. 만약, PST가 생성될 경우 C의 원소인 PST 간 (inter-PST) 간선들 만을 대상으로 축소시키고, 최소 가중치 간선을 연결하여 ST를 생성한다. 일단 ST가 생성되면 k-opt (k개 간선 삭제, k개 대체 간선 추가)로 최적화 과정을 수행한다.

d-MST의 최적화 과정에서 적용된 k-opt는 Stougie[18]가 제시한 그림 4의 개념을 변형시켜 적용하였다.

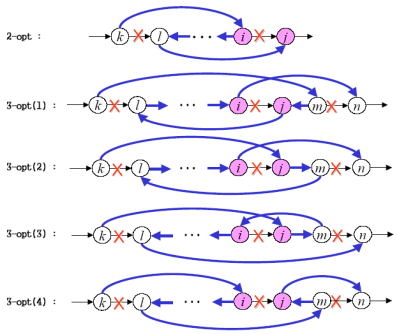


그림 4. k-간선 교환 방법 (2-opt와 3-opt)
Fig. 4. k-Edges Swap Method (2-opt and 3-opt)

일반적인 k-opt는 외판원 문제 (traveling salesperson problem, TSP)와 같은 해밀턴 사이클 (Hamiltonian cycle)에 적합하도록 설계되었다. 반면에, d-MST의 특징은 해밀턴 경로로, 2-MST의 2-opt의 경우 그림 5와 같이 6가지 경우가, 3-MST의 2-opt의 경우 그림 6과 같이 10가지 경우가 발생하여 수많은 경우수를 고려해야 하는 단점이 있다.

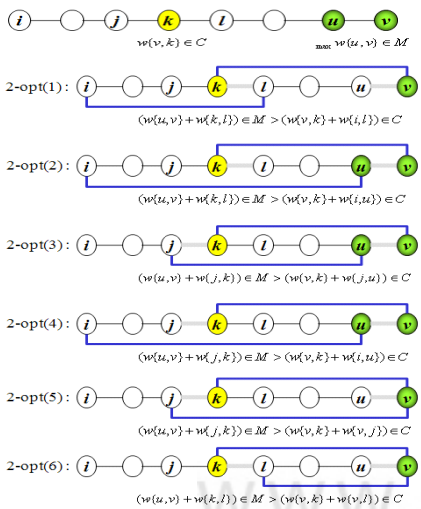


그림 5. 2-MST의 2-opt 경우의 수
Fig. 5. Possible Number of 2-opt for 2-MST

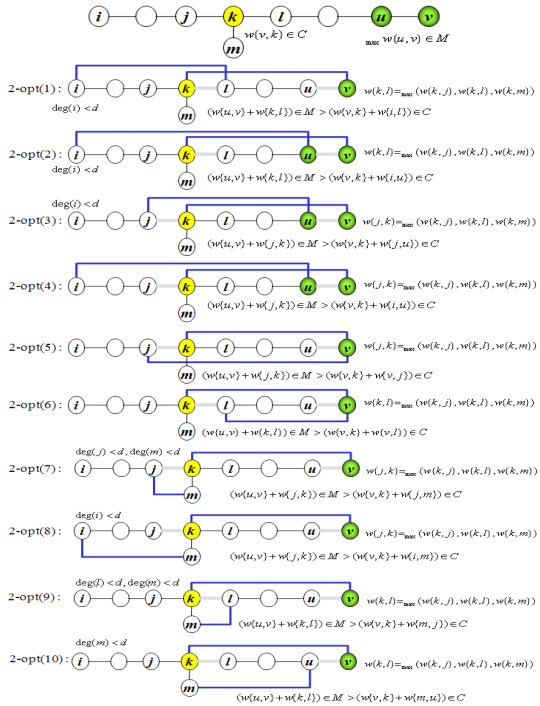


그림 6. d-MST ($d \geq 3$)의 2-opt 경우의 수
Fig. 6. Possible Number of 2-opt for d-MST ($d \geq 3$)

따라서, 본 알고리즘에서는 $\max w\{u,v\} \in M$을 삭제하고, $\min w\{i,j\} \in C$를 추가하는 경우로 한정시켜 가능한 경우를 최소화 시키도록 k-opt를 적용하였다. d-MST에는 k-opt ($k=1,2,3$)가 적용되었다.

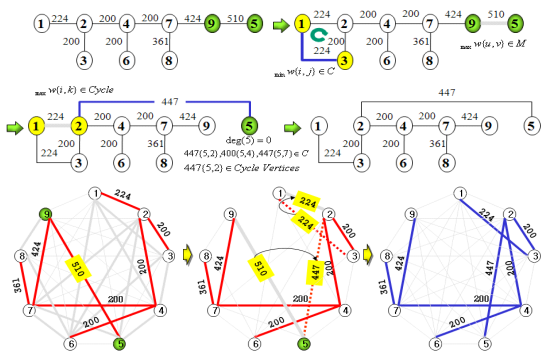
G_1 그래프에 대해 k-opt 3-MST를 구한 결과는 그림 7에, k-opt 2-MST를 구한 결과는 그림 8에 제시되어 있다.

그림 7의 3-MST 초기 해에서 $\max w\{u,v\} \in M$인 510{5,9}가 삭제되면 $d(5)=0$이 된다. 이때 M에 추가될 후보 $w\{5,v\} \in C$는 400{5,4}, 447{5,2}와 447{5,7}이다. $\min w\{u,v\} \in C$는 224{1,3} 추가시 ①-②-③-①의 사이클이 발생하며, 사이클의 $\max w\{i,k\}$인 224{1,2}가 삭제된다. 따라서, $d(5)=0$인 ⑤ 정점을 연결할 후보들 400{5,4}, 447{5,2}와 447{5,7}중 447{5,2}가 추가되면 2-opt가 완료되며, 초기 해에 비해 (510+224)-(447+224) =63의 비용이 절감된다.

그림 8의 2-MST 초기 해에서 $\max w\{u,v\} \in M$인 510{5,9}가 삭제되면 $d(5)=0$이 된다. 이때 M에 추가될 후보 $w\{5,v\} \in C$는 400{5,4}, 447{5,2}와 447{5,7}이다. $\min w\{u,v\} \in C$인 200{4,7}이 추가되면 ④-⑥-⑦-④의 사이클

이 발생하며, 최대 가중치 간선 283{6,7}이 삭제된다. 400{5,4}, 447{5,2}와 447{5,7} 중에서 $\min w\{u,v\}\in C$인 ④와 ⑦을 제외한 447{5,2}가 추가된다. 이 경우 deg(2) = 3, deg(4) = 3이 발생하여 200{2,4}가 삭제된다. 이로 인해 ①-③-②-⑤와 ⑥-④-⑦-⑧-⑨의 분리된 PST가 발생하며, 2개 PST간 최소 가중치 간선인 300{1,6}$\in C$로 연결된다. 따라서 3-opt가 수행되었으며, (510+283+200)-(447+200+300) = 46의 비용이 절감된다.

E		3-MST 초기 해		
미 정렬	오름차순 정렬	C	M	E
{1,2} = 224	{2,3} = 200	-	{2,3} = 200	-
{1,3} = 224	{2,4} = 200	-	{2,4} = 200	-
{1,4} = 361	{4,6} = 200	-	{4,6} = 200	-
{1,5} = 671	{4,7} = 200	-	{4,7} = 200	-
{1,6} = 300	{1,2} = 224	-	{1,2} = 224	-
{1,7} = 539	{1,3} = 224	**{1,3} = 224**	-	-
{1,8} = 800	{2,6} = 283	{2,6} = 283	-	-
{1,9} = 943	{6,7} = 283	{6,7} = 283	-	-
{2,3} = 200	{1,6} = 300	{1,6} = 300	-	-
{2,4} = 200	{1,4} = 361	{1,4} = 361	-	-
{2,5} = 447	{7,8} = 361	-	{7,8} = 361	-
{2,6} = 283	{2,7} = 400	{2,7} = 400	-	-
{2,7} = 400	{3,4} = 400	{3,4} = 400	-	-
{2,8} = 728	{4,5} = 400	**{4,5} = 400**	-	-
{2,9} = 762	{7,9} = 424	-	{7,9} = 424	-
{3,4} = 400	{2,5} = 447	**{2,5} = 447**	-	-
{3,5} = 566	{3,6} = 447	{3,6} = 447	-	-
{3,6} = 447	{5,7} = 447	**{5,7} = 447**	-	-
{3,7} = 600	{6,8} = 500	{6,8} = 500	-	-
{3,8} = 922	{8,9} = 500	{8,9} = 500	-	-
{3,9} = 949	{5,9} = 510	-	**{5,9} = 510**	-
{4,5} = 400	{1,7} = 539	-		
{4,6} = 200	{4,8} = 539	-		
{4,7} = 200	{3,5} = 566	-		{3,5} = 566
{4,8} = 539	{4,9} = 583	-		
{4,9} = 583	{3,7} = 600	-		
{5,6} = 600	{5,6} = 600	-		{5,6} = 600
{5,7} = 447	{1,5} = 671	-		{1,5} = 671
{5,8} = 781	{6,9} = 707	-		-
{5,9} = 510	{2,8} = 728	-		-
{6,7} = 283	{2,9} = 762	-		-
{6,8} = 500	{5,8} = 781	-		{5,8} = 781
{6,9} = 707	{1,8} = 800	-		{1,8} = 800
{7,8} = 361	{3,8} = 922	-		{3,8} = 922
{7,9} = 424	{1,9} = 943	-		-
{8,9} = 500	{3,9} = 949	-		-



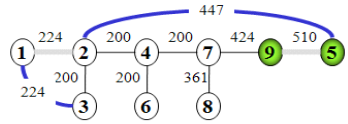
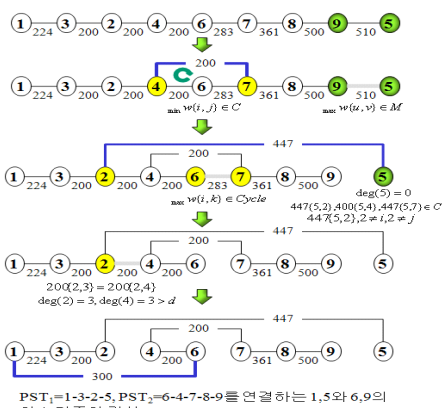


그림 7. G_1 그래프의 2-opt를 적용한 3-MST
Fig. 7. 3-MST Using 2-opt for G_1 Graph

E		2-MST 초기 해		
미 정렬	오름차순 정렬	C	M	E
{1,2} = 224	{2,3} = 200	-	{2,3} = 200	-
{1,3} = 224	{2,4} = 200	-	{2,4} = 200	-
{1,4} = 361	{4,6} = 200	-	{4,6} = 200	-
{1,5} = 671	{4,7} = 200	**{4,7} = 200**	-	-
{1,6} = 300	{1,2} = 224	{1,2} = 224	-	-
{1,7} = 539	{1,3} = 224	-	{1,3} = 224	-
{1,8} = 800	{2,6} = 283	{2,6} = 283	-	-
{1,9} = 943	{6,7} = 283	-	{6,7} = 283	-
{2,3} = 200	{1,6} = 300	{1,6} = 300	-	-
{2,4} = 200	{1,4} = 361	{1,4} = 361	-	-
{2,5} = 447	{7,8} = 361	-	{7,8} = 361	-
{2,6} = 283	{2,7} = 400	{2,7} = 400	-	-
{2,7} = 400	{3,4} = 400	{3,4} = 400	-	-
{2,8} = 728	{4,5} = 400	**{4,5} = 400**	-	-
{2,9} = 762	{7,9} = 424	{7,9} = 424	-	-
{3,4} = 400	{2,5} = 447	**{2,5} = 447**	-	-
{3,5} = 566	{3,6} = 447	{3,6} = 447	-	-
{3,6} = 447	{5,7} = 447	**{5,7} = 447**	-	-
{3,7} = 600	{6,8} = 500	{6,8} = 500	-	-
{3,8} = 922	{8,9} = 500	-	{8,9} = 500	-
{3,9} = 949	{5,9} = 510	-	**{5,9} = 510**	-
{4,5} = 400	{1,7} = 539	-	-	-
{4,6} = 200	{4,8} = 539	-	-	-
{4,7} = 200	{3,5} = 566	-	-	-
{4,8} = 539	{4,9} = 583	-	-	-
{4,9} = 583	{3,7} = 600	-	-	-
{5,6} = 600	{5,6} = 600	-	-	-
{5,7} = 447	{1,5} = 671	-	-	{1,5} = 671
{5,8} = 781	{6,9} = 707	-	-	-
{5,9} = 510	{2,8} = 728	-	-	-
{6,7} = 283	{2,9} = 762	-	-	-
{6,8} = 500	{5,8} = 781	-	-	-
{6,9} = 707	{1,8} = 800	-	-	-
{7,8} = 361	{3,8} = 922	-	-	-
{7,9} = 424	{1,9} = 943	-	-	-
{8,9} = 500	{3,9} = 949	-	-	-



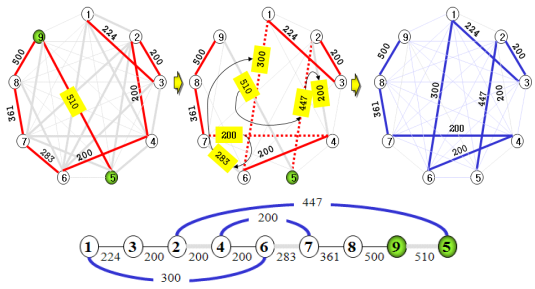


그림 8. G_1 그래프의 3-opt를 적용한 2-MST
Fig. 8. 2-MST Using 3-opt for G_1 Graph

IV. 실험 및 결과 분석

본 장에서 적용된 실험 데이터는 그림 9에 제시되어 있다. G_2와 G_4는 Peiper[18]에서, G_3는 Wikipedia[5]에서 인용되었다. 실험 데이터에 대해 Kruskal 알고리즘으로 구한 차수 무제약 MST는 G_2와 G_3는 3-MST, G_4는 4-MST이다. 따라서, 우리가 구하고자 하는 d-MST는 G_2와 G_3에 대해서는 2-MST를, G_4에 대해서는 3-MST와 2-MST이다.

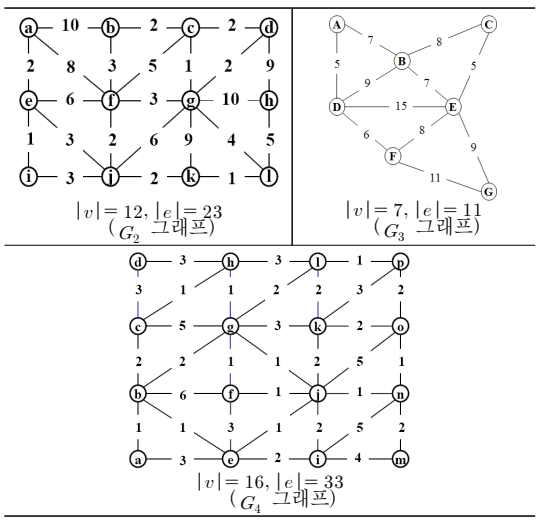
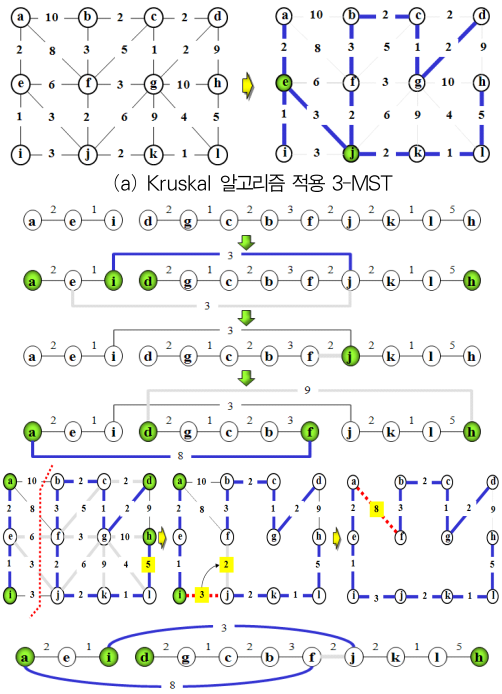


그림 9. 실험 그래프
Fig. 9. Benchmarking Data

G_2 그래프에 대해 2-MST를 1-opt로 구한 결과는 그림 10에 제시되어 있다. 여기서는 Kruskal 알고리즘으로 구한 차수 무제약 MST도 함께 제시하였다. 2-MST 초기 해는 2개의 분리된 신장트리를 형성함을 알 수 있다. 이는 M에서 임의의 정점 u을 선택하여 $\{u,v\},\{v,w\},\{w,x\},\cdots$로 계속적으로 탐색하면서 V의 모든 정점을 방문하지 않으면 신장트리가 분리되었음을 쉽게 검증할 수 있다.

G_2 그래프의 2-MST 초기 해에 대해 PST간 간선을 대상으로 $\deg(u)=1\in PST_1$과 $\deg(v)=2\in PST_2$인 후보 간선들은 10{a,b}, 8{a,f}와 3{i,j}이다. 이들 중 최소 가중치인 3{i,j}를 추가하면 $\deg(j)=3>(d=2)$가 된다. 따라서 j 정점의 부속 간선들 중 최대 가중치 간선인 2{j,f}와 2{j,k}중 어느 하나가 삭제된다. 여기서는 2{j,f}를 삭제하였다. 이 결과 다시 2개의 PST로 분리되었으며, $\deg(u)=1\in PST_1$과 $\deg(v)=1\in PST_2$인 8{a,f}, 9{d,h}중 최소 가중치 간선인 8{a,f}가 연결되어 하나의 신장트리가 완성된다. 즉, G_2 그래프는 d-MST의 초기 해인 분리된 PST를 일단 연결하고, 1-opt를 수행하여 최적 해를 구하였다.



(a) Kruskal 알고리즘 적용 3-MST

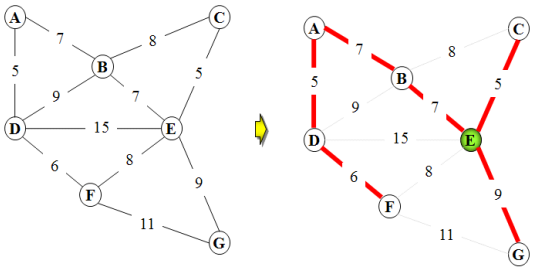
(b) 1-opt 적용 2-MST
그림 10. G_2 그래프
Fig. 10. G_2 Graph

G_3 그래프에 대해 2-MST를 2-opt로 구한 결과는 그림 11에 제시되어 있다. 2-MST 초기 해에 대해 $\max w\{u,v\}\in M$인 11{F,G}가 삭제되면 $d(G)=0$이 된다. 이때 M에 추가될 후보는 $w\{G,v\}\in C$인 9{G,E}만이 존재한다. $\min w\{i,j\}\in C$인 8{B,C}, 8{E,F}에서 E는 신규로 추가되는 9{G,E}에 존재하여 8{B,C}가 추가되는 것으로 결정된다. 따라서, B-C-E-B 사이클

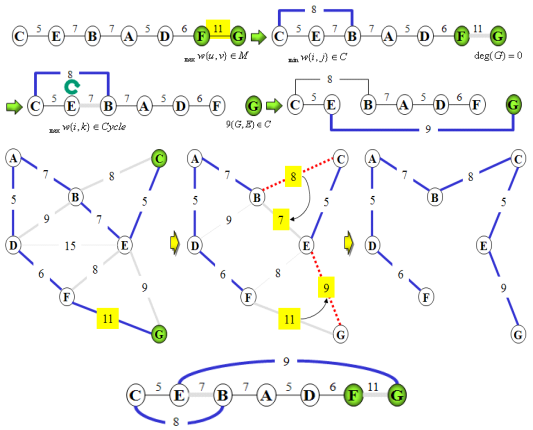
의 최대 가중치 간선인 7{B,E}가 삭제되는 2-opt가 수행되었으며, (11+7)-(9+8)=1의 비용이 절감된다.

G_4 그래프의 차수 무제약 MST는 그림 12에서 제시된 바와 같이 4-MST이다. 먼저, 3-MST 초기 해에 대해 $\max w\{u,v\} \in M$인 3{d,c}를 삭제하면 $\deg(d)=0$이 되며, $w\{d,i\} \notin C$로 해 개선과정이 수행되지 않으며, 초기 해가 최적 해로 결정된다. 다음으로, 2-MST 초기 해는 2개 PST로 분리된 상태이다.

PST간 간선을 대상으로 $\deg(u)=1 \in PST_1$과 $\deg(v)=2 \in PST_2$인 후보 간선들은 2{i,j}, 2{i,c}이다. 이들 중 최소 가중치 간선 2{i,c}를 추가하면 $\deg(j)=3>(d=2)$이 된다.



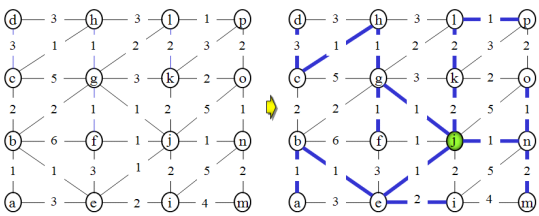
(a) Kruskal 알고리즘 적용 3-MST



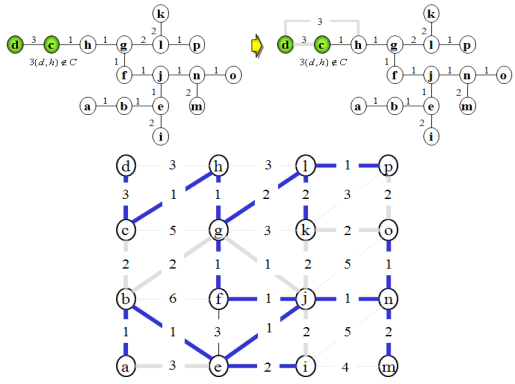
(b) 2-opt 적용 2-MST

그림 11. G_3 그래프

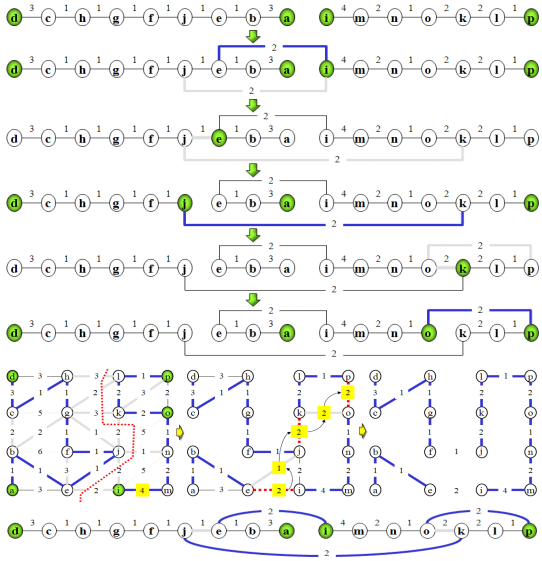
Fig. 11. G_3 Graph



(a) Kruskal 알고리즘 적용 4-MST



(b) k-opt 미적용 3-MST



(c) 2-opt 적용 2-MST

그림 12. G_4 그래프

Fig. 12. G_4 Graph

따라서 j 정점의 부속 간선인 {j,c}가 삭제된다. 이 결과 다시 2개의 PST가 발생하였으며, PST간 간선을 대상으로 $\deg(u)=1 \in PST_1$과 $\deg(v)=2 \in PST_2$인 후보 간선들은 2{j,i}, 2{j,k}이다. 여기서 2{j,k}는 비용 증가 없이 2{k,o}와 교환이 가능하므로 2{j,k}가 추가되었다. 이로 인해 $\deg(k)=3>d$이 발생하여 2{k,o}가 삭제되어 다시 2개 PST가 발생하였으며, $\deg(u)=1 \in PST_1$과 $\deg(v)=1 \in PST_2$인 2{o,p}로 연결된다. 따라서 일단 분리된 PST를 ST로 연결한 후 2-opt가 수행되었다.

본 실험에 적용된 4개 그래프에 대해 k-opt d-MST를 수행한 결과는 표 1에 제시되어 있다.

표 1. 알고리즘 비교
Table 1. Compare of the Algorithms

그래프	$\Sigma w(e)$				
	차수 무제약	3-MST		2-MST	
		알려진 최적 해	k-opt	알려진 최적 해	k-opt
G_1	4-MST 2,209	**GA 2,256**	2-opt **2,256**	x	3-opt 2,432
G_2	3-MST 24	-	-	x	1-opt 30
G_3	3-MST 39	-	-	x	2-opt 40
G_4	4-MST 20	x	0-opt 21	x	2-opt 25

본 실험에 적용된 G_2, G_3, G_4 그래프들에 대해 GA를 적용한 사례는 찾지 못하였다. 이와 같은 이유로 인해 제안된 k-opt 알고리즘과 GA와의 성능 비교 없이 알려진 최적 해와 제안된 k-opt의 결과만을 비교하였다.

제안된 k-opt d-MST는 0-opt부터 3-opt까지 수행하였으며, 2-opt를 가장 많이 수행하였음을 알 수 있다. 또한 G_1 그래프의 3-MST에 대해 변형된 Kruskal 알고리즘으로 얻은 d-MST 초기 해에 대해 2-opt를 수행하고도 알려진 최적 해인 유전자 알고리즘의 2,256과 동일한 결과를 얻었음을 알 수 있다. 특이한 점은 G_4 그래프의 3-MST는 변형된 Kruskal 알고리즘으로 얻은 d-MST가 최적 해로 바로 결정된다는 점이다.

V. 결론

DCMST는 지금까지 정확한 해법을 다항시간에 찾는 알고리즘이 존재하지 않는 NP-완전 문제로 알려져 근사 알고리즘만을 연구하였다. 본 논문은 차수제약과 사이클 검증 기능을 가진 변형된 Kruskal 알고리즘으로 d-MST의 초기 해를 수행복잡도 $O(E\log E)$로 구하였다. 다음으로 $\max w\{u,v\}\in M$을 삭제하고, $\min w\{i,j\}\in C$를 추가하는 k-opt를 수행 복잡도 $O(E)$로 수행하였다. 제안된 알고리즘을 DCMST의 실예로 적용된 Savelsberg와 Volgenant[17] 그래프에 적용하여 3-MST를 쉽게 구하였다. 추가적으로 3개의 그래프에 대해 무제약 MST인 k-MST에 대해 $2 \le d < k$인 d-MST를 쉽게 구할 수 있음을 보였다. 따라서, 본 논문은 DCMST 문제가 NP-완전이 아닌 다항시간 알고리즘이 존재하는 P 문제일 가능성이 높음을 실증 사례를 바탕으로 증명하는 계기를 마련하였다.

REFERENCES

[1] O. Borůvka, "O Jistem Problemu Minimalnim," Prace Mor. Prrodved. Spol. V Brne (Acta Societ. Natur. Moravicae), Vol. Ⅲ, No. 3, pp. 37-58, Mar. 1926.

[2] J. Nešetřil, E. Milková, and H. Nešetřilová, "Otakar Borůvka on Minimum Spanning Tree Problem (Translation of the both 1926 Papers, Comments, History)," DMATH: Discrete Mathematics, Vol. 233, No. 1-3, Apr. 2001.

[3] R. C. Prim, "Shortest Connection Networks and Some Generalisations," Bell System Technical Journal, Vol. 36, No. 6, pp. 1389-1401, Nov. 1957.

[4] J. B. Kruskal, "On the Shortest Spanning Subtree and The Traveling Salesman Problem," Proceedings of the American Mathematical Society, Vol. 7, pp. 48-50, 1956.

[5] Wikipedia, "Minimum Spanning Tree," http://en.wikipedia.org/wiki/Minimum_spanning_tree, 2014.

[6] Wikipedia, "Degree-constrained Spanning Tree," http://en.wikipedia.org/wiki/Degree-constrained_Spanning_Tree, 2014.

[7] G. R. Raidl, "An Efficient Evolutionary Algorithm for the Degree-Constrained Minimum Spanning Tree Problem," Proceedings of the 2000 IEEE Congress on Evolutionary Computation, pp. 104-111, IEEE Press, 2000.

[8] S. C. Narula and C. A. Ho, "Degree-Constrained Minimum Spanning Tree," Computational Operations Research, Vol. 7, pp. 239-249, 1980.

[9] M. Gen, "Evolutionary Algorithms and Optimization: Theory and Its Application," Software Computing Lab., Waseda University, IPS., 2004.

[10] C. H. Chu, G. Premkumar, C. Chou, and J. Sun, "Dynamic Degree Constrained Network Design: A Genetic Algorithm Approach," School of Computer Science, University of Birmingham, UK., 1999.

[11] G. Zhou and M. Gen, "A Note on Genetic Algorithms for Degree-Constrained Spanning Tree Problems," Networks, Vol. 30, No. 2, pp. 91-95, Sep. 1997.

[12] J. Knowles, D. Corne, and M. Oates, "A New Evolutionary Approach to the Degree Constrained Minimum Spanning Tree Problem," Trans. on Evolutionary Computation, Vol. 4, No. 2, pp. 125-134,

Jul. 2000.
〔13〕 X. Duan, C. Wang, X. Lue, and N. Wang, "A Hybrid Algorithm Based on Particle Swarm Optimization," International Journal of Information and Systems Science, Vol. 1, No. 3-4, pp. 275-282, Apr. 2005.
〔14〕 L. Gouveia and P. Moura, "Models for the Degree Constrained Minimum Spanning Tree Problem with Node-Degree Dependent Costs," International Network Optimization Conference, Spa, Belgium, 2007.
〔15〕 M. Krishnamoorthy, A. T. Ernst, and Y. M. Sharaiha, "Comparison of Algorithms for the Degree Constrained Minimum Spanning Tree," Journal of Heuristics, Vol. 7, No. 6, pp. 587-611, Nov. 2001.
〔16〕 G. Zhou and M. Gen, "Approach to the Degree-Constrained Minimum Spanning Tree Problem Using Genetic Algorithms," Engineering Design and Automation, Vol. 3, No. 2, pp. 156-165, Feb. 1997.
〔17〕 M. Savelsbergh and T. Volgenant, "Edge-Exchanges in the Degree-Constrained Minimum Spanning Tree Problem," Computers and Operations Research, Vol. 12, No. 4, pp. 341-348, Apr. 1985.
〔18〕 L. Stougie, "2P350: Optimaliseringsmethoden," http://www.win.tue.nl/~leen/OW/2P350/Week8/week8.pdf, College Wordt ggeven op vinjdagmiddag, 2001.

저 자 소 개

이 상 운(Sang-Un Lee)
1983년 ~ 1987년 :
한국항공대학교 항공전자공학과 (학사)
1995년 ~ 1997년 :
경상대학교 컴퓨터과학과 (석사)
1998년 ~ 2001년 :
경상대학교 컴퓨터과학과 (박사)
2003.3 ~ 2015.3 :
강릉원주대학교 멀티미디어공학과 부교수
2015.4 ~ 현 재 :
강릉원주대학교 멀티미디어공학과 정교수
관심분야 : 소프트웨어 프로젝트 관리, 소프트웨어 개발 방법론, 소프트웨어 신뢰성, 그래프 알고리즘
e-mail : sulee@gwnu.ac.kr