논문 2011-5-22

방향그래프의 최소신장트리 알고리즘

A Minimum Spanning Tree Algorithm for Directed Graph

최명복*, 이상운**

Myeong-Bok Choi, Sang-Un Lee

요 약 본 논문에서는 방향 그래프의 최소신장트리(Directed Graph Minimum Spanning Tree, DMST)를 구하는 알고리즘을 제안하였다. 기존의 Chu-Liu/Edmonds DMST 알고리즘은 DMST를 찾지 못하거나 ST의 가중치 합이 최소가 되지 못하는 경우가 발생한다. 제안된 알고리즘은 Chu-Liu/Edmonds DMST 알고리즘의 단점을 보완하여 항상 DMST를 찾을 수 있도록 하였다. 먼저, 근 노드를 포함한 모든 노드의 최소 가중치를 갖는 유입 호 (Minimum-Weight Arc, MWA)를 선택하여 오름차순으로 정렬시킨 후 사이클이 발생하는 호를 제거하는 과정을 거쳤다. 이 과정에서 최소신장 포레스트 (Minimum Spanning Forest, MSF)가 얻어진다. 만약 MSF가 1개이면 DMST가 얻어지며, MSF가 2개 이상인 경우, MSF 유입 호들 중 최소 가중치를 갖는 호를 결정하기 위해 직접 가중치 합을 계산하는 방법을 택하여 Chu-Liu/Edmonds DMST 알고리즘의 사이클 해결을 위한 유입 호 가중치 수정 과정을 단순화 시켰다. 제안된 Sulee DMST 알고리즘은 근 노드가 지정되어 있거나 미 지정된 경우 모두 항상 호들의 가중치를 최소화 시키는 DMST를 얻을 수 있으며, 그래프의 가중치가 최소화된 ST의 근 노드를 찾는 장점도 갖고 있다.

Abstract This paper suggests an algorithm that obtains the Minimum Spanning Tree for directed graph (DMST). The existing Chu-Liu/Edmonds DMST algorithm has chances of the algorithm not being able to find DMST or of the sum of ST not being the least. The suggested algorithm is made in such a way that it always finds DMST, rectifying the disadvantage of Chu-Liu/Edmonds DMST algorithm. Firstly, it chooses the Minimum-Weight Arc (MWA) from all the nodes including a root node, and gets rid of the nodes in which cycle occurs after sorting them in an ascending order. In this process, Minimum Spanning Forest (MST) is obtained. If there is only one MSF, DMST is obtained. And if there are more than 2 MSFs, to determine MWA among all MST nodes, it chooses a method of directly calculating the sum of all the weights, and hence simplifies the emendation process for solving a cycle problem of Chu-Liu/Edmonds DMST algorithm. The suggested Sulee DMST algorithm can always obtain DMST that minimizes the weight of the arcs no matter if the root node is set or not, and it is also capable to find the root node of a graph with minimized weight.

Key Words : Directed Graph, Minimum Spanning Tree, Directed Graph Minimum Spanning Tree Cycle, Minimum Spanning Forest

Ⅰ. 서 론

그래프 (Graph)는 정점들 (Vertices)과 간선들(Edges)로 구성되어 있으며, 정점들이 간선들로 연결되어 있고 (Connected), 간선들은 방향성 (Directed)과 무방향성 (Undirected)으로 구분된다. 또한 가중치가 있는 경우 (Weighted)와 없는 경우 (Unweighted)로 구분된다. 무방향 그래프 (Undirected Graph)는 $G=(V,E)$로 표기하며, 방향 그래프 (Directed Graph)는 $G=(N,A)$는 로 표기한

*종신회원, 강릉원주대학교 멀티미디어공학과
**정회원, 강릉원주대학교, 멀티미디어공학과
접수일자 2011.7.27, 수정일자 2011.9.25
게재확정일자 2011.10.14

다. 노드 (Nodes, N)는 정점 (Vertices, V)이라 하며, 호 (Arcs, A)는 간선 (Edges, E)또는 화살표 (Arrows)라고도 한다.

그래프가 연결되어 있고, 무방향성이며, 가중치를 갖고 있는 경우 신장트리 (Spanning Tree, ST)를 구할 수 있다. ST는 사이클이 발생하지 않는 상태에서 그래프의 모든 정점들을 간선으로 연결한 트리이다. 하나의 그래프에서는 다수의 ST가 존재할 수 있다. 최소신장트리 (Minimum Spanning Tree, MST)는 그래프가 갖고 있는 다수의 ST들 중 모든 정점들을 연결하는 간선들의 가중치의 합이 최소가 되면서 사이클이 발생하지 않는 ST를 찾는 것으로 전기, 전화, 가스 또는 수도 분야에 활용될 수 있다.[1] 대표적인 MST 알고리즘으로는 Borůvka[2,3], Prim[4]과 Kruskal[5]이 있다. Borůvka MST 알고리즘[2,3]은 모든 간선들의 가중치가 서로 상이한 (Distinct) 그래프에서 MST를 찾는 알고리즘으로 제안되었으며, 현재는 Prim과 Kruskal MST 알고리즘이 널리 사용되고 있다.[1]

MST 알고리즘을 방향 그래프 (Directed Graph)에 적용할 수 있는가? 방향 그래프는 일반적으로 근 (Root) r이 지정되어 있으며, 근 r로부터 방향 경로 (Directed path)를 통해 G의 모든 노드에 도달할 수 있어야 한다. 이 조건을 만족시키기 위해서는 근을 제외한 모든 노드는 유입 차수 (In-degree, 유입되는 호의 수)가 반드시 1개이어야 하며, 유출 차수 (Out-degree, 유출되는 호의 수)는 0 또는 1개 이상이어도 상관없다. 방향그래프의 신장트리 (Directed Spanning Tree, DST)를 찾는 문제는 근 노드 r에서 임의의 모든 모드로 직접 경로를 갖도록 하는 트리를 형성하는 것이며, DMST (Directed Minimum-weight ST)는 방향그래프에 대해 사이클이 없는 신장트리를 형성하면서, 연결된 호들의 가중치 합이 최소가 되도록 하는 것이다.[6-10]

DMST를 찾는 대표적인 알고리즘으로는 Chu-Liu/Edmonds 알고리즘[9,11]을 적용할 수 있다. 그러나 Chu-Liu/Edmonds 알고리즘[9,11]을 적용할 경우 어떤 그래프에서는 DMST를 찾지 못하는 경우도 발생하며, 사이클 발생시 이를 해결하는 과정이 복잡한 단점을 갖고 있다. 또한, 방향성이 없는 간선들과 방향성이 있는 호의 차이로 인해 MST 알고리즘을 DMST를 찾는데 그대로 적용할 수는 없으며, 방향 그래프에 적합하도록 수정하여 적용하여야만 한다. 본 논문에서는 무방향 그래프의 MST를 찾는 Prim MST 알고리즘[4]과 Chu-Liu/Edmonds DMST 알고리즘[9,11]을 적용하여 방향 그래프의 DMST를 구하는 문제점을 고찰해 보고, 간단하면서도 항상 DMST를 찾을 수 있는 새로운 알고리즘을 제안한다.

2장에서는 방향 그래프 MST에 대한 기본 개념을 살펴보고, 실제 그래프에 Prim[4] 과 Chu-Liu/Edmonds 알고리즘[9,11]을 적용한 문제점을 고찰해 본다. 3장에서는 DMST를 보다 쉽게 찾을 수 있는 알고리즘을 제안한다. 4장에서는 15개의 다양한 실제 그래프들에 대해 제안된 DMST 알고리즘을 Prim[4]과 Chu-Liu/Edmonds 알고리즘[9,11]과 비교하여 알고리즘의 적용성을 평가해 본다.

II. 관련 연구와 연구 배경

1. 개념 정의

무방향 그래프 $G=(V,E)$의 간선은 $e=\{x,y\}$로, 방향 그래프 $G=(N,A)$의 호는 방향성이므로 순서쌍 $a=(x,y)$로 표기한다. 여기서, x는 꼬리 (Tail), y는 머리 (Head)라 하며, 노드 n은 유출 차수와 유입 차수를 갖는다. 방향 그래프 $G=(N,A)$를 트리로 볼 때, 하나의 노드를 근 (Root, r) 으로 지정한 경우를 “근이 있는 트리 (Rooted Tree)”라 하며, 근을 지정하지 않은 경우를 “자유 트리 (Free Tree)”라 한다.[12]

방향 그래프는 유입 분기 (In-branching)와 유출 분기 (Out-branching)로 구분할 수 있다. 유입 분기는 근 r에 근원을 두고 분기하는 트리로 임의의 노드에서 근 r로 방향 경로를 갖고 있는 경우이며, 유출 분기는 근 r은 방향 경로를 통해 G의 모든 노드에 도달할 수 있다.[13] 방향그래프의 신장트리 (DST)를 찾는 문제는 근 노드 r에서 임의의 모든 노드로 방향 경로를 갖도록 하는 트리를 형성하는 것이다. 여기서 방향 경로는 경로의 합이 최소가 되는 신장트리 (Minimum ST)와 최대가 되는 신장 트리 (Maximum ST)로 구분할 수 있다. 그러나 일반적으로는 최소신장트리를 찾고자 한다. 이는 도로와 전화망 분야에 활용될 수 있다.[14]

유입 분기 DMST에 대한 정의는 Krishnan과 Raghavachari[13]와 UNO[14]가 있다. Krishnan와 Raghavachari[13]는 ”방향그래프 $G(N,A)$와 근 노드 $r \in N$이 주어졌을 때, G의 모든 노드에서 근 노드 r로

도달 가능하다. r을 제외한 각 노드는 정확히 하나의 유출 호 (꼬리)를 가져야 하며, r은 유출 호를 갖지 않는다“고 정의하고 있다. UNO[14]는 ”DST는 2개의 호가 꼬리를 공유하지 않는 ST로 근 r을 제외한 각 노드는 정확히 한 호의 꼬리이다“고 정의하고 있다. 이들은 유입 분기 DMST를 정의하고 있으며, 근 r을 제외한 모든 노드는 $In-degree \geq 1$, $Out-degree = 1$, 근 r은 $In-degree \geq 1$, $Out-degree = 0$ 조건을 만족시켜야 한다.

반면에, 유출 분기 DMST에 대해서는 Jensen[6], Boyd[7], Duhamen et al.[8], Yang[9]과 Schrijver[10]가 있다. Jensen[6]은 “DST는 $n-1$개의 호로 구성되어 있으며, 호들은 지정된 근 노드 r로부터 모든 다른 노드로 유일한 방향 경로를 갖고 있다.”고 하였다. Boyd[7]와 Duhamen et al.[8]은 “MCA (Minimum Cost Arborescence)는 가중치를 갖는 방향 그래프 $G(N,A)$와 근 노드 $r \in N$가 지정되어 있을 때, r에 근원을 둔 G의 나뭇가지 (Arborescence)는 r에 근원을 둔 G의 DST이다. 즉, 트리에서 근 r로부터 모든 다른 노드로 유일한 방향경로가 존재한다. 따라서 MCA 문제는 r에 근원을 둔 G의 MCA를 찾는 것이다.”고 하였다. Yang[9]은 방향 그래프 $G(N,A)$에서, RDST (Rooted Directed Spanning Tree) $G(N,S)$: $S \in A$는 S에 있는 모든 호의 비용 (가중치)의 합이 최소인 ST이며, $G(N,S)$는 근을 제외한 각 노드는 단지 하나의 유입 호만을 가져야 하며, 연결되어 있고, 사이클이 없으며, $n-1$개의 호를 가진 모든 노드들로 구성된 그래프이다.“고 정의하고 있다. 즉, 근 r을 제외한 모든 노드가 $In-degree = 1$, $Out-degree \geq 1$를 갖는 ST이다. Schrijver[10]는 ”OB (Optimum Branchings)는 근 r을 가진 방향 그래프 $G(N,A)$에서, (N,A')가 근 r을 가진 트리라면, A의 부분집합 S는 근 r을 가진 분기이며, 이를 r-분기 (r-branching) 또는 r-나무가지 (r-arborescence)라 한다. 만약 A'가 r-분기라면, 각 $s \in N$에 대해 A'에는 단지 하나의 r-s 경로가 존재하며, r로부터 D에 있는 각 노드로 도달할 수 있으면, D는 r-분기이다.”고 정의하고 있다. 이와 같이 방향 그래프의 정의에 대해 유입 분기와 유출 분기로, 최소신장트리를 MDST, DMST, MCA, RDST, OB 등 다양한 용어로 정의하고 있다. 본 논문에서는 유출 분기 방향 그래프의 최소 신장트리를 찾는 문제에 초점을 맞추며, 이를 “DMST” 용어로 통일한다.

2. 무방향 그래프의 최소신장트리 (MST)

최소신장트리 (MST)는 일반적으로 무방향 그래프의 모든 정점을 연결하는 간선들의 가중치 합이 최소가 되는 신장트리를 찾는 문제로 알려져 있다. MST를 찾는 대표적인 알고리즘으로는 Borůvka[2,3], Prim[4]과 Kruskal[5]이 있다. Borůvka MST 알고리즘[2,3]은 각 정점에 연결된 MWE (Minimum-Weight Edge)를 선택하고 사이클 (루프)이 발생하는 간선들을 삭제한다. 선택된 MWE들을 연결하여 MSF (Minimum Spanning Forest)를 생성한다. 다음으로 MSF 간 MWE를 선택하여 사이클 (병렬)이 발생하는 간선들을 삭제하면 최종적으로 MST를 얻는다. Prim MST 알고리즘[4]은 임의의 정점을 선택하고, 이에 연결된 간선들 중에서 MWE를 선택한다. 이 간선에 연결된 정점의 간선들 가중치와 기존에 선택된 정점에서 선택되지 않은 간선들 가중치 중에서 MWE를 선택하는 방법이다. (단, 이 과정에서 사이클이 발생하는 간선은 무시한다.) 이 방법을 모든 정점들이 선택될 때까지 수행한다. Kruskal MST 알고리즘[5]은 그래프의 모든 간선들을 대상으로 오름차순으로 정렬시키고, 첫 번째 MWE부터 시작하여 사이클이 발생하지 않는 한 간선들을 $v-1$개가 될 때까지 선택하는 방법이다.

Borůvka[2,3]와 Kruskal[5] MST 알고리즘은 그래프의 간선들을 대상으로 최소 가중치를 갖는 간선들을 선택하여 신장시키는 방법이며, Prim MST 알고리즘[4]은 정점들을 최소 가중치를 갖는 간선들로 신장시키는 방법이다. 무방향 그래프에 적용된 MST 알고리즘을 방향 그래프에 적용시키려면 Borůvka[2,3]와 Kruskal[5] MST 알고리즘으로 DMST를 찾는 과정에서 “방향그래프의 한 노드의 $In-degree = 1$이 되어야 하며, 근 r로부터 다른 모든 노드로 방향경로가 설정되어야 한다”는 조건을 만족시키지 못하는 경우가 빈번히 발생할 수 있다. 따라서 Borůvka[2,3]와 Kruskal[5] MST 알고리즘으로는 DMST에 적용할 수 없다. 반면에 Prim MST 알고리즘[4]은 근 노드로부터 방향 경로를 갖도록 노드들을 연결하는 방법으로 방향그래프의 신장트리를 얻는 조건은 만족시킬 수 있다. Prim MST 알고리즘[4]을 방향 그래프에 적용하기 위해 수정된 형태의 DMST 알고리즘은 그림 1에 제시하였다.

Candidate Arcs : ST를 구성하기 위해 연결에 이용될 후보 Arc들 $(a_r + a_u - a_d)$
a_u : 새로 추가되는 Node의 Out-degree Arcs, a_r : 삭제되지 않고 남아 있는 Arcs
a_d : MSF Nodes로 이미 선택되었거나 사이클 발생으로 삭제되는 Arcs
MWA : Minimum-Weight Arc (x, y), MSF = MSF Nodes + MSF Arcs
$|n| \times |n|$ Adjacency 행렬 작성
Step 1. N에서 근 노드 선택, MSF Nodes에 저장, N에서 근 노드 삭제
Step 2. FOR $|n|$ *to* 0 do
MSF Nodes에 추가된 노드의 Out-degree Arcs들을 Candidate Arcs에 추가(a_u)
IF Candidate Arcs $(a_r + a_u)$ 중 $y \in MSF\,Nodes$ THEN Candidate Arcs에서 삭제(a_d)
/* 사이클 발생 Arc 삭제
END
Candidate Arcs에 남아 있는 Arcs들$(a_r + a_u - a_d)$ 중 MWA 선택**(단, x가 동일한 값이 다수 존재시 모두 선택, y가 동일한 값이 다수 존재시 하나만 선택, 나머지는 삭제)**
선택된 MWA를 MSF Arcs에 추가, Candidate Arcs에서 삭제
선택된 MWA의 (x, y)에서 y를 MSF Nodes에 저장, N에서 삭제
END

그림 1. Prim DMST 알고리즘
Fig. 1. Prim DMST Algorithm

3. 방향 그래프의 최소신장트리 (DMST)

Chu-Liu/Edmonds DMST 알고리즘[9,11]은 다음과 같이 수행된다.

Step 1. ① 만약 근 노드로 유입되는 호가 있다면 삭제한다.
② 근을 제외한 모든 노드에 대해 최소 가중치를 가진 유입 호를 선택하고, 선택된 $n-1$개의 호로 S 집합을 생성한다.
③ S가 사이클이 형성되지 않으면 $G(N,S)$는 MST가 되며, 알고리즘을 종료한다. 만약 그렇지 않으면 Step 2를 수행한다.

Step 2. ④ 각 사이클에 대해, 사이클에 있는 노드들을 Pseudo-node k로 축소시키고, 사이클 외부의 어떤 노드 i에서 사이클에 있는 노드 j로 유입 호의 가중치를 $c(i,k) = c(i,j) - [c(x_j,j) - \min_j[c(x_j,j)]$로 수정한다. 여기서 $c(x_j,j)$는 사이클에 있는 임의의 노드에서 j 노드로 유입되는 호의 가중치이다.

Step 3. ⑤ 각 Pseudo-node에 대해, 수정된 최소 가중치를 가진 유입 호를 선택한다.
⑥ 선택된 사이클 유입 호를 사이클에 존재하는 동일한 실제 노드로 유입되는 호와 교체한다.
⑦ 만약 사이클이 추가로 존재하면 Step 2로 복귀한다.

4. 알고리즘 적용 문제점과 연구 배경

그림 2의 3개 그래프에 대해 Prim[4] MST 알고리즘과 Chu-Liu/Edmonds DMST 알고리즘[9,11]을 적용하여 보자. G_1 그래프는 Yang[9]에서 인용되었으며, $In-degree = 0$인 근 노드가 ①임을 알 수 있다. G_2 그래프는 Llewelly[15]에서, G_3 그래프는 Ikeda[16]에서 인용되었다. G_2와 G_3 그래프는 근 노드가 없어 DMST를 구하기 위해서는 임의로 근 노드를 설정하여야 한다. 본 논문에서는 편의상 노드 ⓐ를 근 노드로 설정하고 알고리즘을 적용하였다.

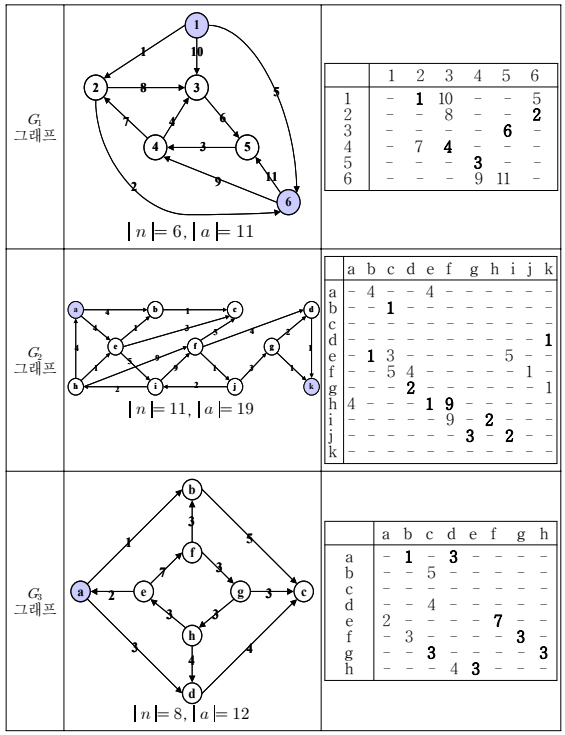


그림 2. 알고리즘에 사용된 그래프
Fig. 2. Graphs for Algorithm

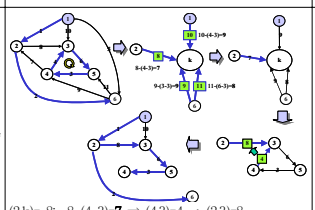
G_1 그래프에 대해 Prim DMST 알고리즘[4]과 Chu-Liu/Edmonds DMST 알고리즘[9,11]을 적용한 결과는 표 1에 제시되어 있다. Prim DMST 알고리즘[4]은 근 노드 ①로부터 모든 노드로 유일한 방향 경로가 설정되었고, 사이클이 없고, 근 노드를 제외한 모든 노드의 $In-degree = 1$이며, $\Sigma w(e) = 20$인 MST를 얻었다. Chu-Liu/Edmonds DMST 알고리즘[9,11]은 ③-④-⑤에서 사이클이 발생하여 이 사이클로 유입되는 최소 가중치 호인 $(2,3) = 8$이 선택되고 ③ 노드로 유입되는 $(4,3) = 4$가 삭제되었다. 따라서 근 노드 ①로부터 모든 노드로 유일한 방향 경로가 설정되었으며, 사이클이 발생하지 않았으며, 근 노드를 제외한 모든 노드의 $In-degree = 1$이며, $\Sigma w(e) = 20$인 MST를 얻을 수 있었다.

표 1. G_1 그래프의 DMST 알고리즘 적용
Table 1. DMST Algorithm Application of Graph G_1

(a) Prim DMST 알고리즘

N	MSF Nodes	Candidate Arcs			MWA 선정 Candidate Arcs $(a_r + a_a - a_d)$	MSF Arcs (MWA)
		a_a	a_r	Cycle (a_d)		
{1,2,3,4,5,6}	{ ∅ }	{ ∅ }	{ ∅ }	{ ∅ }	{ ∅ }	{ ∅ }
{2,3,4,5,6}	{1}	(1,2)=1,(1,3)=10 (1,6)=5	{ ∅ }	{ ∅ }	**(1,2)=1**,(1,3)=10 (1,6)=5	**(1,2)=1**
{3,4,5,6}	{1,2}	(2,3)=8,(2,6)=2	(1,3)=10,(1,6)=5	{ ∅ }	(1,3)=10,(1,6)=5 (2,3)=8,**(2,6)=2**	**(2,6)=2**
{3,4,5}	{1,2,6}	(6,4)=9,(6,5)=11	(1,3)=10,**(1,6)=5** (2,3)=8	**(1,6)=5**	(1,3)=10,**(2,3)=8** (6,4)=9,(6,5)=11	**(2,3)=8**
{4,5}	{1,2,6,3}	(3,5)=6	**(1,3)=10**,(6,4)=9 (6,5)=11	**(1,3)=10**	(6,4)=9,(6,5)=11 **(3,5)=6**	**(3,5)=6**
{4}	{1,2,6,3,5}	(5,4)=3	(6,4)=9,**(6,5)=11**	**(6,5)=11**	(6,4)=9,**(5,4)=3**	**(5,4)=3**
{ ∅ }	{1,2,6,3,5,4}	**(4,2)=7**,**(4,3)=4**	**(6,4)=9**	**(6,4)=9**,**(4,2)=7** **(4,3)=4**	{ ∅ }	-

(b) Chu-Liu/Edmonds DMST 알고리즘

Node	노드 유입 MWA		MSF Arcs	MSF 유입 MWA	DMST Arcs
	ALL	선택			
1	-	-	-		-
2	(1,2)=1	(1,2)=1	(1,2)=1		(1,2)=1
3	(4,3)=4	(4,3)=4	**(4,3)=4**		**(2,3)=8**
4	(5,4)=3	(5,4)=3	(5,4)=3		(5,4)=3
5	(3,6)=6	(3,6)=6	(3,6)=6		(3,6)=6
6	(2,6)=2	(2,6)=2	(2,6)=2		(2,6)=2

(2,k)= 8: 8-(4-3)=**7** ⇒ (4,3)=4 → (2,3)=8
(1,k)=10: 10-(4-3)=9
(6,k)= 9: 9-(3-3)=9
(6,k)=11: 11-(6-3)=8

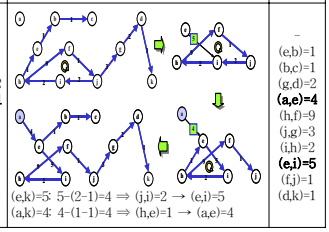
G_2 그래프에 대해 Prim DMST 알고리즘[4]과 Chu-Liu/Edmonds DMST 알고리즘[9,11]을 적용한 결과는 표 2에 제시되어 있다.

표 2. G_2 그래프의 DMST 알고리즘 적용
Table 2. DMST Algorithm Application of Graph G_2

(a) Prim DMST 알고리즘

N	MSF Nodes	Candidate Arcs			MWA선정 Candidate Arcs $(a_r + a_a - a_d)$	MSF Arcs (MWA)
		a_a	a_r	Cycle (a_d)		
{a,b,c,d,e,f,g,h,i,j,k}	{ ∅ }	{ ∅ }	{ ∅ }	{ ∅ }	{ ∅ }	{ ∅ }
{b,c,d,e,f,g,h,i,j,k}	{a}	(a,b)=4,(a,e)=4	{ ∅ }	{ ∅ }	**(a,b)=4**,**(a,e)=4**	**(a,b)=4** **(a,e)=4**
{c,d,f,g,h,i,j,k}	{a,b,e}	(b,c)=1,**(e,b)=1** (e,c)=3,(e,i)=5	{ ∅ }	**(e,b)=1**	**(b,c)=1**,(e,c)=3 (e,i)=5	**(b,c)=1**
{d,f,g,h,i,j,k}	{a,b,e,c}	{ ∅ }	**(e,c)=3**,(e,i)=5	**(e,c)=3**	**(e,i)=5**	**(e,i)=5**
{d,f,g,h,j,k}	{a,b,e,c,i}	(i,f)=9,(i,h)=2	{ ∅ }	{ ∅ }	(i,f)=9,**(i,h)=2**	**(i,h)=2**
{d,f,g,j,k}	{a,b,e,c,i,h}	**(h,a)=4**,**(h,e)=1** (h,f)=9	(i,f)=9	**(h,a)=4**,**(h,e)=1**	**(i,f)=9**,**(h,f)=9**	**(i,f)=9**
{d,g,j,k}	{a,b,e,c,i,h,f}	**(f,c)=5**,(f,d)=4 (f,j)=1	{ ∅ }	**(f,c)=5**	(f,d)=4,**(f,j)=1**	**(f,j)=1**
{d,g,k}	{a,b,e,c,i,h,f,j}	(j,g)=3,**(j,i)=2**	(f,d)=4	**(j,i)=2**	(f,d)=4,**(j,g)=3**	**(j,g)=3**
{d,k}	{a,b,e,c,i,h,f,j,g}	(g,d)=2,(g,k)=1	(f,d)=4	{ ∅ }	(f,d)=4,(g,d)=2 **(g,k)=1**	**(g,k)=1**
{d}	{a,b,e,c,i,h,f,j,g,k}	{ ∅ }	(f,d)=4,(g,d)=2	{ ∅ }	(f,d)=4,**(g,d)=2**	**(g,d)=2**
{ ∅ }	{a,b,e,c,i,h,f,j,g,k,d}	**(d,k)=1**	**(f,d)=4**	**(f,d)=4**,**(d,k)=1**	{ ∅ }	-

(b) Chu-Liu/Edmonds DMST 알고리즘

Node	노드 유입 MWA		MSF Arcs	MSF 유입 MWA	DMST Arcs
	ALL	선택			
a	(h,a)=4	-	-		-
b	(e,b)=1	(e,b)=1	(e,b)=1		(e,b)=1
c	(b,c)=1	(b,c)=1	(b,c)=1		(b,c)=1
d	(g,d)=2	(g,d)=2	(g,d)=2		(g,d)=2
e	(h,e)=1	(h,e)=1	**(h,e)=1**		**(a,e)=4**
f	(h,f)=9,(i,f)=9	(h,f)=9	(h,f)=9		(h,f)=9
g	(j,g)=3	(j,g)=3	(j,g)=3		(j,g)=3
h	(i,h)=2	(i,h)=2	(i,h)=2		(i,h)=2
i	(j,i)=2	(j,i)=2	**(j,i)=2**		**(e,i)=5**
j	(f,j)=1	(f,j)=1	(f,j)=1		(f,j)=1
k	(d,k)=1,(g,k)=1	(d,k)=1	(d,k)=1		(d,k)=1

(e,k)=5: 5-(2-1)=4 ⇒ (j,i)=2 → (e,i)=5
(a,k)=4: 4-(1-1)=4 ⇒ (h,e)=1 → (a,e)=4

Prim DMST 알고리즘[4]은 근 노드 ⓐ로부터 모든 노드로 방향경로가 설정되었으며, $\Sigma w(e) = 32$를 얻었다. Chu-Liu/Edmonds DMST 알고리즘은 최소 가중치를 가진 유입 호가 다수 발생하였을 경우 어떤 호를 선택하는지에 대해 명확히 제시하지 않고 있다. 그러나 DMST는 근을 제외한 모든 노드의 유입 호가 1개만이 존재해야 하므로 본 논문에서는 1개만을 선택하였다.

Chu-Liu/Edmonds DMST 알고리즘[9,11]은 근 노드를 ⓐ라 할 때, ⓗ-ⓕ-ⓘ-ⓙ에 사이클이 발생하였다. 사이클로 유입되는 호는 $(e,i) = 5$만 존재하며, 이를 연결하는 경우 ⓘ 노드로 유입되는 $(j,i) = 2$ 호가 삭제된다. 이 경우 다시 ⓗ-ⓔ-ⓘ가 사이클이 발생하였으며, 이 사이클로 유입되는 호는 $(a,e) = 4$만 존재하여 $(h,e) = 1$이 $(a,e) = 4$로 대체되었다. 결국, $\Sigma w(e) = 29$인 MST를 얻었다. 결국 Chu-Liu/Edmonds DMST 알고리즘이 Prim DMST 알고리즘보다 호들의 가중치 합이 작은 DMST를 얻을 수 있었다.

G_3 그래프에 대해 Prim DMST 알고리즘[4]과 Chu-Liu/Edmonds DMST 알고리즘[9,11]을 적용한 결과는 표 3에 제시되어 있다. Prim DMST 알고리즘[4]은 ⓒ 노드에서 더 이상의 유출 호가 존재하지 않아 알고리즘이 종료되었으며, ST를 얻는데 실패하였다. 또한, Chu-Liu/Edmonds DMST 알고리즘[9,11]도 ⓔ-ⓕ-ⓖ-ⓗ에서 사이클이 발생하였으나 이 사이클로 유입되는 호가 없어 알고리즘이 수행되지 못하였으며, ST를 얻는데 실패하였다.

표 3. G_3 그래프의 DMST 알고리즘 적용
Table 3. DMST Algorithm Application of Graph G_3

(a) Prim DMST 알고리즘

N	MSF Nodes	Candidate Arcs			MWA 선정 Candidate Arcs $(a_r + a_u - a_d)$	MSF Arcs (MWA)
		a_u	a_r	Cycle (a_d)		
{a,b,c,d,e,f,g,h}	{ ø }	{ ø }	{ ø }	{ ø }	{ ø }	{ ø }
{b,c,d,e,f,g,h}	{a}	(a,b)=1,(a,d)=3	{ ø }	{ ø }	**(a,b)=1**,(a,d)=3	**(a,b)=1**
{c,d,e,f,g,h}	{a,b}	(b,c)=5	(a,d)=3	{ ø }	**(a,d)=3**,(b,c)=5	**(a,d)=3**
{c,e,f,g,h}	{a,b,d}	(d,c)=4	(b,c)=5	{ ø }	(b,c)=5,**(d,c)=4**	**(d,c)=4**
{e,f,g,h}	{a,b,d,c}	{ ø }	**(b,c)=5**	**(b,c)=5**	{ ø }	-
		알고리즘 수행 불가				

(b) Chu-Liu/Edmonds DMST 알고리즘

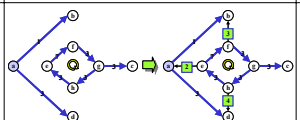
Node	노드 유입 MWA		MSF Arcs	MSF 유입 MWA	DMST Arcs
	ALL	선택			
a	(e,a)=2	-	-		실패
b	(a,b)=1	(a,b)=1	(a,b)=1		
c	(g,c)=3	(g,c)=3	(g,c)=3		
d	(a,d)=3	(a,d)=3	(a,d)=3		
e	(h,e)=3	(h,e)=3	(h,e)=3		
f	(e,f)=7	(e,f)=7	(e,f)=7		
g	(f,g)=3	(f,g)=3	(f,g)=3		
h	(g,h)=3	(g,h)=3	(g,h)=3	ⓔ-ⓕ-ⓖ-ⓗ 사이클로 유입되는 호가 없어 MST를 구하지 못함	

그림 2의 3개 그래프에 대해 Prim[4]과 Chu-Liu/Edmonds DMST 알고리즘[9,11]을 적용하여 얻은 DMST는 그림 3에 제시되어 있다.

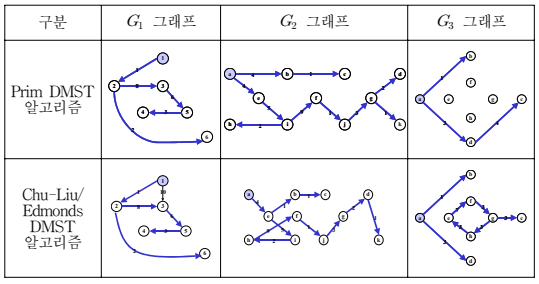


그림 3. DMST
Fig. 3. DMST

결국, 방향 그래프에 Prim[4]과 Chu-Liu/Edmonds MST 알고리즘[9,11]을 적용시 다음과 같은 의문이 제기된다.

(1) Chu-Liu/Edmonds MST 알고리즘처럼 그래프에서 직접 MST를 찾는 수동적 방법에서 인접 행렬(Adjacency Matrix)을 이용하여 MST를 구하는 프로그램을 구현할 수 있는가?

(2) G_2 그래프처럼 근 노드가 명확히 정의되지 않은 경우, 근 노드를 임의로 노드 ⓐ로 설정하지 않고 가중치 합을 최대로 줄일 수 있는 MST를 얻으면서 근 노드를 찾는 방법은 없는가?

(3) G_3 그래프처럼 근 노드가 명확히 정의되지 않은 경우, 근 노드를 임의로 노드 ⓐ로 설정하고 알고리즘을 수행시 MST를 얻지 못하는 문제점을 해결할 수 있는가?

(4) Chu-Liu/Edmonds MST 알고리즘의 사이클을 제거하는 과정을 단순화 시킬 수 있는 방법은 없는가?

Ⅲ. DMST 알고리즘

1. 알고리즘 제안

방향 그래프의 MST (DMST)는 다음 조건을 만족시켜야만 한다. (1) 근 노드 r의 $In-degree=0$가 되어야 한다. 만약에 근 노드의 $In-degree$가 존재하면 이를 무시한다. (2) 근 노드 r로부터 다른 모든 노드로 유일한 방향 경로가 설정되어야 한다. 따라서 근 노드를 제외한 모든 노드는 반드시 $In-degree=1$이 되어야만 한다. 단, $Out-degree \geq 1$또는 $Out-degree=0$이어도 상관없다. (3) 사이클 (닫힌 경로)이 발생하지 않아야 한다. (4) 모든 호들의 가중치 합이 다른 ST와 비교시 최소가 되어야 한다. 이러한 조건을 만족시키기 위해 Chu-Liu/Edmonds DMST 알고리즘[9,11]은 한 노드의 유입 호 중 최소의 가중치를 갖는 호를 선택하는 방법으로 제안되었다. 그러나 G_3 그래프처럼 사이클이 발생하지 않고 알고리즘이 종료되었음에도 불구하고 MST를 얻지 못하는 문제점을 발생시켰다. 따라서 이러한 문제점을 해결할 수 있는 DMST 알고리즘을 다음과 같이 제안한다.

Step 1. ① 근 노드를 포함한 모든 노드에 대해 최소 가중치를 갖는 유입 호 (In-degree MWA)를 선택한다. 단, 근 노드의 유입 호를 삭제하지 않는다. 또한 만약 하나의 노드에서 동일한 가중치를 가진 유입 호가 다수 존재시 1개만 선택한다.)

Step 2. ② 선택된 MWA들을 오름차순으로 정렬한다.
③ 최소 가중치를 갖는 MWA부터 순서대로 노드들을 연결하는 MSF를 생성하면서 사이클이 발생하는 호 (하나의 MSF 에 머리와 꼬리가 모두 속한 경우)를 제거한다.

④ 만약 MSF가 1개이면 DMST가 얻어지며, 알고리즘을 종료한다. 만약 그렇지 않으면 Step 3을 수행한다.

Step 3. ⑤ 모든 MSF의 유입 호들을 선택한다.

⑥ 유입 호가 2개 이상인 MSF에 대해, MSF 유입 호와 머리가 동일한 MSF의 호 (각 노드는 유입 호가 1개만이 존재해야 함)는 제거한다. MSF 유입 호와 MSF의 나머지 호들의 가중치를 합한다.

⑦ 각 MSF에서 가중치 합이 최소인 유입 호를 동일한 머리를 가진 MSF 호와 대체시킨다.

제안된 알고리즘을 Sulee DMST 알고리즘이라 하며, 그림 4에 제시되어 있다. 제안된 알고리즘은 $In-degree=0$인 노드가 존재하는 그래프의 경우 Chu-Liu/Edmonds DMST 알고리즘과 동일한 DMST를 얻는다. 그러나 $In-degree=0$인 노드가 없는 그래프에서는 가중치 합을 최소로 하는 DMST를 얻으면서 이에 따른 근 노드도 함께 구할 수 있는 장점이 있다. 또한, Chu-Liu/Edmonds DMST 알고리즘의 사이클 발생 문제점을 해결하는 가중치 수정 과정을 생략할 수 있다.

2. 적용 방법

그림 2의 3개 그래프에 Sulee DMST 알고리즘을 적용한 과정은 표 4에, 적용 결과 얻은 MST는 그림 5에 제시되어 있다. G_1 그래프는 $\Sigma w(e)=20$으로 Chu-Liu/Edmonds DMST 알고리즘[9,11]과 동일한 가중치 합과 MST를 얻었으며, Pseudo-node를 적용하고 유입 호의 가중치를 수정하는 알고리즘이 생략되어 알고리즘이 간략화 되었음을 알 수 있다. G_2 그래프는 $\Sigma w(e)=18$을 얻어 Chu-Liu/Edmonds DMST 알고리즘[9,11]의 $\Sigma w(e)=29$를 능가하는 실질적인 MST를 얻었으며, 근 노드는 ⓐ가 아닌 ⓕ로 변경되었다. 따라서 근 노드를 ⓕ로 설정하면 가중치 합이 가장 적은 MST를 얻을 수 있다. G_3 그래프에 대해, Chu-Liu/Edmonds DMST 알고리즘[9,11]은 MST를 얻는데 실패하였으나 Sulee DMST 알고리즘은 $\Sigma w(e)=18$인 MST를 얻었으며, 근 노드는 ⓐ가 아닌 ⓕ로 변경되었다.

Adjacency 행렬 $(i,j), i,j=1,2,\cdots n$ 작성
MWA : Minimum-weight Arc, MSF : Minimum Spanning Forest
MST : Minimum Spanning Tree
a_c : Candidate Arcs (Node Incoming Minimum Arcs)
n_f : MSF Nodes
a_{in} : MSF Incoming Arcs

/* **1st Stage : 각 열 노드에 대해 MWA (노드 유입 MWA) 선택, MSF생성**
1-1. 근 노드를 포함한 **모든 열 노드의 최소 가중치를 갖는 호 선택 (단, 동일한 가중치가 다수 존재시 1개만 선택)**, Candidate Arcs에 저장
1-2 **Candidate Arcs (a_c)를 오름차순으로 정렬**
1-3 사이클 발생 호 제거, MSF (n_f) 생성
FOR 1 to $|a_c|$ do /* Candidate Arcs의 최소 가중치를 가진 호부터 MSF Arcs 선택
IF x와 y ∈ 하나의 n_f THEN **Skip** /* 사이클 발생 호 제거
ELSE IF x와 y ∈ MSF의 두 n_f 원소
해당 Arc를 MSF Arcs에 추가, 해당 Arc를 해당 MSF에 연결
ELSE IF x와 y ∉ n_f THEN
해당 Arc를 MSF Arcs에 추가, 해당 Arc로 새로운 MSF 생성
ENDIF
END
IF $|n_f|=1$ THEN 알고리즘 종료
ELSE IF $|n_f|\geq 2$ THEN 2nd Stage 수행
ENDIF

/* **2nd Stage : MSF를 연결하는 MWA (MSF 유입 MWA) 선택, MST를 찾음**
2-1. FOR 1 to $|n_f|$ do /* MSF의 유입 호 선택
각 MSF를 형성하는 노드들에 대해 인접행렬의 열 (유입) 호 선택
선택된 호들의 Tail이 MSF의 노드와 동일한 호 삭제
END
2-2. FOR 1 to $a_{in}\geq 2$ do /* 유입 호가 2개 이상인 MSF의 MFT 유입 MWA 선택
MSF에 포함된 호들 중 MSF 유입 호와 Head가 동일한 호 삭제
MSF 유입 호와 MSF의 나머지 호들의 가중치 합 계산, 최소 가중치를 갖는 외부 유입 호 선택
가중치 합이 최소인 MSF 유입 호 계산에 사용된 호들을 MSF Arcs에 추가, MSF 유입 호와 동일한 Head를 가진 호 삭제
END

그림 4. Sulee DMST 알고리즘
Fig. 4. Sulee DMST Algorithm

만약 G_2와 G_3에 대해 모든 노드의 최소 가중치를 갖는 In-degree MWA를 선택한 후 이들을 오름차순으로 정렬하지 않을 경우, 사이클 검증 과정에서 가중치가 보다 큰 호가 선택되어 MST를 얻지 못하는 경우가 발생한다. 이에 대한 예는 표 5에 제시하였다.

표 4. Sulee DMST 알고리즘 적용
Table 4. Sulee DMST Algorithm Application

(a) G_1 그래프

Node	노드 유입 MWA				MSF	
	ALL	선택	정렬	Cycle Check	Nodes	Arcs
1	-	-	-	-		
2	(1,2)=1	(1,2)=1	(1,2)=1	x		(1,2)=1
3	(4,3)=4	(4,3)=4	(2,6)=2	x	1-**2**	(2,6)=2
4	(5,4)=3	(5,4)=3	(5,4)=3	x	1-2-6	(5,4)=3
5	(3,5)=6	(3,5)=6	(4,3)=4	x	1-2-6, **4**-5	(4,**3**)=4
6	(2,6)=2	(2,6)=2	**(3,5)=6**	O	1-2-6, **3**-4-**5** 1-2-6, 3-4-5	-

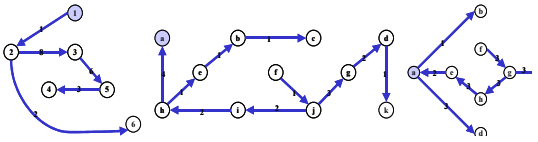
MSF	MSF 유입 MWA				MSF Arcs	DMST Arcs
	유입 호	대상	선택	가중치 계산		
F_a: (1,2,6) (1,2)=1,(2,6)=2	(**1**,2)=1,(4,2)=7 (**1**,6)=5,(**2**,6)=2	(4,2)=7	-	-	(1,2)=1 (2,6)=2	(1,2)=1 (2,6)=2
F_b: (3,4,5) (5,4)=3,(4,3)=4 (3,5)=6	(1,3)=10,(2,3)=8 (**4**,3)=4,(**5**,4)=3 (6,4)=9,(**3**,5)=6 (6,5)=11	(1,3)=10 (2,3)= 8 (6,4)= 9 (6,5)=11	(1,3)=10 (2,3)= 8 (6,4)= 9 (6,5)=11	(1,**3**)+(3,5)+(5,4)=10+6+3=19 **(2,3)+(3,5)+(5,4)=8+6+3=17** (6,**4**)+(4,3)+(3,5)=9+4+6=19 (6,**5**)+(5,4)+(4,3)=11+3+4=18	(5,4)=3 (4,**3**)=4	(5,4)=3 **(2,3)=8** (3,5)=6

(b) G_2 그래프

Node	노드 유입 MWA				MSF	
	ALL	선택	정렬	Cycle Check	Nodes	Arcs
a	(h,a)=4	(h,a)=4	(e,b)=1	x	-	(e,b)=1
b	(e,b)=1	(e,b)=1	(b,c)=1	x	**b**-e	(b,c)=1
c	(b,c)=1	(b,c)=1	(h,e)=1	x	b-c-**e**	(h,e)=1
d	(g,d)=2	(g,d)=2	(f,j)=1	x	b-c-e-h	**(f,j)=1**
e	(h,e)=1	(h,e)=1	(d,k)=1	x	b-c-e-h, f-j	(d,k)=1
f	(h,f)=9,**(i,f)=9**	(h,f)=9	(g,d)=2	x	b-c-e-h, f-j, **d**-k	(g,d)=2
g	(j,g)=3	(j,g)=3	(i,h)=2	x	b-c-e-**h**, f-j, d-g-k	(i,h)=2
h	(i,h)=2	(i,h)=2	(j,i)=2	x	b-c-e-h-**i**, f-**j**, d-g-k	(j,i)=2
i	(j,i)=2	(j,i)=2	(j,g)=3	x	b-c-e-f-h-i-**j**, d-**g**-k	(j,g)=3
j	(f,j)=1	(f,j)=1	(h,a)=4	x	b-c-d-e-f-g-**h**-i-j-k	(h,a)=4
k	(d,k)=1,**(g,k)=1**	(d,k)=1	(h,f)=9	O	a-b-c-d-e-**f**-g-**h**-i-j-k	-
					a-b-c-d-e-f-g-h-i-j-k	

(c) G_3 그래프

Node	노드 유입 MWA				MSF	
	ALL	선택	정렬	Cycle Check	Nodes	Arcs
a	(e,a)=2	(e,a)=2	(a,b)=1	x	-	(a,b)=1
b	(a,b)=1	(a,b)=1	(e,a)=2	x	**a**-b	(e,a)=2
c	(g,c)=3	(g,c)=3	(g,c)=3	x	a-b-e	(g,c)=3
d	(a,d)=3	(a,d)=3	(a,d)=3	x	**a**-b-e, c-g	(a,d)=3
e	(h,e)=3	(h,e)=3	(h,e)=3	x	a-b-d-**e**, c-g	(h,e)=3
f	(e,f)=7	(e,f)=7	(f,g)=3	x	a-b-d-e-h, c-**g**	(f,g)=3
g	(f,g)=3	(f,g)=3	(g,h)=3	x	a-b-d-e-**h**, c-f-**g**	**(g,h)=3**
h	(g,h)=3	(g,h)=3	(e,f)=7	O	a-b-c-d-**e**-**f**-g-h	-
					a-b-c-d-e-f-g-h	



(a) G_1 그래프 (b) G_2 그래프 (c) G_3 그래프

그림 5. Sulee DMST 알고리즘의 MST
Fig. 5. MST of the Sulee DMST

표 5. 정렬을 하지 않은 Sulee DMST 알고리즘 적용
Table 5. Sulee DMST Algorithm Application without sorting

(a) G_2 그래프

Node	In-degree MWA			MSF	
	ALL	선택	Cycle Check	Nodes	Arcs
a	(h,a)=4	(h,a)=4	x	-	(h,a)=4
b	(e,b)=1	(e,b)=1	x	a-h	(e,b)=1
c	(b,c)=1	(b,c)=1	x	a-h, **b**-e	(b,c)=1
d	(g,d)=2	(g,d)=2	x	a-h, b-c-e	(g,d)=2
e	(h,e)=1	(h,e)=1	x	a-**h**, b-c-**e**, d-g	(h,e)=1
f	(h,f)=9,**(i,f)=9**	(h,f)=9	x	a-b-c-e-**h**, d-g	**(h,f)=9**
g	(j,g)=3	(j,g)=3	x	a-b-c-e-f-h, d-**g**	(j,g)=3
h	(i,h)=2	(i,h)=2	x	a-b-c-e-f-**h**, d-g-j	(i,h)=2
i	(j,i)=2	(j,i)=2	x	a-b-c-e-f-h-**i**, d-g-**j**	(j,i)=2
j	(f,j)=1	(f,j)=1	O	a-b-c-d-e-**f**-g-h-i-**j**	-
k	(d,k)=1,**(g,k)=1**	(d,k)=1	x	a-b-c-**d**-e-f-g-h-i-j	(d,k)=1
				a-b-c-d-e-f-g-h-i-j-k	

(b) G_3 그래프

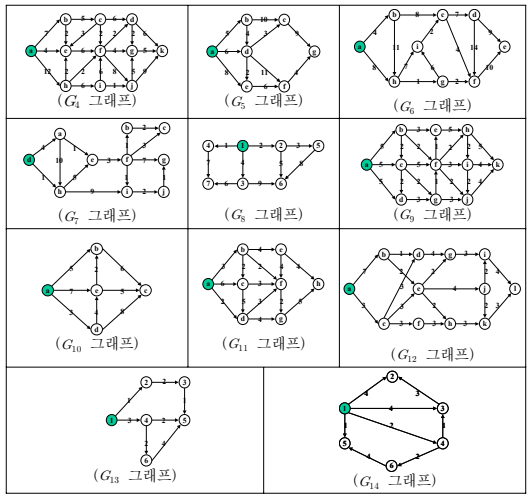
Node	In-degree MWA			MSF	
	ALL	선택	Cycle Check	Nodes	Arcs
a	(e,a)=2	(e,a)=2	x	-	(e,a)=2
b	(a,b)=1	(a,b)=1	x	**a**-e	(a,b)=1
c	(g,c)=3	(g,c)=3	x	a-b-e	(g,c)=3
d	(a,d)=3	(a,d)=3	x	**a**-b-e, c-g	(a,d)=3
e	(h,e)=3	(h,e)=3	x	a-b-d-**e**, c-g	(h,e)=3
f	(e,f)=7	(e,f)=7	x	a-b-d-e-h, c-**g**	**(e,f)=7**
g	(f,g)=3	(f,g)=3	x	a-b-d-e-**f**-h, c-**g**	(f,g)=3
h	(g,h)=3	(g,h)=3	O	a-b-c-d-e-f-**g**-**h**	-
				a-b-c-d-e-f-g-h	

표 5에서 MWA를 정렬시키지 않은 경우, G_2 그래프는 $\Sigma w(e)=26$, G_3 그래프는 $\Sigma w(e)=22$로 ST는 얻었지만 MST 구하는데 실패하였다. 따라서 Sulee DMST 알고리즘은 MWA를 오름차순 (가중치, Head 노드 순)으로 정렬시키는 방법을 채택한다.

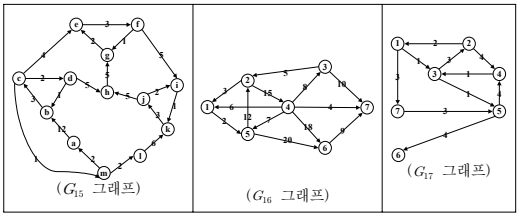
Ⅳ. Sulee MST 알고리즘 적용성 평가

1. 실험에 적용된 그래프

본 절에서는 그림 6과 같이 그래프 상으로 근 노드가 명확히 알 수 있는 11개 그래프와 근 노드를 명확히 알 수 없는 3개 그래프를 대상으로 Sulee DMST 알고리즘의 적용성을 평가해 본다. G_4와 G_5 그래프는 Chen[17]에서, G_6 그래프는 Wenger[18], G_7 그래프는 Llewellyn[15]에서, G_8 그래프는 Forbes[19]에서, $G_9, G_{10}, G_{11}, G_{12}$ 그래프는 Ikeda[16]에서, G_{13}, G_{14}, G_{17} 그래프는 List[20]에서, G_{15}, G_{16} 그래프는 Boyd[7]에서 인용되었다.



(a) 근 노드가 지정된 그래프



(b) 근 노드가 지정되지 않은 그래프

그림 6. 실험에 적용된 그래프
Fig. 6. Graphs for Experiment

2. 근 노드가 지정된 그래프 적용 결과

근 노드가 지정된 그래프는 인접행렬에서 $In-degree=0$인 열 노드가 반드시 하나는 존재하는 경우이며, 예로 G_4 그래프에서는 ⓐ가 이에 해당된다. 노드 ⓐ는 $In-degree=0$ $Out-degree=3$인 경우이다. 11개의 그래프 모두 근 노드가 지정된 경우이며, 각 열 노드의 MWA를 선택(단, 동일한 가중치를 갖는 In-degree가 다수 존재하면 DST의 조건을 만족시키지 못하므로 하나만 선택한다.)하면 사이클이 발생하지 않는 DMST를 얻을 수 있다. 즉, Chu-Liu/Edmonds와 Sulee DMST 알고리즘이 동일한 MST를 얻는다. 이렇게 얻은 DMST를 그림 7에 제시하였다.

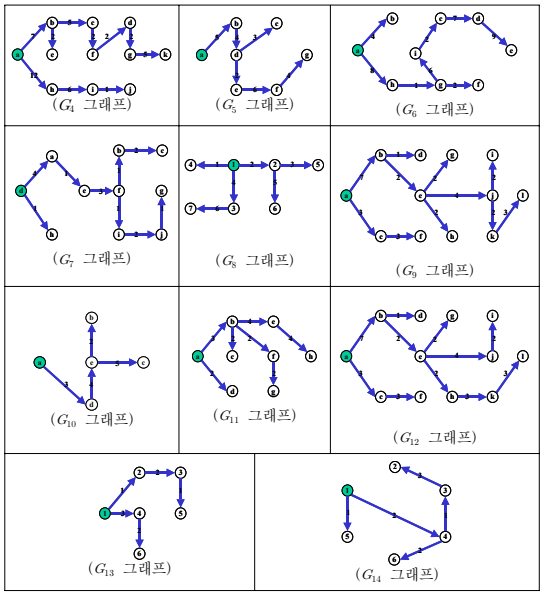


그림 7. 근 노드가 지정된 그래프의 DMST
Fig. 7. DMST of Graph with Root Node

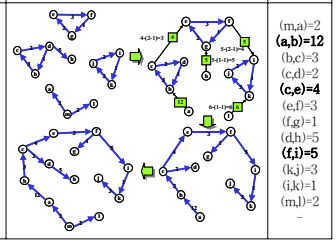
3. 근 노드가 지정되지 않은 그래프 적용 결과

근 노드가 지정되지 않은 그래프 (즉, 인접행렬에서 $In-degree=0$인 열 노드가 없을 경우)에 대해 Chu-Liu/Edmonds 알고리즘은 먼저 임의의 노드를 근으로 가정하고 알고리즘을 수행하였으며, Sulee DMST 알고리즘은 근 노드를 가정하지 않고 알고리즘을 수행하였다. 마지막으로 Sulee DMST 알고리즘으로 얻은 근에 대해 Chu-Liu/Edmonds 알고리즘을 수행한 결과를 제시하였다.

G_{15} 그래프에 대한 Chu-Liu/Edmonds와 Sulee DMST 알고리즘을 적용한 결과는 표 6에 제시하였다. G_{15} 그래프의 경우 근을 ⓜ으로 가정한 Chu-Liu/Edmonds DMST 알고리즘은 $\Sigma w(e)=43$를 얻은 반면, 근을 가정하지 않은 Sulee DMST 알고리즘은 $\Sigma w(e)=30$를 얻고 근을 ⓒ로 결정할 수 있었다. Sulee DMST 알고리즘에서 얻은 노드 ⓒ를 근으로 가정한 Chu-Liu/Edmonds DMST 알고리즘은 Sulee DMST 알고리즘과 동일하게 $\Sigma w(e)=30$을 얻을 수 있었다.

표 6. G_{15} 그래프의 DMST
Table 6. DMST of Graph G_{15}

(a) Chu-Liu/Edmonds DMST 알고리즘 (근 노드= m)

Node	노드 유입 MWA		MSF Arcs	MSF 유입 MWA	DMST Arcs
	ALL	선택			
a	(m,a)=2	(m,a)=2	(m,a)=2		(m,a)=2
b	(d,b)=1	(d,b)=1	**(d,b)=1**		**(a,b)=12**
c	(b,c)=3	(b,c)=3	(b,c)=3		(b,c)=3
d	(c,d)=2	(c,d)=2	(c,d)=2		(c,d)=2
e	(g,e)=2	(g,e)=2	**(g,e)=2**		**(c,e)=4**
f	(e,f)=3	(e,f)=3	(e,f)=3		(e,f)=3
g	(f,g)=1	(f,g)=1	(f,g)=1		(f,g)=1
h	(d,h)=5	(d,h)=5	(d,h)=5		(d,h)=5
i	(j,i)=2	(j,i)=2	**(j,i)=2**		**(f,i)=5**
j	(k,j)=3	(k,j)=3	(k,j)=3		(k,j)=3
k	(i,k)=1	(i,k)=1	(i,k)=1		(i,k)=1
l	(m,l)=2	(m,l)=2	(m,l)=2		(m,l)=2
m	-	-	-		-

(b) Sulee DMST 알고리즘

Node	노드 유입 MWA				MSF	
	ALL	선택	정렬	Cycle	Nodes	Arcs
a	(m,a)=2	(m,a)=2	(d,b)=1	x	-	(d,b)=1
b	(d,b)=1	(d,b)=1	(f,g)=1	x	b-d	(f,g)=1
c	(b,c)=3	(b,c)=3	(i,k)=1	x	b-d, f-g	(i,k)=1
d	(c,d)=2	(c,d)=2	(c,m)=1	x	b-d, f-g, i-k	(c,m)=1
e	(g,e)=2	(g,e)=2	(m,a)=2	x	b-d, f-g, i-k, c-**m**	(m,a)=2
f	(e,f)=3	(e,f)=3	(c,d)=2	x	b-**d**, f-g, i-k, a-**c**-m	(c,d)=2
g	(f,g)=1	(f,g)=1	(g,e)=2	x	a-b-c-d-m, f-**g**, i-k	(g,e)=2
h	(d,h)=5	(d,h)=5	(j,i)=2	x	a-b-c-d-m, e-f-g, **i**-k	(j,i)=2
i	(j,i)=2	(j,i)=2	(m,l)=2	x	a-b-c-d-**m**, e-f-g, i-j-k	(m,l)=2
j	(k,j)=3	(k,j)=3	(b,c)=3	O	a-**b-c**-d-l-m, e-f-g, i-j-k	-
k	(i,k)=1	(i,k)=1	(e,f)=3	O	a-b-c-d-l-m, **e-f**-g, i-j-k	-
l	(m,l)=2	(m,l)=2	(k,j)=3	O	a-b-c-d-l-m, e-f-g, i-**j-k**	-
m	(c,m)=1	(c,m)=1	(d,h)=5	x	a-b-c-**d**-l-m, e-f-g, i-j-k	(d,h)=5
					a-b-c-d-h-l-m, e-f-g, i-j-k	

MSF	MSF 유입 MWA				Candidate MWA	DMST Arcs
	유입 호	MWA				
		대상	선택	가중치 계산		
F_a {a,h,c,d,l,m}	(**m**,a)=2,(**a**,b)=12 (**d**,h)=1,(**b**,c)=3 (**c**,d)=2,(**d**,h)=5 (j,h)=5,(**m**,l)=2 (**c**,m)=1	(j,h)=5	–	–	(d,b)=1 (f,g)=1 (i,k)=1 (c,m)=1 (m,a)=2 (c,d)=2 **(g,e)=2** (j,i)=2 (m,l)=2 **(b,c)=3** (e,f)=3 (k,j)=3 (d,h)=5	(d,b)=1 (f,g)=1 (i,k)=1 (c,m)=1 (m,a)=2 (c,d)=2 **(c,e)=4** **(f,i)=5** (m,l)=2 – (e,f)=3 (k,j)=3 (d,h)=5
F_b: {e,f,g} (g,e)=2,(e,f)=3 (f,g)=1	(c,e)=4,(**g**,e)=4 (**e**,f)=3,(h,g)=5 (**f**,g)=1	(c,e)=4 (h,g)=5	(c,e)=4 (h,g)=5	(c,**e**)+(e,f)+(f,g)=4+3+1= **8** [추가]:(c,**e**)=4, 삭제:(g,**e**)=2 (h,**g**)+(g,e)+(e,f)=5+2+3=10		
F_c: {i,j,k} (j,i)=5,(k,j)=3 (i,k)=1	(f,i)=5,(**j**,i)=5 (**k**,j)=3,(**i**,k)=1 (l,k)=6	(f,i)=5 (l,k)=6	(f,i)=5 (l,k)=6	(f,**i**)+(i,k)+(k,j)=5+1+3= **9** [추가]:(f,**j**)=5, 삭제:(k,**j**)=3 (l,k)+(j,i)+(k,j)=6+2+3=11		

(c) Chu-Liu/Edmonds DMST 알고리즘 (근 노드=c)

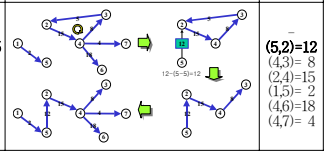
Node	노드 유입 MWA		MSF Arcs	MSF 유입 MWA	DMST Arcs
	ALL	선택			
a	(m,a)=2	(m,a)=2	(m,a)=2		(m,a)=2
b	(d,b)=1	(d,b)=1	(d,b)=1		(d,b)=1
c	–	–	–		–
d	(c,d)=2	(c,d)=2	(c,d)=2		(c,d)=2
e	(g,e)=2	(g,e)=2	**(g,e)=2**		**(c,e)=4**
f	(e,f)=3	(e,f)=3	(e,f)=3		(e,f)=3
g	(f,g)=1	(f,g)=1	(f,g)=1		(f,g)=1
h	(d,h)=5	(d,h)=5	(d,h)=5		(d,h)=5
i	(j,i)=2	(j,i)=2	**(j,i)=2**		**(f,i)=5**
j	(k,j)=3	(k,j)=3	(k,j)=3		(k,j)=3
k	(i,k)=1	(i,k)=1	(i,k)=1		(i,k)=1
l	(m,l)=2	(m,l)=2	(m,l)=2		(m,l)=2
m	(c,m)=1	(c,m)=1	(c,m)=1		(c,m)=1

G_{16} 그래프에 대한 Chu-Liu/Edmonds와 Sulee DMST 알고리즘을 적용한 결과는 표 7에 제시하였다. G_{16} 그래프의 경우 근을 ①로 가정한 Chu-Liu/Edmonds DMST 알고리즘은 $\Sigma w(e)=59$를 얻은 반면, 근을 가정하지 않은 Sulee DMST 알고리즘은 $\Sigma w(e)=40$를 얻고 근을 ④로 결정할 수 있었다. Sulee DMST 알고리즘에서 얻은 노드 ④를 근으로 가정한 Chu-Liu/Edmonds DMST 알고리즘은 Sulee DMST 알고리즘과 동일하게 $\Sigma w(e)=40$을 얻을 수 있었다.

G_{17} 그래프에 대한 Chu-Liu/Edmonds와 Sulee DMST 알고리즘을 적용한 결과는 표 8에 제시하였다. G_{17} 그래프의 경우 근을 ①로 가정한 Chu-Liu/Edmonds DMST 알고리즘은 $\Sigma w(e)=16$을 얻은 반면, 근을 가정하지 않은 Sulee DMST 알고리즘은 $\Sigma w(e)=15$를 얻고 근을 ②로 결정할 수 있었다. Sulee DMST 알고리즘에서 얻은 노드 ②를 근으로 가정한 Chu-Liu/Edmonds DMST 알고리즘은 Sulee DMST 알고리즘과 동일하게 $\Sigma w(e)=15$를 얻을 수 있었다.

표 7. G_{16} 그래프의 DMST
Table 7. DMST of Graph G_{16}

(a) Chu-Liu/Edmonds DMST 알고리즘 (근 노드=1)

Node	노드 유입 MWA		MSF Arcs	MSF 유입 MWA	DMST Arcs
	ALL	선택			
1	(2,1)=3	–	–		–
2	(3,2)= 5	(3,2)= 5	**(3,2)= 5**		**(5,2)=12**
3	(4,3)= 8	(4,3)= 8	(4,3)= 8		(4,3)= 8
4	(2,4)=15	(2,4)=15	(2,4)=15		(2,4)=15
5	(1,5)= 2	(1,5)= 2	(1,5)= 2		(1,5)= 2
6	(4,6)=18	(4,7)= 4	(4,6)=18		(4,6)=18
7	(4,7)= 4		(4,7)= 4		(4,7)= 4

(b) Sulee DMST 알고리즘

Node	노드 유입 MWA				MSF	
	ALL	선택	정렬	Cycle Check	Nodes	Arcs
1	(2,1)= 3	(2,1)= 3	(1,5)= 2	x	–	(1,5)= 2
2	(3,2)= 5	(3,2)= 5	(2,1)= 3	x	**1**–5	(2,1)= 3
3	(4,3)= 8	(4,3)= 8	(4,7)= 4	x	1–2–5	(4,7)= 4
4	(2,4)=15	(2,4)=15	(3,2)= 5	x	1–**2**–5, 4–7	(3,2)= 5
5	(1,5)= 2	(1,5)= 2	(4,3)= 8	x	1–2–**3**–5, **4**–7	(4,3)= 8
6	(4,6)=18	(4,6)=18	(2,4)=15	O	1–**2**–3–**4**–5–7	–
7	(4,7)= 4	(4,7)= 4	(4,6)=18	x	1–2–3–**4**–5–7 1–2–3–4–5–6–7	(4,6)=18

(c) Chu-Liu/Edmonds DMST 알고리즘 (근 노드=4)

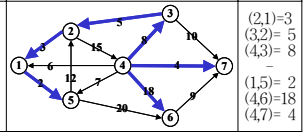
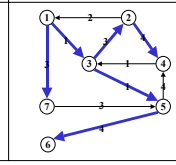
Node	노드 유입 MWA		MSF Arcs	MSF	DMST Arcs
	ALL	선택			
1	(2,1)=3	(2,1)=3	(2,1)=3		(2,1)=3
2	(3,2)= 5	(3,2)= 5	(3,2)= 5		(3,2)= 5
3	(4,3)= 8	(4,3)= 8	(4,3)= 8		(4,3)= 8
4	–	–	–		–
5	(1,5)= 2	(1,5)= 2	(1,5)= 2		(1,5)= 2
6	(4,6)=18	(4,6)=18	(4,6)=18		(4,6)=18
7	(4,7)= 4	(4,7)= 4	(4,7)= 4		(4,7)= 4

표 8. G_{17} 그래프의 DMST
Table 8. DMST of Graph G_{16}

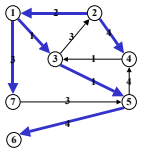
(a) Chu-Liu/Edmonds DMST 알고리즘 (근 노드=1)

Node	노드 유입 MWA		MSF Arcs	MSF	DMST Arcs
	ALL	선택			
1	–	–	–		–
2	(3,2)=3	(3,2)=3	(3,2)=3		(3,2)=3
3	(1,3)=1,(4,1)=1	(1,3)=1	(1,3)=1		(1,3)=1
4	(2,4)=4,(5,4)=4	(2,4)=4	(2,4)=4		(2,4)=4
5	(3,5)=1	(3,5)=1	(3,5)=1		(3,5)=1
6	(5,6)=4	(5,6)=4	(5,6)=4		(5,6)=4
7	(1,7)=3	(1,7)=3	(1,7)=3		(1,7)=3

(b) Sulee DMST 알고리즘

Node	노드 유입 MWA				MSF	
	ALL	선택	정렬	Cycle Check	Nodes	Arcs
1	(2,1)=2	(2,1)=2	(1,3)=1	x	–	(1,3)=1
2	(3,2)=3	(3,2)=3	(3,5)=1	x	1–**3**	(3,5)=1
3	(1,3)=1,(4,1)=1	(1,3)=1	(2,1)=2	x	**1**–3–5	(2,1)=2
4	(2,4)=4,(5,4)=4	(2,4)=4	(3,2)=3	O	1–**2**–**3**–5	–
5	(3,5)=1	(3,5)=1	(1,7)=3	x	**1**–2–3–5	(1,7)=3
6	(5,6)=4	(5,6)=4	(2,4)=4	x	1–**2**–3–5–7	(2,4)=4
7	(1,7)=3	(1,7)=3	(5,6)=4	x	1–2–3–4–**5**–7 1–2–3–4–5–6–7	(5,6)=4

(c) Chu-Liu/Edmonds DMST 알고리즘 (근 노드=2)

Node	노드 유입 MWA		MSF Arcs	MSF	DMST Arcs
	ALL	선택			
1	(2,1)=2	(2,1)=2	(2,1)=2		(2,1)=2
2	-	-	-		-
3	(1,3)=1,(4,1)=1	(1,3)=1	(1,3)=1		(1,3)=1
4	(2,4)=4,(5,4)=4	(2,4)=4	(2,4)=4		(2,4)=4
5	(3,5)=1	(3,5)=1	(3,5)=1		(3,5)=1
6	(5,6)=4	(5,6)=4	(5,6)=4		(5,6)=4
7	(1,7)=3	(1,7)=3	(1,7)=3		(1,7)=3

4. 적용 결과 분석

본 논문에 적용된 17개 그래프에 대한 알고리즘 수행 결과를 종합한 데이터는 표 9와 표 10에 제시하였다. 근 노드가 지정된 12개 그래프는 Chu-Liu/Edmonds와 Sulee DMST 알고리즘 모두 동일한 결과를 얻음을 알 수 있다. 반면에 Prim MST 알고리즘은 12개 그래프 중 7개 그래프만이 동일한 MST를 얻었으며, 5개 그래프에서는 MST를 얻지 못하였음을 알 수 있다. 근 노드가 지정되지 않은 그래프의 경우, Sulee DMST 알고리즘은 항상 MST를 얻은 반면 근 노드를 임의로 지정한 Chu-Liu/Edmonds와 Prim MST 알고리즘은 5개 그래프 모두 MST를 얻는데 실패하였다. 반면에 Chu-Liu/Edmonds을 Sulee DMST 알고리즘에서 얻은 근 노드로 설정하였을 경우 Sulee DMST 알고리즘과 동일한 MST를 얻는데 성공하였다. 그러나 Prim MST 알고리즘은 5개 그래프 중 3개만이 MST를 얻을 수 있었다. 따라서 Sulee DMST 알고리즘은 근 노드가 설정된 그래프와 설정되지 않은 그래프 모두에서 적용이 가능하며, 특히 근 노드가 설정되지 않은 그래프의 경우 MST를 얻으면서 근 노드도 함께 찾을 수 있는 장점을 갖고 있다.

표 9. 근 노드가 지정된 그래프의 MST 알고리즘 비교
Table 9. MST Algorithm Comparison of Graph with Root Node

그래프	$\Sigma w(e)$		
	Chu-Liu/Edmonds DMST 알고리즘	Sulee DMST 알고리즘	Prim MST 알고리즘
G_1	**20**	**20**	**20**
G_4	**44**	**44**	57
G_5	**24**	**24**	**24**
G_6	**39**	**39**	45
G_7	**16**	**16**	**16**
G_8	**21**	**21**	**21**
G_9	**23**	**23**	30
G_{10}	**14**	**14**	**14**
G_{11}	**19**	**19**	20
G_{12}	**31**	**31**	36
G_{13}	**9**	**9**	**9**
G_{14}	**9**	**9**	**9**

표 10. 근 노드가 지정되지 않은 그래프의 MST 알고리즘 비교
Table 10. MST Algorithm Comparison of Graph without Root Node

구분	그래프	근 노드 → $\Sigma w(e)$ 또는 $\Sigma w(e)$ → 근 노드				
		Chu-Liu/Edmonds DMST 알고리즘		Sulee DMST 알고리즘	Prim MST 알고리즘	
		근노드 임의 지정	Sulee DMST 알고리즘과 동일		근노드 임의 지정	Sulee DMST 알고리즘과 동일
근 노드 미지정 그래프	G_2	ⓐ (29)	ⓕ **(18)**	**18 (ⓕ)**	ⓐ (32)	ⓕ **(18)**
	G_3	ⓐ (실패)	ⓕ **(18)**	**18 (ⓕ)**	ⓐ (실패)	ⓕ (20)
	G_{15}	ⓜ (43)	ⓒ **(30)**	**30 (ⓒ)**	ⓜ (47)	ⓒ **(30)**
	G_{16}	① (59)	④ **(40)**	**40 (④)**	① (59)	④ (43)
	G_{17}	① (16)	② **(15)**	**15 (②)**	① (16)	② **(15)**

V. 결론 및 향후 연구과제

본 논문에서는 방향그래프의 최소신장트리를 얻는 알고리즘을 제안하였다. 먼저 무방향 그래프에서 일반적으로 적용되고 있는 Prim MST 알고리즘을 방향 그래프에 적용하기 위해 수정된 알고리즘을 제안하였다. 다음으로, 근 노드가 지정된 1개 그래프와 근 노드가 지정되지 않은 2개 그래프에 대해Prim DMST 알고리즘과 Chu-Liu/Edmonds DMST 알고리즘을 적용하여 문제점을 고찰하였다. 이러한 문제점을 해결하기 위해 Chu-Liu/Edmonds DMST 알고리즘을 수정한 Sulee DMST 알고리즘을 제안하였다. Sulee DMST 알고리즘은 근 노드의 유입 호를 제거하지 않으며, 모든 노드에서 최소 가중치를 갖는 유입 호들을 선택하고 이들을 오름차순으로 정렬시킨 후 사이클이 발생하는 호를 제거하는 알고리즘을 추가시켰다. 또한, Chu-Liu/Edmonds DMST 알고리즘은 사이클을 제거하기 위해 사이클 발생 호들에 유입되는 호들 중 최소 가중치를 갖는 호를 선택하는 과정에서 유입 호들의 가중치를 수정하는 과정을 거친다. 반면에 Sulee DMST 알고리즘은 MSF들을 연결하는 최소 가중치를 가진 유입 호를 선택하는 과정에서 유입 호들의 가중치를 수정하지 않고 직접 계산하는 방법을 채택하였다. 이와 같이 제안된 Sulee DMST 알고리즘은 근 노드가 지정된 그래프의 경우, Chu-Liu/Edmonds DMST 알고리즘과 동일한 DMST를 얻을 수 있으며, 근 노드가 지정되지 않은 그래프에서는 항상 DMST를 얻음과 동시에 근 노드를 결정할 수 있는 장점을 갖고 있다.

본 논문에서 제안된 Sulee DMST 알고리즘은 방향 그래프의 근 노드로부터 모든 노드로 유일한 방향 경로를

가진 DMST를 구할 수 있었다. 이를 기반으로 하여 추후 임의의 노드로부터 시작하여 종점 노드로의 최단거리 경로 (Shortest Path)를 찾는 알고리즘으로 적용할 수 있는지를 연구하고자 한다.

참고문헌

[1] Wikipedia, http://en.wikipedia.org/wiki/Minimum_spanning_tree, Wikimedia Foundation Inc., 2007.

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

[3] 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, 2001.

[4] R. C. Prim, "Shortest Connection Networks and Some Generalisations," Bell System Technical Journal, Vol. 36, pp. 1389-1401, 1957.

[5] 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.

[6] P. A. Jensen, "Operations Research Models and Methods," John Wiley and Sons, 2003.

[7] S. Boyd, "Applications of Combinational Optimization: Optimal Paths and Trees," http://www.site.uottawa.ca/~sylvia/csi5166web/5166tespg26to61.pdf, School of Information Technology and Engineering (SITE), University of Ottawa, Canada, 2005.

[8] C. Duhamel, L. Gouveia, P. Moura, and M. C. Souza, "Models and Heuristics for a Minimum Arborescence Problem," Research Report LIMOS/RR-04-13, http://www.isima.fr/limos/publi/RR-04-13.pdf, 2004.

[9] S. J. Yang, "The Directed Minimum Spanning Tree Problem," http://www.ce.rit.edu/~sjyeec/dmst.html, 2000.

[10] A. Schrijver, "Advanced Graph Theory and Combinatorial Optimization," Dept. of Mathematics, University of Amsterdam, Netherlands, http://citeseer.ist.psu.edu/schrijver01advanced.html, 2001.

[11] N. Bezroukov, "Minimum Spanning Trees," Softpanorama, http://www.softpanorama.org/Algorithms/Digraphs/mst.shtml, 2006.

[12] Wikipedia, "Tree(Graph Theory)," http://en.wikipedia.org/wiki/Tree_graph_Theory/, 2007.

[13] R. Krishnam, B and Raghavachari, "The Directed Minimum-Degree Spanning Tree Problem," FSTTCS 2001, LNCS 2245, pp. 232-243, 2001.

[14] T. UNO, "An Algorithm for Enumerating all Directed Spanning Trees in a Directed Graph," International Symposium on Algorithm and Computation(ISAAC96), pp. 166-173, 1996.

[15] M. Llewellyn, "COP 3503: Computer Science II - Introduction to Graphs," http://www.cs.ucf.edu/courses/cop3503/summer04, 2004.

[16] K. Ikeda, "Mathematical Programming," Dept. Information Science and Intelligent Systems, The University of Tokushima, http://www-b2.is.tokushima-u.ac.jp/~ikeda/suuri/maxflow/Maxflow App.shtml. 2005.

[17] WWL. Chen, "Discrete Mathematics," Department of Mathematics, Division of ICS, Macquarie University, Australia, http://www.maths.mq.edu.au/~wchen/lndmfolder/lndm.html, 2003.

[18] R. Wenger, "CIS 780: Analysis of Algorithms," http://www.cse.ohio-state.edu/~wenger/cis780/shortest _path.pdf, 2004.

[19] J. R. N. Forbes, "CPS196.2 Robotics: Graphs and Matrices," http://www.cs.duke.edu/courses/cps196.2/fall03/pdf/graphs_and_ matrices.pdf

[20] C. List, "MT303: Operations Research Graphs & Networks - Handout 3," Department of Mathematics, National University of Ireland, http://www,maths.may.ie/clist/teching/netw_

handout_3. pdf, 2003.
[21] 이용진, 이동우, “WSN의 최소비용 신장트리 문제와 휴리스틱 알고리즘,” 한국정보기술학회논문지, 제7권, 제4호, pp. 275-282, 2009.
[22] 박양재, 이정현, "무선 이동 에드 혹 네트워크에서 효율적인 코어- 기반 멀티캐스트 트리," 한국정보기술학회논문지, 제1권, 제1호, pp. 32-44, 2003.
[23] 임은기, "자료흐름도로부터 데이터베이스 테이블 설계순서를 추출하는 알고리즘," 한국정보기술학회논문지, 제5권, 제4호, pp. 226-233, 2007.

저자 소개

최 명 복(종신회원)

- 1992년 : 호서대학교 전자계산학과
- 1994년 : 아주대학교 컴퓨터공학과
- 2001년 : 아주대학교 컴퓨터공학과
- 1997~현재 : 강릉원주대학교 멀티미디어공학과 교수
- 2004. 1~현재 : 한국인터넷방송통신학회 이사
- <주관심분야 : 지능형 정보검색, 소프트웨어 공학, 알고리즘>
- e-mail : cmb5859@gmail.com

이 상 운(정회원)

- 1983년~1987년 : 한국항공대학교 항공전자공학과 (학사)
- 1995년 ~ 1997년 : 경상대학교 컴퓨터과학과 (석사)
- 1998년 ~ 2001년 : 경상대학교 컴퓨터과학과 (박사)
- 2004년 ~ 2007.2 : 국립 원주대학 여성교양과 조교수
- 2007.3 ~ 현재 : 강릉원주대학교 과학기술대학 멀티미디어공학과 부교수
- <주관심분야 : 소프트웨어 시험 및 품질보증, 소프트웨어 신뢰성, 신경망, 뉴로-퍼지, 그래프 알고리즘>
- e-mail : sulee@gwnu.ac.kr