An Enhanced Heuristic Using Direct Steiner Point Locating and Distance Preferring MST Building Strategy for GOSST Problem

†,††*Inbum Kim*, †*Chae-kak Kim*

†Dep.of Internet Information, Kimpo College, Gimpo-si, Kyonggi-do, South Korea
††Dep.of Electrical Engineering and Computer Science, University of Wisconsin-Milwaukee, Milwaukee, Wisconsin, USA

Summary

An enhanced heuristic for the GOSST (Grade of Services Steiner Minimum Tree) problem is proposed in this paper. GOSST problem is to look for a network topology satisfying the *G-condition* with minimum construction cost of the network. In previous research, we proposed a heuristic for the problem, which employed two Steiner point locating strategies with Naïve MST (Minimum Spanning Tree) building strategy. The GOSST heuristic of this paper employs new Steiner point locating strategy and MST building strategy, which are Direct Locating strategy and Distance Preferring MST building strategy. Based on the results of this research, we can assert the Direct Steiner point Location strategy is better than Global or Local location strategy of our previous research, and the Distance Preferring MST building strategy is better than previous Naïve MST building strategy. And the Distance Direct GOSST method employing the Distance Preferring MST building strategy and the Direct Steiner point Locating strategy entails the least network construction cost and acquires 13.2% enhancement by comparison to the Naïve Global method, the best GOSST method in our previous research [19].

Key words:

GOSST, Steiner Point, Minimum Spanning Tree, Network Construction Cost, Locating Strategy, MST building Strategy, Direct locating, Global Locating, Local Locating, Distance Preferring, G-condition, G-MST

1. Introduction

There is a vital subject for the physical construction of a network structure in the network design, which is to save the building costs with sufficient transmission capabilities. The solutions always answer paths providing enough communication capacities between any two sites of the network, expending less network construction costs. A solution for the GOSST (Grade of Services Steiner Minimum Tree) problem might be good answer such as problems. GOSST problem also be applied to transportation, for road constructions or to some more potential uses of CAD in terms of interconnecting the elements on a plane such that to provide enough flux between any two elements.

The researches on GOSST have focused on the optimization. In Optimization on GOSST, Ding-Zhu, Du and his colleagues have the authority [1,5,9,15]. They have solved the difficult problems related to Steiner Tree problem, such as Gilbert-Pollak conjecture on the Steiner ratio [15], and then deduced many problems related to Steiner Tree problem. One of them is GOSST problem. As this problem might be regarded to reach the theoretical limitation of the optimization solution, many people seem not to try the theoretical research any more.

Most of optimization algorithms for GOSST have constructed to prove that the GOSST problem is belonged to PTAS (Polynomial Time Approximation Scheme) problem [5, 6]. In NP-hard problems, PTAS problem is defined that though spending very large polynomial time, the problem could reach the (1+e) approximation solution. When the problem is asserted as PTAS problem, the problem classification, the main issue of theory of computing could be accomplished.

Though the running time is polynomial time, in fact, the size might be exponential time because exponential function could be approached by polynomial time. The purpose of drawing the approximation algorithm is to show the given problem has the constructively polynomial time algorithm in theoretical view, not to apply to practical area. In PTAS problem, if the running time of approximation algorithm decreases to practical level, the approximation ratio could be worse drastically. In most case, the approximation ratio becomes meaningless.

In Heuristic on GOSST, as people might consider this problem as theoretical problem related to Steiner Tree problem and not expand their sight to the true worth anymore, they seem not to do their researches on practical applications. But as the practical solutions of this problem could be applied to the network or circuit design area, the

Manuscript received
Manuscript revised

research to implement a heuristic with more feasible time complexity is necessary.

More improved heuristic than previous heuristic for GOSST problems is presented in this paper. With the results, the possibility of applying the heuristic to practical and useful applications could be augmented in real world.

2. Problem Definition

The GOSST problem asks for a minimum cost network interconnecting terminal points in set P and some Steiner Points. The definition of GOSST problem [1] is as follows: Let $\mathbf{P}=\{p_1, p_2, \ldots, p_n\}$ be a set of n terminal points in the Euclidean plane, where each point p_i has a service request of grade as $grade(p_i) \in \{1, 2, \ldots, r\}$. Let $0 < c(1) < c(2) < \ldots < c(r)$ be r real numbers, where $c(\boldsymbol{i})$ is the cost for providing service $\boldsymbol{i}$. Each edge in the network might be assigned a specific grade of service, which is a number in $\{1, 2, \ldots, r\}$. Let $grade(e)$ denote the service grade of edge e. Between each pair of terminal points p_i and p_j in a network, there is a path whose minimum grade of service is at least as large as $\min(grade(p_i), grade(p_j))$ and the construction cost of the network is minimum. The construction cost of an edge with service of grade $\boldsymbol{g}$ in the network is the product of the Euclidean length of the edge by $c(\boldsymbol{g})$. The GOSST problem is a generalization of the ESMT problem where all terminal points have the same service request of grade.

3. Background

For implement a GOSST heuristic, the backgrounds on Minimum Spanning Tree, Steiner Minimum Tree and Grade Of Services Steiner minimum Tree are necessary.

3.1 Minimum Spanning Tree

For each edge in an edge set **E**, weight $\boldsymbol{w}$ between two nodes is specified by the cost for communicating with each other. There is a case to find acyclic subset that connects all of the vertices and whose total weight sum is minimized. Since the subset is acyclic and connects all vertices, it forms a tree. The tree grows until the tree spans all vertices and therefore, it is called as a spanning tree. The problem of determining such a tree is called as a minimum spanning tree problem. There are two dominant algorithms for solving minimum spanning tree problem; Kruskal's algorithm and Prim's algorithm. In this research, the Prim's algorithm is adopted to build a Minimum Spanning Tree for the heuristic of GOSST problem.

For interconnecting of a set of n nodes, a selection of n-1 edges can be used; the one using the least amount of edges is usually craved for. With **V**, a set of nodes and **E**, a set of possible interconnections between pairs of nodes, a connected, undirected graph **G** can be described as $\mathbf{G}=(\mathbf{V}, \mathbf{E})$. For each edge $(u, v) \in \mathbf{E}$, weight $\boldsymbol{w}(u, v)$ is defined by the cost for connecting u and v. MST(Minimum Spanning Tree) problem is to find acyclic subset $\mathbf{T} \subseteq \mathbf{E}$ that connects all of the vertices with minimum total weights. As **T** is an acyclic tree and connects all vertices, it is called a spanning tree. A minimum spanning tree problem is to look for the tree like **T**.

There are two prominent algorithms for a minimum spanning tree; Kruskal's algorithm and Prim's algorithm. In Kruskal's algorithm, the set **A** is a forest and an edge added to **A** should be always a minimum weight edge. The tree connects to a vertex not belong to the tree at that time and only one edge is added at a time in each iteration for growing the minimum spanning tree. A set of edges **A** is a subset of final minimum spanning tree during the iterations.

In Prim's algorithm, the edges in the set **A** always form a single tree. The tree starts from an arbitrary root vertex $\boldsymbol{r}$ called first vertex and grows until the tree spans all the vertices in **V**. At each step, a least weighted edge is added to the tree **A** for connecting **A** to an isolated vertex.

3.2 Steiner Minimum Tree

There is a problem to find the point **P** that minimizes the sum of the distance from **P** to each of three given points in the plane or to find the point **P** in a triangle so that the total distance from **P** to each of the triangle's vertices is minimized. This problem can be expended even further by allowing the addition of an arbitrarily number of points to find the shortest network connecting all points. Adding each point called Steiner Point and producing a tree to create the minimal network is the Minimum Steiner Tree problem. In this research, this Steiner point and Minimum Steiner Tree are employed to implement the heuristic for the GOSST problems.

Minimizing a network's length has been one of the important optimization problems. One of the problems is to find the point **P** that minimizes the sum of the distance from **P** to each of three given points in the plane or to find the point **P** in a triangle so that the total distance from **P** to the triangle's vertices is minimized. This problem expanded it to include an arbitrarily large set of points in the plane. As this involved only one point, forming a star-like shape, it is called Steiner star, when **P** was joined to each other of the points. This problem was expended even further by allowing the addition of an arbitrarily number of points to find the shortest network connecting all points. By adding the points called Steiner Point, the minimal network or a tree as the final result could be yielded

instead of Steiner star.

This problem is NP-Hard or NP-Complete when discrete points are used. Therefore, the problem cannot be solved in polynomial time, so appreciate heuristic for Steiner Tree is required [19].

3.3 Grade Of Services Steiner minimum Tree

GOSST problem is a variation of the ESMT (Euclidean Steiner Minimum Tree) which is a problem to find a minimum cost network interconnecting a set of given points in the Euclidean planes. The works for ESMT problem could be found in [11,12,13,14,15,16,17]. The GOSST problem is known to be NP-Complete. Therefore, to solve even small-scale problems needs tremendous computations and memory spaces. The previous many researches on GOSST have interested in geometric analysis and improvements of approximation algorithms. But the heuristics for GOSST have not been published lively.

The GOSST (Grade Of Services Steiner minimum Tree) problem asks for a network interconnecting the point set P and some Steiner points. Between each pair of terminal point p_i and p_j in GOSST, there is a path whose minimum grade of service is at least as large as minimum value between *grade*(p_i) and *grade*(p_j), which is called as *G-condition* [19]. And the construction cost of the network is minimum among all interconnecting networks satisfying the *G-condition*, where the cost of an edge with service of grade *u* is the product of the Euclidean edge length by the expenditure for service grade *u*.

In Fig 1, there are three terminal nodes **A**, **B** and **C** having its own processing capacities considering as service grades. Though node **A**'s processing capacity is 2, the edge capacity from node **A** to node **B** could not be more than 1 as node **B**'s processing capacity is 1. The capacity of path from **B** to **C** or from **A** to **C** via **B** could not be more than 1, either. If the edge capacity is regarded as bandwidth, the bandwidth waste of this network is inevitable.

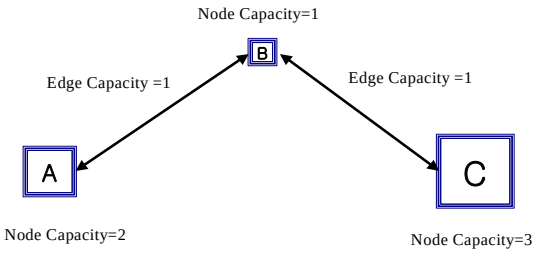


Fig 1 Three terminal nodes A, B, and C having their own processing capacities want to communicate with other nodes.

To avoid this dissipation or to satisfy *G-condition* described above, this research proposes that new created Steiner Point connects to terminal node **A**, **B** and **C** as Fig 2. In this technique, the possible capacity of path from **A** to **C** increases to 2, though the sum of edge length might be increased. The determination of the Steiner point position is a key to reduce the sum of edge length. In this research, the Steiner point positions are also considered. To minimize the network construction cost and to satisfy the *G-condition* for all paths of pairs of terminal node concurrently is a goal of GOSST problem.

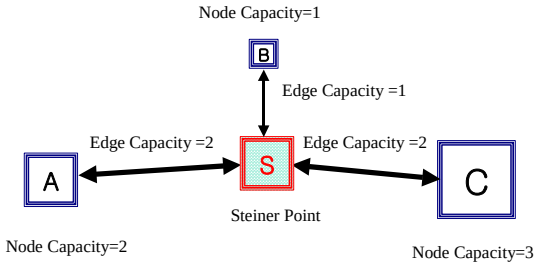


Fig 2 A Steiner point is created and makes connects to terminal nodes A, B and C to satisfy the *G-condition*.

4. Previous Heuristic Methods for GOSST

Previous proposed heuristic could be classified according to applied connecting strategies. Two connecting strategies are global connecting and local connecting [19].

4.1 Naïve MST Building Strategies

Naïve building method for MST (Minimum Spanning Tree) is as follows;

Step 1 Calculate the weight of each connection as its Length×Capacity.
Step 2 Select a node, not first node, to which first node connects and the connection's weight is least, and making the connection as an edge of a final tree.
Step 3 Select a not touched node, to which a touched node connects and the connection's weight is least. Make the connection as an edge of a final tree.
Step 4 Repeat step 3 until all terminal nodes are connected by selected edges. A Naive Minimum Spanning Tree is constructed by collecting the selected edges.

4.2 Global and Local Steiner Points Locating Strategies

Fig 3 shows the global locating strategy. A candidate

Steiner point is created, if a path from *start* node **S** to *end* node **E** violates *G-condition*. And it connects to *start* node **S**, *end* node **E** and *g-node*, where *g-node* is a selected node on the path from *start* node **S** and *end* node **E**. To guarantee loop-free, no necessary existing edges are removed.

In fig 4, the local locating strategy could be found. A *G-condition* violation of sub path between *fparent* node **A** and *bparent* node **Z** on the path between *start* node **S** and *end* node **E** is investigated. If a violation is found, a candidate *Steiner point* is created and connects to *fparent* **A**, *bparent* **Z** and *g-node*, where *g-node* is a selected node being on the section between *fchild*, the next node of *fparent* and *bchild*, the previous node of *bparent*. For loop-free, unnecessary edges are swept away.

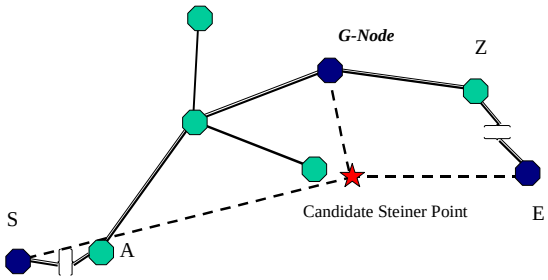
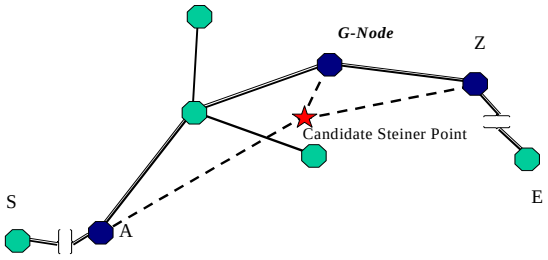


Fig 3 Global Locating Strategy



.

Fig 4 Local Locating Strategy

4.3 Naïve Global GOSST Method

Step 1: Build a first **Naïve Minimum Spanning Tree** for given terminal nodes and edges with their distances and capacities. Step 2: Check the ***g-condition*** for every pair of terminal nodes of give network.

Step 3: If ***g-condition*** is violated on a pair of ***start*** and ***end*** terminal node, created **Steiner points** are located by **Global Locating** method.

Step 4: Rebuild new **Naïve Minimum Spanning Tree** with changed nodes and edges.

Step 5: Repeat from step 2 to 4 until all paths of given network satisfy ***g-condition***.

Fig 5 presents the result of **Naïve Global GOSST** method described above with 100 terminal nodes, 3 max connections per node and 3 service kinds.

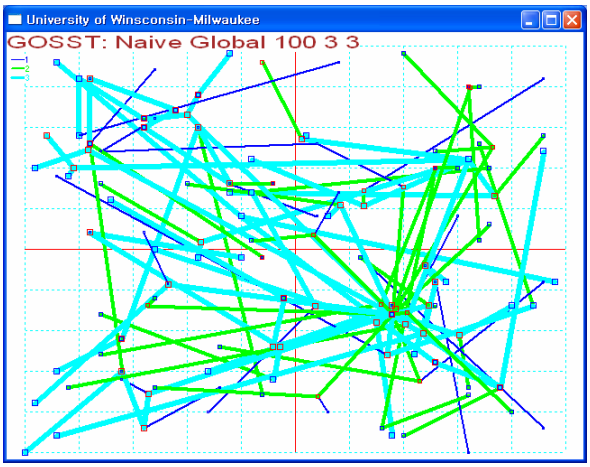


Fig 5 The result of Naïve Global GOSST method with 100 terminal nodes

4.4 Naïve Local GOSST Method

Step 1: Build a first **Naïve Minimum Spanning Tree** for given terminal nodes and edges with their distances and capacities. Step 2: Check the ***g-condition*** for every pair of terminal nodes of give network.

Step 3: If ***g-condition*** is violated on a pair of ***start*** and ***end*** terminal node, created **Steiner points** are located by **Local locating** method.

Step 4: Rebuild new **Naïve Minimum Spanning Tree** with changed nodes and edges.

Step 5: Repeat from step 2 to 4 until all paths of given network satisfy ***g-condition***.

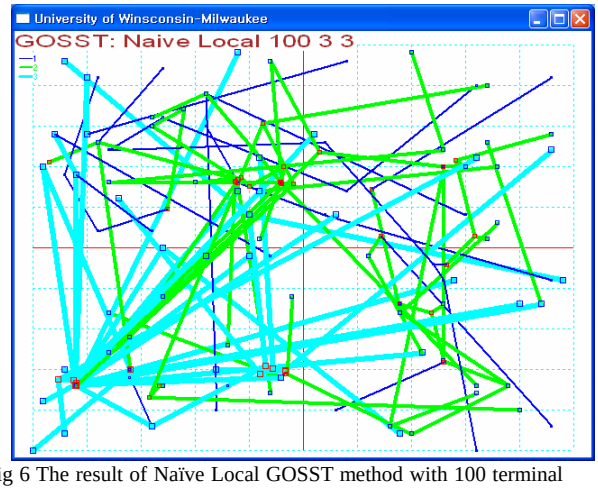


Fig 6 The result of Naïve Local GOSST method with 100 terminal nodes

Fig 6 presents the result of **Naïve Local GOSST** method described above with 100 terminal nodes, 3 max connections per node and 3 service kinds.

5. Enhanced Heuristic Methods for GOSST

5.1 Direct Steiner Points Locating Strategies

Step 1 Find a *G-condition* violation sub path between *fparent* node and *bparent* node on the path between *start* and *end*.

Step 2 Create a candidate Steiner point with 3 sequence nodes on the path between *fparent* and *bparent*.

Step 3-A If *Steiner Cost* is less than or equal to *Spanning Cost*, the candidate becomes a Steiner Point and appropriate connections and disconnections are conducted for loop- free

Step 3-B Else *Steiner Cost* is more than *Spanning Cost*, the candidate is discarded and the Steiner points are created on the nodes locating the path from *fchild* to *bchild* and appropriate connections and disconnections are conducted for loop- free.

Step 4 Repeat from step 1 to step 3 until there is no *G-condition* violation sub path between *start* and *end*.

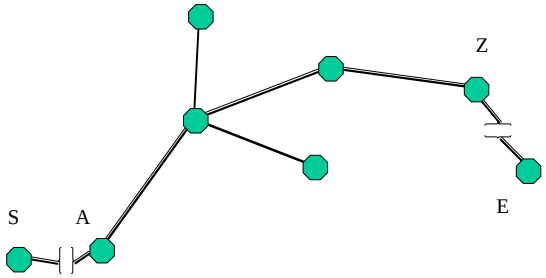


Fig 7 An example sub network for Direct locating Method. There is a violation of g-condition on a intermediate sub path starting A to Z

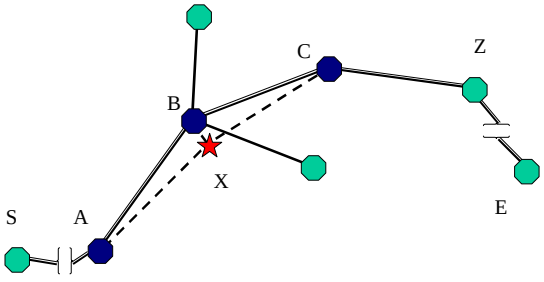


Fig 8 Finding a candidate Steiner Point X with 3 sequence node A, B and C

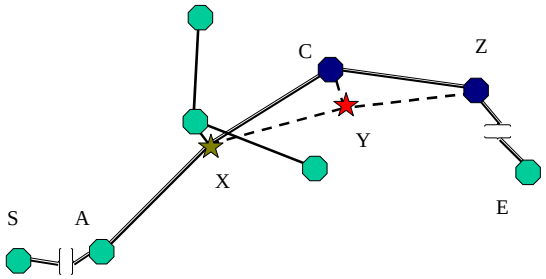


Fig 9 Finding next candidate Steiner point Y with 3 sequence node X, C and Z on the path between A and Z. If Y satisfies the Steiner Point qualification, Y becomes a Steiner Point.

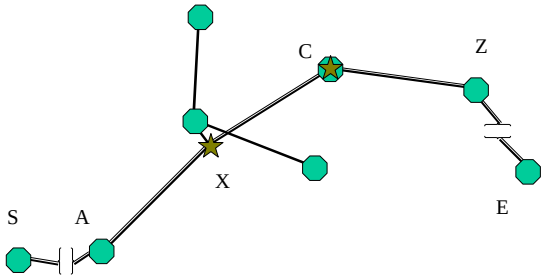


Fig 10 If Y does not satisfy the qualification, the candidate Y is discarded and new Steiner point is created on node C.

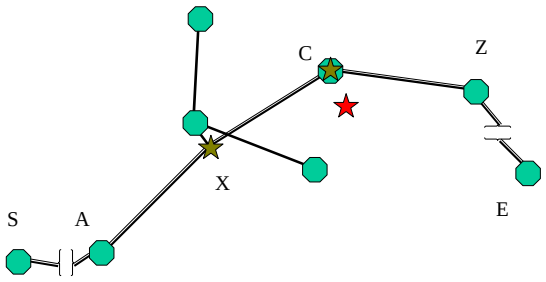


Fig 11 The result of Direct location strategy on the path between S and E. There is node any violation of g-condition on the path between S and E

Fig 7 shows an example sub network for Direct locating Method. There is a violation of G-condition on a sub_ path starting A to Z between S and E.

Fig 8 displays the process of the looking for a candidate Steiner Point for 3 sequence nodes A, B and C on the path between A and Z. If *Steiner Cost* is less than or equal to *Spanning Cost*, the candidate becomes a Steiner Point and appropriate connections and disconnections are conducted as shown in Fig 9. While if *Steiner Cost* is more than

Spanning Cost, the candidate is discarded and a new Steiner point is created on the node C. The capacities of the Steiner points are the same value of the target path capacity between S and E as shown Fig 10. This processes repeat until there is no G-condition violation between S and E as shown Fig 11.

5.2 Distance Preferring building method for Minimum Spanning Tree

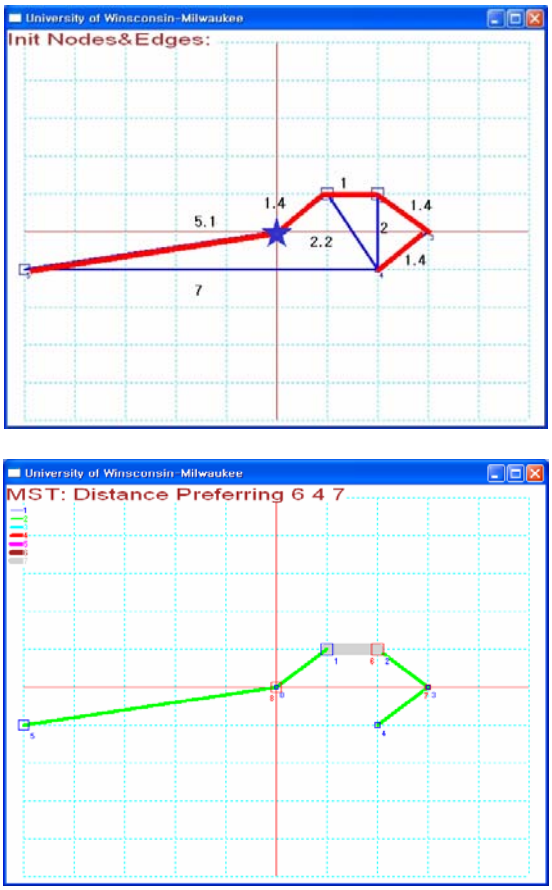


Fig 12 Input network (top) and the result of Distance Preferring Minimum Spanning Tree building strategy started by point 0 as first node (bottom).

Step 1 Select a node, not first node, to which first node connects and the connection's length is shortest, and make the connection as an edge of a final tree.

Step 2 Select a not touched node, to which touched node connects and the connection's length is shortest. Make the connection as an edge of a final tree.

Step 3 If there are one more connections of same shortest length, select the connection of most capacity as an edge of a final tree.

Step 4 Repeat step 2 and step 3 until all terminal nodes are connected by selected edges. A Distance Preferring Minimum Spanning Tree is constructed by collecting the selected edges.

Fig 12 shows the Distance Preferring MST built by these steps with given network.

5.3 Naïve Direct GOSST

Step 1: Build a first **Naïve Minimum Spanning Tree** for given terminal nodes and edges with their distances and capacities. Step 2: Check the ***g-condition*** for every pair of terminal nodes of give network.

Step 3: If ***g-condition*** is violated on a pair of ***start*** and ***end*** terminal node, created **Steiner points** are located by **Direct locating** method.

Step 4: Rebuild new **Naïve Minimum Spanning Tree** with changed nodes and edges.

Step 5: Repeat from step 2 to 4 until all paths of given network satisfy ***g-condition***.

Fig 13 presents the result of **Naïve Direct GOSST** method described above with 100 terminal nodes, 3 max connections per node and 3 service kinds. The method employs Naïve MST building strategy and Direct locating strategy for created Steiner points.

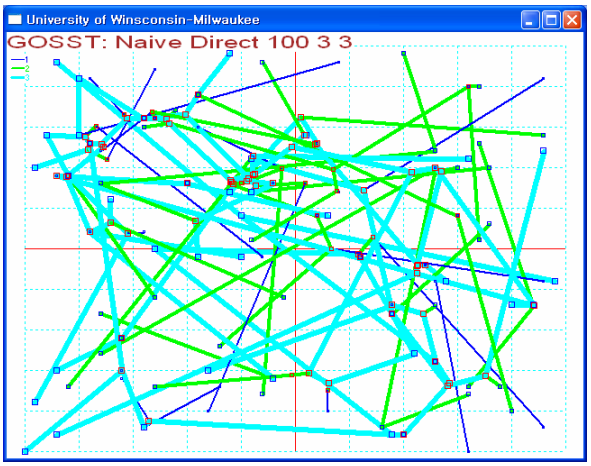


Fig 13 The result of Naïve Direct GOSST method with 100 terminal nodes

5.4 Distance Direct GOSST

Step 1: Build a first **Distance Preference Minimum Spanning Tree** for given terminal nodes and edges with their distances and capacities.Step 2: Check the ***g-condition*** for every pair of terminal nodes of give network.

Step 3: If ***g-condition*** is violated on the pair of ***start*** and ***end*** terminal node, created **Steiner**

points are located by **Direct locating** method.

Step 4: Rebuild new **Distance Preference Minimum Spanning Tree** with changed nodes and edges.

Step 5: Repeat from step 2 to 4 until all paths of given network satisfy ***g-condition***.

Fig 14 presents the result of **Distance Direct GOSST** method described above with 100 terminal nodes, 3 max connections per node and 3 service kinds. The method employs Distance Preferring MST building strategy and Direct locating strategy for created Steiner points.

5.5 Distance Global GOSST Method

Step 1: Build a first **Distance Preference Minimum Spanning Tree** for given terminal nodes and edges with their distances and capacities. Step 2: Check the ***g-condition*** for every pair of terminal nodes of give network.

Step 3: If ***g-condition*** is violated on the pair of *start* and *end* terminal node, created **Steiner points** are located by **Global locating** method.

Step 4: Rebuild new **Distance Preference Minimum Spanning Tree** with changed nodes and edges.

Step 5: Repeat from step 2 to 4 until all paths of given network satisfy ***g-condition***.

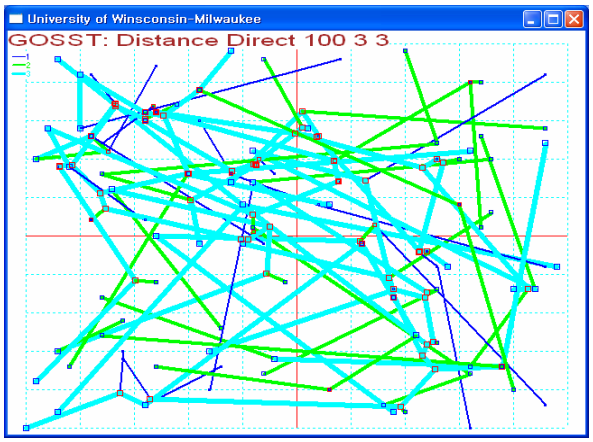


Fig 14 The result of Distance Direct GOSST method with 100 terminal nodes

Fig 15 presents the result of **Distance Global GOSST** method described above with 100 terminal nodes, 3 max connections per node and 3 service kinds. The method employs Distance Preferring MST building strategy and Global locating strategy for created Steiner points.

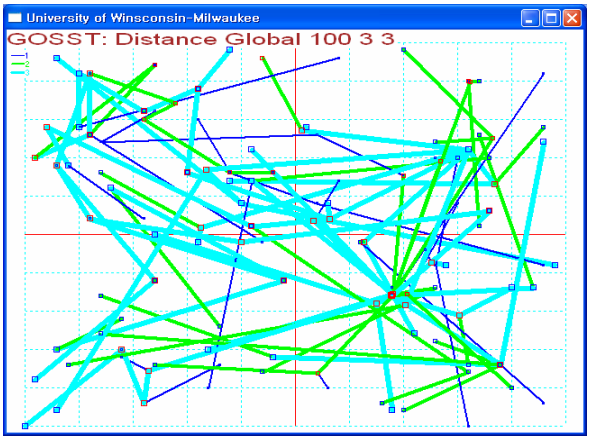


Fig 15 The result of Distance Global GOSST method with 100 terminal nodes

5.6 Distance Local GOSST Method

Step 1: Build a first **Distance Preference Spanning Tree** for given terminal nodes and edges with their distances and capacities. Step 2: Check the ***g-condition*** for every pair of terminal nodes of give network.

Step 3: If ***g-condition*** is violated on a pair of ***start*** and ***end*** terminal node, created **Steiner points** are located by **Local locating** method.

Step 4: Rebuild new **Distance Preference Minimum Spanning Tree** with changed nodes and edges.

Step 5: Repeat from step 2 to 4 until all paths of given network satisfy g-***condition***.

Fig 16 presents the result of **Distance Local GOSST** method described above with 100 terminal nodes, 3 max connections per node and 3 service kinds. The method employs Distance Preferring MST building strategy and Local locating strategy for created Steiner points

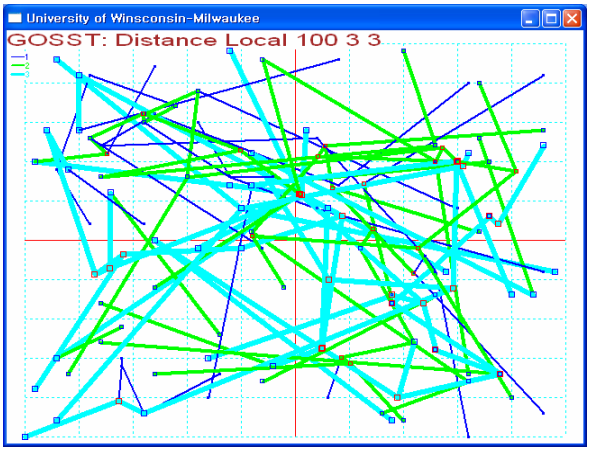


Fig 16 The result of Distance Local GOSST method with 100 terminal nodes

6. Experiments and Analysis

The original problem does not provide edge information; that is to find a GOSST with only given set of terminal nodes having own capacity. But in this research, to generalize the original GOSST problem, an edge set is also provided as input.

Table 1 represents the main module for finding the GOSST with input. The input file has the given network information, that is, the number of node, max connections per node, the number of capacity kinds, nod information and edge information. The node information consists of its name, 2-dimensional *x*, *y* position and capacity. The module for producing a MST is based on Prim's algorithm.

The functions are required to create new Steiner points and their new connections to the related partners. New Steiner points might be created whenever the *G-condition* is not satisfied. Removing process of an unnecessary established connections for loop-free and reconstructing new MST module are carried out after Steiner Point and new connections are created.

```
MAIN(){
    readNodeEdgeData();
    buildingMinSpanningTree(startNode);
    …
    WHILE ( violateGonditionOnAllPairNode()){
        …
        buildingMinSpanningTree(startNode);
        …
    } END-WHILE
    …
    produceMSTInform();
    produceGOSSTInform();
}
```

Table 1 Main module of proposed heuristic for GOSST

The factors of these experiments are as follows;

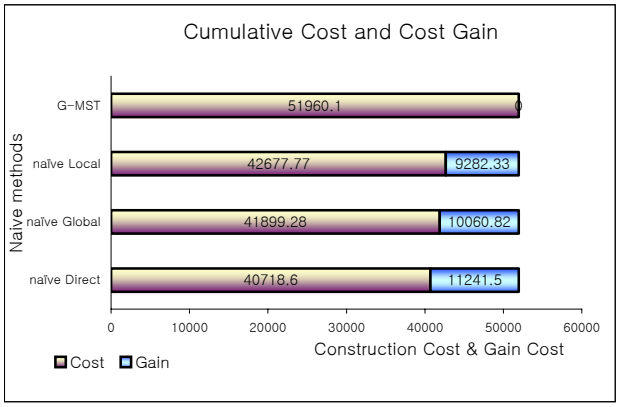
- Node Number = (25, 50, 75, 100)
  - Terminal nodes are generated randomly within this constraint...
- Max Connections per Node = (3, 5, 7, 9)
  - Edges (connections) are generated randomly within this constraint.
- Capacity Kinds = (3, 5, 7, 9)
  - A node's capacity is assigned randomly within this constraint.
- Applied Methods= 6 Methods;
  - Naïve Global GOSST
  - Distance Global GOSST
  - Naïve Local GOSST
  - Distance Local GOSST
  - Naïve Direct GOSST
  - Distance Direct GOSST

The experiments execute 384 (4×4×4×6) cases, for each of 4 kinds terminal node number (25, 50, 75, 100), each of 4 kinds max connections per node (3, 5, 7, 9), each of 4 kinds capacities (3, 5, 7, 9) and each of 6 methods. The each method results of the experiments are yielded by sum of network construction cost and created Steiner point number and execution times of 64 (4×4×4) experiments results of the method. The each method cost saving ratios are achieved by the average values of 64 experiments of the method.

The control of this research is G-MST (Grade of service Minimum Spanning Tree). G-MST can be built by adjusting the capacities of the nodes of a MST for satisfying the G-conditions.

6.1 Experiments for Proposed Direct Locating Strategy

Fig 17 shows the results of cumulative network constructing cost for 3 locating strategies applied Naïve or MST building strategy in order to check the improvement to the Direct Locating Strategy proposed in this paper.



Method (A)	Cost (B)	Gain (C-A)	Improvement (T-B)/B
Naïve Direct (a)	40718.6 (T)	11241.5	0.0%
Naïve Global (b)	41899.3	10060.8	-2.8%
Naïve Local (c)	42677.8	9282.3	-4.6%
Average improvement			-3.7%
G-MST	51960.1 (C)	0	-21.6%

Fig 17 The improvement of network construction cost saving by Direct locating Strategy. Network construction cost sums, gain sums, and cost saving ratios to G-MST for 64 experiments of each method employing Naïve MST building strategy.

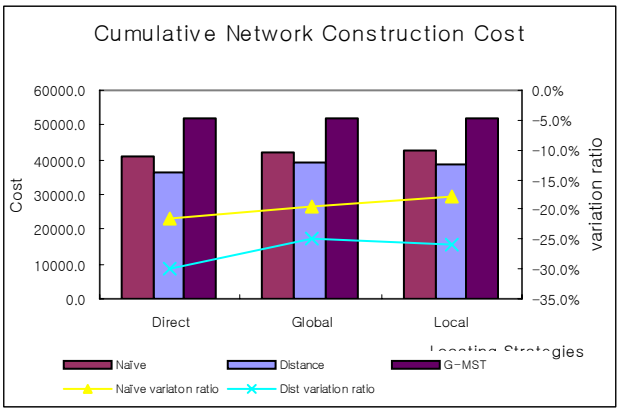
From the results of Fig 17, we find the facts as follows;

- New locating strategy, Naïve Direct locating strategy requires least network construction cost and shows most cost gain.
- While Local locating strategy requires most network construction cost and reveals least cost gain.
- The costs are saved by average 3.7% when employing Direct Locating strategy instead of previous proposed 2 locating strategies.

6.2 Experiments for Proposed Distance Preferring MST Building Strategy

Fig 18 shows the results of cumulative network constructing cost for 3 locating strategies about Naïve or Distance Preferring MST building strategy, in order to check the improvement to the Distance Preferring MST building Strategy proposed in this paper.

.



	Naïve (A)	Distance (B)	Improvement from (A) to (B)
Direct	40718.6	36387.3	10.6%
Global	41899.3	39093.3	6.7%
Local	42677.8	38409.8	10.0%
Average improvement			9.1%

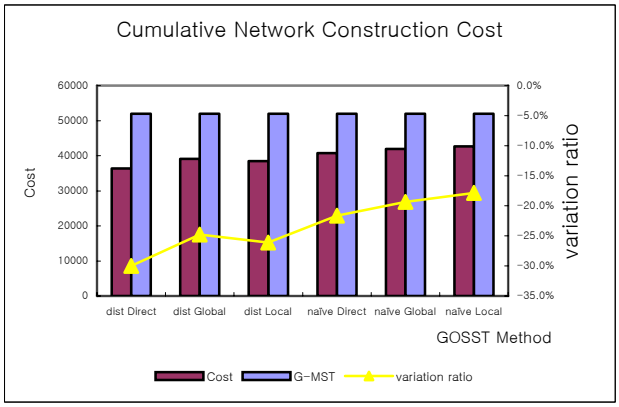
Fig 18 The improvement of network construction cost saving by Distance Preferring MST building Strategy. Cumulative network construction cost sums and cost variation ratios to G-MST is for 64 experiments of each method.

From the result of Fig 18, we find the facts as follows;

- For 3 location strategies, the Distance Preferring MST building strategy proposed in this paper is better than Naïve MST building strategy proposed in previous research.
- The costs are saved by average 9.1% when employing Distance Preferring MST building strategy instead of Naïve MST building strategy.
- Direct locating strategy shows the most gain (10.6%) by replacing MST building strategy, while Global locating strategy does the least.

6.3 Experiments for comparing the Network Construction Cost of 6 GOSST methods

Fig 19 shows the results of the cumulative network constructing costs for 64 experiments of each GOSST method and the cost variation ratios to G-MST.



Method	Cost (A)	Saving (C-A)/A	Difference (T-A)/A
Dist Direct	36387.3 (T)	30.0%	0
Dist Global	39093.3	24.8%	-6.9%
Dist Local	38409.8	26.1%	-5.3%
Naïve Direct	40718.6	21.6%	-10.6%
Naïve Global	41899.3	19.4%	-13.2%
Naïve Local	42677.8	17.9%	-14.7%
G-MST	51960.1 (C)	0.0%	-30.0%

Fig 19 The improvement of network construction cost saving by Distance Direct GOSST method. Network construction cost sums and cost saving ratios to G-MST for 64 experiments of each method.

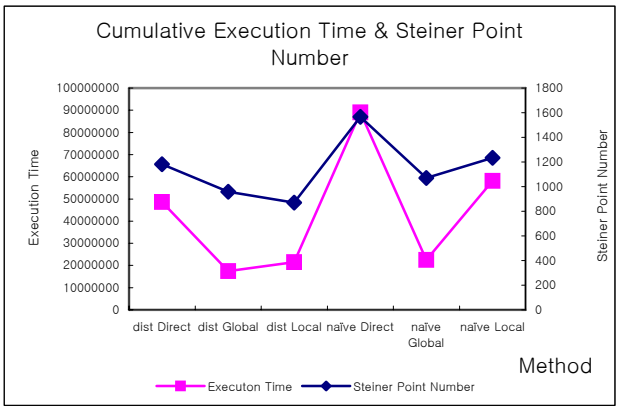
From the results of Fig 19, we find the facts as follows;

- Distance Direct GOSST method shows the most cost saving (30%), while Naïve Local GOSST method does least (17.9%).
- Distance Direct GOSST method obtains 6.9% improvement comparing to Distance Local GOSST method which is the most cost saving in previous research.
- Distance MST building strategy is better than Naïve MST building strategy in savings of the network construction cost.
- Direct locating, Local locating and Global locating is the decreasing order of savings of the

network construction cost in Distance MST building strategy, while Direct locating, Global locating and Local locating is in Naïve MST building strategy.

6.4 Experiments on the Execution Time and Created Steiner Points Number

Fig 20 shows the cumulative results of 64 experiments for execution time and created Steiner point number of each GOSST method.



Method	Execution Time	Steiner Point Number
Dist Direct	48640220	1181
Dist Global	17432218	959
Dist Local	21495384	869
Naïve Direct	89101592	1565
Naïve Global	22541437	1069
Naïve Local	58197638	1234

Fig 20 The accumulative of execution times and created Steiner points numbers for 64 experiments of each method

From the results of Fig 20, we find the facts as follows;

- Naïve Direct GOSST method needs most execution times among 6 methods, while Distance Global does least.
- Naïve Direct GOSST method needed most Steiner points for a GOSST solution, while Distance Local does least.
- Naïve MST building strategy requires more execution time and Steiner points than Distance MST building strategy.
- Direct locating strategy requires more Steiner points than Local locating strategy and Global locating strategy.
- Direct locating strategy, Local locating strategy and Global locating strategy is the decreasing order of execution time.

7. Conclusions

This research is to propose more improved heuristic than previous heuristic for GOSST problems [19]. With the results, the possibility of applying the heuristic to some practical and useful applications could be increased in real world.

GOSST problem is to seek for a network topology, which interconnects all points in set **P** and some Steiner points with satisfying G-condition and Minimum network constructing costs. This GOSST could provide paths with enough communication capacities between any two sites, with the least network construction costs. The researches on GOSST have focused on the optimization. In Optimization on GOSST, Ding-Zhu, Du and his colleagues have solved the difficult problems related to Steiner Tree problem and deduced many problems related to Steiner Tree problem. One of them is GOSST. As this problem might be regarded to reach the theoretical limitation of the optimization solution, many people seem not to try the theoretical research.

Most of optimization algorithms for GOSST have constructed to prove that the GOSST problem is belonged to PTAS (Polynomial Time Approximation Scheme) problem. When the problem is asserted as PTAS problem, problem classification, the main issue of theory of computing, could be accomplished. While the running time is polynomial time, in fact, the size might be exponential time because exponential function could be approached by polynomial time. The purpose of drawing the approximation algorithm is to show the given problem has the constructively polynomial time algorithm in theoretical view, not to apply to practical area. In PTAS problem, if the running time of approximation algorithm decreases to practical level, the approximation ratio could be worse drastically. In most case, the approximation ratio becomes meaningless.In Heuristic on GOSST, as people might consider this problem as theoretical problem related to Steiner Tree problem and not to expand their sight to the value anymore, they seem not to do their researches on practical applications. But as this problem could be applied to the network or circuit design area practically, the research to implement a heuristic with more practical time complexity is necessary.

For the Direct Locating strategy, the costs are saved by average 3.7% when employing Direct Locating strategy instead of previous proposed 2 locating strategies such as Global Locating and Local Locating. For he costs are saved by average 9.1% when employing Distance Preferring MST building strategy instead of Naïve MST building strategy.

By combination of Locating strategy and MST building strategy, we suggest new 4 GOSST methods addition to 2 GOSST methods proposed in previous

research [19]. The new methods employ direct locating strategy and distance preferring MST building strategy. Among the 6 GOSST methods for the problem, Distance Direct GOSST method shows the most cost saving (30%), while Naïve Local GOSST method does least (17.9%). Distance Direct GOSST method obtains 13.2 % improvement comparing to Naïve Global GOSST method which is the most cost saving in previous research.

For execution time and required Steiner points number, Naïve Direct GOSST method needs most execution times among 6 methods, while Distance Global does least. Naïve Direct GOSST method needed most Steiner points for a GOSST solution, while Distance Local does least. Naïve MST building strategy requires more execution time and Steiner points than Distance MST building strategy. Direct locating strategy requires more Steiner points than Local locating strategy and Global locating strategy.

Our further works are more elaborate analysis and evaluation of our experiments results and more studies and improvement of the proposed methods. For performance improvement, Execution time & required Steiner points number, cost and cost saving ratio, gain of edge capacity sum, and overhead of edge length sum are considered. Also we will study for discovering network application problem domains to apply this GOSST heuristic to the area such as networks design and construction, wireless network and QoS.

References

[1] Xue, G.L., Lin, G.H. and Du, D.z. (2001), Grade of Service Steiner Minimum Trees in Euclidean Plane, Algorithmica, 31, 479-500.

[2] Duin, C. and Volgenant, A. (1991), The Multi-weighted Steiner Tree problem, Annals of Operations Research 33, 451-469

[3] Balakrishnan, A., Nagnanti, T.L. and Mirchandani, P. (1994), Modeling and Heuristic Worst Case Performance Analysis of the two level Network Design Problem, Management Science 40, 846-867

[4] Current, J.R., Revelle, C.S. and Cohon,J.L.(1986), The hierarchical network design problem, European Journal of Operational Research 27, 57-66.

[5] J.Kim, M.Cardei, I.Cardei and X.Jia, A polynomial Time Approximation Scheme for the Grade of Service Steiner Minimum Tree Problem

[6] Arora, S.(1996), Polynomial-Time Approximation Schemes for Euclidean TSP and other Geometric Problems, Proceeding of 37th IEEE Symposium on Foundations of Computer Science, 2-12.

[7] Winter, P and Zachariasen, M.(1998), Large Euclidean Steiner Minimum trees : an improved exact algorithm, Networks 30, 149-166.

[8] G.-H. Lin and G.L. Xue (1999), Steiner tree problem with minimum number of Steiner points and bounded edge-length, Information Processing Letters, 53-57

[9] Joonmo Kim and Inbum Kim (2003), Approximation ratio 2 for the Minimum Number of Steiner Points, Journal of KISS, 387-396

[10] T.H. Cormen, C.E Leiserson, R.L. Rivest and C.Stein, Introduction to Algorithms, McGraw-Hill Book Company

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

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

[13] E.J. Cockayne and D.E. Hewgrill (1986), Exact Computation of Steiner minimal trees in the plane, Information Processing Letters, Vol.22, pp.151-156

[14] E.J. Cockayne and D.E. Hewgrill (1992), Improved computation of plane Steiner minimal tree, Algorithmica, Vol.7, pp.219-229

[15] D.Z. Du and F.K. Hwang (1990), An approach for providing lower bounds; solution of Gilbert-Pollak conjecture on Steiner ratio, Proceedings of IEEE 31sy FOCS, pp.76-85

[16] M.J. Smith and B. Toppur (1997), Euclidean Steiner minimal trees, minimum energy configurations and the embedding problem of weighted graph in E3, Discrete Applied Mathematics, Vol.71, pp.187-215

[17] P. Winter (1985), An algorithm for the Steiner problem in the Euclidean plane, Networks, Vol.15, pp.323-345

[18] P. Mirchandani(1996), The multi-tier tree problem, INFORMS Journal on Computing, Vol.8, pp.202-218

[19] Inbum Kim, Chae-kak Kim and S.H. Hosseini (2006), A Heuristic using GOSST with 2 Connecting Strategies for Minimum Construction Cost of Network, International Journal of Computer Science and Network Security, Vol.6, pp.60-72

Inbum Kim received the B.S. and M.S. degree in computer engineering from Seoul national university, South Korea, respectively. He worked as research engineer at Daewoo Telecom Ltd. and Oracle Korea, Seoul, South Korea. He is currently an associate processor at the department of Internet Information, Kimpo College, Kyonggi-do, South Korea and is working toward his Ph.D degree at university of Wisconsin Milwaukee. His interests are in network construction, mobile ad hoc networks, computer architecture, computer theory and database system.

Chae-kak Kim received his B.S., M.S. and Ph.D. degrees in computer science from Soongsil university, South Korea, Yonsei university, South Korea and Soongsil university, South Korea, respectively. He worked as system engineer and software development team manager at LG Electronics and TriGem Computer Inc.. He is currently an associate processor at the Dept. of Internet Information, Kimpo College, Kyonggi-do, South Korea. His interests are in mobile and network security.