Journal of Global Optimization **24:** 437–448, 2002.

A Polynomial Time Approximation Scheme for the Grade of Service Steiner Minimum Tree Problem

JOONMO KIM[1], MIHAELA CARDEI[1], IONUT CARDEI[1] and XIAOHUA JIA[2]
[1]*Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN 55455, USA (e-mail: {jkim,mihaela,ionut}@cs.umn.edu);* [2]*Department of Computer Science, City University of Hong Kong, Kowloon Tong, Hong Kong (e-mail: jia@cs.cityu.edu.hk)*

Abstract. In this paper, we present the design of a Polynomial Time Approximation Scheme (PTAS) for the Grade of Service Steiner Minimum Tree (GOSST) problem, which is known to be NP-Complete. Previous research has focused on geometric analyses and different approximation algorithms have been designed. We propose a PTAS that provides a polynomial time, near-optimal solution with performance ratio $1 + \epsilon$. The GOSST problem has some important applications. In network design, a fundamental issue for the physical construction of a network structure is the interconnection of many communication sites with the best choice of the connecting lines and the best allocation of the transmission capacities over these lines. Good solutions should provide paths with enough communication capacities between any two sites, with the least network construction costs. Also, the GOSST problem has applications in transportation, for road constructions and some potential uses in CAD in terms of interconnecting the elements on a plane to provide enough flux between any two elements.

Key words: Approximation algorithms, Rectangular partitions, Grade of service Steiner minimum tree, Network design

1. Introduction

The Grade of Service Steiner Minimum Tree (GOSST) problem [22] can be defined as follows: Let $P = \{p_1, p_2, \ldots, p_n\}$ be a set of n terminal points in the Euclidean plane, where point p_i has a service request of grade $grade(p_i) \in \{1, 2, \ldots, r\}$. Let $0 < c(1) < c(2) < \cdots < c(r)$ be r real numbers. Each edge in the network is assigned a specific grade of service, which is a number in $\{1, 2, \ldots, r\}$. We use $grade(e)$ to denote the grade of service of edge e. The cost-per-unit-length (CPUL) for an edge with service grade u is $c(u)$. The GOSST problem asks for a minimum cost network interconnecting the point set P and some Steiner points with service request of grade 0 such that (1) between each pair of terminal points p_i and p_j there is a path whose minimum grade of service is at least as large as $\min(grade(p_i), grade(p_j))$ and (2) the cost of the network is minimum among all interconnecting networks satisfying the condition (1), where the cost of an edge with service of grade u is the product of the Euclidean length of the edge with $c(u)$.

The *GOSST* problem is a generalization of the Euclidean Steiner Minimum Tree (ESMT) problem, which is obtained when all terminal points have the same grade of service request. The ESMT problem [12, 15] asks for a minimum cost network interconnecting a set of given points in the Euclidean plane, where the network cost is defined as the sum of the edge lengths. Reference [22] describes its history and some major applications. Additional research results can be found in [4, 5, 8, 9, 11, 12–14, 17–19, 20, 21, 23].

In the GOSST problem, the minimum cost interconnecting network is determined by the combination of two factors: (i) the grade of service for each edge, and (ii) the choice of Steiner points. That is, (i) and (ii) cannot be determined one before another. Most research results refer to the case when the number of grades of service request is either 2 or 3 [10], or some other particular cases [3, 7]. For the general case, Michandani [16] proposed an approximation algorithm with performance ratio $r\rho + 1$, where ρ is the best performance ratio of a Steiner tree heuristic and r is the number of different grades of service request. In [6], Colburn and Xue presented the GOSST problem on a series-parallel network and proposed an $O(r^3n)$ time algorithm, where n is the number of vertices and r is the number of grades of service.

Based on the problem characteristics, we propose a Polynomial Time Approximation Scheme (PTAS) to the GOSST problem using the dynamic programming approach to accomplish the general case solution. Still, using a classic dynamic programming is impossible in this case. In our approach, we provide some adjustments and use *rectangular partitions* [1, 2] to the problem instance to facilitate the implementation of the dynamic programming. Fortunately, the GOSST problem fits well to the Arora's technique [1, 2], so we may proceed straightforward to the $(1+\epsilon)$-approximation.

The rest of this paper is organized as follows. Overall, Section 2 presents a PTAS for the GOSST problem. Subsection 2.1 presents some definitions and the Structure Theorem. Then, the PTAS is described in Subsection 2.2. Subsection 2.3 contains the proof for the Structure Theorem and Section 3 concludes the paper.

2. The GOSST Algorithm

As mentioned earlier, we apply Arora's framework [1, 2] to the GOSST problem. The basic idea of the PTAS is sketched next. For a given problem instance, the Structure Theorem (*Theorem 1*) uses a recursive partition of the square containing all given points and guarantees the existence of a $(1+\varepsilon)$-approximate solution that crosses each line of the partition at most m times, where $m = O((\log n)/\epsilon)$, such that all the crossings happen at some prespecified points, called *portals*. A solution characterized by these properties can be found with dynamic programming. In other words, we decompose the instance of our problem into a large number of smaller problems and apply dynamic programming to obtain a solution which is

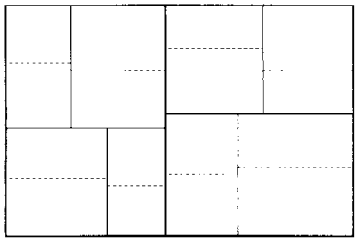

Figure 1. A 1/3:2/3 tiling

guaranteed by the Structure Theorem to be a $(1 + \epsilon)$-approximate of the optimal solution of the given instance.

2.1. DEFINITIONS AND THE STRUCTURE THEOREM

In this paper we understand by a *rectangle* an axis-aligned rectangle. All the definitions that follow and apply to a rectangle are also valid for a square. The *size* of a rectangle is the length of its longer edge. The *bounding box* of a set of terminal points is the smallest square enclosing them. The *optimal structure* represents the optimal cost network the GOSST problem asks for.

The *cost* of an interconnecting network is the sum of the products of the Euclidean edge length with the cost-per-unit-length (CPUL) of that edge. The CPUL for an edge with service grade u is $c(u)$. The minimum CPUL is $c(1)$ and the maximum is $c(r)$, where $c(r) = C \cdot c(1)$ and C is a constant.

A *line separator* of a rectangle R is a straight line segment parallel with R's shorter edge that partitions R into two rectangles, each having area at least one-third of R's area. For example, if R's width W is greater than the height, then a *line separator* is any verticle line in the middle $W/3$ of R.

Next we define a recursive partition of a rectangle, which is used by the dynamic program algorithm.

DEFINITION 1 (1/3:2/3-tiling). A 1/3:2/3-tiling of a rectangle R is a binary tree (a hierarchy) of sub-rectangles of R. The rectangle R is at the root. If the size R is $\leqslant 1$, then the hierarchy contains nothing else. Otherwise the root contains a line separator for R, and has two subtrees that are 1/3:2/3-tilings of the two rectangles, into which the line separator divides R (see Figure 1).

The rectangles obtained in the tiling procedure at depth d form a partition of the root rectangle. Also, all rectangles at depth $d + 1$ are a refinement of the depth d partition, obtained by applying a *line separator* to each depth d rectangle of size > 1.

DEFINITION 2 (portals). A portal in a 1/3:2/3-tiling is any point that lies on an edge of any rectangle in the tiling. If m is any positive integer, then a set of portals T is called m-regular for the tiling, if there are exactly m equidistant portals on the

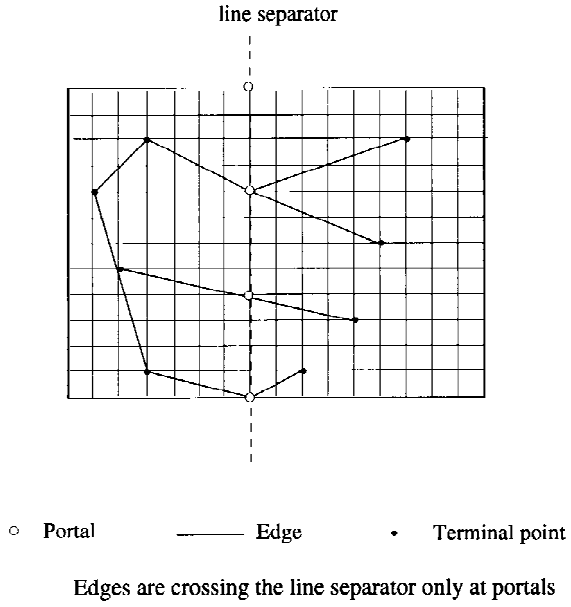

Figure 2. An *m-light* graph.

line separator of each rectangle of the tiling. We assume that the end points of the line separator are also portals. In other words, the line-separator is partitioned into exactly $m - 1$ equal parts by the portals on it.

DEFINITION 3 (m-light structure). Let $m \in Z^+$ and π be an interconnecting network instance for the GOSST problem that satisfies the conditions in the problem definition. Let S be a 1/3:2/3-tiling of the bounding box and T be an m-regular set of portals on this tiling. Then π is a m-light structure with respect to S if the following are true: (i) in each rectangle of tiling S, the edges cross the line separator of that rectangle at most m times (ii) the edges cross the line separator only at portals in T (see Figure 2).

Next we present the Structure Theorem which guarantees that every COSST instance has a $(1 + \epsilon)$-approximate m-light structure. This Theorem will be proved later, in subsection 2.3.

THEOREM 1 (Structure Theorem). *The following is true for every* $\epsilon > 0$. *Each GOSST problem instance has a* $(1 + \epsilon)$*-approximate* m*-light structure, where* $m = O((\log n)/\epsilon)$.

In this paper we use the following abbreviations: *APX*, Approximation Structure; *OPT*, Optimal Structure.

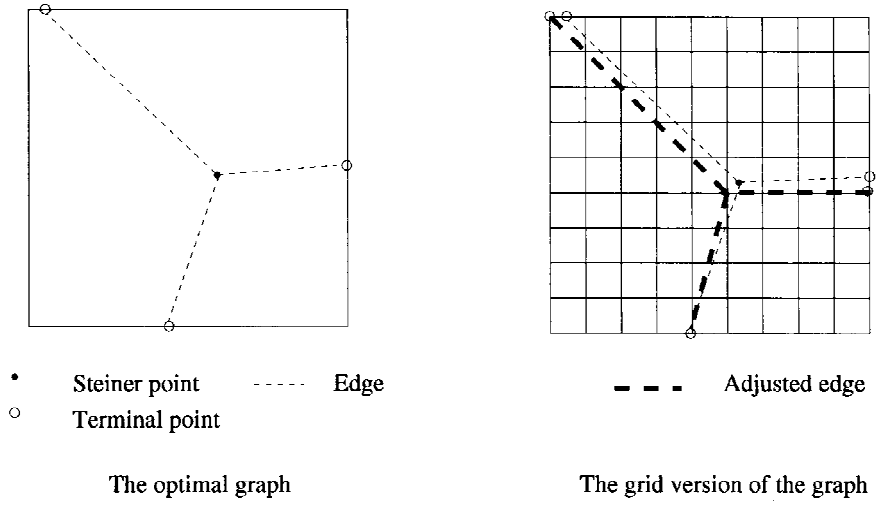

Figure 3. Difference between the optimal and grid graph.

2.2. A PTAS FOR THE GOSST PROBLEM

In order to facilitate the design of the dynamic programming, we adjust the problem instance to grids and then rescale the problem instance, making it well-rounded. In addition we assume that the Steiner points lie only at the grids points. A well-rounded instance will have the coordinates of the network points integral and the minimum non-zero internode distance 1. Let us suppose the bounding box is a square with edge length L. We equally divide the initial square into a $g \times g$ grid, where $g = O(n^2)$ and then move each terminal point to its nearest gridpoint (see Figure 3). Note that each point is moved by a distance less than L/g. Finally, divide all distances in the new instance by L/g, so that the smallest internode distance is 1 and the size of the bounding box is g. Also, in the 1/3:2/3 *tiling* mechanism the *line separators* will lie over the grid lines.

The next proposition shows that with the adjustments described above, a PTAS of the grid instance is a PTAS of the original instance.

PROPOSITION 1. *$(1 + \epsilon)$-approximation over the grid instance implies $(1 + \hat{\epsilon})$-approximation over the original instance.*

Proof.

$$|\text{total\_cost}^{\text{Original}}_{\text{OPT}} - \text{total\_cost}^{\text{Grid}}_{\text{OPT}}| \leqslant 2 \cdot (2n - 3) \cdot c(r)$$

where the number of edges of the tree structure is $2n - 3$. Note that the maximum number of points for the problem instance is $2n - 2$, that is the sum of terminal points and Steiner points. Both end points of an edge can move within the distance 1 (after the rescaling step), each with the highest *service request of grade* $c(r)$.

$$|\text{total\_cost}^{\text{Original}}_{\text{APX}} - \text{total\_cost}^{\text{Grid}}_{\text{APX}}| \leqslant 2 \cdot n^2 \cdot c(r)$$

where n^2 is greater than the worst case number of edges $\binom{n}{2}$, when the approximation graph is a complete network.

We know that:

$$\text{total\_cost}_{\text{APX}}^{\text{Grid}} \leqslant (1+\epsilon) \cdot \text{total\_cost}_{\text{OPT}}^{\text{Grid}}$$

Based on these relations we get:

$$\begin{aligned}
\text{total_cost}_{\text{APX}}^{\text{Original}} &\leqslant \text{total_cost}_{\text{APX}}^{\text{Grid}} + 2n^2c(r) \\
&\leqslant (1+\epsilon) \cdot \text{total_cost}_{\text{OPT}}^{\text{Grid}} + 2n^2c(r) \\
&\leqslant (1+\epsilon)(\text{total_cost}_{\text{OPT}}^{\text{Original}} + 2(2n-3)c(r)) + 2n^2c(r) \\
&= \text{total_cost}_{\text{OPT}}^{\text{Original}} \cdot \left(1+\epsilon+\frac{2(1+\epsilon)(2n-3)c(r)+2n^2c(r)}{\text{total_cost}_{\text{OPT}}^{\text{Original}}}\right)
\end{aligned}$$

The optimal cost of the original instance is at least $c(1)$ multiplied with the size of the bounding box:

$$\text{total\_cost}_{\text{OPT}}^{\text{Original}} \geqslant c(1) \cdot g, \text{ where } g = \hat{C} \cdot n^2 \text{ and } \hat{C} \text{ is a constant.}$$

So we obtain:

$$\begin{aligned}
\text{total_cost}_{\text{APX}}^{\text{Original}} &\leqslant \text{total_cost}_{\text{OPT}}^{\text{Original}} \cdot \left(1+\epsilon+\frac{2C}{\hat{C}} \cdot \left(1+\frac{(1+\epsilon)(2n-3)}{n^2}\right)\right) \\
&\leqslant \text{total_cost}_{\text{OPT}}^{\text{Original}} \cdot (1+\hat{\epsilon})
\end{aligned}$$

The value of $\hat{\epsilon}$ can be chosen correspondingly by properly choosing the value of $\hat{C}$. Note that g can be set to greater values (like $O(n^3)$, etc) by choosing a smaller grid granularity. □

Based on this proposition, we will work on the grid version of the problem instance.

PROPOSITION 2. *If the bounding box square has the size $\hat{C} \cdot n^2$, then every 1/3:2/3 tiling has depth $O(\log n)$.*

Proof. The area of any depth i rectangle is at most $(2/3)^i$ times the area of the bounding box. Also, in order to apply the tilling over a rectangle it has to have the size greater than 1. Therefore, for a rectangle at the last level, say the d^{th} level, we have $(2/3)^{d-1}.\ (\hat{C}n^2)^2 > 1$. That is $d = O(\log n)$. □

We describe next the PTAS. Since $\epsilon > 0$ is arbitrary, based on the results of *Proposition 1*, we adjust the problem instance to the grid such that the minimum interpoint distance is 1 and the bounding box has size $O(n^2)$. The Structure Theorem guarantees the existence of a $(1+\epsilon)$-approximate *m-light structure* π and a tiling S, where $m = O((\log n)/\epsilon)$. By *Proposition 2*, the depth of such a tiling is

$O(\log n)$. We describe next the dynamic programming that finds a minimum cost *m-light structure* and its corresponding tiling in $n^{O(1/\epsilon)}$ time.

The network in the GOSST problem contains both terminal points and Steiner points. During the dynamic programming, the Steiner points that lie on a *line separator* are allowed to have assigned a grade of service different of 0, such that they will behave more like temporary terminal points. The temporarily assigned grades of service represent the potential effect of terminal points from some other partitions on assigning grades of service to the edges. When the dynamic programming ends, these Steiner points lying on some *line separator* will be reassigned the grade of service 0. This final change does not affect the current grade of service of any edge, so the network cost is unchanged.

First, we want to determine how a subproblem is defined. The conditions that need to be satisfied are:

(1) It should contain the original problem as a special case.

(2) It should satisfy the following recursive relation:

$$\text{Opt}(P) = \min_{(P', P'')} (\text{Opt}(P') + \text{Opt}(P'') + \text{cost}(P', P''))$$

where P is a subproblem that can be partitioned into two subproblems P' and P'' and $\text{cost}(P', P'')$ is the cost of this partition.

To satisfy (2), we need to have

(2.1) For every partition (P', P'') of P,

$$\text{Opt}(P) \leqslant \text{Opt}(P') + \text{Opt}(P'') + \text{cost}(P', P'')$$

(2.2) There exists a partition (P', P'') of P such that

$$\text{Opt}(P) = \text{Opt}(P') + \text{Opt}(P'') + \text{cost}(P', P'')$$

For the considered problem, we use only portals. This yields that $cost(P', P'') = 0$. To satisfy (2.1), a simple way is to ask that

$$\text{Opt}(P') + \text{Opt}(P'') = \text{Feasible}(P)$$

A subproblem P is feasible if between each pair of terminal points p_i and p_j from P there is a path whose minimum grade of service is at least as large as $\min(grade(p_i), grade(p_j))$.

Now, let us specify the definition of a subproblem. Each subproblem P contains the following parameters:

(a) A rectangle R inside the bounding box.

(b) A multiset of portals on the four sides of R. One side contains exactly m portals and each of the other three contains less than m portals.

(c) A multiset X of $k(\leqslant 4m)$ portals on the four sides. These are the crosspoints.

(d) A connection pattern $(X_1, X_2, \ldots, X_p)$ of X, where each set X_i identifies a connected component (a tree). There is no edge crossing between different components.
(e) A grade of service for each crosspoint in X.

The entry in the lookup table stores the lowest cost *m-light structure* found by the algorithm for this instance of the subproblem P. In this *m-light structure*, the i^{th} connected component contains all portals from X_i and all connected components contain all terminal points inside R. Each such instance can be broken into many smaller and simpler instances, by choosing a *line separator* according to the 1/3:2/3-*tiling* mechanism. Continuing this way we obtain smaller and smaller instances of the problem, and we stop when the instance is small enough to allow a brute-force solution.

The size of the lookup table is at most

$$\#(a) \times \#(b) \times \#(c) \times \#(d) \times \#(e).$$

where #(x) represents the number of choices for (x). The number of choices in (a) is $O(n^8)$. The number of choices in (b) is $O(n^{12})$. The m portals on a *line separator* are evenly spaced, so they are completely determined once we know the *line separator*. Also the number of choices for a *line separator* is at most the number of pairs of nodes, which is $\binom{n^2}{2}$. One side of the rectangle R is a complete *line separator*, so its portals are known, whereas the other three sides are parts of the *lines separator* of some ancestor, accounting for the factor $4 \cdot O((n^4)^3) = O(n^{12})$. Once we have identified the portals, the number of ways of choosing a multiset of k out of them is $\#(c) = O(2^{4m+k})$.

The number of choices for (d) is $O(2^{O(k)})$ and the number of choices for (e) is at most r^k, where r is the number of different grades of service request.

Therefore the total number of entries in the lookup table is upper bounded by

$$\begin{aligned} & n^8 \times n^{12} \times \sum_{k=1}^{4m} 2^{4m+k} \cdot 2^k \cdot r^k \\ &= n^{20} \times 2^{4m} \times \sum_{k=1}^{4m} (4r)^k \\ &\leqslant n^{20} \times 2^{4m} \times \sum_{k=1}^{4m} (4r)^{4m} \\ &= n^{20} \times (8r)^{4m} \times 4m \end{aligned}$$

which is $n^{O(1/\epsilon)}$, since $m = O((\log n)/\epsilon)$ and $r^{O((\log n)/\epsilon)} = n^{O(1/\epsilon)}$.

We already mentioned that a leaf rectangle has a limited number of terminal points, such that it can be solved by the brute-force in polynomial time. Solving

a subproblem implies computing the cost of the network inside the corresponding rectangle, by assigning grades of service to the edges. The grade of service of Steiner points which are not portals is 0. If the lookup entry corresponds to the root rectangle of the 1/3:2/3-*tiling*, then no set of portals or crosspoints need to be considered for each side. This is the result of the dynamic programming and is an *m-light structure*. For each other entry, we need to calculate an optimal *m-light structure* recursively. Let R be a nonleaf rectangle. Each choice of (b), (c), (d) and (e) for R decides an instance for the subproblem P. Let us see how the table entry corresponding to the subproblem P is computed. The objective of the subproblem P is to compute a minimum cost network G satisfying the following conditions:

- There exists a *1/3:2/3-partition* of R such that G crosses the *line separator* only at portals.
- For each pair of terminal points p_i and p_j in X_i there is a path whose minimum grade of service is at least as large as $\min(grade(p_i), grade(p_j))$.
- Each terminal point lying inside R is connected to a component X_i for some i.

To get a partition (P', P'') of the subproblem P, we need to do the following:

(a′) Choose a *line separator* to cut R into two subrectangles R' and R''.

(b′) On the *line separator*, choose $k'(\leqslant m)$ portals. These are the crosspoints, where the network crosses the *line separator*.

(c′) In R' and R'' choose connection patterns $(X'_1, X'_2, \ldots, X'_{p'})$ and $(X''_1, X''_2, \ldots, X''_{p''})$ for crosspoints on the boundaries of R' and R'', respectively. These two connection patterns should fit with the connection pattern $(X_1, X_2, \ldots, X_p)$ in the following way:

(1) If X_i lies entirely in R' (or R''), then $X_i \subseteq X'_j$ (or $X_i \subseteq X''_j$) for some j.

(2) If X_i is partitioned into two nonempty subsets A and B, where A lies in R' and B lies in R'', then there exists a crosspoint c such that $A \cup c \subseteq X'_j$ for some j and $B \cup c \subset X''_h$ for some h.

(d′) Choose a grade of service for each k' crosspoints of the *line separator*. For the case (2), when X_i is partitioned into two nonempty subsets A and B, the grade of service chosen for c needs to satisfy the inequality:

$$g(c) \geqslant \min(\max\{grade(p_i) | p_i \in A\}, \max\{grade(p_j) | p_j \in B\})$$

Each choice of (a′), (b′), (c′) and (d′) decides two smaller instances whose optimal cost network solutions are already in the lookup table, so they do not need to be computed again. From all the possible partition choices that issue a valid instance for the subproblem P, the one with the smallest cost will be stored into the lookup table.

Since $\#(a') = O(n^2)$, $\#(b') = O(2^{m+k'})$, $\#(c') = O(2^{O(m)})$, $\#(d') = O(r^{k'})$, the running time for each entry takes $= n^{O(1/\epsilon)}$.

Therefore the dynamic programming runs in time $n^{O(1/\epsilon)}$.

REMARK 1. The dynamic programming could also return the tiling S used in obtaining the optimal cost *m-light structure* π. For this, every entry in the lookup table, but the ones corresponding to the leaf rectangles, needs to store the *line separator* used in computing the optimal value.

Note that through the dynamic programming all valid *m-light structures* have been checked, while the Structure Theorem ensures that the minimum cost *m-light structure* is within the expected approximation ratio.

2.3. PROOF OF THE STRUCTURE THEOREM

The Structure Theorem guarantees the existence of an *m-light structure*, within a small error allowance from the *optimal structure* network. Once the existence of such an *m-light structure* is shown, the minimum cost *m-light structure* is also within the error allowance. Finding the minimum cost *m-light structure* is the goal of the dynamic programming.

We prove the Structure Theorem constructively. We start from the optimal cost network and modify it successively with respect to a specific tiling S until it becomes *m-light* in report with S. We describe next the way we transform the optimal network into an *m-light structure* within a factor $(1+\epsilon)$ of the optimal. We start with the optimal network and an empty tiling. For each level in the tiling mechanism we choose the *line separator* that crosses the current network the least number of times. Then the *m-regular* portals on this *line separator* are identified and each crossing point is moved to its closest portal. The tiling mechanism has depth $O(\log n)$ and the resulting network is an *m-light structure*.

Let us see now how much does this modification of the original optimal solution increases the network cost. Let R be a rectangle at a certain level, with size W. We look at each feasible *line separator* and choose the one that intersects the current network the smallest number of points, say x points. Then the cost of the network lying in the rectangle R is at least $x \cdot c(1) \cdot W/3$. Moving x crosspoints to portals require to add some segments of cost at most $x \cdot c(r) \cdot W/(m-1)$. Let us note with $total\_cost(R)$ and $total\_cost^*(R)$ the cost of the network lying inside R before and after applying the modification. Then we get:

$$\frac{total\_cost^*(R) - total\_cost(R)}{total\_cost(R)} \leqslant \frac{x \cdot c(r) \cdot W/(m-1)}{x \cdot c(1) \cdot W/3} = \frac{3 \cdot C}{m-1}$$

where C is a constant, $C = c(r)/c(1)$. This implies:

$$total\_cost^*(R) \leqslant \left(1 + \frac{3 \cdot C}{m-1}\right) \cdot total\_cost(R),$$

and because the tiling has depth $O(\log n)$, the resulting *m-light* network will be an $(1 + 3 \cdot C/(m-1))^{O(\log n)}$ approximate of the original optimal network. Since $m = O((\log n)/\epsilon)$, this validates $(1 + 3 \cdot C/(m-1))^{O(\log n)} \leqslant 1 + \epsilon$, so we have obtained an $(1+\epsilon)$ approximation ratio. □

This completes the proof of the Structure Theorem, theorem that has a key role in this framework.

3. Conclusions

In this paper we studied the Grade of Service Steiner Minimum Tree problem. This is a recent problem in literature, with some important applications in network design and road interconnection areas. We presented a PTAS for the GOSST problem. We adjusted the problem instance to grids and used rectangular partitions to the problem instance in order to facilitate the design of the dynamic programming algorithm. We showed that, given a problem instance and a positive error parameter ϵ, (i) the running time of the dynamic programming is polynomial in the size of problem instance and (ii) based on the Structure Theorem, the output of the dynamic programing is guaranteed to be a $(1+\epsilon)$-approximate solution.

References

1. Arora, S. (1996), Polynomial-Time Approximation Schemes for Euclidean TSP and other Geometric Problems, *Proceedings of 37th IEEE Symposium on Foundations of Computer Science*, pp. 2–12.
2. Arora, S. (1997), Nearly Linear Time Approximation Schemes for Euclidean TSP and other Geometric Problems, *Proceedings of 38th IEEE Symposium on Foundations of Computer Science*, pp. 554–563.
3. Balakrishnan, A., Magnanti, 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. Cockayne, E.J. and Hewgill, D.E. (1986), Exact Computation of Steiner Minimal Trees in the Plane, *Information Processing Letters* 22, 151–156.
5. Cockayne, E.J. and Hewgill, D.E., (1992), Improved Computation of Plane Steiner Minimal Trees, *Algorithmica* 7, 219–229.
6. Colbourn, C.J. and Xue, G. (2000), Grade of Service Steiner Trees on a Series-Parallel Network, in Du, D.Z. Smith, J.M. and Rubinstein, J.H. (eds)., *Advances in Steiner Trees*, Kluwer Academic Publishers, Dordrecht, pp. 163–174.
7. Current, J.R., Revelle, C.S. and Cohon, J.L. (1986), The hierarchical network design problem, *European Journal of Operational Research* 27, 57–66.
8. Du, D.Z. and Hwang, F.K. (1990), An Approach for Proving Lower Bounds: Solution of Gilbert-Pollak Conjecture on Steiner Ratio, *Proceedings of IEEE 31st FOCS*, 76–85.
9. Du, D.Z., Lu, B., Ngo, H. and Pardalos, P.M. (2000), Steiner Tree Problems, *manuscript*.
10. Duin, C. and Volgenant, A. (1991), The Multi-weighted Steiner Tree Problem, *Annals of Operations Research* 33, 451–469.
11. Garey, M.R., Graham, R.L. and Johnson, D.S. (1997), The Complexity of Computing Steiner Minimal trees, *SIAM Journal of Applied Mathematics* 32, 835–859.
12. Gilbert, E.N. and Pollack, H.O. (1968), Steiner Minimal Trees, *SIAM Journal on Applied Mathematics* 16, 1–29.
13. Hwang, F.K. (1991), A Primer of the Euclidean Steiner Problem, *Annals of Operations Research* 33, 73–84.
14. Hwang, F.K., Richard, D.S. and Winter, P. (1992), The Steiner Tree Problem, in *Annals of Discrete Mathematics* 53, North-Holland, Amsterdam.

15. Melzak, Z.A. (1961), On the Problem of Steiner, *Canadian Mathematics Bulletin* 4, 143–148.
16. Mirchandani, P. (1996), The Multi-tier Tree Problem, *INFORMS Journal on Computing* 8, 202–218.
17. Sarrafzadeh, M., Lin, W.-L. and Wong, C.K. (1998), Floating Steiner Trees, *IEEE Transactions on Computers* 47, 197–211.
18. Smith, M.J. and Toppur, B. (1997), Euclidean Steiner Minimal Trees, Minimum Energy Configurations, and the Embedding Problem of Weighted Graphs in E^3, *Discrete Applied Mathematics* 71, 187–215.
19. Winter, P. (1985), An Algorithm for the Steiner problem in the Euclidean Plane, *Networks* 15, 323–345.
20. Winter, P. and Zachariasen, M. (1996), Large Euclidean Steiner Minimum Trees in an Hour, *Technical report 96/34*, http://www.diku.dk/pawel/publications.html.
21. Winter, P. and Zachariasen, M. (1998), Large Euclidean Steiner Minimum Trees: an improved exact algorithm, *Networks* 30, 149–166.
22. Xue, G.L., Lin, G.H. and Du, D.Z. (2001), Grade of Service Steiner Minimum Trees in the Euclidean Plane, *Algorithmica*, 31, 479–500.
23. Zelikovsky, A. (1993), An $\frac{11}{6}$-Approximation Algorithm for the Network Steiner Problem, *Algorithmica* 9, 463–470.