An interactive programme for weighted Steiner trees

Marcelo Zanchetta do Nascimento[1], **Valério Ramos Batista**[2], **Wendhel Raffa Coimbra**[3]

[1] *FACOM-UFU, Av. João Naves de Ávila 2121, Bl.A, 38400-902 Uberlândia-MG, Brazil*
[2] *CMCC-UFABC, R. Sta. Adélia 166, Bl.B, 09210-170 St. André-SP, Brazil*
[3] *UFMS, rod. BR 497 km 12, 79500-000 Paranaiba-MS, Brazil*

E-mail: `nascimento@facom.ufu.br`

Abstract. We introduce a fully written programmed code with a supervised method for generating weighted Steiner trees. Our choice of the programming language, and the use of well-known theorems from Geometry and Complex Analysis, allowed this method to be implemented with only 764 lines of effective source code. This eases the understanding and the handling of this beta version for future developments.

1. Introduction

One of the main problems at implementing multicast in Wide Area Networks (WAN) is the high cost of transmissions between terminals. Cost reduction is attained by adding routers to the network, but this increases complexity (see [1, 2]). Steiner trees have long been used in order to optimise routes, aiming at the lowest cost possible (see [3, 4]).

The Steiner Minimal Tree (SMT) problem is NP-hard [5] but can be exactly solved for terminals in thousands. GeoSteiner does it amazingly fast for terminals at random. Its strategy is to look for terminals that (almost) make equilateral triangles, and then to prune sub-optimal trees. See `www.diku.dk/hjemmesider/ansatte/martinz/geosteiner` for details. But when terminals follow a pattern, GeoSteiner can be very slow. For instance, we use 4GB of RAM, microprocessor Intel Core i5 3.2GHz, and operating system Linux Ubuntu 12.04. With this setting GeoSteiner takes 73.02s to generate Figure 1 (compare it with Figure 2). This time drops to only 0.06s when the 31 terminals are at random.

Moreover, the GeoSteiner algorithm cannot be easily adapted to find *weighted* minimal Steiner trees. This fact, together with the slowness in patterned cases, is precisely due to the strategy of looking for equilateral triangles.

Given a graph $G = (V, E, w)$, a subset $S \subset V$ and a weight function w, a *weighted minimal Steiner tree* (WSMT) $T \subset G$ is one that spans all vertices of S and also minimises the total weight. WSMTs have many applications. Among others, they are used in network formation games, computational sustainability and electric power networks [6, 7, 8]. The problem has further variations, like for unity disk graphs and restrictions on w, that have been studied recently [9, 10, 11, 12]. These and other works use heuristics for non-supervised methods. They are fast at generating weighted Steiner Trees with good chances to be minimal. But if one really seeks *the minimum*, non-supervised methods will not be reliable unless applied to few terminals.

In the last paragraph from §6 of [13] the authors commented that skill and good guesses of a trained user can help find the SMT. Their work is previous to GeoSteiner by three decades but

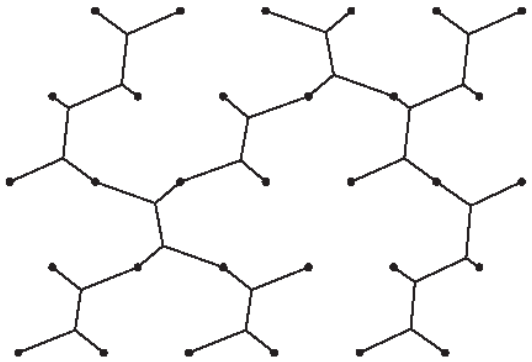
Figure 1: Non-patterned GeoSteiner output.

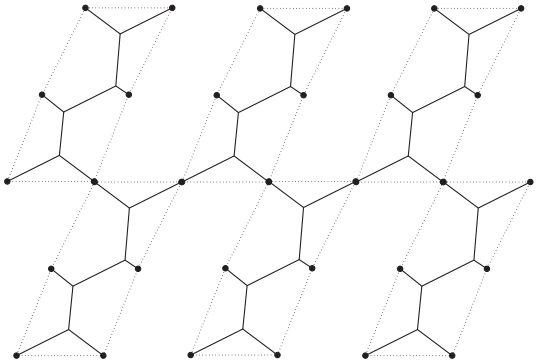
Figure 2: Patterned SMT drawn with Xfig.

their comment is still pertinent. Indeed, one cannot always predict whether a fast algorithm exists to solve a given problem. In that same work, Gilbert and Pollak conjectured that the length L of an SMT always verify $L \geq Lprim \cdot \sqrt{3}/2$, where *Lprim* is the length of the Minimal Spanning Tree (MST). This one can be obtained by Prim's algorithm [14]. According to [15] their conjecture is true. If so, any Steiner tree obtained from the terminals of an MST can improve it of at most 13.4%.

It seems rather little, but if a connection is used extensively it represents a great saving in the long-term. Moreover, in [16] the authors claim that the Gilbert and Pollak's conjecture is not completely answered yet. Thus it might even exist an SMT of which $L < Lprim \cdot \sqrt{3}/2$.

A supervised programme is not suitable for thousands of terminals, except for a long-term project distributed to a team of users and terminals in hundreds. This is still the case of sound and video cards. Their frequent access makes it desirable to bring delay to a minimum (even a 1% decrease would count). For them we need to focus on rectilinear Steiner trees, a variation we shall work on in future. However, in VLSI-design we have millions of terminals. For such extreme cases supervised methods are unsuitable.

We present `stree`, a fully programmed code to generate WSMTs. It is written in Matlab/Octave to spare the GUI. Moreover, many original ideas are within the code. They are based on theorems from Geometry and Complex Analysis. This produces a very short programme (only 855 lines, or 764 without graphical input/output), easier to understand and to handle for future developments. The present version of our programme has didactic purposes. It is even accessible to undergraduate students. The reader can access the link *Softwares* of our webpage

`http://www.facom.ufu.br/~nascimento`

to download `stree.zip`, which contains the executable and some source codes. The link also contains a *preprint* (this one with all technical detais and a user's manual).

2. Methodology

As we mentioned in the Introduction, the MST is frequently used to construct a Steiner tree. Prim's algorithm is easily adaptable to find the MST for terminals that are weighted as follows:

Definition 2.1. Consider the tree $T = (V, E)$ with weight function $w : V \to \mathbb{R}^+$ and 0-1 adjacency matrix a_{ij}. Then T is an MST if it minimises the total cost C given by

$$C = \sum_{i<j} a_{ij}(w_i + w_j)||V_i - V_j||, \tag{1}$$

where $||V_i - V_j|| = \frac{1}{2}(w_i + w_j)|V_i - V_j|$ is the connection cost between terminals V_i and V_j.

When we have $w : V \to \{1\}$, then it is the Euclidean non-weighted case. For a given set of terminals $V = \{V_1, \ldots, V_n\}$ and a weight function $w : V \to \mathbb{R}^+$, we can find the edges E that minimise the cost C in (1) and result in an MST $T = (V, E)$. By adding extra points $S = \{S_1, \ldots, S_m\}$ to V, namely $\tilde{V} = V \cup S$, we shall find the corresponding $\tilde{T} = (\tilde{V}, \tilde{E})$ such that $\tilde{C} \leq C$. We claim that $\tilde{C} < C$ exactly when $S \neq \emptyset$, providing one chooses a suitable extension $\tilde{w} : \tilde{V} \to \mathbb{R}^+$ of the weight function. The following lemma shows how to make this choice when $n = 3$ and $m = 1$.

Lemma 2.1. *Consider a triangle with vertex weights a, b and c, and suppose it admits a classical (Euclidean) Steiner point. If $s \leq \min\{a, b, c\}$ is the weight of the Steiner point, then its connection with the vertices will cost less than any other connection through the vertices only.*

Its proof and all the others' are in our preprint (see the link *Softwares* of our webpage).

Lemma 2.2. *Consider a full Steiner tree with $n \geq 3$ terminals $A_1, \ldots, A_n$ and Steiner points $S_1, \ldots, S_{n-2}$. Add weights to their respective terminals as a_i, $1 \leq i \leq n$, and to the Steiner points as $s_i = \min\{a_1, \ldots, a_n\} \, \forall \, i$. The resulting weighted tree is then a local minimum of C.*

The most important consequence of Lemmas 2.1 and 2.2 is that Steiner points can be added exactly as in the Euclidean case for arbitrary $n \geq 3$. The resulting MST $\tilde{T} = (\tilde{V}, \tilde{E})$ coincides with a Euclidean non-weighted Steiner tree, which will not be necessarily the SMT. Anyway, many results proved in [13] still hold for $\tilde{T}$: *Convex hull*, *Maxwell's Theorem*, *Lune* and *Wedge* properties.

3. Results

Our pseudocode in Figure 3 works recursively while the connection matrix is incomplete.

Input: Set of terminals
Output: Steiner tree
1. Pts ← terminals, Connexion_Matriz ← 0
2. Tree_of_Prim ← Compute_Tree_of_Prim(Pts)
3. Steiner_hull ← Compute_Steiner_hull(Pts)
4. While Connexion_Matriz implies Stree non-connected
  i. Subset_Pts ← User's Choice of a Subgroup(Pts)
  ii. Subtree ← Compute_Connection(Subset_Pts)
  iii. If Subtree not OK, redo Step i
  iv. Else Connexion_Matriz ← Connection(Subset_Pts)
5. return Steiner tree (and its length)

Figure 3: The pseudocode of the `stree` algorithm.

The main result is our programme `stree`, of which some outputs are depicted in Figure 4. There we start with a *Weighted* Minimal Spanning Tree (WMST) obtained by the adapted Prim's algorithm (see Definition 2.1). Notice that a WMST can have self-intersections, but this never happens to a WSMT because Lemmas 2.1 and 2.2 imply that any self-intersection can be split into two Steiner points in order to lower the cost.

Basically, one uses the WMST as a reference to draw Steiner subtrees. These will build an entire tree with good chances of being the WSMT. The weighted length is printed on the Matlab terminal window. The user gets some feedback while drawing and can also undo some choices. Sections *3* and *4* of our preprint are a detailed manual with hints, tutorials and illustrations.

4. Conclusions

Differently from trying a fully automated method, we propose to take advantage of the good choices that a user can make. Many attributes like intuition, guess, practice and a bird's-eye view are valuable means that one cannot translate into any programming language. As long as

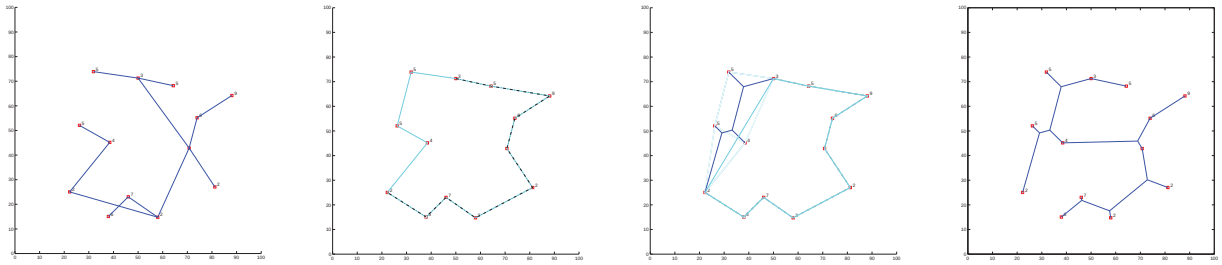

Figure 4: Some steps at running the `stree` programme.

a task is feasible with supervision we suggest taking it into account, besides the fully automated methods. This proposal is not new, but we endeavour to obtain a code that is both easy to run and to understand. The programme `stree` is still in the beta version. Further improvements will include more feedback to the user.

5. Acknowledgements

Many improvements in this paper were due to the careful analyses carried out by referees. We thank them for their valuable help. We are also grateful to Cláudio Nogueira de Meneses, professor at the Federal University of ABC, for his assistance with weighted graphs.

References

[1] Jia X, Hu X-D, Lee M, Du D-Z and Gu J 2001 Optimization of wavelength assignment for QoS multicast in WDM networks *IEEE Trans. on Communications* **49** 341–350
[2] Sahasrabuddhe L H and Mukherjee B 2001 Multicast routing algorithms and protocols: a tutorial *IEEE Network* **14** 90–102
[3] Ausiello G, Crescenzi P, Gambosi G, Kann V, Marchetti-Spaccamela A and Protasi M 1999 *Complexity and approximation: combinatorial optimization problems and their approximability properties* (Berlin: Springer)
[4] Hu X-D, Shuai T-P, Jia X and Zhang M-H 2004 Multicast routing and wavelength assignment in WDM networks with limited drop-offs *INFOCOM 2004: Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies* **1** (Hong-Kong: IEEE)
[5] Garey M R, Graham R L and Johnson D S 1977 The complexity of computing Steiner minimal trees *SIAM Journal of Applied Mathematics* **32** 835–859
[6] Dilkina B and Gomes C P 2010 *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems* (Berlin: Springer) p 102–116
[7] Guha S, Moss A, Naor J S and Schieber B 1999 Efficient recovery from power outage *Proceedings of the 31st Annual ACM Symposium on Theory of Computing* (Atlanta: ACM) p 574–582
[8] Anshelevich E, Dasgupta A, Kleinberg J, Tardos E, Wexler T and Roughgarden T 2008 The price of stability for network design with fair cost allocation *SIAM Journal on Computing* **38** 1602–1623
[9] Angelopoulos S 2006 The node-weighted Steiner problem in graphs of restricted node weights *Algorithm Theory-SWAT* (Berlin: Springer) p 208–219
[10] Demaine E D, Hajiaghayi M and Klein P N 2009 Node-weighted Steiner tree and group Steiner tree in planar graphs *Automata, Languages and Programming* (Berlin: Springer) p 328–340
[11] Li X, Xu X-H, Zou F, Du H, Wan P, Wang Y and Wu W 2009 A PTAS for node-weighted Steiner tree in unit disk graphs *Combinatorial Optimization and Applications* (Berlin: Springer) p 36–48
[12] Zou F, Li X, Gao S and Wu W 2009 Node-weighted Steiner tree approximation in unit disk graphs *Journal of Combinatorial Optimization* **18** 342–349
[13] Gilbert E N and Pollak H 0 1968 Steiner minimal trees *SIAM Journal of Applied Mathematics* **16** 1–29
[14] Prim R C 1957 Shortest connection networks and some generalizations *Bell System Technical Journal* **36** 1389–1401
[15] Du D Z and Hwang F K 1992 A proof of the Gilbert-Pollak conjecture on the Steiner ratio *Algorithmica* **7** 121–135
[16] Innami N, Kim B H, Mashiko Y and Shiohama K 2010 The Steiner ratio conjecture of Gilbert-Pollak may still be open *Algorithmica* **57** 869–872