Dynamic Programming for Minimum Steiner Trees

B. Fuchs [a] W. Kern [b] D. Mölle [c] S. Richter [c] P. Rossmanith [c] X. Wang [b,1]

[a] *University of Cologne, Center for Applied Computer Science Cologne, Group AFS, Weyertal 80, 50931 Köln, Germany*
bfuchs@zpr.uni-koeln.de

[b] *University of Twente, Department of Applied Mathematics, Faculty of EEMCS, P.O.Box 217, 7500 AE Enschede, The Netherlands*
{w.kern,xinhuiw}@math.utwente.nl

[c] *Computer Science Department, RWTH Aachen University, Germany*
{moelle,richter,rossmani}@cs.rwth-aachen.de

Abstract

We present a new dynamic programming algorithm that solves minimum Steiner tree problems with k terminals in time $O^*(c^k)$ for any $c > 2$. This improves the running time of the previously fastest exponential time algorithms (Dreyfus-Wagner [2]) of order $O^*(3^k)$ and the so-called "full set dynamic programming" algorithm, *cf.* [3], solving rectilinear instances in time $O^*(2.38^k)$.

Key words: Steiner tree, exact algorithm, dynamic programming, Dreyfus-Wagner

1 Introduction

The *Steiner tree problem* is one of the most well-known NP-hard problems: Given a graph $G = (V, E)$ of order $n = |V|$, edge costs $c : E \to \mathbb{R}_+$ and a set $Y \subseteq V$ of $k = |Y|$ *terminals*, we are to find a minimum cost tree $T \subseteq E$ connecting all terminals. Note that here and in the following, we identify a subtree of the underlying graph with its edge set $T \subseteq E$. The node set of the

[1] Supported by Netherlands Organization for Scientific Research (NWO) grant 613.000.322 (Exact Algorithms).

tree is denoted by $V(T)$. So an optimal Steiner tree for Y is a tree $T = T(Y)$ that minimizes $c(T)$ subject to $Y \subseteq V(T)$.

The Steiner tree problem has been investigated extensively *w.r.t.* approximation (*cf.*, *e.g.*, [1]) and computational complexity, both from a theoretical and practical point of view ([3], [7]). The most popular algorithm for computing minimum Steiner trees is the dynamic programming procedure proposed by Dreyfus and Wagner [2] which we shortly present below to make our presentation self-contained.

First note that (since $c \geq 0$) every leaf of a minimum Steiner tree must be a terminal. Every interior node is either a terminal or a *Steiner node*. To describe the Dreyfus-Wagner algorithm, let us adopt the following (rather ambiguous) notation: For $X \subseteq V$ we let $T(X)$ denote both the cost of minimum Steiner tree for X as well as the minimum Steiner tree itself.

The Dreyfus-Wagner algorithm recursively computes $T(X \cup v)$ for all $X \subseteq Y$ and $v \in V$. In the "generic case", the new terminal $v \in V$ is a leaf of the Steiner tree $T = T(X \cup v)$ and v is joined by a min cost path P_{vw} to an interior node $w \in V(T)$ of degree at least 3. The node w splits $T \backslash P_{vw}$ into two parts, namely $T(X' \cup w)$ and $T(X'' \cup w)$ for some nontrivial bipartition $X = X' \cup X''$. Hence we may write (slightly misusing the notation as announced earlier)

$$T(X \cup v) = \quad \min \quad P_{vw} \cup T(X' \cup w) \cup T(X'' \cup w), \tag{1}$$

where the minimum is taken over all nontrivial bipartitions $X = X' \cup X''$ and all $w \in V$.

The above recursion is also valid in the"non-generic cases", when the new terminal v is not a leaf of T (choose $w = v$) or when v is joined by P_{vw} to a leaf of $T(X)$, *i.e.*, when w has only degree 2 in T (take $X' = \{w\}$ in this case).

Summarizing, the above recursion correctly computes an optimal tree for $Y \subseteq V$. As to the running time, observe that there are less than $n\binom{k}{i}$ sets of type $X \cup v$ with $|X| = i$ and each such X has less than 2^i nontrivial bipartitions. Hence we get

$$n \sum_{i \leq k} \binom{k}{i} 2^i = n3^k = O^*(3^k)$$

as an upper bound on the running time.

2 Improving the exponential time bound

Let us fix some minimum Steiner tree $T = T(Y)$ for Y. Every leaf of T is a terminal. In case T has interior nodes which are terminals, these *interior* *terminals* split T into *components*, *i.e.*, maximal subtrees without any interior terminals. The basic idea of our approach is to add (a few) additional terminals so as to ensure that T is split into many "small" components and then recursively reconstruct T from these small components. Here and in the following, the *size* of a component equals the number of terminals (leaves) of the component.

Lemma 2.1 For $\epsilon > 0$, it suffices to add $a \leq 1/\epsilon$ additional terminals, splitting T into components of size at most $\epsilon k + 1$ each.

Proof. We may assume *w.l.o.g.* that T has no interior terminals. For an interior node $u \in V(T)$, let $k_u \leq k - 1$ denote the maximum size (including u) of the components induced by u. There exists a node u with $k_u \leq k/2 + 1$. Hence there also exists a node u^* that maximizes k_{u^*}, subject to $k_{u^*} \leq k - k\epsilon$. Observe that u^* splits T into one large component of size k_{u^*} and one or more components of size (including u^*) at most $\epsilon k + 1$ each. By induction, the large component can be split into components of size at most $\frac{\epsilon}{1-\epsilon} k_{u*} + 1 \leq \epsilon k + 1$ with no more than $\frac{1-\epsilon}{\epsilon} = \frac{1}{\epsilon} - 1$ additional terminals. ■

To describe our algorithm that reconstructs T (or any other optimal Steiner tree for Y) by successively attaching small components, we adopt the following notation from [3] for terminal sets X_1, X_2 and X:

$$X := X_1 \bowtie X_2 \quad \Leftrightarrow \quad X = X_1 \cup X_2 \text{ and } |X_1 \cap X_2| = 1.$$

Assume that $A \subseteq V(T), |A| = \left\lfloor \frac{1}{\epsilon} \right\rfloor$ has been added as a set of additional terminals, splitting $T = T(Y)$ into components of size at most $\epsilon k + 1$. Let $\tilde{Y} = Y \cup A$, so that $T = T(\tilde{Y})$. If $X_1 \subseteq \tilde{Y}$ is the terminal set of a connected union of components, then the subtree of T induced by X_1 must be a minimum Steiner tree for X_1 (otherwise T would not be optimal). Hence the following algorithm indeed composes recursively an optimal tree for Y by successively attaching small components, once an appropriate set A of additional terminals is chosen.

Algorithm ASC ("Attach Small Components")

For each $\tilde{Y}, Y \subseteq \tilde{Y} \subseteq V, |\tilde{Y}| = k + \left\lfloor \frac{1}{\epsilon} \right\rfloor$ do:

1) Compute $T(X)$ for all $X \subseteq \tilde{Y}, |X| \leq \epsilon k + 1$.

2) For all $X \subseteq \tilde{Y}, |X| > \epsilon k + 1$, compute $T(X)$ recursively, according to

$$T(X) = \min\{T(X_1) \cup T(X_2) | X = X_1 \bowtie X_2, |X_2| \leq \epsilon k + 1\}. \tag{2}$$

There are $O(n^{\frac{1}{\epsilon}})$ choices for $\tilde{Y}$ of size $\tilde{k} = k + \left\lfloor \frac{1}{\epsilon} \right\rfloor$. The time needed for 1) (using Dreyfus-Wagner) is negligible for reasonably small $\epsilon > 0$. So the total running time is bounded by

$$n^{\frac{1}{\epsilon}} \sum_i \binom{\tilde{k}}{i}\binom{i}{\epsilon k+1} \;\leq\; n^{\frac{1}{\epsilon}}\, \tilde{k}\, 2^{\tilde{k}} \binom{\tilde{k}/2}{\epsilon \tilde{k}}. \tag{3}$$

This yields our main result.

Theorem 2.1 Algorithm ASC correctly computes a minimum Steiner tree for $Y \subseteq V, |V| = k$ in time $O^*(c^k)$ for any $c > 2$ by an appropriate choice of $\epsilon > 0$.

Proof. By Stirling's Formula, the binomial in (3) can be approximated (up to a polynomial) as

$$\binom{\tilde{k}/2}{\epsilon\tilde{k}} \approx \left[(\frac{1}{2\epsilon})^{\epsilon}(\frac{1}{1-2\epsilon})^{\frac{1}{2}-\epsilon}\right]^{\tilde{k}} < (1+\delta)^{\tilde{k}}$$

for any prescribed $\delta > 0$, provided $\epsilon > 0$ is small enough. Hence indeed the running time is $O^*\left[(2+2\delta)^{\tilde{k}}\right] = O^*\left[(2+2\delta)^{k}\right]$. ■

Remark. The idea of composing the optimal tree from its components has been used by Ganley and Cohoon [4] in the rectilinear case, *i.e.*, when $Y \subseteq \mathbb{R}^2$ and (V, E) is the grid graph induced by Y endowed with the Manhattan metric $|x - y| = |x_1 - y_1| + |x_2 - y_2|$. The currently fastest algorithms for minimal Steiner tree in the rectilinear case are based on this (de-)composition (*cf.*[3]). The point is that in the rectilinear case, a lot can be said about the structure of these components. Indeed, it can be assumed *w.l.o.g.* that each component of the optimal tree consists of a straight line (the *Steiner Chain*), which starts at a terminal node and has edges (*legs*) attached to it alternatively from left and right. In addition, the last leg may have an additional edge attached to it, *cf.* Figure 1.

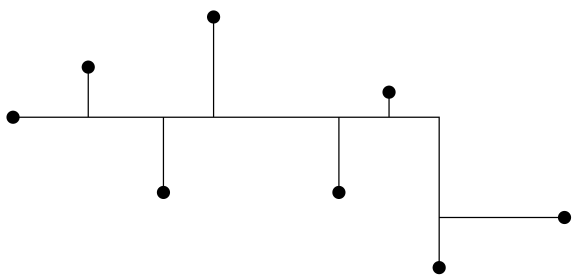

Fig. 1. A component in the rectilinear case

This structure of components in the rectilinear case in known as *Hwang topology* (*cf.* [6]). The components in the rectilinear case are called *full components* and, correspondingly, a subset $X \subseteq Y$ is called a *full set* if X is the terminal set of a Steiner tree as in Fig. 1, *i.e.*, of a component with Hwang topology.

Exploiting additional structural properties of full components, Fößmeier and Kaufmann [3] could show that the number of full sets is bounded by 1.38^k. Full sets can be identified easily, so the recursive construction of the optimal tree according to

$$T(X) = \min\{T(X_1) \cup T(X_2) | X = X_1 \bowtie X_2 \subseteq Y, X_2 \text{ full}\}$$

takes time

$$\sum_i \binom{k}{i} 1.38^i = O^*(2.38^k).$$

It remains to be analyzed, whether our idea of splitting the optimal tree can be fruitfully applied to the rectilinear case to yield an algorithm with complexity $O^*(c^k)$ without such a prohibitively large polynomial factor of $n^{\frac{1}{\epsilon}}$.

Remark. An analogue of ASC can be designed to solve the *directed Steiner tree problem*, where the underlying graph (V, E) is a directed graph and we seek for a directed rooted tree (with prescribed root terminal) connecting Y. Indeed, it suffices to compute rooted Steiner trees $T_r(X)$ (rooted in $r \in X$) for all small $X \subseteq \tilde{Y}$ and then attach these successively in the obvious way.

3 Improving the polynomial factor

In this section we show how to improve the polynomial factor of $n^{\frac{1}{\epsilon}}$ to roughly $n^{\frac{1}{\sqrt{\epsilon}}}$. The basic idea is as follows. Instead of recursively constructing the optimal tree T by adding components of size ϵk in each step, we allow the addition of larger pieces at levels $i \ll k/2$ and $i \gg k/2$. Only when $i \approx k/2$, we proceed by adding small components of size ϵk as before.

To work this out in detail, we need the following technical result:

Lemma 2 For sufficiently small $\alpha > 0$ and $\epsilon' < \alpha^2$ we have

$$\binom{k}{i}\binom{i}{\epsilon' k} \leq 2^k$$

for all i such that $|k/2 - i| \geq \alpha k$.

Proof. If suffices to prove the claim for $i = (\frac{1}{2} + \alpha)k$. By Stirling's Formula, we compute

$$\binom{k}{(\frac{1}{2}+\alpha)k}\binom{(\frac{1}{2}+\alpha)k}{\epsilon' k} = \left[\left[\frac{1}{\frac{1}{2}-\alpha}\right]^{(\frac{1}{2}-\alpha)}\left[\frac{1}{\epsilon'}\right]^{\epsilon'}\left[\frac{1}{\frac{1}{2}+\alpha-\epsilon'}\right]^{(\frac{1}{2}+\alpha-\epsilon')}\right]^k$$

(up to polynomial factors). Hence, setting $\epsilon' = \alpha^\beta$ with $\beta > 2$, our claim can be restated as

$$f(\alpha) := (\frac{1}{2} - \alpha)^{(\frac{1}{2}-\alpha)} (\alpha^\beta)^{\alpha^\beta} (\frac{1}{2} + \alpha - \alpha^\beta)^{(\frac{1}{2}+\alpha-\alpha^\beta)} \geq \frac{1}{2}.$$

Note that $f(0) = \frac{1}{2}$. Elementary calculus yields

$$f'(\alpha) = f(\alpha) \left[-ln(\frac{1}{2} - \alpha) + \beta^2 \alpha^{\beta-1} ln\alpha + (1 - \beta\alpha^{\beta-1}) ln(\frac{1}{2} + \alpha - \alpha^\beta) \right].$$

Let $g(\alpha)$ denote the term in brackets. Then $g(0) = 0$ (as $\lim_{\alpha\to 0} \alpha^{\beta-1} ln\alpha = 0$) and

$$\begin{aligned} g'(\alpha) = {} & (\tfrac{1}{2} - \alpha)^{-1} + \beta^2(\beta-1)\alpha^{\beta-2} ln\alpha + \beta^2\alpha^{\beta-2} - \beta(\beta-1)\alpha^{\beta-2} ln(\tfrac{1}{2} - \alpha - \alpha^\beta) \\ & + (1 - \beta\alpha^{\beta-1})^2 (\tfrac{1}{2} - \alpha - \alpha^\beta). \end{aligned}$$

Now $\beta > 2$ implies $\lim_{\alpha\to 0} \alpha^{\beta-2} ln\alpha = 0$, showing that $g'(\alpha) \approx 4 > 0$ for sufficiently small $\alpha > 0$. Hence also $g(\alpha)$ and $f'(\alpha) = f(\alpha)g(\alpha)$ are positive for sufficiently small values of α. So indeed $f(\alpha) \geq \frac{1}{2}$ for sufficiently small $\alpha > 0$. ■

Let us call - relative to a value of α to be determined below - a level i *critical* if $|k/2 - i| \leq \alpha k$, and *uncritical* otherwise. We modify the recursion (2) in ASC such that, as long a i is uncritical, we replace ϵ by $\epsilon' > \epsilon$, whereas for critical i, everything remains unchanged. Lemma 2 ensures that our upper bound on the running time of (2) remains unchanged:

$$\sum_{i \text{ uncritical}} \binom{\tilde{k}}{i} \binom{i}{\epsilon'\tilde{k}} + \sum_{i \text{ critical}} \binom{\tilde{k}}{i} \binom{i}{\epsilon\tilde{k}} \leq \tilde{k}\, 2^{\tilde{k}} \binom{\tilde{k}/2}{\epsilon\tilde{k}}.$$

Corresponding to the modified recursion, we investigate non-homogeneous subdivisions of the optimal tree T into (many) components of size at most $\epsilon' k$ and (a few) components of size at most ϵk. We first add $\frac{1}{\epsilon'}$ additional terminals to ensure component size of at most $\epsilon' k$. Now add such components (in some order), one at a time, until the constructed subtree spans $(\frac{1}{2} - \alpha)\tilde{k}$ terminals.

At this point, we enter the *critical phase*. We keep adding $\epsilon'-$components until the current subtree has $(\frac{1}{2} + \alpha)\tilde{k}$ terminals. At this point the critical phase stops. The *critical subtree*, *i.e.*, the subtree of T that we added during the critical phase spans at most $2\alpha\tilde{k}$ (plus possibly $\epsilon'\tilde{k}$) terminals. We can subdivide this critical subtree into $\epsilon-$components with at most $(2\alpha/\epsilon)$ additional terminals. After the critical phase, we complete the optimal tree by adding

ϵ'−components, one at a time. This shows that, in order to make the modified recursion (2) work, it suffices to add

$$a \le 2\alpha/\epsilon + 1/\epsilon'$$

additional terminals.

Now choose $\rho < \frac{1}{2}$ close to $\frac{1}{2}$ and observe that $\alpha := \epsilon^\rho$ and $\epsilon' := \epsilon^\varsigma$ with $\varsigma = \frac{1}{2} + \rho$ satisfy the assumption of Lemma 2 (for sufficiently small $\epsilon > 0$). The number of necessary additional terminals is thus

$$a \le 2\epsilon^\rho/\epsilon + \frac{1}{\epsilon^\varsigma}.$$

Applying the same trick to ϵ' instead of ϵ, we can further reduce the necessary number of additional terminals to

$$a \le 2\epsilon^\rho/\epsilon + 2(\epsilon^\rho/\epsilon)^\varsigma + \frac{1}{\epsilon^{\varsigma^2}}.$$

Continuing this way, we arrive at

Proposition For every $\rho < 1/2$, the number of necessary additional terminals can be reduced to $O(\epsilon^{\rho-1})$.

Proof. Since $\varsigma = \frac{1}{2} + \rho < 1$, there exists a constant $r > 0$ such that $\varsigma^r \le 1/2$. Hence if we apply our trick $(r-1)$ times, we arrive at

$$a \le 2\left[\epsilon^\rho/\epsilon + (\epsilon^\rho/\epsilon)^\varsigma + \cdots + (\epsilon^\rho/\epsilon)^{\varsigma^{r-1}}\right] + \frac{1}{\epsilon^{\varsigma^r}} \;\le\; 2r\epsilon^\rho/\epsilon + \frac{1}{\sqrt{\epsilon}} = O(\epsilon^{\rho-1}).$$

References

[1] C. Gröpl, S. Hougardy, T. Nierhoff and H. J. Prömel, (2001) Approximation Algorithms for the Steiner Tree Problem in Graphs, in: *Steiner Trees in Industries*, X. Cheng and D.-Z. Du (eds.), Kluwer.

[2] S. E. Dreyfus and R. A. Wagner, (1972) The Steiner problem in graphs. *Networks*, 1, 195–207.

[3] U. Fößmeier and M. Kaufmann, (2000) On exact solutions for the rectilinear Steiner tree problem Part 1: Theoretical results. *Algorithmica*, 26, 68–99.

[4] J. L. Ganley and J. P. Cohoon, (1994) Optimal rectilinear Steiner minimal trees in $O(n^2 2.62^n)$ time, in: *Proceedings of the Sixth Canadian Conference on Computational Geometry*, 308–313.

[5] M. Garey and D. Johnson, (1977) The Rectilinear Steiner Tree Problem is NP-Complete, *SIAM Journal on Applied Mathematics* 32(4), 826–834.

[6] F. K. Hwang, (1976) On Steiner Minimal Trees with Rectilinear Distance. *SIAM Journal on Applied Mathematics*, 30(1), 104–114.

[7] D. M. Warme, P. Winter and M. Zachariasen, (2000) Exact Algorithms for Plane Steiner Tree Problems: A Computational Study, in: *Advances in Steiner Trees*, D.Z. Du, J.M. Smith and J.H. Rubinstein (eds.), Kluwer.