Dynamic Programming for Minimum Steiner Trees

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

January 23, 2006

[a] University of Cologne, Center for Applied Computer Science Cologne, Group AFS, Weyertal 80, 50931 Köln, Germany
[b] University of Twente, Department of Applied Mathematics, Faculty of EEMCS, P.O.Box 217, 7500 AE Enschede, The Netherlands
[c] Computer Science Department, RWTH Aachen University, Germany

Abstract

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

1 Introduction

The *Steiner tree problem on graphs* is one of the best-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

*Supported by the Deutsche Forschungsgemeinschaft (DFG) under grant RO 927/6 (TAPI).

†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 (*e.g.*, [1]) and computational complexity, both from a theoretical and practical point of view ([3], [8]). 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 a minimum Steiner tree for X as well as the minimum Steiner tree itself.

The Dreyfus-Wagner algorithm recursively computes[1] $T(X \cup v)$ 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 cheapest path P_{vw} to an interior node $w \in V(T)$ of degree at least three. 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)

$$T(X \cup v) = \min\bigl(P_{vw} \cup T(X' \cup w) \cup T(X'' \cup w)\bigr),$$

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 two 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 of the critical part of the algorithm.

In this paper we develop a new algorithm that constructs an optimal solution from parts that have ony a small number of terminals. This is only

[1] We abbreviate $X \cup \{v\}$ by $X \cup v$ from now on.

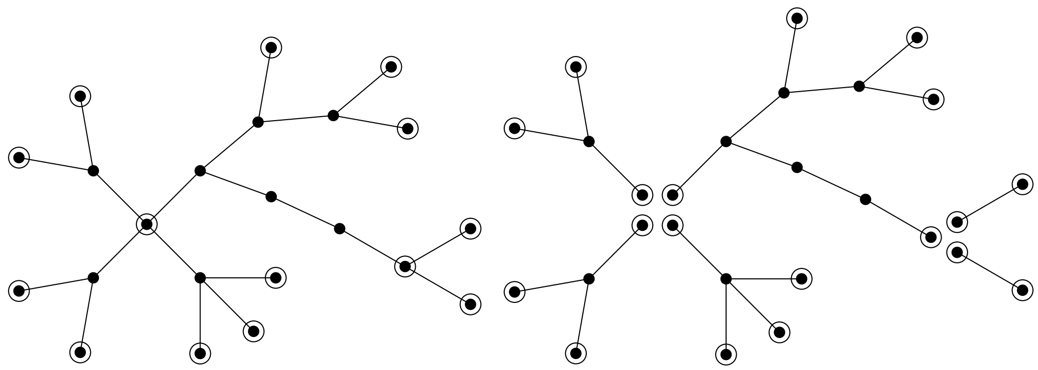

Figure 1: An optimum Steiner tree and its components

possible after adding additional terminals that split this optimal solution into small parts. Since the solution is unknown and $1/\varepsilon$ additional terminals are added, we have to try all $n^{1/\varepsilon}$ possibilities, which results in an extra polynomial factor. The first algorithm we present has running time $O\Big((2+\delta)^k n^{11\ln(\delta)/\delta} = O^*\left((2+\delta)\right)$ for any $1 < \delta < 2$. The second algorithm is more complicated and constructs the solution from parts of varying sizes thus needing less additional terminals. Its running time is then $= O\big((2+\delta)^k n^{12/\sqrt{\delta/\ln(1/\delta)}}\big)$ for sufficiently small δ.

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 1 *For $\varepsilon > 0$, adding at most $1/\varepsilon$ additional terminals suffices in order to split T into components of size at most $\varepsilon k + 1$ 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 of the components induced by u, including 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

$k_{u^*} \leq k - k\varepsilon$. Observe that u^* splits T into one large component of size k_{u^*} and one or more components of size (including u^*) at most $\varepsilon k + 1$ each. By induction, the large component can be split into components of size at most $\frac{\varepsilon}{1-\varepsilon} k_{u*} + 1 \leq \varepsilon k + 1$ with no more than $\frac{1-\varepsilon}{\varepsilon} = \frac{1}{\varepsilon} - 1$ additional terminals. $\square$

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

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

The following algorithm recursively composes 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 $\tilde{Y}, Y \subseteq \tilde{Y} \subseteq V, |\tilde{Y}| = k + \left\lfloor \frac{1}{\varepsilon} \right\rfloor$ do:

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

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

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

We can now establish our main results:

Theorem 1 *Algorithm ASC correctly computes a minimum Steiner tree for $Y \subseteq V$.*

Proof. Let T be a minimum Steiner tree for Y. Assume that $A \subseteq V(T)$ with $|A| = \lfloor 1/\varepsilon \rfloor$ has been added as a set of additional terminals, splitting $T = T(Y)$ into components of size at most $\varepsilon k + 1$. 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 T' of components, then T' must be a minimum Steiner tree for X_1—otherwise, T would not be optimal.

We show by induction on $|X|$ that the algorithm computes a minimum Steiner tree for $X \subseteq \tilde{Y}$ if X is the terminal set of a connected union T_X of components of T. Since $\tilde{Y}$ itself has this property, the correctness follows.

For $|X| \leq \varepsilon k + 1$ we can make use of the Dreyfus–Wagner algorithm and thus get optimal trees even for *all* such X.

Otherwise we can decompose X into $X = X_1 \bowtie X_2$, where X_1 and X_2 are terminal sets of connected components T_1 resp. T_2 of T. Moreover, $|X_2| \leq \varepsilon k + 1$. Then T_X is a minimum Steiner tree for X. By the induction hypothesis the algorithm computes minimum Steiner trees $T(X_1)$ and $T(X_2)$

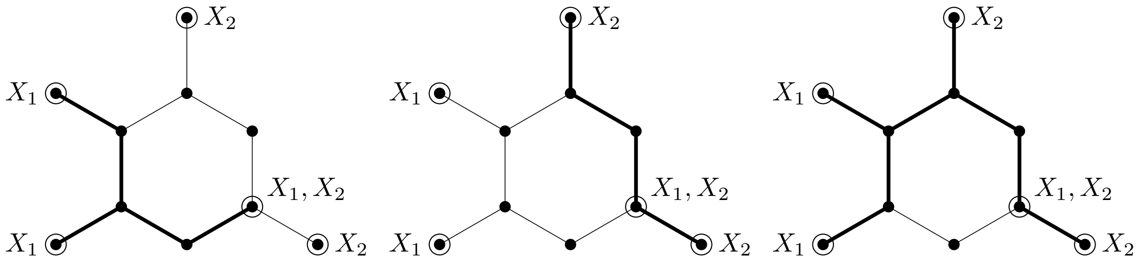

Figure 2: Two optimal Steiner trees for X_1, X_2 that do not combine to form an optimal Steiner tree for $X_1 \bowtie X_2$ (assuming unit-weight edges), compared to an optimal Steiner tree for $X_1 \bowtie X_2$

for X_1 and X_2. It is easy to verify that we can replace T_1 by $T(X_1)$ and T_2 by $T(X_2)$ in T_X to get another minimum Steiner tree for X. Obviously, this replacement yields $T(X_1) \cup T(X_2)$ as calculated by the algorithm. $\square$

Note that if $X = X_1 \bowtie X_2$ and $T(X_1)$, $T(X_2)$ are minimum Steiner trees for X_1 and X_2 then $T(X_1) \cup T(X_2)$ is in general *not* a minimum Steiner tree for X. Figure 2 contains a counterexample.

Theorem 2 *For $Y \subseteq V$ and $k = |Y|$ Algorithm ASC runs in time $O^*((2+\delta)^k)$ for any $0 < \delta < 1$ for an appropriate choice of $\varepsilon > 0$. In particular, the running time can be bounded by $O\big((2+\delta)^k n^{-11\ln(\delta)/\delta}\big)$.*

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

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

By a variant of Stirling's formula, $n! = \sqrt{2\pi n}(n/e)^n \theta$ with $1 < \theta < 1.1$, the binomial coefficient in (3) can be bounded by

$$\binom{\tilde{k}/2}{\varepsilon \tilde{k}} \leq \left[\left(\frac{1}{2\varepsilon}\right)^{\varepsilon} \left(\frac{1}{1-2\varepsilon}\right)^{\frac{1}{2}-\varepsilon} \right]^{\tilde{k}}.$$

Since we desire a running time of the form $O^*((2+\delta)^k)$, let us choose $\varepsilon = \frac{5}{18}\delta / \ln \frac{1}{\delta^3}$. Then $\delta^3 < 2\varepsilon$ for $\delta < 1$ and

$$\varepsilon \ln \frac{1}{2\varepsilon} \leq \frac{5}{18}\delta \leq \frac{\delta}{3} - \frac{\delta^2}{18} \leq \ln(1 + \frac{\delta}{3})$$

using Taylor expansion.[2] Exponentiating yields $(1/2\varepsilon)^\varepsilon \leq 1 + \delta/3$. On the other hand,

$$\left(\frac{1}{1-2\varepsilon}\right)^{1/2-\varepsilon} \leq 1+\varepsilon \leq 1+\delta/8,$$

because $\ln(1+x) \geq x - x^2/2$ for $0 < x \leq 1$ and consequently

$$\left(\frac{1}{2}-\varepsilon\right)\ln\left(\frac{1}{1-2\varepsilon}\right) \leq \left(\frac{1}{2}-\varepsilon\right)(2\varepsilon - 2\varepsilon^2) \leq \varepsilon - \varepsilon^2/2 \leq \ln(1+\varepsilon).$$

Hence

$$\left(\frac{1}{2\varepsilon}\right)^\varepsilon \left(\frac{1}{1-2\varepsilon}\right)^{\frac{1}{2}-\varepsilon} \leq (1+\delta/3)(1+\delta/8) \leq 1+\delta/2,$$

which lets us bound the running time at $O\big((2+\delta)^k n^{-\frac{54\ln\delta}{5\delta}}\big)$. $\square$

Remark 1 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 [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.

This structure of components in the rectilinear case is known as *Hwang topology* [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 that is 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

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

takes time

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

[2] For $\delta \leq 1/8$ we might take $\varepsilon = \frac{5}{18}\delta/\ln\frac{1}{\delta^2}$ or even smaller to get a running time of $O\big((2+\delta)^k n^{-\frac{36}{5}\ln\delta/\delta}\big)$, but in order to ease the analysis we exchange tightness for generality.

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 $n^{\frac{1}{\varepsilon}}$.

Remark 2 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 $n^{1/\varepsilon}$ to $n^{O(1/\sqrt{\varepsilon})}$ for sufficiently small δ and $\varepsilon = \frac{5}{36}\delta/ln(1/\delta)$. 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 $\varepsilon' < \alpha^2$ we have*

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

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

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

Hence, setting $\varepsilon' = \alpha^\beta$ with $\beta > 2$, our claim can be restated as

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

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

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

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

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

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 $\alpha > 0$. $\square$

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 as i is uncritical, we replace ε by $\varepsilon' > \varepsilon$, whereas for critical i, everything remains unchanged. Lemma 2 ensures that our upper bound on the running time of (2) remains unchanged:

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

Corresponding to the modified recursion, we investigate non-homogeneous subdivisions of the optimal tree T into (many) components of size at most $\varepsilon' k$ and (a few) components of size at most εk. We first add $\frac{1}{\varepsilon'}$ additional terminals to ensure a component size of at most $\varepsilon' k + 1$. 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 $\varepsilon'-$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 $\varepsilon'\tilde{k}$ terminals). We can subdivide this critical subtree into $\varepsilon-$components with at most $(2\alpha/\varepsilon)$ 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

$$a \leq 2\alpha/\varepsilon + 1/\varepsilon'$$

additional terminals.

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

$$a \leq 2\varepsilon^\rho/\varepsilon + \frac{1}{\varepsilon^\varsigma}.$$

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

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

Continuing this way, we arrive at

Proposition 1 *The number of necessary additional terminals can be reduced to* $4.38\varepsilon^{-1/2}$.

Proof. Let $r = -1/\ln\varepsilon$ and $\zeta = \frac{1}{2} + \rho$. If we apply our trick $(r-1)$ times, we arrive at

$$a \leq 2\left[\varepsilon^\rho/\varepsilon + (\varepsilon^\rho/\varepsilon)^\varsigma + \cdots + (\varepsilon^\rho/\varepsilon)^{\varsigma^{r-1}}\right] + \frac{1}{\varepsilon^{\varsigma^r}}.$$

Since

$$\frac{\left(\varepsilon^{\rho-1}\right)^{\zeta^i}}{\left(\varepsilon^{\rho-1}\right)^{\zeta^{i+1}}} = \left(\frac{1}{e}\right)^{(1/2-1/\ln\varepsilon)(1-1\ln\varepsilon)^i} \leq \left(\frac{1}{e}\right)^{1/2e},$$

we have

$$\begin{aligned}\varepsilon^\rho/\varepsilon + (\varepsilon^\rho/\varepsilon)^\varsigma + \cdots + (\varepsilon^\rho/\varepsilon)^{\varsigma^{r-1}} &\leq \sum_{i=0}^{\infty} \frac{\varepsilon^\rho}{\varepsilon}\left(\frac{1}{e}\right)^{k/2e} = \\ &= \frac{\varepsilon^\rho}{\varepsilon}\frac{1}{1-e^{-1/2e}} = \frac{1}{\sqrt{\varepsilon}e(1-e^{-1/2e})} < \frac{2.19}{\sqrt{\varepsilon}}\end{aligned}$$

and a short calculation shows $1/\varepsilon^{\zeta^r} < 1$ (so nothing more is required after $r-1$ recursive applications of the refinement trick). This means that $a \leq 4.38/\sqrt{\varepsilon}$ additional terminals suffice. □

The overall running time to compute optimal Steiner trees then becomes

$$O((2+\delta)^k n^{4.38/\sqrt{\varepsilon}}) = O\big((2+\delta)^k n^{12/\sqrt{\delta/\ln(1/\delta)}}\big).$$

Historical Note. Since 1972 there had been no improvement on the classical Dreyfus–Wagner algorithm for the Steiner tree problem. Then, in 2004, at least four groups began work on this problem independently. Two of them—one in Aachen and one in Enschede and Cologne—eventually found algorithms that improved the running time to $O^*\big((2+\varepsilon)^k\big)$ [7] and $O^*(2.68^k)$. They met in Dagstuhl at the Workshop on Parameterized Complexity and Exact Algorithms in 2005, joining their work thereafter.

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. Mölle, S. Richter, and P. Rossmanith. A faster algorithm for the Steiner tree problem. In Proc. of 23rd STACS. 2006.

[8] 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.