Approximation Algorithms for Bandwidth Problems on some large Graph Classes

Jürgen Wirtgen*

Abstract

The *bandwidth problem* is the problem of numbering the vertices of a given graph G so that the maximum difference between the numbers of adjacent vertices is *minimal*. The *topological bandwidth problem* is a natural extension of the bandwidth problem. It is the problem of numbering the vertices of a homeomorphic image of a given graph G so that the maximum difference between the numbers of adjacent vertices is *minimal*, over all numberings and images. Both problems have a long history and they are known to be NP-hard [Pa 76], [MPS 85].

In this paper we present the first $PTAS$ for the topological bandwidth of trees. Furthermore we construct n^ϵ-approximation algorithms for the bandwidth of graphs with minimum degree n^δ, for any $\delta, \epsilon > 0$.

*Dept. of Computer Science, University of Bonn, 53117 Bonn. Research partially supported by the ESPRIT BR Grants 7097 and EC-US 030. Email: wirtgen@cs.bonn.edu

1 Introduction

Graph layout problems are a collection of simple graph problems, motivated as models for VLSI layout problems: given a set of modules, in VLSI we have the problem to place these modules on a board in a non overlapping manner and wiring the terminals on the different modules according to a given wiring specification and in such a way that the wires do not interfere among them. In the modification as a graph problem we have an input graph $G = (V, E)$ on n vertices. A layout or numbering of G is a one-to-one mapping $f : V \rightarrow \{1, ..., n\}$. Some well known layout problems are the following:

Given a graph $G = (V, E)$, find a layout f that

Bandwidth Problem: minimizes the maximum of $|f(u) - f(v)|$ over all $\{u, v\} \in E$.

Topological Bandwidth Problem: minimizes the bandwidth of G' over all G' that are a homeomorphic image of G.

Minimum Cut Problem: minimizes the maximum over all $i, 1 \leq i \leq n$, of the number of edges that cross over i.

In this paper we study the *bandwidth* and the *topological bandwidth problem*. Despite their very long history and their technical importance, it is not much known about efficient approximability of these problems.

Formally, the bandwidth minimization problem is defined as follows. Let $G = (V, E)$ be a simple graph on n vertices. A numbering (layout) of G is a one-to-one mapping $f : V \rightarrow \{1, ..., n\}$. The *bandwidth* $B(f, G)$ of this numbering is defined by

$$B(f, G) = \max\{|f(v) - f(w)| : \{v, w\} \in E\},$$

the maximal distance between adjacent vertices in G corresponding to f.

The bandwidth $B(G)$ is then

$$B(G) = \min\{B(f, G) : f \text{ is a layout of } G\}$$

Clearly the bandwidth of G is the maximal bandwidth of its components. Therefore, we assume without loss of generality that the input graph is connected. There is also a reformulation of the bandwidth problem in the manipulation of sparse matrices. Let $A = (a_{ij})$ be a matrix. One can define the graph $G(A)$ to be the graph with the adjacency matrix A: $a_{ij} \neq 0$ if and only if $\{i, j\} \in E(G(A))$. $G(A)$ has bandwidth b if and only if there is a permutation matrix P such that in PAP^t all nonzero entries appear within b of the main diagonal. In this case A has bandwidth b.

The problem of constructing the bandwidth of a graph is NP-hard [Pa 76], even for trees with maximum degree 3 [GGJK 78]. There are only few cases where we can construct the optimal layout in polynomial time. Saxe [Sa 80] designed an algorithm which decides whether a given graph has bandwidth at most k in time $O(n^{k+1})$ by dynamic programming. His algorithm can be turned into a construction algorithm of the optimum layout. Bandwidth two can be checked in linear time [GGJK 78]. Smithline [Sm 95] (see also [Ch 88]) proved that the bandwidth of the complete k-ary tree $T_{k,d}$ with d levels and k^d leaves is exactly $\lceil k(k^d-1)/(k-1)(2d) \rceil$. The proof is constructive and entails a polynomial time algorithm for this problem.

The *topological bandwidth* is a natural generalization of the bandwidth. A graph G' is said to be a *homeomorphic image* of a graph G if G' can be obtained from G by subdividing edges in G with an arbitrary number of degree two vertices. The topological bandwidth of G is formally defined by

$$TB(G) = \min_{G' \text{ is a homeomorphic image of } G} B(G')$$

This problem is also known to be NP-complete [MPS 85]. For some special cases it is solvable in polynomial time [MPS 85]: binary trees in time $O(n \log n)$, topological bandwidth k can be checked in time $O(n^k)$ (See Theorem 2.1), topological bandwidth 2 in linear time. As the bandwidth the topological bandwidth has also an interesting sparse matrix interpretation. Let A be a matrix arising from a linear system $a_i x = b_i$. It is possible that the bandwidth of this matrix is quite large, meaning that there is no permutation matrix P such that PAP^t has all its nonzero entries close to its diagonal. To reduce the bandwidth we may replace a term $a_{ij}x_j$ by a new variable y and add a new equation of the form $a_{ij}x_j = y$ which has the same effect as adding a degree 2 vertex into the edge $\{i, j\}$ of $G(A)$.

The *minimum cut problem* is defined as follows. Let f be a layout. The cutwidth $cw(f, G)$ of this layout for G is

$$cw(f, G) = \max_{i=1,\ldots,n} |\{\{v, w\} \in E(G) | f(v) \leq i < f(w)\}|.$$

The *modified cutwidth* $mcw(f, G)$ of this layout for G is

$$mcw(f, G) = \max_{i=1,\ldots,n} |\{\{v, w\} \in E(G) | f(v) \leq i \leq f(w)\}|.$$

The (modified) cutwidth of G is defined as

$$cw(G) = \min_f cw(f, G) \text{ and } mcw(G) = \min_f mcw(f, G).$$

The problem is NP-complete, even for planar graphs with maximum degree 3 [Ga 77] but in polynomial time solvable for trees [Ya 85] (See Theorem 2.6).

The design of approximation algorithms for NP-hard optimization problems became an important field of research in the last decade. In the best of situations we are able to find approximation algorithms which work in polynomial time and approximate optimal solutions within an arbitrary given constant. Such (meta-) algorithms are called polynomial time approximation schemes ($PTAS$s), cf.eg., [Ho 97].

In this paper we present the first $PTAS$ for the topological bandwidth of trees. Furthermore we construct n^ϵ-approximation algorithms for the bandwidth of graphs with minimum degree n^δ, for any $\delta, \epsilon > 0$. For the general bandwidth problem there are two approximation algorithms known:

Feige [Fe 98] An $O(\log^{\lceil 11/2 \rceil} n)$-approximation algorithm and

Blum, Konjevod, Ravi and Vempala [BKRV 98] an $O(\sqrt{n/B(G)} \log n)$-approximation algorithm.

While the approximation-ratio of our algorithm is not as good as the approximation ratio of [Fe 98], for graphs with minimum degree n^δ it will be better than the approximation ratio of [BKRV 98].

It is a very surprising fact that there are $PTAS$s for the topological bandwidth problem on trees, since there is no such algorithm possible for the related bandwidth problem on trees. Blache, Karpinski and Wirtgen [BKW 98] showed that there is no 4/3-approximation algorithm for the bandwidth problem, unless $P = NP$.

The paper is organized as follows. In Section 2 we develop an approximation scheme for the topological bandwidth of trees, by relating the topological bandwidth to the cutwidth. In Section 3 we use random sampling techniques of Dessmark, Dorgerloh, Lingas and Wirtgen [DDLW 98] to construct spanner like trees. By using an approximation algorithm for the Bandwidth of $h(k)$-trees [HM 97] we get good approximations for the bandwidth on graphs with guaranteed minimum degree.

2 A $PTAS$ for the Topological Bandwidth Problem on Trees

For a constant k it is possible to check whether the topological bandwidth of a given graph is k or not. This is done by a modification of the dynamic programming algorithm given in [GS 84] and [Sa 80], which shows that deciding whether a graph G with n vertices has bandwidth $k \in O(1)$ can be checked in time $O(n^k)$.

Theorem 2.1 ([MPS 85]) *For all $k \geq 3$, it is possible to recognize graphs with topological bandwidth k in time $O(n^k)$. For $k = 2$, it can be done in linear time. We can construct such a layout in the same time bounds.*

We use this result in our approximation algorithm for the case that the bandwidth of the input graph is small.

2.1 Cutwidth and Topological Bandwidth of Trees

In this section we relate the topological bandwidth problem to the cutwidth problems.

Lemma 2.2 ([MPS 85]) *For any graph G we have*

$$cw(G) \geq TB(G).$$

Especially if G is a tree.

PROOF: We show how to construct for any cutwidth layout f of G a layout f' and a homomorphic image G' of G, such that

$$cw(f, G) \geq B(f', G').$$

Choose a subgraph G_1 of G as follows

1. $C = \emptyset$
2. Choose an edge $\{u, v\}$ such that $f(u) \leq f(v)$ and u is the smallest vertex with $f(u) \geq f(w)$ for any vertex w which is contained in an edge of C. Put $\{u, v\}$ into C. If there is no such edge, stop.

Clearly, the graph $G_1 = G[C]$ has cutwidth 1. Also the graph $G - G_1$ satisfies $cw(f, G - G_1) = cw(f, G) - 1$. Assume, that i is the least number with

$$|\{\{u, v\} \in E(G - G_1) | f(u) \leq i < f(v)\}| = cw(f, G).$$

The edges $\{u, v\}$ in G with $f(u) \leq i < f(v)$ are not in G_1. From the second step we know that there is no edge $\{u, v\}$ in G with $f(u) = i < f(v)$. Thus there are $cw(f, G)$ edges $\{u, v\}$ with $f(u) < i < f(v)$. Now we have

$$|\{\{u, v\} \in E(G - G_1) | f(u) \leq i - 1 < f(v)\}| = cw(f, G).$$

This is a contradiction to the minimality of i.

Repeat the above process to partition G into graphs $G_1, G_2, ..., G_{cw(f,G)}$, such that $cw(f, G_i) = 1$ for all $i = 1, ..., cw(f, G)$. Construct the refinement G' of G and the layout f' of G' as follows. For any edge $\{u, v\}$ in G_i with $f(u) < f(v)$, subdivide $\{u, v\}$ into a path $u = u_0, u_1, ..., u_t = v$ of $f(v) - f(u) + 1$ vertices. Define

$$f'(u_j) = cw(f, G)(f(u) + j) + i - 1 \quad j = 0, ..., f(v) - f(u).$$

Clearly f' is a proper layout. It easy to see, that $B(f', G') = cw(f, G)$. Thus we have $cw(G) \geq TB(G)$. ■

Corollary 2.3 *For any cutwidth layout f of a graph G, we can construct in polynomial time a homeomorphic image G' and layout f' of G', such that*

$$cw(f, G) \geq B(f', G').$$

PROOF: We have to check the running time of the algorithm included in Theorem 2.2. Let be n the number of vertices of G.

$cw(f, G)$ is at most $O(n^2)$. Thus we have at most $O(n^2)$ graphs G_i. For each graph we have at most $O(n^2)$ iterations of cost $O(n^2)$. Therefore the construction of G' and f' is polynomial in n. ■

Makedon et al. improved this bound by relating the topological bandwidth to the modified cutwidth

Lemma 2.4 ([MPS 85])

$$TB(G) \leq mcw(G) + 1$$

Their proof gives a polynomial time algorithm, which finds for any modified cutwidth layout f of a graph G, a homeomorphic image G' and layout f' of G', such that

$$B(f', G') \leq mcw(G) + 1.$$

However, this gives the same bound as in Theorem 2.2, since $mcw(G) \leq cw(G) - 1$.

Theorem 2.5 ([Ch 85]) *For any tree T the following inequality holds:*

$$TB(T) \leq cw(T) \leq TB(T) + \log TB(T) + 2.$$

For trees the modified cutwidth problem is polynomial solvable.

Theorem 2.6 ([Ya 85]) *There is an $O(n \log n)$ time algorithm, which finds for any tree T a layout f, such that $mcw(f, T) = mcw(T)$.*

2.2 The Scheme

Theorem 2.7 *There is a PTAS for the topological bandwidth problem on trees.*

PROOF: Let be T the input tree with n vertices and $(1+\epsilon)$ the desired approximation ratio.

First we run the algorithm of Theorem 2.6 to construct in polynomial time a layout f, such that $mcw(f,T) = mcw(T)$. Using the Algorithm of Lemma 2.4, we get a layout f' for a homeomorphic image T' of T, such that

$$\begin{aligned} B(f',T') &\leq mcw(T)+1 \quad \text{(Lemma 2.4)} \\ &\leq cw(T) \\ &\leq TB(T) + \log TB(T) + 2 \quad \text{(Theorem 2.5)} \\ &= (1 + \frac{\log TB(T)+2}{TB(T)})TB(T) \end{aligned}$$

Observe that for $\frac{\log TB(T)+2}{TB(T)} \leq \epsilon$, we have the desired approximation bound. It is easy to see, that the solution of $\frac{\log t+2}{t} \leq \epsilon$ is a constant $t(\epsilon)$ only dependent of ϵ.

Run the algorithm of Theorem 2.1 to check in time $O(n^{t(\epsilon)})$, whether the tree T has topological bandwidth at most $t(\epsilon)$ or not. If not, $TB(f',T') \leq (1+\epsilon)TB(T)$. Otherwise we construct with this algorithm in the time $O(n^{t(\epsilon)})$ such a layout.
■

3 An Approximation Algorithm for Graphs with Guaranteed Minimum Degree

First we state some Chernoff like bounds, which can be found in standard books (See e.g. [H 64] [MR 95]).

Lemma 3.1 *Let be X_i, $i = 1, ..., k$ independent random variables such that $0 \leq X_i \leq 1$. Let $X = \sum_i X_i$ and $\mu = E[X]$ then*

(a) $\Pr[|X-\mu| > \mu] < 2\exp(-\mu/3)$.

(b) $\Pr[X < (1-\delta)\mu] < \exp(\mu\delta^2/2)$.

We need for our approximation algorithm on *non sparse graphs* as a subroutine an approximation algorithm for a class of trees, the so called $h(k)$-trees: given any tree vertex v, the depth depth difference of any two nonempty subtrees rooted at v is bounded by a constant k. Formally a tree T is a *generalized height balanced-* or $h(k)$-tree, whenever:

1. k is a nonnegative integer, and
2. if a vertex v of T has more than one subtrees, then for any two such subtrees, $|d_i - d_j| \leq k$, where d_i, d_j denote the respective depths of the subtrees T_i, T_j.

For any $h(k)$-tree Haralamides and Makedon [HM 97] presented an $O(\log d)$-approximation algorithm, where d is the depth of the input tree.

Theorem 3.2 ([HM 97]) *Let be T a $h(k)$-tree with depth d, for some constant k. There is a polynomial time $O(\log d)$-approximation algorithm for the bandwidth of T.*

A basic tool in our algorithm is the random sampling of vertices of the input graph. We restrict ourselves to graphs with a guaranteed minimum degree of n^δ for some delta. In these graphs a *small* random vertex subset suffices to construct a dominating set.

Lemma 3.3 *Let $G = (V, E)$ be a graph with minimum degree $d(n)$. A set of*

$$k = \Theta(\log(n)n/d(n))$$

randomly chosen vertices R forms a dominating set with high probability.

Clearly if the minimum degree is in $\Theta(n)$, then k can be chosen to be $\Theta(\log n)$ ([KWZ 97], [DDLW 98]). Such graphs are called *dense.*

PROOF: The probability, that one particular vertex v will be dominated by one of the randomly chosen vertices, is at least $d(n)/n$. Define the random variable $X_{v,i}$ to be 1, if the ith random vertex dominates this vertex and

$$\begin{aligned} X_v &= \sum_i^k X_{v,i} \\ \mu_v &= E[X_v] \\ &= \Omega(\log n) \end{aligned}$$

Using Lemma 3.1 (a) we get

$$\Pr[\mu_v < 1] < 1/\omega(n)$$

Using Lemma 3.1 (b) we get that with high probability each vertex will be dominated. ∎

If we choose a small sample set R in a dense graph, $G[R]$ will be again dense with high probability.

Lemma 3.4 *Let $G = (V, E)$ be a δ-dense graph and $R \subset V$ be a set with $|R| = \beta \log n$. Then $G[R]$ is $(1-\gamma)\delta$-dense with high probability.*

PROOF: We define the random variable X_v for $v \in V$ to be the number of neighbors of v in R. Then $\mathrm{E}[X_v] \geq c \log n$, where $c = \delta\beta$. Now we bound the probability that X_v deviates far from its expectation by applying the Chernoff bound (Lemma 3.1 (b)).

$$\begin{aligned} \Pr[X_v < (1-\gamma)\mathrm{E}[X_v]] &< \exp(-\mathrm{E}[X_v]\gamma^2/2) \\ &\leq 1/\exp(c \log n \gamma^2/2) \\ &= 1/2^{c \log n \log e \gamma^2/2} \\ &= 1/n^{c \log e \gamma^2/2} =: p_1(n). \end{aligned}$$

Define Y_v as

$$Y_v := \begin{cases} 0 & \text{if } X_v \geq (1-\gamma)\mathrm{E}[X_v] \\ 1 & \text{otherwise.} \end{cases}$$

and set $Y = \sum_{v \in R} Y_v$. Then the probability that a vertex $v \in R$ has fewer than $(1-\gamma)c \log n$ neighbors in R is at most $\Pr[Y > 1]$. By applying Markov's inequality, we obtain that

$$\begin{aligned} \Pr[Y \geq 1] &\leq \mathrm{E}[Y] \\ &= \sum_{v \in R} \underbrace{\mathrm{E}[Y_v]}_{<p_1(n)} \\ &< p_1(n)\beta \log n \\ &= \frac{\beta \log n}{n^{c \log e \gamma^2/2}} =: p_2(n). \end{aligned}$$

Thus $G[R]$ is $(1-\gamma)\delta$-dense with probability $1 - p_2(n) = 1 - o(1)$.

■

We can generalize this lemma as follows.

Lemma 3.5 *Let $G_0 = G = (V, E)$ be a graph with minimum degree $d_0(n) = n^\delta$ $(\delta > 0)$. A set of*

$$n_1 = \Theta(n^{\delta'+1-\delta}) \quad (\delta' = \frac{\delta(1-\delta)}{2(1-1/2\delta)})$$

randomly chosen vertices R forms a dominating set with high probability. Furthermore $G_1 = G[R]$ has with high probability a minimum degree of $d_1(n_1) = \Omega(n_1^{\delta/2})$.

PROOF: Clearly R is a dominating set (Lemma 3.3). We define again a random variable X_v for $v \in V$ to be the number of neighbors of v in R.

$$\mathrm{E}[X_v] = \sum_i^{n_1} \frac{|N(v)|}{n} \geq n^{\delta'} \frac{n}{d_0(n)} \frac{d_0(n)}{n} = n^{\delta'}$$

By Lemma 3.1 (b) we get

$$\Pr[X_v < (1-\gamma)\mathrm{E}[X_v]] < 1/\exp(O(n^{\delta'}))$$

Thus each vertex in $G[R]$ has with high probability at least

$$\begin{aligned} d_1(k) = n_1^{\delta/2} &= (n^{\delta'+1-\delta})^{\delta/2} \\ &= n^{\delta/2(\frac{\delta(1-\delta)}{2(1-1/2\delta)}+(1-\delta))} \\ &= n^{\delta/2(1-\delta)(\frac{\delta+2-\delta}{2(1-1/2\delta)})} \\ &= n^{\frac{\delta(1-\delta)}{2(1-1/2\delta)}} \\ &= n^{\delta'} \end{aligned}$$

neighbors. ■

Let $G_0 = G = (V, E)$ be a connected graph with minimum degree $d_0(n) = n^\delta$ $(\delta > 0)$. Let be k the smallest integer satisfying

$$\left(1 - \frac{\delta^2}{4^k}\right)^k \leq \epsilon \tag{1}$$

Clearly $k = k(\epsilon, \delta)$ is for fixed ϵ and δ a constant.

Consider the following algorithm

Step 1: Set $i = 0$, $n_0 = n$, $G_0 = G$, $\delta_0 = \delta$.

Step 2: Choose a random subset $R_i \subset V(G_i)$ of size $n_{i+1} = n_i^{\frac{\delta_i(1-\delta_i)}{2(1-1/2\delta_i)}+1-\delta_i}$.

Step 3: Set $i = i+1$, $G_i = G_{i-1}[R_{i-1}]$, $\delta_i = \delta_{i-1}/2$.

Step 4: If $i \leq k$, goto 2.

Lemma 3.6 *G_{k+1} has at most n^ϵ vertices.*

PROOF: We have following inequality:

$$\begin{aligned}
\underbrace{\frac{\delta_i(1-\delta_i)}{2(1-\underbrace{1/2\delta_i}_{\leq 1/2})}}_{\geq 1} + 1 - \delta_i &\leq \delta_i(1-\delta_i) + 1 - \delta_i \\
&\leq 1 - \delta_i^2
\end{aligned}$$

Thus we have

$$\begin{aligned}
n_1 &\leq n_0^{1-\delta_0^2} \\
n_2 &\leq n_1^{1-\delta_1^2} \leq n_0^{(1-\delta_0^2)(1-\delta_1^2)} \\
&\ldots
\end{aligned}$$

This gives us

$$\begin{aligned}
n_{k+1} &\leq n^{\Pi_{i=0}^{k}(1-(\delta/2^i)^2)} \\
&\leq n^{(1-(\delta/2^k)^2)^k} \\
(1) &\leq n^\epsilon
\end{aligned}$$

■

Lemma 3.5 ensures, that the minimum degree of G_i is $|V(G_i)|^{\delta/2^i}$. Thus the random chosen vertices R_i of each iteration build a dominating set in the graph G_i.

Step 5: Find spanning trees $T_1, ..., T_c$ for the components of G_{k+1}. Set $i = k$.

Step 6: For each vertex v in $V(G_i)$ find a neighbor w in R_i. If w belongs to T_j augment T_j by the edge $\{v, w\}$.

Step 7: Set $i = i - 1$. If $i \geq 0$, goto 6.

Step 8: For each tree T_i we choose one vertex t_i of this tree as its representant (See Figure 1).

Step 9: Set $E_{connect} = \{\{t_1, t_i\} | i = 2, ..., c\}$.

Step 10: $T = \bigcup_i^c T_i \cup E_{connect}$ (See Figure 2).

Clearly T is connected and the depth of T is $O(n^\epsilon)$, since the depth of each T_i $(i = 1, ..., c)$ is $O(n^\epsilon)$.

In the next step we modify T to get a $h(0)$-tree:

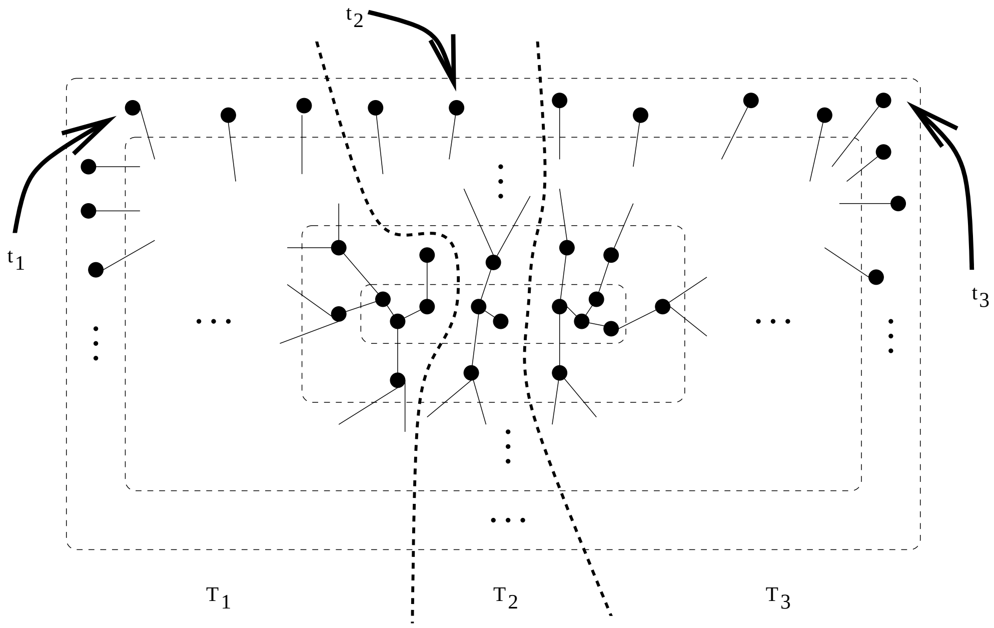

Figure 1: The Forest of **Step** 8 with rooted trees $T_1, ..., T_c$ ($c \in n^{1-\delta}$).

Step 11: Let be h the height of T. For each leaf l of T we denote by h_l the distance of l to t_1 in T. We replace the edge $\{l, a_l\}$ in T by a path of length $h - h_l + 1$. This gives us T'.

Lemma 3.7 *T' is a $h(0)$-tree.*

PROOF: By the construction, the distance of any leaf to t is the same. ■

Lemma 3.8 *The bandwidth $B(T)$ of T is at most $O(n^{2\epsilon}B(G))$.*

PROOF: Suppose we have some layout f of G with bandwidth $B(G)$. Clearly

$$B(f, T \setminus E_{connect}) \leq B(G)$$

If we insert in the layout f for each leaf l of T the vertices of the corresponding path just before l (See Figure 3), we get a layout f', such that

$$B(f', T' \setminus E_{connect}) \leq O(n^{\epsilon}B(G)),$$

since the length of this path is $O(n^{\epsilon})$.

For each vertex $v \in V(G)$ there is some tree $T_{i(v)}$ to which it belongs to. Thus there is some representant $t_{i(v)}$. Since the depth of each tree is at most $O(n^{\epsilon})$, the distance of v and $t_{i(v)}$ is at most $O(n^{\epsilon})$. Thus the distance for connected vertices is at most $O(n^{\epsilon})$. This gives us

$$B(f', T') \leq O(n^{2\epsilon}B(G))$$

■

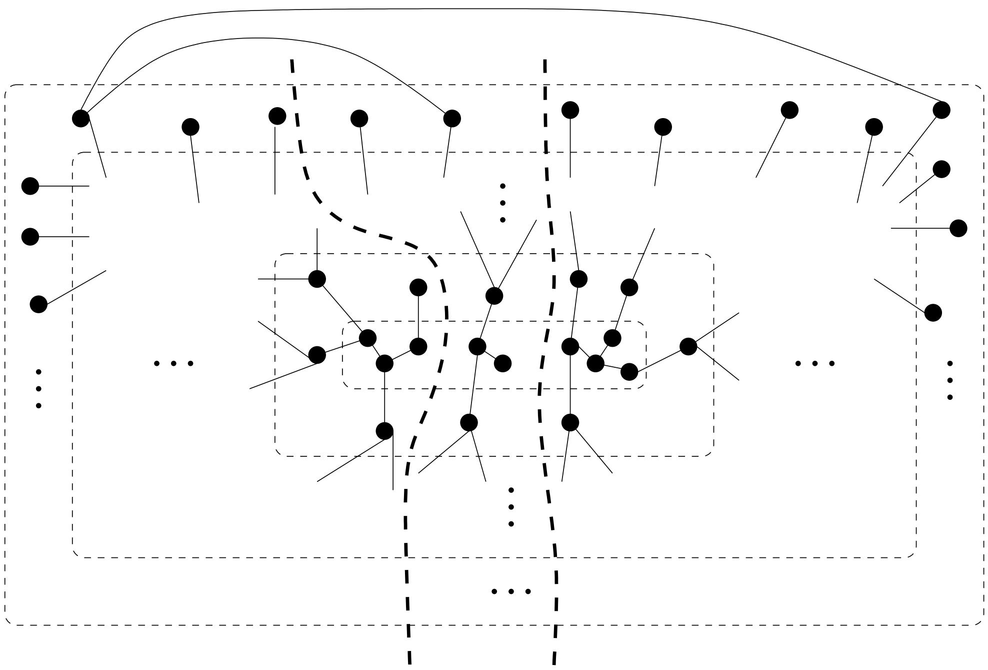

Figure 2: The Tree T of **Step** 10 build of the rooted trees $T_1, ..., T_c$.

Lemma 3.9 *Suppose an algorithm gives us a layout f' of T'. Then we can transform this layout to a layout f of G such that,*

$$B(f, G) \leq O(n^\epsilon B(f', T'))$$

PROOF: We get f by taking the same order for the vertices of T as in f'. Thus

$$B(f, G) \leq O(n^\epsilon B(f', T')),$$

since each edge of G is represented by a path of length $O(n^\epsilon)$ in T'. ■

Theorem 3.10 *Let $G_0 = G = (V, E)$ be a connected graph with minimum degree $d_0(n) = n^\delta$ ($\delta > 0$). For any $\epsilon > 0$ there is a polynomial time approximation algorithm for the bandwidth which finds a layout f such that*

$$B(f, G) \leq O(n^{3\epsilon} \log n) B(G).$$

PROOF: For given G and ϵ, run the above algorithm to get T'. Clearly this can be done in polynomial time. Now we get with the algorithm of Theorem 3.2 for $h(0)$-trees a layout f', such that

$$B(f', T') \leq O(\log n) B(T').$$

Using Lemma 3.8 and Lemma 3.9 we get a layout f, such that

$$\begin{aligned} B(f, G) &\leq O(n^\epsilon B(f', T')) \\ &\leq O(n^\epsilon \log n B(T')) \\ &\leq O(n^{3\epsilon} \log n B(G)) \end{aligned}$$

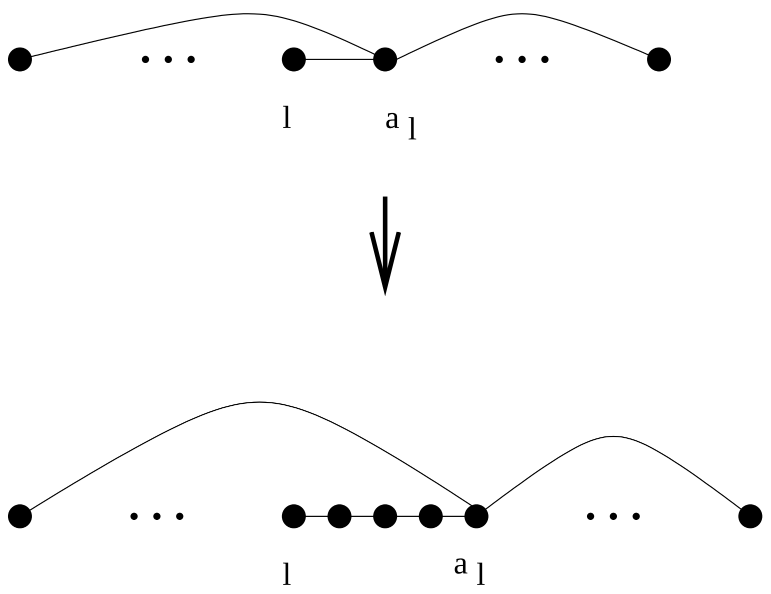

Figure 3: A possible layout for T'.

■

For dense graphs we can even improve this result, by using Lemma 3.4 and the results of [DDLW 98].

Theorem 3.11 *Let G be a dense graph. For any integer $k > 0$ there is a polynomial time algorithm which finds a layout f, such that*

$$B(f, G) \leq \log^{(k)} \log^{(k+1)} B(G),$$

where $\log^{(k)}$ is the k-times iterated logarithm ($\log^{(1)} = \log, \log^{(k+1)} = \log \log^{(k)}$)

This theorem gives a somewhat worse approximation ratio (a 3-approximation) for dense graphs, than given in [KWZ 97]. But the running time is significantly more practical.

Acknowledgment

I thank Gunter Blache, Carsten Dorgerloh and Marek Karpinski for a number of interesting discussions on the subject of this paper.

References

[BKW 98] Blache, G., Karpinski, M., Wirtgen, J., *On Approximation Intractability of the Bandwidth Problem*, Technical Report ECCC TR 98-014, 1998.

[BKRV 98] Blum, A., Konjevod, G., Ravi, R., Vempala, S., *Semi-Definite Relaxations for Minimum Bandwidth and other Vertex-Ordering Problems*, Proc. 30^{th} ACM STOC (1998), to appear.

[Ch 85] Chung, F., *On the Cutwidth and the Topological Bandwidth of a Tree*, SIAM Journal on Algebraic Discrete Methods **6**(2) (1985), pp. 268–277.

[Ch 88] Chung, F., *Labelings of Graphs*, Beineke, L., Wilson, R., ed, Selected Topics in Graph Theory, pp. 151–168, Acadamic Press, 1988.

[DDLW 98] Dessmark, A., Dorgerloh, C., Lingas, A., Wirtgen, J., *Ultrafast Randomized Parallel Construction- and Approximation Algorithms for Spanning Forests in Dense Graphs*, Proc. 3^{rd} Workshop on Randomized Parallel Computing (1998).

[Fe 98] Feige, U., *Approximating the Bandwidth via Volume Respecting Embeddings*, Proc. ACM STOC (1998), to appear.

[GGJK 78] Garey, M., Graham, R., Johnson, D., Knuth, D., *Complexity Results For Bandwidth Minimization*, SIAM J. Appl. Math. **34** (1978), pp. 477–495.

[Ga 77] Gavril, F., *Some NP-complete problems on graphs*, Proc. 11^{th} Annual Conference on Information Sciences and Systems (1977), pp. 91–95.

[GS 84] Gurari, E., Sudborough, I., *Improved Dynamic Programming Algorithms for Bandwidth Minimization and the MinCut Linear Arrangement Problem*, Journal of Algorithms **5** (1984), pp. 531–546.

[HM 97] Haralamides, J., Makedon, F., *Approximation Algorithms for the Bandwidth Minimization Problem for a Large Class of Trees*, Theory of Computing Systems **30** (1997), pp. 67–90.

[H 64] Höffding, W., *Probability inequalities for sums of bounded random variables*, Journal of the American Statistical Association (1964).

[Ho 97] Hochbaum, D., ed, *Approximation Algorithms for NP-hard Problems*, PWS Publ. Co., 1997.

[KWZ 97] Karpinski, M., Wirtgen, J., Zelikovsky, A., *An Approximation Algorithm for the Bandwidth Problem on Dense Graphs*, Proc. 1^{st} RALCOM (1997).

[MPS 85] Makedon, F., Papadimitriou, C., Sudborough, I., *Topological Bandwidth*, SIAM Journal on Algebraic Discrete Methods **6**(3) (1985), pp. 418–444.

[MR 95] Motwani, R., Raghavan, P., *Randomized Algorithms*, Cambridge University Press, 1995.

[Pa 76] Papadimitriou, C., *The NP-Completness of the Bandwidth Minimization Problem*, Computing **16** (1976), pp. 263–270.

[Sa 80] Saxe, J., *Dynamic Programming Agorithms for Recognizing Small-Bandwidth Graphs*, SIAM Journal on Algebraic Methods **1** (1980), pp. 363–369.

[Sm 95] Smithline, L., *Bandwidth of the Complete k-ary Tree*, Discrete Mathematics **142** (1995), pp. 203–212.

[Ya 85] Yannakakis, M., *A Polynomial Algorithm for the Min-Cut Linear Arrangement of Trees*, Journal of the ACM **32**(4) (1985), pp. 950–988.