A Tight Upper Bound on the Probabilistic Embedding of Series-Parallel Graphs*

Yuval Emek† David Peleg‡

Abstract

We prove that every unweighted series-parallel graph can be probabilistically embedded into its spanning trees with logarithmic distortion. This is tight due to an $\Omega(\log n)$ lower bound established by Gupta, Newman, Rabinovich, and Sinclair on the distortion required to probabilistically embed the $n$-vertex diamond graph into a collection of dominating trees. Our upper bound is gained by presenting a polynomial time probabilistic algorithm that constructs spanning trees with low expected stretch. This probabilistic algorithm can be derandomized to yield a deterministic polynomial time algorithm for constructing a spanning tree of a given (unweighted) series-parallel graph $G$, whose communication cost is at most $O(\log n)$ times larger than that of $G$.

---

* A preliminary version of this paper appeared in SODA 2006.
† School of Electrical Engineering, Tel Aviv University, Tel Aviv, 69978 Israel. E-mail: yuvale@eng.tau.ac.il. This work was done while the author was in the Weizmann Institute of Science.
‡ Department of Computer Science and Applied Mathematics, The Weizmann Institute of Science, Rehovot, 76100 Israel. E-mail: david.peleg@weizmann.ac.il. Supported in part by a grant from the Israel Science Foundation.
1 Introduction

The problem. Consider a connected $n$-vertex graph $G = (V(G), E(G))$. For any two vertices $u, v \in V(G)$, let $\text{dist}_G(u, v)$ denote the distance between $u$ and $v$ in $G$, namely, the length of a shortest path connecting them in $G$. Let $T$ be a spanning tree of $G$ and let $u$ and $v$ be two vertices in $V(G)$. The stretch of $u$ and $v$ in $T$, denoted $\text{str}_T(u, v)$, is the ratio of the distance between $u$ and $v$ in $T$ to their distance in $G$, i.e., $\text{str}_T(u, v) = \frac{\text{dist}_T(u, v)}{\text{dist}_G(u, v)}$.

Spanning trees achieving low stretch for every pair of vertices can be very useful in many applications. However, there exist some simple graphs for which in any spanning tree, there exists a pair of vertices with stretch $\Omega(n)$. This motivates the following definition introduced in [Bart96]. Let $D$ be a probability distribution over the spanning trees of $G$. We say that $G$ can be probabilistically embedded into its spanning trees (under $D$) with distortion $\alpha$ if the expected stretch of any two vertices is at most $\alpha$, that is, $\mathbb{E}[\text{str}_T(u, v)] \leq \alpha$ for every two vertices $u$ and $v$ in $V(G)$, where $T$ is chosen according to $D$. Given a graph $G$, our goal is to find a distribution $D$ over the spanning trees of $G$ that minimizes the distortion $\alpha$.

The support of a probabilistic embedding can be extended to dominating trees: a dominating tree of the graph $G$ is a tree $T$ that satisfies (1) $V(T) \supseteq V(G)$; and (2) $\text{dist}_T(u, v) \geq \text{dist}_G(u, v)$ for every two vertices $u, v \in V(G)$. Indeed, many works consider probabilistic embeddings into a collection of dominating trees that are not necessarily spanning trees of $G$ as they may contain vertices and edges that do not exist in $G$. However, in some applications, and in particular in settings where $G$ represents a physical network, relying on vertices or edges that do not exist in $G$ does not make any sense.

The deterministic form of the problem described above is essentially the minimum communication spanning tree (MCT) problem introduced in [Hu74]. Given a connected graph $G$ and a non-negative real communication requirement $c(u, v)$ for every pair of vertices $u$ and $v$ in $G$, the goal of the MCT problem is to find a spanning tree $T$ of $G$ that minimizes the communication cost of $T$, defined as

$$\text{com-cost}(T) = \sum_{u \neq v} c(u, v) \cdot \text{dist}_T(u, v).$$

A special case of the MCT problem is the minimum average stretch spanning tree (MAST) problem studied in [AKPW95, EEST08]. Given a graph $G = (V(G), E(G))$, the goal of the MAST problem is to find a spanning tree $T$ of $G$ that minimizes the average stretch of $T$, defined as

$$\text{av-str}(T) = \frac{1}{|E(G)|} \sum_{(u, v) \in E} \text{str}_T(u, v).$$

Using the von Neumann minimax principle of game theory, it is shown in [AKPW95] that a graph

---

1 Throughout this paper, the term “spanning tree” is reserved to subgraph spanning trees, namely, spanning trees whose edges occur in the graph itself.
$G$ can be probabilistically embedded into a its spanning trees with distortion $\alpha$ if and only if every multigraph obtained from $G$ by replicating its edges has a spanning tree with average stretch $\alpha$.

**Series-parallel graphs.** Given a (not necessarily connected) graph $G$ and two vertices $u$ and $v$ in $G$, we say that the graph $H$ is the product of the identification of $u$ and $v$ if $H$ is identical to $G$ except from a new vertex $w$ that replaces the vertices $u$ and $v$, where the neighbors of $w$ in $H$ are the union of the neighbors of $u$ and $v$ in $G$. The class of series-parallel graphs can now be recursively defined as follows (see, for example, [RS42, VTL82]). The graph consisting of a single edge $(x, y)$ is series-parallel with terminals $x$ and $y$. Let $G_1 = (V(G_1), E(G_1))$ and $G_2 = (V(G_2), E(G_2))$ be two series-parallel graphs with terminals $x_1, y_1 \in V(G_1)$ and $x_2, y_2 \in V(G_2)$, respectively. Then the graphs obtained from the following two composition rules are also series-parallel.

- **Series composition:** consider $G_1$ and $G_2$ as a single (disconnected) graph and identify the vertices $y_1$ and $x_2$, forming a new vertex $z$, to obtain the (connected) graph $G_s$ with terminals $x = x_1$ and $y = y_2$.

- **Parallel composition:** consider $G_1$ and $G_2$ as a single (disconnected) graph and identify the vertices $x_1$ and $x_2$, forming a new vertex $x$, and identify the vertices $y_1$ and $y_2$, forming a new vertex $y$, to obtain the (connected) graph $G_p$ with terminals $x$ and $y$.

Observe that by definition, $G_1$ and $G_2$ are (strictly) subgraphs of both $G_s$ and $G_p$.

Given a sequence of series and parallel compositions that resulted in some series-parallel graph $G$, it will be convenient to consider this composition sequence as a tree $\mathcal{T}$, directed toward its root. This directed tree is referred to as a composition trace of $G$ (a series-parallel graph may have many different composition traces). The leaves of $\mathcal{T}$ correspond to the edges of $G$ and each internal node $u$, corresponds to the subgraph of $G$ that was obtained by a (series or parallel) composition of the children of $u$ in $\mathcal{T}$. (The root of the tree corresponds to $G$ itself.) We group together consecutive series compositions and consecutive parallel compositions, transforming $\mathcal{T}$ into a (possibly) shallower directed tree so that every leaf to root path in $\mathcal{T}$ consists of alternating series and parallel arcs. Consequently, a typical node in $\mathcal{T}$ may have an arbitrary number of children.

**Related work.** Probabilistic embedding was first used in [AKPW95], where it is shown that every $n$-vertex graph can be $e^{O(\sqrt{\ln n \ln \ln n})}$-probabilistically embedded into a probability distribution over its spanning trees. (Formally, that paper proved that every $n$-vertex multigraph has a spanning tree of average stretch $e^{O(\sqrt{\ln n \ln \ln n})}$; the notion of probabilistic embedding is only implicit therein.) Moreover, a lower bound of $\Omega(\log n)$ is established for the probabilistic embedding of an unweighted graph into its spanning trees and it is conjectured that this lower bound is tight.

The notion of probabilistic embedding was explicitly introduced in [Bart96], where it is shown that every $n$-vertex graph can be probabilistically embedded into a collection of dominating trees with distortion $O(\log^2 n)$. The same paper establishes an $\Omega(\log n)$ lower bound for this task. The gap left by [Bart96] was narrowed in [Bart98, CCG+98] by presenting an improved upper bound of
Finally, [FRT04] proved a tight upper bound by showing that every graph can be probabilistically embedded into a collection of dominating trees with logarithmic distortion.

By employing a technique presented in [Gupt01] for eliminating Steiner vertices, the result of [FRT04] can be viewed as an $O(\log n)$ upper bound on the distortion required to probabilistically embed an arbitrary $n$-node metric graph (i.e., a complete graph with edge weights obeying the triangle inequality) into its spanning trees, thus resolving the conjecture of [AKPW95] in the affirmative for such graphs. However, it is still unknown if this conjecture holds for arbitrary graphs. A significant step towards proving this conjecture was made in [EEST08], where it is shown that every $n$-vertex graph can be probabilistically embedded into its spanning trees with distortion $O(\log^2 n \log \log n)$. This was recently improved in [ABN08] by introducing an upper bound of $O(\log n \log \log n \log^3 \log \log n)$.

Some papers study probabilistic embeddings of specific classes of graphs. In [GNRS04] it is proved that while every series-parallel graph can be embedded into $\ell_1$ with constant distortion, the $n$-vertex (unweighted) diamond graph, which is a canonical example of a series-parallel graph, cannot be probabilistically embedded into dominating trees with distortion better than $\Omega(\log n)$. Moreover, the authors of [GNRS04] also prove that outerplanar graphs can be probabilistically embedded into their spanning trees with constant distortion. A tight logarithmic upper bound on the distortion required to probabilistically embed a planar graph into a collection of dominating trees is established in [KRS01] by using the decomposition technique of [KPR93] for graphs excluding small minors. [CGN+06] show that every $k$-outerplanar graph can be probabilistically embedded into a collection of dominating trees with constant distortion (assuming that $k$ is fixed).

A motivation for embedding a graph into a distribution over many trees rather than into a single tree is found in [RR98, Gupt01], where it is shown that embedding a graph with girth $g$ into a dominating tree must necessarily suffer an $\Omega(g)$ distortion. This implies that the $n$-cycle cannot be approximated by a single dominating tree unless some edges are allowed to have $\Omega(n)$ stretch.

The MCT problem was introduced in [Hu74] and extensively studied since. NP-hardness of the problem was established in [JLR78], even for the case of uniform requirements over unweighted graphs. In [WLB+98] it was proved that the problem remains NP-hard in the case of uniform requirements over complete weighted graphs. The special case of uniform MCT, i.e., uniform requirements for all pairs of vertices, was studied in [Wong80], where it is shown that there exists a vertex $v$ such that the shortest paths tree rooted at $v$ provides a 2-approximation for the problem. This result was improved in [WLB+98], where a PTAS for the uniform MCT problem is established. The authors of [ABN07] show that every instance of the uniform MCT problem admits a solution (i.e., a spanning tree of the given graph) with constant communication cost. Note that in contrast to the results of [Wong80, WLB+98], the upper bound of [ABN07] is absolute (rather than relative to the optimal solution).

A deterministic $O(\log n \log \log n)$ approximation algorithm for the MCT problem on $n$-points
metric spaces was presented in [PR98]. The performance guarantee of this algorithm is adapted to $O(d \log n)$ for Euclidean spaces of $d$ dimensions. This result was improved in [CCG+98] where it is shown that metrics induced by $n$ points in $\mathbb{R}^d$ can be $O(f(d, p) \log n)$-probabilistically approximated by a distribution over a polynomial number of dominating tree metrics, where $f(d, p) = d^{1/p}$ for $1 \leq p \leq 2$ and $f(d, p) = d^{1-1/p}$ for $p > 2$. A deterministic $O(\log n)$ approximation algorithm for the MCT problem on arbitrary metric spaces was established in [FRT04] by derandomizing their tree sampling algorithm.

Contributions. In this paper we prove that every unweighted $n$-vertex series-parallel graph can be probabilistically embedded into its spanning trees with distortion $O(\log n)$. This is tight due to the lower bound of [GNRS04]. Our proof is constructive: we present a polynomial time algorithm, referred to as Algorithm Construct_Tree, that constructs spanning trees with logarithmic expected stretch. Our probabilistic algorithm can be derandomized to yield a deterministic polynomial time algorithm for constructing a spanning tree of a given unweighted series-parallel graph $G$, whose communication cost is at most $O(\log n)$ times larger than that of $G$. In fact, Algorithm Construct_Tree can be applied to weighted graphs, yielding an $O(\log(n + \Delta))$ upper bound on the distortion, where $\Delta$ is the ratio of the largest edge weight to the smallest one.

Outline of the paper. In Section 2 we present the basic notation and definitions used throughout this paper. Our randomized algorithm is presented in Section 3. In Section 4 it is proved that if our algorithm is invoked on an unweighted $n$-vertex series-parallel graph, then the expected stretch of every pair of vertices is $O(\log n)$. The algorithm is derandomized in Section 5 using the method of conditional expectation. In Appendix A we show that our construction implies an $O(\log(n + \Delta))$ upper bound on the distortion of probabilistically embedding an arbitrarily weighted series-parallel graph into its spanning trees.

2 Preliminaries

Consider an undirected graph $G$. Let $V(G)$ and $E(G)$ denote the vertex and edge sets of $G$, respectively. Each edge $e \in E(G)$ is assumed to be assigned with a positive weight $\omega_G(e)$. The ratio of the largest edge weight to the smallest one is denoted by $\Delta$. We say that $G$ is unweighted if $\omega_G(e) = 1$ for every $e \in E(G)$.

The length of a path $P$ in the graph is the sum of weights of the edges in the path, denoted by $\text{len}(P) = \sum_{e \in E(P)} \omega_G(e)$. For two vertices $u, v$ in $V(G)$, let $\text{dist}_G(u, v)$ denote the distance between them in $G$, i.e., the length of a shortest path between $u$ and $v$.

Although the notion of stretch can be defined for every spanning subgraph, our focus in the current paper is on spanning trees only. Consider some spanning tree $T$ of $G$. Denote the stretch

\footnote{When supplied with a composition trace of the input graph, the algorithm runs in linear time. Given an $n$-vertex series-parallel graph and two terminal vertices, a valid composition trace can be computed in time $O(n^2)$.}
of \(u\) and \(v\) in \(T\) with respect to \(G\) by

\[
\text{str}_{T,G}(u,v) = \frac{\text{dist}_T(u,v)}{\text{dist}_G(u,v)}.
\]

When the graph \(G\) is clear from the context we may omit it and write simply \(\text{str}_T(u,v)\).

Consider a non-negative real communication requirement \(c(u,v)\) for every pair of vertices \(u, v \in V(G)\) and let \(H\) be a connected spanning subgraph of \(G\). The communication cost of \(H\) is defined as

\[
\text{com-cost}(H) = \sum_{u \neq v} c(u,v) \cdot \text{dist}_H(u,v).
\]

Let \(\mathcal{S}\) be a composition trace of some series-parallel graph. Consider a node \(u\) in \(\mathcal{S}\) and let \(v^1, \ldots, v^k\) be the children of \(u\) in \(\mathcal{S}\). If the graph corresponding to the node \(u\) is a series (respectively, parallel) composition of the graphs corresponding to its children in \(\mathcal{S}\), then we say that the arc \((v^j, u)\) is a series arc (resp., parallel arc) for every \(1 \leq j \leq k\).

3 Sampling a spanning tree

The randomized Algorithm \texttt{Construct\_Tree} is recursive. On each recursion level, the algorithm gets an unweighted series-parallel graph \(G\) and terminal vertices \(x, y \in V(G)\) and returns a spanning tree \(T\) of \(G\). Assuming that the graph \(G\) is a (series or parallel) composition of the graphs \(G_1, \ldots, G_k\), Algorithm \texttt{Construct\_Tree} recursively constructs the spanning tree \(T_j\) of \(G_j\) for every \(1 \leq j \leq k\). If the composition of the graphs \(G_1, \ldots, G_k\) is series, then the spanning tree \(T\) is simply the corresponding series composition of the trees \(T_1, \ldots, T_k\). Otherwise (the composition is parallel), Algorithm \texttt{Construct\_Tree} constructs the corresponding parallel composition of the trees \(T_1, \ldots, T_k\), and then transforms it into a spanning tree of \(G\) by removing a single edge (chosen at random) from the unique path between the terminal vertices \(x\) and \(y\) in all the spanning trees \(T_1, \ldots, T_k\) except the one in which this path is the shortest. A formal description of Algorithm \texttt{Construct\_Tree} is given in Table 1.

4 Performance analysis

Consider an unweighted \(n\)-vertex series-parallel graph \(\hat{G}\) with terminal vertices \(\hat{x}\) and \(\hat{y}\) and let \(\hat{T} = \texttt{Construct\_Tree}(\hat{G}, \hat{x}, \hat{y})\). We begin the analysis of Algorithm \texttt{Construct\_Tree} with the observation that \(\hat{T}\) is indeed a spanning tree of \(\hat{G}\). This is true because on every recursion level, a series composition of the spanning trees \(T_1, \ldots, T_k\), obviously results in a spanning tree of \(G\), while a parallel composition of \(T_1, \ldots, T_k\), results in a subgraph of \(G\) that contains a distinct path \(P_j\) between \(x\) and \(y\) for every \(1 \leq j \leq k\). By removing one edge from all but one of the paths \(P_j\), we
**Input:** An unweighted series-parallel graph $G$ with terminal vertices $x$ and $y$.

**Output:** A spanning tree $T$ of $G$.

1. If $E(G) = \{(x, y)\}$, then return $G$.
2. Otherwise, $G$ is a (series or parallel) composition of the graphs $G_1, \ldots, G_k$ with terminal vertices $x_j, y_j \in V(G_j)$ for every $1 \leq j \leq k$.
3. Recursively invoke $T_j = \text{Construct\_Tree}(G_j, x_j, y_j)$ for every $1 \leq j \leq k$.
4. Let $P_j$ be the unique path between $x_j$ and $y_j$ in $T_j$ for $1 \leq j \leq k$. Assume without loss of generality that $\text{len}(P_k) \leq \text{len}(P_j)$ for every $1 \leq j < k$.
5. Case $[G$ is a series composition$]$:
   (a) For every $1 \leq i < j \leq k$:
       if $x_i \in V(G_i)$ and $y_j \in V(G_j)$ are identified in the series composition of $G_1, \ldots, G_k$,
       then identify $x_i \in V(T_i)$ and $y_j \in V(T_j)$.
   (b) Return the resulting spanning tree of $G$.
6. Case $[G$ is a parallel composition$]$:
   (a) Identify the vertices $x_1, \ldots, x_k$ to form a new vertex $x$, and the vertices $y_1, \ldots, y_k$
       to form a new vertex $y$, obtaining a subgraph $H$ of $G$.
   (b) For every $1 \leq j < k$:
       pick an edge uniformly at random in $E(P_j)$ and remove it from $E(H)$.
   (c) Return the resulting spanning tree of $G$.

Table 1: Sampling a spanning tree $T = \text{Construct\_Tree}(G, x, y)$
Figure 1: A parallel composition of the spanning trees $T_j$ (solid lines) and $T_k$ (dashed lines). If the edge $e$ is removed from the path $P_j$, then $P_k$ remains the unique path between the terminal vertices $x$ and $y$.

Get a spanning tree. (Refer to Figure 1 for an illustration of a parallel composition of two spanning trees.) Furthermore, by induction on the depth of the composition trace of $\tilde{G}$ (the one used by the algorithm), we conclude that the designated shortest unique path $P_k$ is also a shortest path between $x$ and $y$ in the graph $G$. This gives rise to the following observation.

**Observation 4.1.** Every subgraph $G$ of $\tilde{G}$ that corresponds to some node in the composition trace $\Sigma$ with terminals $x$ and $y$ contains some designated shortest path $P$ between $x$ and $y$ that remains intact in the spanning tree $T = \text{Construct Tree}(G, x, y)$.

Our main goal is to establish an upper bound on the expected distance between the endpoints of every edge in $E(\tilde{G})$. By the linearity of expectation, this bounds the expected stretch of every pair of vertices in $V(\tilde{G})$. (Recall that as $\tilde{G}$ is unweighted, the stretch of each edge is just the distance between its endpoints in $\tilde{T}$.)

Our proof is based on bounding the maximum possible value of a sum defined as follows. For some positive integers $m$, $\{a_i\}_{i=1}^m$, $\{b_i\}_{i=1}^m$ and $d$, let

$$S = \sum_{i=1}^{m} \frac{a_i}{a_i + b_i} \left( \prod_{j=1}^{i-1} \frac{b_j}{a_j + b_j} \right) (a_1 + \cdots + a_{i-1} + b_i) + \left( \prod_{i=1}^{m} \frac{b_i}{a_i + b_i} \right) (a_1 + \cdots + a_m + d).$$

Define the function $S(n)$ to be the maximum possible value of $S$, where $m$, $\{a_i\}_{i=1}^m$, $\{b_i\}_{i=1}^m$ and $d$ are subject to the following restrictions:

- $m \leq n$.
- $b_1 = 1$.
- $b_{i+1} \leq a_i + b_i$ for every $1 \leq i < m$.
- $d \leq a_m + b_m$.
- $\sum_{i=1}^{m} a_i \leq n$. 

7
In Lemma 4.2 we prove that the expected distance between the endpoints of $e$ in the output spanning tree $\hat{T}$ is at most $S(n)$ for every edge $e \in E(\hat{G})$. The value of $S(n)$ is bounded by $O(\log n)$ in Lemma 4.3 thus yielding the desired upper bound.

**Lemma 4.2.** Let $\hat{T}$ be the output of Algorithm `Construct_Tree` when invoked on an $n$-vertex series-parallel graph $\hat{G}$. Then $E[\text{dist}_{\hat{T}}(u,v)] \leq S(n)$ for every edge $(u,v) \in E(\hat{G})$.

**Proof.** Let $\mathcal{T}$ be the composition trace of $\hat{G}$ used by Algorithm `Construct_Tree`. Consider an edge $e = (u,v) \in E(\hat{G})$ and let $v_e$ be the leaf of $\mathcal{T}$ that corresponds to $e$. Let $\pi = (u^0, u^1, \ldots, u^t)$ be the unique path from $v_e$ to the root in $\mathcal{T}$, where $u^0 = v_e$ and $u^t$ is the root of $\mathcal{T}$ corresponding to the graph $\hat{G}$. Let $e^h = (u^{h-1}, u^h)$ be the $h$th arc in $\pi$ for every $0 < h \leq t$.

Consider the recursive invocation of Algorithm `Construct_Tree` on the graph $G$ corresponding to the node $u^h$ for some $0 < h \leq t$ and let $x$ and $y$ be the terminal vertices of $G$. Let $G_j$ be the composing subgraph of $G$ that contains the edge $e = (u,v)$. Let $\psi^h$ be the unique path between $u$ and $v$ in the spanning tree $T_j$ of $G_j$ and recall that $P_j$ is the unique path between $x$ and $y$ in $T_j$. We say that $\psi^h$ is **settled** if it does not overlap with $P_j$, that is, if $E(\psi^h) \cap E(P_j) = \emptyset$. Observe that if $\psi^h$ is settled, then $\psi^{h'}$ (corresponding to the node $u^{h'}$ in the composition trace) is settled for every $h' > h$, and $\psi^h$ is the unique path between $u$ and $v$ in $\hat{T}$. For non-settled $\psi^h$, let $\chi^h$ be the subpath of $\psi^h$ that overlaps with the path $P_j$. Assuming that $\psi^h$ is not settled, we say that $\chi^h$ is **at risk** if the arc $e^h$ is parallel and $j < k$ (recall that $\text{len}(P_j)$ is minimum for $j = k$), that is, if $\chi^h$ may be disconnected during this recursive invocation (by a removal of one of its edges). Refer to Figure 2 for illustration.

Let $\Lambda = \langle e^h_1, \ldots, e^h_m \rangle$ be the subsequence of $\langle e^1, \ldots, e^t \rangle$ such that $e^h_i$ is a parallel arc and $\chi^h_i$ is at risk with positive probability for every $1 \leq i \leq m$. Every $e^h_i$ in $\Lambda$ corresponds to a recursive invocation of Algorithm `Construct_Tree` in which, depending on the random choices of the algorithm, the distance between $u$ and $v$ may increase. Observe that $\Lambda$ is independent of the coin tosses of the algorithm and it depends solely on the composition trace $\mathcal{T}$. Furthermore, for
Figure 3: During the recursive invocation of the algorithm on the graph corresponding to the node \( u^h \) with terminal vertices \( x \) and \( y \) (solid lines), an edge \( e \in E(\chi^h) \) may be removed. Consequently, the new path between \( u \) and \( v \) (bold lines) consists of the edges in \( E(\psi^h) \oplus E(P_j) \) and of \( P_k \). This new path serves as \( \psi^{h+1} \) when the algorithm is invoked on the graph corresponding to \( u^{h+1} \) with terminal vertices \( x' \) and \( y' \) (solid and dashed lines), while \( P_k \) serves as \( \chi^{h+1} \).

Every \( 0 < h \leq t \), the path \( \psi^h \) is non-settled with positive probability in which case, both \( \psi^h \) and \( \chi^h \) are determined by the composition trace \( T \).

Consider the recursive invocation of Algorithm \texttt{Construct Tree} on the graph \( G \) corresponding to the node \( u^h \) for some \( e^h \in \Lambda \). Let \( G_j \) be the composing subgraph of \( G \) that contains the edge \( e \). Assuming that \( \psi^h \) is not settled, we define the variables \( a_i, b_i, c_i \) and \( d_i \) as follows.

- \( a_i = \text{len}(P_j) - \text{len}(\chi^h) \),
- \( b_i = \text{len}(\chi^h) \),
- \( c_i = \text{len}(\psi^h) - \text{len}(\chi^h) \) and
- \( d_i = \text{len}(P_k) \).

The values of these variables depend solely on the composition trace \( T \).

Note that \( b_1 = 1 \), \( c_1 = 0 \) and \( m \leq t \leq n \). Since \( \text{len}(P_k) \leq \text{len}(P_j) \) for every \( 1 \leq j < k \), it follows that \( d_i \leq a_i + b_i \) for every \( 1 \leq i \leq m \). Moreover, if \( \chi^h \) is disconnected during the recursive invocation of Algorithm \texttt{Construct Tree} on \( G \), then the new path between \( u \) and \( v \) consists of the edges in the symmetric difference \( E(\psi^h) \oplus E(P_j) \) and of \( P_k \), the latter serves as \( \chi^{h+1} \) (unless \( i = m \)). It follows that \( b_{i+1} = d_i \) and \( c_{i+1} = a_i + c_i \) for every \( 1 \leq i < m \). Refer to Figure 3 for illustration.

Every edge in \( P_j \) is removed from \( T \) with probability \( 1/\text{len}(P_j) \). Therefore, given that \( \psi^h \) is not settled, the path \( \chi^h \) is disconnected with probability \( \text{len}(\chi^h)/\text{len}(P_j) = b_i/(a_i + b_i) \). Note that for every \( 1 \leq i < m \), if \( \chi^h \) is disconnected, then \( \psi^{h+1} \) is not settled. On the other hand, with probability \( a_i/(a_i + b_i) \), some edge in \( E(P_j) \setminus E(\chi^h) \) is removed and \( \chi^h \) remains connected, in that case, the unique path between \( u \) and \( v \) in \( T \) is determined to be \( \psi^h \).
For every $1 \leq i \leq m$, let $A_i$ denote the event in which $\psi^{h_i}$ is not settled and the unique path between $u$ and $v$ in $\hat{T}$ is $\chi^{h_i}$. Let $B_i$ denote the event in which $\chi^{h_i}$ was disconnected for every $1 \leq l \leq i$, namely, the event $\neg A_1 \land \cdots \land \neg A_i$. Using conditional probability, we get

$$\Pr[A_i] = \Pr[A_i \land B_{i-1}] = \Pr[A_i \mid B_{i-1}] \cdot \Pr[B_{i-1}]$$

and

$$\Pr[B_i] = \Pr[B_i \land B_{i-1}] = \Pr[B_i \mid B_{i-1}] \cdot \Pr[B_{i-1}] .$$

Consequently we have the following equation for the probability of $A_i$:

$$\Pr[A_i] = \Pr[A_i \mid B_{i-1}] \cdot \left( \prod_{j=2}^{i-1} \Pr[B_j \mid B_{j-1}] \right) \cdot \Pr[B_1] = \frac{a_i}{a_i + b_i} \cdot \prod_{j=1}^{i-1} \frac{b_j}{a_j + b_j} .$$

For the expected distance between $u$ and $v$ in the output spanning tree, we now get

$$\mathbb{E}[\text{dist}_\mathcal{T}(u, v)] = \sum_{i=1}^{m} \frac{a_i}{a_i + b_i} \left( \prod_{j=1}^{i-1} \frac{b_j}{a_j + b_j} \right) (b_i + c_i) + \left( \prod_{i=1}^{m} \frac{b_i}{a_i + b_i} \right) (a_m + c_m + d_m) ,$$

where the rightmost term corresponds to the event in which $\chi^{h_i}$ was disconnected for every $1 \leq i \leq m$. The lemma follows by substituting $c_{i+1} = a_1 + \cdots + a_i$ for every $1 \leq i < m$ and by renaming $d_m = d$.

It remains to bound the value of $S(n)$.

**Lemma 4.3.** The function $S(\cdot)$ satisfies $S(n) = O(\log n)$.

**Proof.** We begin by proving that the rightmost term in $S$ is $O(1)$. This is done by separating that term to

$$\left( \prod_{i=1}^{m} \frac{b_i}{a_i + b_i} \right) (a_1 + \cdots + a_m) + \left( \prod_{i=1}^{m} \frac{b_i}{a_i + b_i} \right) d .$$

(1)

Since $b_{i+1} \leq a_i + b_i$ for every $1 \leq i < m$, and since $d \leq a_m + b_m$, it follows, by telescoping, that the right term in (1) is at most $b_1 = 1$. For the left term, we use the inequality $b_{i+1} \leq a_i + b_i$ to conclude that $b_{i+1} \leq 1 + a_1 + \cdots + a_i$ for every $1 \leq i < m$. Now, by substituting the $b_i$s in the left term of (1), accordingly, and by telescoping, it follows that this term is at most $b_1$ as well.

It remains to bound the sum

$$S' = \sum_{i=1}^{m} \frac{a_i}{a_i + b_i} \left( \prod_{j=1}^{i-1} \frac{b_j}{a_j + b_j} \right) (a_1 + \cdots + a_{i-1} + b_i) .$$

In attempt to do so, we first show that $S'$ is maximized when $m = n$ and $a_1 = a_2 = \cdots = a_n = 1$. Indeed, by increasing the value of $a_m$, we increase $S'$, hence it is sufficient to consider $a_i$'s with $\sum_{i=1}^{m} a_i = n$. Now, suppose towards deriving contradiction that $S'$ is maximized when $a_l > 1$ for
some $1 \leq l \leq m$. Since $\sum_{i=1}^{m} a_i = n$, it follows that $m < n$. We show that $S'$ can be increased by replacing $a_1, \ldots, a_{l-1}, a_l, a_{l+1}, \ldots, a_m$ and $b_1, \ldots, b_{l-1}, b_l, b_{l+1}, \ldots, b_m$ with $a_1, \ldots, a_{l-1}, a_l - 1, a_{l+1}, \ldots, a_m$ and $b_1, \ldots, b_{l-1}, b_l + 1, b_{l+1}, \ldots, b_m$, respectively (this is a valid replacement as $m + 1 \leq n$), thus yielding a contradiction. That is, the sum

$$S'' = \sum_{i=1}^{l-1} \frac{a_i}{a_i + b_i} \left( \prod_{j=1}^{i-1} \frac{b_j}{a_j + b_j} \right) (a_1 + \cdots + a_{i-1} + b_i)$$

$$+ \frac{1}{b_l + 1} \left( \prod_{j=1}^{l-1} \frac{b_j}{a_j + b_j} \right) (a_1 + \cdots + a_{l-1} + b_l)$$

$$+ \frac{a_l - 1}{a_l + b_l} \left( \prod_{j=1}^{l-1} \frac{b_j}{a_j + b_j} \right) \cdot \frac{b_l}{b_l + 1} \cdot (a_1 + \cdots + a_{l-1} + 1 + b_l + 1)$$

$$+ \sum_{i=l+1}^{m} \frac{a_i}{a_i + b_i} \left( \prod_{j=1}^{i-1} \frac{b_j}{a_j + b_j} \right) \cdot \frac{b_l}{b_l + 1} \cdot \frac{b_l + 1}{a_l + b_l} \cdot \left( \prod_{j=l+1}^{i-1} \frac{b_j}{a_j + b_j} \right) (a_1 + \cdots + a_{i-1} + b_i)$$

is strictly greater than $S'$. Observe that the first $l - 1$ terms in $S''$ are identical to the first $l - 1$ terms in $S'$, and that the last $m - l$ terms in $S''$ are identical to the last $m - l$ terms in $S'$. Therefore we only have to verify that the sum of the $l$th term and the $(l+1)$st term in $S''$ is strictly greater than the $l$th term in $S'$. This is done by a straightforward calculation.

So, in what follows we assume that $m = n$ and $a_1 = a_2 = \cdots = a_n = 1$. Therefore we can also assume that $b_i \leq i$ for every $1 \leq i \leq n$. We separate the sum $S'$ to two sums

$$S_a = \sum_{i=1}^{n} \frac{1}{b_i + 1} \left( \prod_{j=1}^{i-1} \frac{b_j}{b_j + 1} \right) \cdot (i - 1) = \sum_{i=1}^{n} \frac{i - 1}{b_i} \cdot \prod_{j=1}^{i} \frac{b_j}{b_j + 1}$$

and

$$S_b = \sum_{i=1}^{n} \frac{1}{b_i + 1} \left( \prod_{j=1}^{i-1} \frac{b_j}{b_j + 1} \right) \cdot b_i = \sum_{i=1}^{n} \prod_{j=1}^{i} \frac{b_j}{b_j + 1} \leq \sum_{i=1}^{n} \prod_{j=1}^{i} \frac{1}{j + 1},$$

where $S' = S_a + S_b$. The sum $S_b$ telescopes to the harmonic sum, thus it is at most $\ln(n + 1)$. The rest of the proof is dedicated to bounding the sum $S_a$. We will show that the sum of the last $n/2$ terms of $S_a$ is $O(1)$. The proof is completed by repeating this argument, showing that the set of terms in $S_a$ can be partitioned into $\log n$ subsets, where the sum of the terms in each subset is $O(1)$.

Let $\tilde{S}_a$ be the sum of the last $n/2$ terms of $S_a$. We first cancel the dependency of $\tilde{S}_a$ on the
value of $n$:

\[
\hat{S}_n = \sum_{i=\lceil \frac{n}{2} \rceil + 1}^{n} \frac{i - 1}{b_i} \cdot \prod_{j=1}^{i} \frac{b_j}{b_j + 1}
\]

\[
\leq n \sum_{i=\lceil \frac{n}{2} \rceil + 1}^{n} \frac{1}{b_i} \cdot \prod_{j=1}^{i} \frac{b_j}{b_j + 1}
\]

\[
\leq n \sum_{i=\lceil \frac{n}{2} \rceil + 1}^{n} \frac{1}{b_i} \cdot \frac{2}{n} \cdot \prod_{j=\lceil \frac{n}{2} \rceil + 1}^{i} \frac{b_j}{b_j + 1}
\]

\[
= 2 \sum_{i=\lceil \frac{n}{2} \rceil + 1}^{n} \frac{1}{b_i} \cdot \prod_{j=\lceil \frac{n}{2} \rceil + 1}^{i} \frac{b_j}{b_j + 1},
\]

where inequality (2) holds by telescoping since $b_1 = 1$ and $b_{i+1} \leq b_i + 1$. Now, we can consider the simpler sum

\[
R = \sum_{i=1}^{k} \frac{1}{r_i} \cdot \prod_{j=1}^{i} \frac{r_j}{r_j + 1}.
\]

Our goal is to show that $R = O(1)$ for positive integers $k$ and $\{r_i\}_{i=1}^{k}$, where there are no restrictions on $k$ and the sole restriction on $\{r_i\}_{i=1}^{k}$ is that $r_{i+1} \leq r_i + 1$ for every $1 \leq i < k$.

We are interested in those $i$s that do not satisfy $r_i = r_{i-1} + 1$ (including $i = 1$). Suppose that there are $t$ such is, named $i_1, \ldots, i_t$, where $i_1 = 1$. Let $\delta_j = i_{j+1} - i_j$ for every $1 \leq j < t$ and let $\delta_t = k - i_t + 1$. Let $r_j = r_{i_j}$ for every $1 \leq j \leq t$. Observe that $\sum_{j=1}^{t} \delta_j = k$ and that $r_{j+1} < r_j + \delta_j$ for every $1 \leq j < t$.

Let $\sigma = (\langle r_1, \delta_1 \rangle, \langle r_2, \delta_2 \rangle, \ldots, \langle r_t, \delta_t \rangle)$. We rewrite the sum $R$, using the notations $\delta_j$ and $r_j$ and
define the function $f(\sigma)$ as follows:

$$
R = \sum_{i=1}^{\delta_1} \frac{1}{\rho_1 + i - 1} \cdot \prod_{j=1}^{i} \rho_1 + j - 1 \cdot \prod_{j=1}^{i+1} \rho_1 + j + 1 + \sum_{i=1}^{\delta_2} \frac{1}{\rho_2 + i - 1} \cdot \prod_{j=1}^{i} \rho_2 + j - 1 \cdot \prod_{j=1}^{i+1} \rho_2 + j + 1
+ \ldots + \sum_{i=1}^{\delta_i} \frac{1}{\rho_t + i - 1} \cdot \prod_{j=1}^{i} \rho_t + j - 1 \cdot \prod_{j=1}^{i+1} \rho_t + j + 1
$$

$$
= \sum_{i=1}^{\delta_1} \frac{1}{\rho_1 + i - 1} \cdot \rho_1 + i \cdot \prod_{j=1}^{i} \rho_1 + j + 1 + \sum_{i=1}^{\delta_2} \frac{1}{\rho_2 + i - 1} \cdot \rho_2 + i \cdot \prod_{j=1}^{i} \rho_2 + j + 1
+ \ldots + \sum_{i=1}^{\delta_i} \frac{1}{\rho_t + i - 1} \cdot \rho_t + i \cdot \prod_{j=1}^{i} \rho_t + j + 1
$$

$$
\leq 2\rho_1 \sum_{i=1}^{\delta_1} \left( \frac{1}{\rho_1 + i} \right)^2 + 2\rho_2 \sum_{i=1}^{\delta_2} \left( \frac{1}{\rho_2 + i} \right)^2
+ \ldots + 2\rho_t \sum_{i=1}^{\delta_t} \left( \frac{1}{\rho_t + i} \right)^2
\leq f(\sigma)
.$$

Note that if $\sigma = (\langle \rho, \delta \rangle) \circ \sigma'$, then

$$
f(\sigma) = 2\rho \sum_{i=1}^{\delta} \left( \frac{1}{\rho + i} \right)^2 + \frac{\rho}{\rho + \delta} \cdot f(\sigma')
.$$

Using the fact that $\sum_{i=1}^{\delta} \frac{1}{(\rho+i)^2} \leq \frac{1}{\rho} - \frac{1}{\rho + \delta}$, we prove, by induction on $t$, that $f(\sigma) \leq 2$. For $t = 1$, we have

$$
f(\sigma) = 2\rho \sum_{i=1}^{\delta} \left( \frac{1}{\rho + i} \right)^2 \leq 2\rho \left( \frac{1}{\rho} - \frac{1}{\rho + \delta} \right) \leq 2,
$$

and the claim holds. Assume that the claim holds for $t$ and consider some $\sigma = (\langle \rho, \delta \rangle) \circ \sigma'$, where $|\sigma'| = t$. The value of $f(\sigma)$ satisfies

$$
f(\sigma) = 2\rho \sum_{i=1}^{\delta} \left( \frac{1}{\rho + i} \right)^2 + \frac{\rho}{\rho + \delta} \cdot f(\sigma') \leq 2\rho \left( \frac{1}{\rho} - \frac{1}{\rho + \delta} \right) + 2\frac{\rho}{\rho + \delta} = 2,
$$

and the claim holds.

\[ \square \]

**Theorem 4.4.** For every unweighted $n$-vertex series-parallel graph $G$, there exists a probability distribution $\mathcal{D}$ over the spanning trees of $G$ such that for every two vertices $u, v \in V(G)$, we have

$$
\mathbb{E}[\text{str}_T(u, v)] = O(\log n),
$$

where $T$ is chosen according to the distribution $\mathcal{D}$.  

13
5 Derandomization

The goal in probabilistic embedding is to find a probability distribution over the spanning trees of a given graph so that the expected stretch of every pair of vertices is small. The dual of probabilistic embedding is essentially the MCT problem, where the goal is to find a single spanning tree with small communication cost. In this section we use the method of conditional expectation to derandomize Algorithm Construct Tree (see [AS92]), presenting a deterministic algorithm that on input unweighted \( n \)-vertex series-parallel graph \( G \) and communication requirement \( c(u, v) \) for every two vertices \( u, v \in V(G) \), outputs a spanning tree \( T \) of \( G \) that satisfies

\[
\text{com-cost}(T) = O(\log n) \cdot \sum_{u \neq v} c(u, v) \cdot \text{dist}_G(u, v) .
\]

In particular, this can be viewed as a deterministic algorithm that given an unweighted series-parallel \( n \)-vertex multigraph \( G \), constructs a spanning tree \( T \) of \( G \) with \( \text{av-str}(T) = O(\log n) \).

We begin by showing that it is sufficient to consider instances of the MCT problem with 0 communication requirement for every two vertices that are not an edge in the graph. Consider a graph \( G \) and communication requirement \( c(u, v) \) for every two vertices \( u, v \in V(G) \). We say that the communication requirements \( c(\cdot, \cdot) \) are confined if \( c(u, v) = 0 \) for every \((u, v) \notin E(G)\). Given the communication requirements \( c(\cdot, \cdot) \), we define the new communication requirements \( c'(\cdot, \cdot) \) as follows. Let \( Q_{u,v} \) be a designated shortest path between \( u \) and \( v \) in \( G \) for every two vertices \( u,v \in V(G) \). The new communication requirements are set to

\[
c'(u,v) = \sum_{x,y \mid (x,y) \in E(Q_{x,y})} c(x,y)
\]

for every \( u,v \in V(G) \). Clearly, the communication requirements \( c'(\cdot, \cdot) \) are confined.

**Proposition 5.1.** The communication requirements \( c'(\cdot, \cdot) \) satisfy

- \( \sum_{u \neq v} c'(u,v) \cdot \text{dist}_G(u,v) = \sum_{u \neq v} c(u,v) \cdot \text{dist}_G(u,v) \) and
- \( \sum_{u \neq v} c'(u,v) \cdot \text{dist}_T(u,v) \geq \sum_{u \neq v} c(u,v) \cdot \text{dist}_T(u,v) \) for every spanning tree \( T \) of \( G \).

**Proof.** Consider a spanning subgraph \( H \) of \( G \). By the definition of \( c'(\cdot, \cdot) \), we have the following inequality:

\[
\sum_{u \neq v} c(u,v) \cdot \text{dist}_H(u,v) \leq \sum_{u \neq v} c(u,v) \sum_{(x,y) \in E(Q_{u,v})} \text{dist}_H(x,y) \\
= \sum_{(x,y) \in E(G)} \text{dist}_H(x,y) \sum_{u,v \mid (x,y) \in E(Q_{u,v})} c(u,v) \\
= \sum_{x \neq y} \text{dist}_H(x,y) \cdot c'(x,y) ,
\]

where, the inequality is replaced by equality when the subgraph \( H \) is \( G \) itself. The proposition follows. \( \square \)
Next, we present a deterministic algorithm that on input unweighted n-vertex series-parallel graph $\hat{G}$ and confined communication requirements $c(\cdot, \cdot)$, outputs a spanning tree that satisfies [3]. By Proposition 5.1 this guarantees an $O(\log n)$-approximate solution for any instance of the MCT problem on unweighted series-parallel graphs. For the sake of analysis, suppose that we invoke (the randomized) Algorithm \textbf{Construct Tree} on $\hat{G}$, constructing the random spanning tree $\hat{T}$ with

$$\text{com-cost}(\hat{T}) = \sum_{u \neq v} c(u, v) \cdot \text{dist}_T(u, v).$$

Let $\mathcal{T}$ be the composition trace of $\hat{G}$ (the one used by Algorithm \textbf{Construct Tree}). In a preprocess stage, we perform a preorder traversal of $\mathcal{T}$ form the root down to the leaves and efficiently compute the expected distance between $x$ and $y$ in $\hat{T}$ for every pair of vertices $x, y \in V(\hat{G})$ that serve as the terminals of some subgraph in the composition trace of $\hat{G}$. This works as follows. For the terminals $\hat{x}$ and $\hat{y}$ of $\hat{G}$ we have $\text{dist}_\hat{T}(\hat{x}, \hat{y}) = \text{dist}_G(\hat{x}, \hat{y})$ with probability 1, and hence $\mathbb{E}[\text{dist}_\hat{T}(\hat{x}, \hat{y})] = \text{dist}_G(\hat{x}, \hat{y})$.

Consider the graph $G$ which corresponds to some node $u$ in $\mathcal{T}$ and let $G'$ be the graph that corresponds to the parent $u'$ of $u$ in $\mathcal{T}$. Let $x$ and $y$ (respectively, $x'$ and $y'$) be the terminals of $G$ (resp., $G'$). If $(u, u')$ is a parallel arc in $\mathcal{T}$, then $\text{dist}_\hat{T}(x, y) = \text{dist}_\hat{T}(x', y')$ with probability 1, and hence $\mathbb{E}[\text{dist}_\hat{T}(x, y)] = \mathbb{E}[\text{dist}_\hat{T}(x', y')]$. So, assume that $(u, u')$ is a series arc.

Observation 4.1 guarantees that there exists some designated shortest path $P$ between $x$ and $y$ in $G$ that remains intact in the spanning tree $T = \text{Construct Tree}(G, x, y)$. Let $\hat{P}$ be the designated shortest path between $\hat{x}$ and $\hat{y}$ in $\hat{G}$ ($\hat{P}$ remains intact in $\hat{T}$). If $P$ is a subpath of $\hat{P}$, then $P$ remains intact in $\hat{T}$ and $\text{dist}_\hat{T}(x, y) = \text{dist}_G(x, y)$ with probability 1, hence $\mathbb{E}[\text{dist}_\hat{T}(x, y)] = \text{dist}_G(x, y)$. Otherwise ($P$ is not a subpath of $\hat{P}$), let $v$ be the highest ancestor of $u$ in $\mathcal{T}$ such that $P$ is a subpath of the the designated shortest path $P_v$ between the terminals $x_v$ and $y_v$ of the graph $G_v$ that corresponds to $v$. Let $u'$ be the parent of $v$ in $\mathcal{T}$ and let $G'_v$ be the graph that corresponds to $v'$ with terminals $x'_v$ and $y'_v$. (Observe that the arc $(v, u')$ must be a parallel arc.)

The algorithm is designed so that the path $P$ remains intact in the the spanning tree $T'_v = \text{Construct Tree}(G'_v, x'_v, y'_v)$ with probability $\frac{\text{len}(P_v) - \text{len}(P)}{\text{len}(P_v)}$, in which case it remains intact in $\hat{T}$ and $\text{dist}_\hat{T}(x, y) = \text{len}(P) = \text{dist}_G(x, y)$; and it is cut off in $T'_v$ with probability $\frac{\text{len}(P)}{\text{len}(P_v)}$, in which case $\text{dist}_\hat{T}(x, y) = \mathbb{E}[\text{dist}_\hat{T}(x'_v, y'_v)] + \text{len}(P) - \text{len}(P)$. Put together, we have

$$\mathbb{E}[\text{dist}_\hat{T}(x, y)] = \frac{\text{dist}_G(x_v, y_v) - \text{dist}_G(x, y)}{\text{dist}_G(x_v, y_v)} \cdot \text{dist}_G(x, y) + \frac{\text{dist}_G(x, y)}{\text{dist}_G(x_v, y_v)} \cdot (\mathbb{E}[\text{dist}_\hat{T}(x'_v, y'_v)] + \text{dist}_G(x_v, y_v) - \text{dist}_G(x, y)).$$

To conclude, if we already know the expected distance in $\hat{T}$ between the terminals of graphs that correspond to all ancestors of $u$ in $\mathcal{T}$, then the computation of $\mathbb{E}[\text{dist}_\hat{T}(x, y)]$ becomes straightforward.
We now turn to describe the deterministic algorithm after the preprocess stage is completed. This algorithm follows the random choices of Algorithm \texttt{Construct Tree} (from the leaves up to the root). Let $u$ be a node in $\mathcal{T}$ that corresponds to the parallel composition of the subgraphs $G_1, \ldots, G_k$. When invoked on the graph corresponding to $u$, the randomized algorithm tosses coins to pick an edge for removal from the unique path $P_j$ for every $1 \leq j < k$. Choosing to remove the edge $e$ from $P_j$ may affect the contribution of the vertex pair $u, v \in V(\hat{G})$ to $\text{com-cost}(\hat{T})$ only if $(u, v)$ is an edge in $E(\hat{G})$ (this is because the communication requirements are confined) and this edge belongs to $G_j$. As we already computed (in the preprocess stage) the expected distance between the endpoints of $G_j$ in $\hat{T}$, it follows that conditioned on the previous choices of the algorithm, we can efficiently compute the expected contribution of the vertex pair $u, v$ (which depends on $c(u, v)$) to $\text{com-cost}(\hat{T})$ for every such vertex pair $u, v$. Therefore, by choosing to remove the edge $e$ (from the path $P_j$) that minimizes the sum of these contributions, we guarantee that the expected communication cost of our solution is bounded from above by the expected communication cost of the randomized solution which is $O(\log n) \cdot \text{com-cost}(\hat{G})$. 

16
References


A The weighted case

Let $G$ be some class of graphs that is closed under edge subdivision, namely, if $G$ is in $G$ and $e = (x, y)$ is an edge in $E(G)$, then the graph obtained from $G$ by replacing the edge $e$ with the simple path $(x, v_e, y)$, where $v_e$ is a new vertex, is also in $G$. By definition, series-parallel graphs are closed under edge subdivision. We show\(^3\) that if any unweighted $n$-vertex graph in $G$ can be probabilistically embedded into its spanning trees with distortion $O(\log n)$, then any weighted $n$-vertex graph in $G$ can be probabilistically embedded into its spanning trees with distortion $O(\log(n + \Delta))$.

Consider some weighted $n$-vertex graph $G$ in $G$. For simplicity, we assume that the edge weights in $G$ are positive integers and that the smallest edge weight is 1 (this affects the distortion by a factor of at most 2), so $\Delta$ is the largest edge weight. Let $G'$ be the unweighted graph obtained from $G$ by subdividing each edge $e \in E(G)$ of weight $k = \omega_G(e)$ into a (simple) path consisting of $k$ edges $e_1, \ldots, e_k$ of unit weight. By definition, $G'$ is also in $G$. Fix $n' = |V(G')|$. Let $D'$ be a probability distribution over the spanning trees of $G'$ so that if $T'$ is chosen according to $D'$, then $\mathbb{E}[\text{dist}_{T'}(x,y)] = O(\log n')$ for every $x, y \in V(G')$.

For each spanning tree $T'$ of $G'$ in the support of $D'$, we construct a spanning tree $T$ of $G$ by including the edge $e \in E(G)$ if and only if $e_i \in E(T')$ for all $1 \leq i \leq k$. It is easy to verify that the resulting subgraph $T$ is indeed a spanning tree of $G$. Therefore we obtain a probability distribution $D$ over the spanning trees of $G$.

**Lemma.** For every two vertices $u, v \in V(G)$, we have $\mathbb{E}[\text{str}_T(u,v)] = O(\log n')$, where $T$ is chosen according to the distribution $D$.

**Proof.** We establish the lemma by showing that $\mathbb{E}[\text{dist}_T(x,y)] \leq O(\log n') \cdot \omega_G(e)$ for every $e = (x, y) \in E(G)$. Suppose that $\omega_G(e) = k$ and let $e_1, \ldots, e_k$ be the unit weight edges that correspond to $e$ in $E(G')$. By the construction of $T$, it follows that $\text{dist}_T(x,y) = \text{dist}_{T'}(x,y)$, thus it suffices to prove that $\mathbb{E}[\text{dist}_{T'}(x,y)] = O(k \log n')$. Indeed,

$$\mathbb{E}[\text{dist}_{T'}(x,y)] \leq \mathbb{E}\left[\sum_{i=1}^{k} \text{dist}_{T'}(e_i)\right] = \sum_{i=1}^{k} \mathbb{E}[\text{dist}_{T'}(e_i)] = O(k \log n')$$

as required. \(\square\)

As every edge in $G$ introduces at most $\Delta - 1$ new vertices in $G'$, we know that $n' \leq n + \binom{n}{2}(\Delta - 1)$, and hence $\log(n') = O(\log(n + \Delta))$.

---

\(^3\) This is not a new result; we just revisit it for completeness.