Parallelized approximation algorithms for minimum routing cost spanning trees

Ching-Lueh Chang∗ Yuh-Dauh Lyuu†

February 5, 2008

Abstract

Let $G = (V, E)$ be an undirected graph with a nonnegative edge-weight function $w$. The routing cost of a spanning tree $T$ of $G$ is $\sum_{u,v \in V} d_T(u, v)$, where $d_T(u, v)$ denotes the weight of the simple $u$-$v$ path in $T$. The MINIMUM ROUTING COST SPANNING TREE (MRCT) problem [WLB+00] asks for a spanning tree of $G$ with the minimum routing cost. In this paper, we parallelize several previously proposed approximation algorithms for the MRCT problem and some of its variants. Let $\epsilon > 0$ be an arbitrary constant. When the edge-weight function $w$ is given in unary, we parallelize the $(4/3 + \epsilon)$-approximation algorithm for the MRCT problem [WCT00b] by implementing it using an $\mathcal{RNC}$ circuit. There are other variants of the MRCT problem. In the SUM-REQUIREMENT OPTIMAL COMMUNICATION SPANNING TREE (SROCT) problem [WCT00a], each vertex $u$ is associated with a requirement $r(u) \geq 0$. The objective is to find a spanning tree $T$ of $G$ minimizing $\sum_{u,v \in V} (r(u) + r(v)) d_T(u, v)$. When the edge-weight function $w$ and the vertex-requirement function $r$ are given in unary, we parallelize the 2-approximation algorithm for the SROCT problem [WCT00a] by realizing it using $\mathcal{RNC}$ circuits, with a slight degradation in the approximation ratio from 2 to $2 + o(1)$. In the weighted 2-MRCT problem [Wu02], we have additional inputs $s_1, s_2 \in V$ and $\lambda \geq 1$. The objective is to find a spanning tree $T$ of $G$ minimizing $\sum_{v \in V} \lambda d_T(s_1, v) + d_T(s_2, v)$. When the edge-weight function $w$ is given in unary, we parallelize the 2-approximation algorithm [Wu02] into $\mathcal{RNC}$ circuits, with a slight degradation in the approximation ratio from 2 to $2 + o(1)$. To the best of our knowledge, our results are the first parallelized approximation algorithms for the MRCT problem and its variants.

1 Introduction

Let $G = (V, E)$ be an undirected graph with a nonnegative edge-weight function $w$. The routing cost of a spanning tree $T$ of $G$ is $\sum_{u,v \in V} d_T(u, v)$ where $d_T(u, v)$ is the weight of any shortest $u$-$v$ path in $T$, or equivalently, the weight of the

∗Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan. Email: Email: d95007@csie.ntu.edu.tw
†Department of Computer Science and Information Engineering, National Taiwan University, Taipei, Taiwan. Email: lyuu@csie.ntu.edu.tw
The Minimum Routing Cost Spanning Tree (MRCT) problem [WLB+00] asks for a spanning tree $T$ of $G$ with the minimum routing cost. It is also known as the Shortest Total Path Length Spanning Tree problem. The MRCT problem is first proposed by Hu [Hu74], who referred to the problem as the Optimum Distance Spanning Tree Problem. In Hu’s formulation of the more general Optimum Communication Spanning Tree (OCT) problem [Hu74], an additional value $\tau_{u,v} \geq 0$ is given for each pair $(u,v)$ of vertices. The communication cost [Hu74] of a spanning tree $T$ of $G$ is $\sum_{u,v \in V} \tau_{u,v} d_T(u,v)$. The OCT problem asks for a spanning tree of $G$ with the minimum communication cost. When $G$ is a complete graph and the edge-weight function $w$ obeys the triangle inequality, a randomized $O(\log |V|)$-approximation algorithm is known for the OCT problem [Bar98, CCGG98, WLB+00, FRT03]. The MRCT problem is the special case of the OCT problem when $\tau_{u,v} = 1$ for all $u,v \in V$.

The MRCT problem has applications in network design [Hu74, JLK78] as well as multiple sequences alignment in computational biology [FD87, Pev92, Gus93, BLP94, WLB+00]. Unfortunately, it is shown to be $NP$-hard [JLK78], and it is $NP$-hard even when all edge weights are equal [JLK78, GJ79] or when the edge-weight function obeys the triangle inequality [WLB+00].

Exact and approximation algorithms for the MRCT problem have been extensively researched [BFW73, Hoa73, DF79, Won80, WCT00b, WLB+00, FLS02]. Boyce et al. [BFW73], Hoang [Hoa73] and Dionne and Florian [DF79] study branch-and-bound algorithms as well as heuristic approximation algorithms for the Optimal Network Design problem [BFW73], which includes the MRCT problem as a special case. Fischetti et al. [FLS02] give exact algorithms for the MRCT problem while avoiding exhaustive search. Wong [Won80] gives a polynomial-time 2-approximation algorithm for the MRCT problem. That is, he gives a polynomial-time algorithm that, given a graph $G = (V,E)$ with a nonnegative edge-weight function $w$, outputs a spanning tree of $G$ whose routing cost is at most 2 times the minimum. Subsequent work by Wu et al. [WCT00b] shows a different polynomial-time 2-approximation algorithm as well as polynomial-time 15/8, 3/2 and $(4/3 + \epsilon)$-approximation algorithms for the MRCT problem, where $\epsilon > 0$ is an arbitrary constant. Their results are later improved by Wu et al. [WLB+00] to give a polynomial-time approximation scheme (PTAS) [CLRS01] for the MRCT problem. That is, a polynomial-time $(1 + \epsilon)$-approximation algorithm is given for any constant $\epsilon > 0$.

There are other variants of the MRCT problem that also have applications in network design [WLB+00, WCT00a, WCT00c, Wu02]. In the Sum-Requirement Optimal Communication Spanning Tree (SROCT) problem [WCT00a], each vertex $u$ is associated with a requirement $r(u) \geq 0$. The objective is to find a spanning tree $T$ of $G$ minimizing $\sum_{u,v \in V} (r(u) + r(v)) d_T(u,v)$. The Product-Requirement Optimal Communication Spanning Tree (PROCT) [WCT00a] problem is to find a spanning tree $T$ of $G$ minimizing $\sum_{u,v \in V} r(u) r(v) d_T(u,v)$. The SROCT and PROCT problems are clearly generalizations of the MRCT problem.

Wu et al. [WCT00a] give a 2-approximation algorithm for the SROCT problem. They also propose a 1.577-approximation algorithm [WCT00a] for
the PROCT problem. The result is improved by Wu et al. [WCT00c] to yield a polynomial-time approximation scheme (PTAS) for the PROCT problem.

Another variant of the MRCT problem is the 2-MRCT problem [Wu02]. In this problem, except for $G = (V,E)$ and $w : E \rightarrow \mathbb{R}^{+}_0$, we are given two source vertices $s_1, s_2 \in V$. The objective is to find a spanning tree $T$ of $G$ minimizing $\sum_{v \in V} d_T(s_1, v) + d_T(s_2, v)$. This problem is $NP$-hard even when $w$ obeys the triangle inequality [Wu02]. Wu [Wu02] shows a 2-approximation algorithm as well as a PTAS for this problem. A variant of the 2-MRCT problem is the weighted 2-MRCT problem [Wu02] where an additional $\lambda \geq 1$ is given as input. The objective is to find a spanning tree $T$ of $G$ minimizing $\sum_{v \in V} \lambda d_T(s_1, v) + d_T(s_2, v)$.

Wu [Wu02] proposes a 2-approximation algorithm for the weighted 2-MRCT problem. When the edge-weight function $w$ obeys the triangle inequality, there is a PTAS for the weighted 2-MRCT problem [Wu02].

In this paper, however, we will focus on parallelizing the approximation algorithms for the above problems. We first describe our results concerning the MRCT problem. For an arbitrary $\epsilon > 0$ and when the edge-weight function $w$ is given in unary, we show that the $(4/3 + \epsilon)$-approximation algorithm proposed by Wu et al. [WCT00b] can be implemented by an $RNC$ circuit. That is, the approximation algorithm can be performed by a uniform polynomial-size circuit [Pap94] with random gates and poly-logarithmic depth. Indeed, with a small probability our parallelized algorithm may fail to find a $(4/3 + \epsilon)$-approximate solution, in which case it outputs “fail.” Thus, our algorithm does not fail (to find a $(4/3 + \epsilon)$-approximate solution) without ever knowing that it fails, which is a desirable property for randomized algorithms with a small probability of failure.

We now turn to describe our results concerning the SROCT problem. When the edge-weight and the vertex-requirement functions are given in unary, we parallelize the 2-approximation algorithm [WCT00a] by realizing it using $RNC$ circuits, with a slight degradation in the approximation ratio (from the currently best 2 to our $2 + o(1)$). Still, with a small probability our algorithm may fail to output a $(2 + o(1))$-approximate solution, in which case it knows the failure and outputs “fail.”

Finally, for the weighted 2-MRCT problem with the edge-weight function given in unary, we parallelize the 2-approximation algorithm [Wu02] into $RNC$ circuits, with a slight degradation in the approximation ratio (from the currently best 2 to our $2 + o(1)$). Again, there is a small probability that our algorithm fails to find a $(2 + o(1))$-approximate solution, in which case it outputs “fail.”

To the best of our knowledge, our results are the first efforts towards parallelized approximation algorithms for the MRCT problem and its variants. Our results open up new opportunities to compute approximate solutions to the above problems in parallel poly-logarithmic time. In the applications of the MRCT problem to network design [Hu74, JLK78] as well as the applications of the SROCT and PROCT problems to network design [WCT00a, WCT00c], the network is often modeled as a graph with a nonnegative edge-weight function representing the distances between pairs of nodes. Although approximate solutions to the aforementioned problems (MRCT, SROCT, PROCT, weighted 2-MRCT) are attainable in polynomial time, in any real networking environ-
ment, however, the cost of traffic between any pair of nodes may vary over time. Thus, it is highly desirable to be able to compute approximate solutions to these problems as fast as possible, so as to reduce the risk that the traffic costs change during the computation. Our results imply that approximate solutions to the MRCT, SROCT and weighted 2-MRCT problems can indeed be computed in parallel poly-logarithmic time on multiprocessors.

For other applications of the MRCT problem where the data does not change quickly over time, for example multiple sequences alignment in computational biology [FD87, Pev92, Gus93, BLP94, WLB+00], being able to compute approximate solutions to the MRCT problem in parallel sublinear time is still beneficial. Indeed, Fischer [Fis01] argues that in many practical applications today, the input size is so large that even performing linear-time computations is too time-consuming. Certainly, multiple sequences alignment in computational biology constitutes a good example where the input size is usually too large. It is therefore a desirable property that our algorithms operate in parallel sublinear time, and in fact poly-logarithmic time.

The main idea underlying our proofs is that many of the previously proposed approximation algorithms for the MRCT, SROCT and weighted 2-MRCT problems rely heavily on finding shortest paths between pairs of vertices in a graph. This motivates applying the well-known result that $NL \subseteq NC$ to parallelize these algorithms since we can guess a path (possibly the shortest one) between two vertices of a graph in nondeterministic logarithmic space. There is the complication that, in our proofs, we will often need to generate the same shortest path between two vertices $u, v$, whenever a shortest $u-v$ path is needed. For this purpose, we use the isolation technique [Wig94, GW96, RA00] to slightly modify the edge-weight function of the input graph, so that there is exactly one shortest path between each pair of vertices with high probability. We then apply the double-counting technique [RA00] to decide whether the input graph (with the modified edge-weight function) exhibits a unique shortest path between each pair $u, v$ of vertices. If so, we are able use the double counting technique to generate the unique shortest $u-v$ path whenever it is needed. The whole procedure runs in unambiguous logarithmic space and our results follow by $UL \subseteq NL \subseteq NC$. The approximation ratio would be slightly degraded. The degradation comes from randomly modifying the edge-weight function when we apply the isolation technique.

Our paper is organized as follows. Section 2 provides the basic definitions. Section 3 presents the parallelized $(4/3 + \epsilon)$-approximation algorithm for the MRCT problem. Section 4–5 describe our parallelized approximation algorithms for the SROCT and the weighted 2-MRCT problems, respectively. Section 6 concludes the paper. Proofs are given in the appendix for references.

2 Notations and basic facts

Throughout this paper, graphs are simple undirected graphs [Wes01]. That is, we disallow parallel edges and self-loops. There will always be a nonnegative edge-weight function mapping each edge to a nonnegative real number. For a
graph \( G \), \( V(G) \) is its vertex set and \( E(G) \) is its edge set. Let \( R \) be a subgraph of \( G \). A path \( P \) connects a vertex \( v \) to \( R \) (or \( V(R) \)) if one endpoint of \( P \) is \( v \) and the other is in \( V(R) \). An edge connecting two vertices \( u \) and \( v \) is denoted \( uv \). A path connecting two vertices \( u \) and \( v \) is said to be a \( u-v \) path. A path \((v_0, \ldots, v_k)\) is one which traverses \( v_0, \ldots, v_k \), in that order. A simple path is a path that traverses each vertex at most once [Wes01]. A graph \( G \) contains another graph \( G' \) if \( G' \) is a subgraph of \( G \). The set of nonnegative real numbers is denoted \( \mathbb{R}_0^+ \).

**Definition 1.** Let \( G = (V, E) \) be an undirected graph and \( w : E \to \mathbb{R}_0^+ \) be a nonnegative edge-weight function. The lexicographical ordering on \( V \) is that of the encodings of the vertices in \( V \), assuming any reasonable encoding of a graph. Let \( u, v \in V \) and \( R \) be a subgraph of \( G \). The sum of edge weights of \( R \) is denoted \( w(R) \). When \( R \) is a path, \( w(R) \) is called the weight or length of \( R \). For \( x, y \in V \), we use \( d_G(x, y) \) to denote the weight of any shortest \( x-y \) path. We use \( d_G(x, R) \) (or \( d_G(x, V(R)) \)) for \( \min_{z \in V(R)} d_G(x, z) \). The lexicographically first vertex \( x' \in V(R) \) satisfying \( d_G(x, x') = d_G(x, R) \) is denoted \( \text{closest}(x, R) \) (or \( \text{closest}(x, V(R)) \)). The set of all shortest paths connecting \( u \) and \( v \) is denoted \( SP_G(u, v) \). \( SP_G(u, R) \) (or \( SP_G(u, V(R)) \)) is the set of shortest paths connecting \( u \) and \( R \). That is, \( SP_G(u, R) \) is the set of paths that connect \( u \) and \( R \) and have weight equal to \( d_G(u, R) \). For \( k \in \mathbb{N} \), \( S_{k,u} \) denotes the set of vertices reachable from \( u \) with at most \( k \) edges. \( SP_{u,v}^{(k)} \) denotes the set of all shortest paths among those \( u-v \) paths with at most \( k \) edges. That is, \( SP_{u,v}^{(k)} \) is the set of \( u-v \) paths with at most \( k \) edges whose weight is not larger than any other path with at most \( k \) edges. The union of two graphs \( G_1 = (V_1, E_1) \) and \( G_2 = (V_2, E_2) \) is the graph \((V_1 \cup V_2, E_1 \cup E_2)\). The graph \( G \) is strongly min-unique with respect to \( w \) if for all \( k \in \mathbb{N} \) and \( u, v \in V \), we have \( |SP_{u,v}^{(k)}| \leq 1 \). When \( w \) is clear from the context, we may simply say that \( G \) is strongly min-unique without referring to \( w \).

In Definition 1, it is not hard to show that \( G = (V, E) \) is strongly min-unique if \( |SP_{u,v}^{(k)}| \leq 1 \) for all \( k \in \{0, \ldots, |V| - 1\} \) and \( u, v \in V \), provided \( |V| \geq 3 \).

The MRCT of a graph, standing for its Minimum Routing Cost spanning Tree, is defined below.

**Definition 2.** ([WC04]) Given a connected graph \( G = (V, E) \) with a nonnegative edge-weight function \( w : E \to \mathbb{R}_0^+ \), the routing cost \( c_w(T) \) of a spanning tree \( T \) of \( G \) is \( \sum_{u,v \in V} d_T(u,v) \). A spanning tree of \( G \) with the minimum routing cost is an MRCT of \( G \), which is denoted by \( \text{MRCT}(G) \) for convenience.

The MRCT problem asks for \( \text{MRCT}(G) \) on input \( G, w \). The following fact shows that the routing cost of a tree can be computed efficiently.

**Fact 3.** ([WC04]) Let \( G \) be a graph with a nonnegative edge-weight function \( w : E(T) \to \mathbb{R}_0^+ \) and \( T \) be a spanning tree of \( G \). For each edge \( e \in E(T) \), let \( T_{e,1} \) and \( T_{e,2} \) be the two trees formed by removing \( e \) from \( T \). We have \( c_w(T) = \sum_{e \in E(T)} 2|V(T_{e,1})||V(T_{e,2})| w(e) \) and \( c_w(T) \leq |V(T)|^3/2 \cdot \max_{e \in E(T)} w(e) \).

To ease the description, we introduce the following definition.
Definition 4. ([RA00]) A nondeterministic Turing machine $M$ outputs a string $s$ unambiguously on input $x$ if it outputs $s$ on exactly one non-rejecting computation branch, and rejects $x$ on all other computation branches. The unambiguously output string $s$ is also denoted $M(x)$.

Throughout this paper, when a nondeterministic Turing machine $A$ runs or simulates another nondeterministic machine $B$, it means that $A$ runs $B$ and make nondeterministic branches as $B$ does. It does not mean that $A$ enumerates all computation branches of $B$ and simulate them deterministically. For convenience, $A$ does not necessarily have to output $B$’s output. Instead, it may extract portions of $B$’s output for output.

We will need the notion of a general star to introduce the approximation algorithms for the MRCT problem.

Definition 5. ([WC04]) Let $G$ be a connected graph with a nonnegative edge-weight function $w : E \to \mathbb{R}_0^+$ and $S$ be a subtree of $G$. A spanning tree $T$ containing $S$ is a general star with core $S$ if each vertex $u \in V$ satisfies $d_T(u, S) = d_G(u, S)$. When $V(S) = \{v\}$ is a singleton, a general star with core $S$ is also called a shortest path tree rooted at $v$.

Given any subtree $S$ of $G = (V, E)$, a general star with core $S$ exists [WCT00b]. This follows by observing that for any shortest path $P$ connecting $u \in V$ and $S$, the part of $P$ from any vertex $x \in V(P)$ to $S$ constitutes a shortest path connecting $x$ and $S$.

The notion of a metric graph is defined below.

Definition 6. ([WLB+00]) A complete graph $G$ with a non-negative edge-weight function $w$ is metric if $w(xy) + w(yz) \geq w(xz)$ for all $x, y, z \in V(G)$.

3 A parallelized $(4/3 + \epsilon)$-approximation for MRCT

We begin with the following form of the famous isolation lemma. It is implicit in some previous works [Wig94, GW96, RA00].

Theorem 7. ([RA00]) Let $G = (V, E)$ be a graph with a nonnegative edge-weight $w : E \to \mathbb{R}_0^+$ and $w_r : E \to \mathbb{R}_0^+$ assign the weight of each $e \in E$ independently and randomly from the uniform distribution over a set $W \subseteq \mathbb{R}_0^+$. With probability at least $1 - |V|^5/(2|W|)$, the graph $G$ is strongly min-unique with respect to $w + w_r$.

The following theorem is implicit in [RA00]. It uses the double counting technique [RA00] similar to the inductive counting technique used to prove the Immerman-Szelepcsényi theorem [Imm88, Sze88].

Theorem 8. ([RA00]) There is a nondeterministic logarithmic-space Turing machine FIND-PATH that, on input a graph $G = (V, E)$ with a nonnegative edge-weight function $w : E \to \{0, \ldots, \text{poly}(|V|)\}$ and two vertices $s, t \in V$, satisfies the following conditions.
1. If \( G \) is not strongly min-unique, then FIND-PATH outputs “not strongly min-unique” unambiguously.

2. If \( G \) is strongly min-unique and has an \( s\)-\( t \) path, then FIND-PATH outputs the unique path \( P \in \text{SP}_G(s, t) \) and its weight \( w(P) \) unambiguously. The edges in \( P \) are output in the direction going from \( s \) to \( t \).

3. If \( G \) is strongly min-unique and does not have an \( s\)-\( t \) path, then FIND-PATH has no accepting computation branches.

The following theorem is due to Wu et al. [WCT00b].

**Theorem 9.** ([WCT00b]) Let \( r \in \mathbb{N} \) be a constant and \( G = (V, E) \) be a connected, strongly min-unique graph with a nonnegative edge-weight function \( w : E \to \mathbb{R}_0^+ \). For \( 1 \leq k \leq r + 4 \) and \( S = (v_1, \ldots, v_k) \in V^k \), let \( R_{1,S} \) be the subgraph of \( G \) containing only \( v_1 \). For \( 2 \leq i \leq k \), let \( R_{i,S} = R_{i-1,S} \cup P_{i,S} \) where \( P_{i,S} \in \text{SP}_G(v_i, \text{closest}(v_i, R_{i-1,S})) \) is the unique shortest path connecting \( v_i \) and \( \text{closest}(v_i, R_{i-1,S}) \). For some \( 1 \leq k \leq r + 4 \) and \( S \in V^k \), every general star \( T \) with core \( R_{k,S} \) satisfies

\[
c_w(T) \leq \left( \frac{4}{3} + \frac{8}{9r+2} \right) c_w(\text{MRCT}(G)).
\]

That \( R_{i,S} \) in Theorem 9 is a subtree of \( G \) for \( 2 \leq i \leq k \) is easily shown because \( w(e) > 0 \) for each \( e \in E \) by the strong min-uniqueness of \( G \). The core \( R_{k,S} \) in Theorem 9 is unambiguously computable in logarithmic space on strongly min-unique connected graphs. To show this, we need the following lemma.

**Lemma 10.** There is a nondeterministic logarithmic-space Turing machine ADD-PATH that, on input a strongly min-unique connected graph \( G = (V, E) \) with a nonnegative edge-weight function \( w : E \to \{0, \ldots, \text{poly}(|V|)\} \), a subgraph \( R \) of \( G \) and a vertex \( v \in V \), outputs the unique path \( P \in \text{SP}_G(v, \text{closest}(v, R)) \) unambiguously.

With Lemma 10, we are able to compute the core \( R_{k,S} \) in Theorem 9 unambiguously in logarithmic space on strongly min-unique connected graphs.

**Lemma 11.** Let \( r \in \mathbb{N} \) be a constant. There is a nondeterministic logarithmic-space Turing machine CORE that, on input a strongly min-unique connected graph \( G = (V, E) \) with a nonnegative edge-weight function \( w : E \to \{0, \ldots, \text{poly}(|V|)\} \) and \( S = (v_1, \ldots, v_k) \in V^k \) where \( 1 \leq k \leq r + 4 \), unambiguously outputs \( R_{k,S} \) defined below. \( R_{1,S} \) is the subgraph of \( G \) containing only \( v_1 \). For \( 2 \leq i \leq k \), \( R_{i,S} = R_{i-1,S} \cup P_{i,S} \) where

\( P_{i,S} \in \text{SP}_G(v_i, \text{closest}(v_i, R_{i-1,S})) \)

is the unique shortest path connecting \( v_i \) and \( \text{closest}(v_i, R_{i-1,S}) \).

With Theorem 9 and Lemma 11, it is not hard to show the following fact.
Fact 12. Let \( r \in \mathbb{N} \) be a constant and \( G = (V, E) \) be a strongly min-unique, connected graph with a nonnegative edge-weight function \( w : E \to \{0, \ldots, \text{poly}(|V|)\} \). For a sequence \( S \) of at most \( r + 4 \) vertices in \( V \), let \( C_S = \text{CORE}(G, w, S) \) and \( P_u \in \text{SP}_G(u, \text{closest}(u, C_S)) \) for \( u \in V \setminus V(C_S) \). Then

\[
T_S = C_S \cup \bigcup_{u \in V \setminus V(C_S)} P_u
\]

is a general star with core \( S \), and

\[
c_w(T_S) < \left( \frac{4}{3} + \frac{8}{9r + 12} \right) \cdot c_w(\text{MRCT}(G))
\]

for some \( S \).

The general star with a core in Fact 12 can be computed unambiguously in logarithmic space, as the next lemma shows.

Lemma 13. Let \( r \in \mathbb{N} \) be a constant. There is a nondeterministic logarithmic-space Turing machine \( \text{STAR} \) that, on input a strongly min-unique connected graph \( G = (V, E) \) with a nonnegative edge-weight function \( w : E \to \{0, \ldots, \text{poly}(|V|)\} \) and a sequence \( S \) of at most \( r + 4 \) vertices in \( V \), outputs \( C_S = \text{CORE}(G, w, S) \) and each unique path in \( \text{SP}_G(u, \text{closest}(u, C_S)) \) for \( u \in V \setminus V(C_S) \) unambiguously.

The following lemma allows unambiguous logarithmic-space computation of the routing cost of a tree.

Lemma 14. There is a nondeterministic logarithmic-space Turing machine \( \text{ROUT-PAIR} \) that, on input a tree \( T \) with a nonnegative edge-weight function \( w : E(T) \to \{0, \ldots, \text{poly}(|V(T)|)\} \) and \( s, t \in V(T) \), unambiguously outputs the unique simple path \( P^* \) connecting \( s \) and \( t \) in \( T \) and \( w(P^*) \).

Combining Fact 12 and Lemmas 13–14 gives the following lemma.

Lemma 15. Let \( r \in \mathbb{N} \) be a constant. There exists a nondeterministic logarithmic-space Turing machine \( \text{APPROX} \) that, on input a strongly min-unique connected graph \( G = (V, E) \) with a nonnegative edge-weight function \( w : E \to \{0, \ldots, \text{poly}(|V|)\} \), unambiguously outputs a spanning tree \( T \) of \( G \) with

\[
c_w(T) < \left( \frac{4}{3} + \frac{8}{9r + 12} \right) \cdot c_w(\text{MRCT}(G))
\]

for some \( S \).

The following lemma will be useful.

Lemma 16. Let \( \alpha > 0 \) be a constant. Let \( G = (V, E) \) be a graph with a nonnegative edge-weight function \( w : E \to \mathbb{R}_0^+ \), and the minimum nonzero weight assigned by \( w \), if it exists, is at least 1. Let \( T_1 \) and \( T_2 \) be spanning trees of \( G \). Let \( w_r \) assign to each edge \( e \in E \) a nonnegative weight \( w(e) \leq 1/|V|^4 \) and \( w' = w + w_r \). Then

\[
c_{w'}(T_1) \leq \alpha c_{w'}(T_2)
\]

(1)
implies
\[ c_w(T_1) \leq \alpha \left(1 + \frac{1}{2|V|}\right) c_w(T_2) \]  
for sufficiently large \(|V|\).

Combining Theorem 7 and Lemma 15–16 yields the following theorem.

**Theorem 17.** Let \( \epsilon > 0 \) be a constant. There is an RNC\(^2\) algorithm PARALLEL that, on input a weighted undirected graph \( G = (V, E) \) with a nonnegative edge-weight function \( w : E \to \{0, \ldots, \text{poly}(|V|)\} \), satisfies the following.

1. If \( G \) is disconnected, then PARALLEL\((G, w)\) outputs “disconnected.”

2. If \( G \) is connected, then PARALLEL\((G, w)\) outputs a spanning tree of \( G \) unambiguously or outputs “fail” unambiguously. The probability that PARALLEL\((G)\) outputs a spanning tree \( T \) of \( G \) unambiguously is at least \( 1 - \frac{1}{2|V|} \). If PARALLEL\((G)\) outputs a spanning tree \( T \) of \( G \) unambiguously, then
\[ c_w(T) \leq \left(\frac{4}{3} + \epsilon\right) c_w(MRCT(G)). \]

### 4 The SROCT problem

We begin this section with the following definition.

**Definition 18.** ([WCT00a, Wu02]) Let \( G = (V, E) \) be a graph with a nonnegative edge-weight function \( w : E \to \mathbb{R}_{\geq 0}^+ \) and \( r : V \to \mathbb{R}_{\geq 0}^+ \) be a requirement function on vertices. Let \( s_1, s_2 \in V \) be two vertices of \( G \) and \( T \) be a spanning tree of \( G \). The sum-requirement communication (s.r.c.) cost of \( T \) is
\[ c_w^{(s)}(T) = \sum_{u,v \in V} (r(u) + r(v)) d_T(u, v). \]

**The Sum-Requirement Optimal Communication Spanning Tree (SROCT) problem** is to find a spanning tree \( T \) of \( G \) with the minimum value of \( c_w^{(s)}(T) \) over all spanning trees of \( G \). We use SROCT\((G)\) to denote an arbitrary spanning tree of \( G \) with the minimum s.r.c. cost. The two-source routing cost of \( T \) with sources \( s_1, s_2 \) is
\[ c_w^{(2)}(T, s_1, s_2) = \sum_{v \in V} (d_T(s_1, v) + d_T(s_2, v)). \]

The 2-MRCT problem is to find a spanning tree \( T \) of \( G \) with the minimum value of \( c_w^{(2)}(T, s_1, s_2) \) over all spanning trees of \( G \) (in this problem \( s_1 \) and \( s_2 \) are part of the input). We use 2-MRCT\((G)\) to denote an arbitrary spanning tree of \( G \) with the minimum two-source routing cost when the sources \( s_1, s_2 \) are clear from
the context. Let $\lambda \geq 1$. The weighted two-source routing cost of $T$ with sources $s_1, s_2$ and weight $\lambda$ is

$$c_w^{(2)}(T, s_1, s_2, \lambda) = \sum_{v \in V} (\lambda d_T(s_1, v) + d_T(s_2, v)).$$

The weighted 2-MRCT problem is to find a spanning tree $T$ of $G$ with the minimum value of $c_w^{(2)}(T, s_1, s_2, \lambda)$ over all spanning trees of $G$ (in this problem $s_1, s_2$ and $\lambda$ are part of the input). We use $W$-2-MRCT($G$) to denote an arbitrary spanning tree of $G$ with the minimum weighted two-source routing cost when $\lambda$ and the sources $s_1, s_2$ are clear from the context.

The SROCT, 2-MRCT and weighted 2-MRCT problems are all $NP$-hard, even on metric graphs [WLB+00, WCT00a, WCT00c, Wu02].

The following theorem gives a 2-approximation solution to the SROCT problem.

**Theorem 19.** ([WCT00a]) Let $G = (V, E)$ be a connected graph with a nonnegative edge-weight function $w$ and a nonnegative vertex-requirement function $r$. There exists a vertex $x \in V$ such that any shortest path tree $T$ rooted at $x$ satisfies

$$c_w^{(s)}(T) \leq 2c_w^{(s)}(SROCT(G)).$$

Theorems 7–8, 19 and Lemma 14 give the following parallelized 2-approximation solution to the SROCT problem.

**Theorem 20.** There is an $RNC^2$ algorithm PARALLEL-SROCT that, on input a connected graph $G = (V, E)$ with a nonnegative edge-weight function $w : E \to \{0, \ldots, \text{poly}(|V|)\}$ and a nonnegative vertex-requirement function $r : V \to \{0, \ldots, \text{poly}(|V|)\}$, outputs a spanning $T$ of $G$ with

$$c_w^{(s)}(T) \leq (2 + o(1)) c_w^{(s)}(SROCT(G))$$

with high probability. If PARALLEL-SROCT does not output such a spanning tree, it outputs “fail.”

## 5 Weighted 2-MRCT problem

For the weighted 2-MRCT problem, we can assume without loss of generality that the two sources $s_1, s_2$ are such that $d_G(s_1, s_2) > 0$, where $G$ is the input graph. Otherwise, the problem reduces to finding a shortest path tree rooted at $s_1$, which was implicitly done in the proof of Theorem 20. Wu [Wu02] has the following 2-approximation solution for the weighted 2-MRCT problem.

**Theorem 21.** ([Wu02]) Let $G = (V, E)$ be a connected graph with a nonnegative edge-weight function $w : E \to \mathbb{R}^+_0$, two sources $s_1, s_2 \in V$ with $d_G(s_1, s_2) > 0$ and $\lambda \geq 1$. Denote

$$D_1(v) = (\lambda + 1) d_G(v, s_1) + d_G(s_1, s_2)$$
and
\[ D_2(v) = (\lambda + 1) d_G(v, s_2) + \lambda d_G(s_1, s_2) \]
for \( v \in V \). Let \( Z_1^w = \{ v \mid D_1(v) \leq D_2(v) \} \) and \( Z_2^w = V \setminus Z_1^w \). Let \( Q \in SP_G(s_1, s_2) \) be arbitrary. Denote
\[ Q = (q_0 = s_1, \ldots, q_j, q_j+1, \ldots, s_2) \]
where \( q_{j+1} \) is the first vertex on \( Q \) (in the direction from \( s_1 \) to \( s_2 \)) that is not in \( Z_1^w \) (it is easy to see that \( s_1 \in Z_1^w \)). For each \( v \in V \), let \( P_{v,s_1} \in SP_G(v, s_1) \) and \( P_{v,s_2} \in SP_G(v, s_2) \) be arbitrary. If \( T_1 = \bigcup_{v \in Z_1^w} P_{v,s_1} \) and \( T_2 = \bigcup_{v \in Z_2^w} P_{v,s_2} \) are trees, then \( T = T_1 \cup T_2 \cup q_jq_{j+1} \) is a spanning tree of \( G \) and
\[ c_w^{(2)}(T) \leq 2c_w^{(2)}(W-2-MRCT(G)) \).

Theorems 7–8 and 21 and Lemma 14 yield the following theorem.

**Theorem 22.** There is an \( \mathcal{RNC}^2 \) algorithm WEIGHTED-2-MRCT that, on input a graph \( G = (V, E) \) with a nonnegative edge-weight function \( w : E \to \{0, \ldots, \text{poly}(|V|)\} \), \( s_1, s_2 \in V \) and \( \lambda \geq 1 \), with high probability outputs a spanning tree \( T \) with
\[ c_w^{(2)}(T) \leq (2 + o(1)) c_w^{(2)}(W-2-MRCT(G)) \).

If WEIGHTED-2-MRCT does not output such a spanning tree, it outputs “fail.”

We make the following concluding remark. All our algorithms are shown to be \( \mathcal{RNC}^2 \)-computable by showing that they run in unambiguous logarithmic space and succeed in giving an approximate solution when the random input specifies an edge-weight function \( w_r \) such that \( G \) is strongly min-unique with respect to \( w + w_r \). By a method similar to that in [RA00], we can also turn the random weight assignment into polynomially long advices. This is summarized below.

**Corollary 23.** Let \( \epsilon > 0 \) be a constant. There are \( \mathcal{UL}/\text{poly} \) algorithms for \((4/3+\epsilon)\)-approximating the MRCT problem, \((2+o(1))\)-approximating the SROCT problem and \((2+o(1))\)-approximating the weighted 2-MRCT problem, where the respective edge-weight and vertex-requirement functions are given in unary.

**6 Conclusion**

We have given parallelized approximation algorithms for the minimum routing cost spanning tree problem and some of its variants. Our results show that, by exhibiting multiple processors, we can compute approximate solutions to the considered problems in parallel poly-logarithmic time. We hope this will shed light on the many areas in which the considered problems are concerned, for example network design [Hu74, JLK78] and multiple sequences alignment in computational biology [FD87, Pev92, Gus93, BLP94, WLB00].
Acknowledgments

The authors are grateful to Wen-Hui Chen for his helpful comments and suggestions.

Appendix

Proof of Theorem 7. The theorem is clearly true for $|V| \leq 2$. If $G$ is not strongly min-unique with respect to $w + w_r$ and $|V| \geq 3$, we have seen that there exist $0 \leq k \leq |V| - 1, s, t \in V$ such that $|\text{SP}^{(k)}_{s,t}| \geq 2$ where the edge weights are given with respect to $w + w_r$. This implies the existence of an edge $e \in E$ such that at least one path in $\text{SP}^{(k)}_{s,t}$ contains $e$, and at least one does not. In this case we say that $(k, s, t)$ blames $e$. Thus, the probability that $G$ is not strongly min-unique is at most the sum over $0 \leq k \leq |V| - 1, s, t \in V$ and $e \in E$ of the probability that $(k, s, t)$ blames $e$.

For any $k \in \{0, \ldots, |V| - 1\}$, $s, t \in V$, $e \in E$ and any partial weight assignment of $w_r$ to $E \setminus \{e\}$, there is at most the sum over $0 \leq k \leq |V| - 1, s, t \in V$ and $e \in E$ of the probability that $(k, s, t)$ blames $e$.

Proof of Theorem 8. Before describing how FIND-PATH works, we describe a few procedures that are useful. For $k \in \mathbb{N}$, let $c_k = |S_{k,s}|$. Let $P_{s,v}^{(k)} \in \text{SP}_{s,v}^{(k)}$ be arbitrary and $\Sigma_k = \sum_{v \in S_{k,s}} w(P_{s,v}^{(k)})$. Note that the definition of $\Sigma_k$ does not depend on exactly which path in $\text{SP}_{s,v}^{(k)}$ is chosen as $P_{s,v}^{(k)}$. It is clear that $c_0 = 1$ and $\Sigma_0 = 0$.

We first introduce a nondeterministic logarithmic-space subroutine OUTPUT that outputs $S_{k,s}$ unambiguously, given $G, w, s, c_k, \Sigma_k$ and that $|\text{SP}_{s,v}^{(k)}| = 1$ for each $v \in S_{k,s}$. OUTPUT just needs to nondeterministically guess each vertex $x$ to be in or out of $S_{k,s}$, and if the guess is $x \in S_{k,s}$ then it outputs $x$. It verifies each guess of $x \in S_{k,s}$ by nondeterministically guessing an s-x path with at most $k$ edges and rejecting if it fails. Along the way OUTPUT counts the number $c'_{k}$ of vertices verified to be in $S_{k,s}$ and accumulates the weights of the guessed s-x paths (for $x$ verified to be in $S_{k,s}$) in a variable $\Sigma'_k$. It then rejects if $c'_{k} \neq c_k$ or $\Sigma'_k \neq \Sigma_k$. Clearly, guessing any vertex out of $S_{k,s}$ to be in $S_{k,s}$ results
in rejection. For a computation branch of OUTPUT not to reject, it must have \( c'_k \) reach \( c_k \), which requires successfully guessing an \( s-x \) path with at most \( k \) edges for each \( x \in S_{k,s} \). But to have \( \Sigma'_k \) not exceed \( \Sigma_k \), the guessed \( s-x \) path for each \( x \in S_{k,s} \) should be the unique one in \( \text{SP}^{(k)}_{s,x} \). So OUTPUT\((G, w, s, c_k, \Sigma_k)\) has a unique non-rejecting computation branch, on which it correctly guesses whether each vertex \( x \) belongs to \( S_{k,s} \) and if so, correctly guesses the unique path in \( \text{SP}^{(k)}_{s,x} \).

We now describe a procedure \text{INDUCTIVE} that computes \( c_{k+1} \) and \( \Sigma_{k+1} \), and determines whether \(| \text{SP}^{(k+1)}_{s,x} | > 1 \) for some \( x \in V \) unambiguously, given \( c_k \) and \( \Sigma_k \) and that \(| \text{SP}^{(k)}_{s,v} | = 1 \) for each \( v \in S_{k,s} \). For each vertex \( x \in V \), INDUCTIVE runs OUTPUT\((G, w, s, c_k, \Sigma_k)\) to determine whether \( x \in S_{k,s} \) unambiguously and if so, accumulates the weight \( w_x \) of the unique path in \( \text{SP}^{(k)}_{s,x} \) as it is guessed by OUTPUT. For each \( u \) such that \( ux \) is an edge, INDUCTIVE also determines whether \( u \in S_{k,s} \) unambiguously and if so, accumulates the weight \( w_u \) of the unique path in \( \text{SP}^{(k)}_{s,u} \) as it is guessed. INDUCTIVE then computes the weight of any shortest \( s-x \) path with at most \( k + 1 \) edges as

\[
\min Q
\]

where

\[
Q = \{ w_x \} \cup \{ w_u + w(ux) \mid u \in S_{k,s}, ux \in E \}
\]

if \( x \in S_{k,s} \) and

\[
Q = \{ w_u + w(ux) \mid u \in S_{k,s}, ux \in E \}
\]

otherwise. If \( Q = \emptyset \), INDUCTIVE knows that \( x \notin S_{k+1,s} \). Otherwise \( x \in S_{k+1,s} \) and the weight of any path in \( \text{SP}^{(k+1)}_{s,x} \) is known to be \( \min Q \). In case of a tie when computing \( \min Q \), INDUCTIVE knows that \(| \text{SP}^{(k+1)}_{s,x} | > 1 \). Doing the above for all \( x \in V \) allows INDUCTIVE to compute \( c_{k+1} = | S_{k+1,s} | \) and \( \Sigma_{k+1} \) and determine whether \(| \text{SP}^{(k+1)}_{s,x} | > 1 \) for some \( x \in V \) unambiguously. During the computation of \( \min Q \), INDUCTIVE does not store the set \( Q \). Instead, INDUCTIVE computes the elements of \( Q \) one by one and stores the smallest element in \( Q \) that has been computed so far, as well as a flag indicating whether \( \min Q \) is achieved by two elements at any time.

We are now ready to describe how FIND-PATH works. Assume \(| V | \geq 3 \). FIND-PATH starts with \( c_0, \Sigma_0 \) and repeatedly simulates INDUCTIVE until it computes \( c_{|V|}, \Sigma_{|V|} \) or until it determines that \(| \text{SP}^{(k)}_{s,x} | > 1 \) for some \( 0 \leq k \leq |V| - 1 \) and \( x \in V \). Doing the above with each other vertex \( s' \in V \) replacing the role of \( s \) guarantees that if \( G \) is not strongly min-unique, then we must find \( | \text{SP}^{(k)}_{s',x} | > 1 \) for some \( 0 \leq k \leq |V| - 1 \) and \( s', x \in V \). Instead, if \( G \) is strongly min-unique then FIND-PATH will compute all the way from \( c_0, \Sigma_0 \) to \( c_{|V|}, \Sigma_{|V|} \). It then runs OUTPUT\((G, w, s, c_{|V|-1}, \Sigma_{|V|-1})\). As we have seen, OUTPUT\((G, w, s, c_{|V|-1}, \Sigma_{|V|-1})\) has a unique non-rejecting computation branch, on which the unique shortest \( s-t \) path \( P \in \text{SP}^{(|V|-1)}_{s,t} \) is correctly guessed
by OUTPUT. The weight \( w(P) \) is accumulated along the way. The strong uniqueness of \( G \) guarantees that \( P \) is also the unique path in \( SP_{G}(s, t) \).

Proof of Lemma 10. For each \( x \in V(R) \), ADD-PATH runs \( \text{FIND-PATH}(G, w, x, v) \) to unambiguously generate the unique shortest path \( P_{x,v} \in SP_{G}(x, v) \) and its weight \( w(P_{x,v}) \). In this way, ADD-PATH could compute \( \min_{x \in V(R)} w(P_{x,v}) \) as well as \( \text{closest}(v, R) \). Then ADD-PATH simulates \( \text{FIND-PATH}(G, w, v, \text{closest}(v, R)) \) to output the unique path \( P \) in \( SP_{G}(v, \text{closest}(v, R)) \) unambiguously.

Proof of Lemma 11. Clearly, CORE could output \( R_{1,S} \) unambiguously. Let \( 2 \leq j \leq k \). To output \( R_{j,S} = R_{j-1,S} \cup P_{j,S} \) unambiguously, CORE recursively outputs \( R_{j-1,S} \) and then runs \( \text{ADD-PATH}(G, w, R_{j-1,S}, v_{j}) \) to unambiguously output \( P_{j,S} \). There is an additional complication that CORE does not store \( R_{j-1,S} \) before calling ADD-PATH. Instead, whenever ADD-PATH wants to read any bit encoding \( R_{j-1,S} \), CORE recursively outputs \( R_{j-1,S} \) unambiguously on the fly to support the required bit. Each level of recursion uses up logarithmic space and the depth of recursion is at most \( r + 4 \), a constant. The space requirement is therefore logarithmic.

Proof of Lemma 13. STAR begins by running \( \text{CORE}(G, w, S) \) to output \( C_{S} \) unambiguously. For any \( u \in V \), STAR runs \( \text{CORE}(G, w, S) \) to unambiguously determine whether \( u \in V(C_{S}) \). If \( u \notin V(C_{S}) \), STAR needs to output the unique path in \( SP_{G}(u, \text{closest}(u, C_{S})) \). For this purpose, it computes \( \text{closest}(u, C_{S}) \) as follows. For each \( v \in V \), STAR tests if \( v \in V(C_{S}) \), again by running \( \text{CORE}(G, w, S) \). If \( v \notin V(C_{S}) \), STAR goes on with the next \( v \in V \). Otherwise, STAR invokes \( \text{FIND-PATH}(G, w, u, v) \) to generate the unique path \( P_{u,v} \in SP_{G}(u, v) \) and its weight \( w(P_{u,v}) \) unambiguously. STAR records the \( v \in V(C_{S}) \) that has generated the smallest value of \( w(P_{u,v}) \) so far, favoring lexicographically smaller values of \( v \) in case of a tie. In the end, the recorded \( v \in V \) must be \( \text{closest}(u, C_{S}) \) by the definition of \( \text{closest}(u, C_{S}) \). At this time STAR just invokes \( \text{FIND-PATH}(G, w, u, \text{closest}(u, C_{S})) \) to output the unique path in \( SP_{G}(u, \text{closest}(u, C_{S})) \) unambiguously. Doing the above for all \( u \in V \) does the job.

Proof of Lemma 14. If \( s = t \) the task is trivial. We assume otherwise. ROUT-PAIR nondeterministically guesses a path \( P \) that does not enter a vertex immediately after it has left that vertex. If \( P \) is an \( s-t \) path, then ROUT-PAIR accepts, otherwise it rejects. The simple \( s-t \) path in \( T \) is the only \( s-t \) path that does not enter a vertex immediately after leaving it. Its weight \( w(P^{*}) \) can be accumulated as it is guessed.

Proof of Lemma 15. For each sequence \( S \) of at most \( r+4 \) vertices in \( V \), Lemma 13 enables us to unambiguously output \( C_{S} = \text{CORE}(G, w, S) \) and then each unique path \( P_{u} \) in \( SP_{G}(u, \text{closest}(u, C_{S})) \) for \( u \in V \setminus V(C_{S}) \). Furthermore, Fact 12 guarantees that

\[
T_{S} = C_{S} \cup \bigcup_{u \in V \setminus V(C_{S})} P_{u}
\]
is a spanning tree of $G$ and satisfies
\[
c_w(T_S) < \left( \frac{4}{3} + \frac{8}{9r + 12} \right) \cdot c_w(\text{MRCT}(G))
\]
for some $S$. Thus, we need only compute $c_w(T_S)$ unambiguously for each sequence $S$ of at most $r + 4$ vertices, and output $T_S^*$ unambiguously for the sequence $S^*$ of at most $r + 4$ vertices satisfying $c_w(T_S^*) = \min_{S \in \{V^k, 1 \leq k \leq r + 4\}} c_w(T_S)$. By Definition 2, $c_w(T_S)$ can be computed unambiguously by running $\text{ROUT-PAIR}(T_S, w, s, t)$ for all pairs $s, t \in V$ and summing up the weight of the simple paths as they are output. There is a complication that APPROX does not store $T_S$ before calling ROUT-PAIR. Instead, when ROUT-PAIR wants to read any bit in the encoding of $T_S$, APPROX runs $\text{STAR}(G, w, S)$ to generate the required bit unambiguously on the fly. This enables us to obtain $S^*$ unambiguously and thus $T_S^*$ unambiguously by running $\text{STAR}(G, w, S^*)$.

Proof of Lemma 16. It is clear that either $c_w(T_2) = 0$ or $c_w(T_2) \geq 1$. Also, Fact 3 implies that
\[
c_w'(T_2) \leq \frac{1}{2|V|}.
\]
If $c_w(T_2) = 0$, Eq. (3) implies that
\[
c_w'(T_2) = c_w(T_2) \leq \frac{1}{2|V|}
\]
and thus $c_w'(T_1) < 1$ for sufficiently large $|V|$ by Eq. (1). This implies $c_w(T_1) < 1$ and thus $c_w(T_1) = 0$, establishing Eq. (2).

If $c_w(T_2) \geq 1$, then Eq. (1) and (3) imply
\[
c_w(T_1) \leq c_w'(T_1) \\
\leq \alpha (c_w(T_2) + c_w(T_2)) \\
\leq \alpha (1 + \frac{1}{2|V|}) c_w(T_2).
\]

Proof of Theorem 17. We will show that PARALLEL needs only take a poly($|V|$)-long random input and do the rest of the computation in unambiguous logarithmic space. The standard proof technique for showing that $\mathcal{UL} \subseteq \mathcal{NL} \subseteq \mathcal{NC}^2$ [Pap94, Sip05] then completes the proof.

PARALLEL tests the connectedness of $G$ by testing each pair of vertices for connectedness in logarithmic space [Rei05].

Below we assume that $G$ is connected. If we assume that this theorem is true when $w$ is not identically zero, then PARALLEL can also deal with the identically zero case by using the unit edge-weight function instead. The output spanning tree would have zero routing cost under the identically zero edge-weight function, so item 2 is still satisfied. Thus, we can assume without
loss of generality that \( w \) is not identically zero. Furthermore, we can normalize \( w \) to give \( \min_{e \in E, w(e) \neq 0} w(e) \geq 1 \).

The random input to PARALLEL determines an edge-weight function \( w_r : E \rightarrow \mathbb{R}^+_0 \) where for each \( e \in E \), \( w_r(e) \) is independently and randomly chosen from the uniform distribution over \( \{1/|V|^10, \ldots, |V|^6/|V|^{10}\} \). Note that \( w_r(e) \leq 1/|V|^4 \) for every \( e \in E \). Denote \( w' = w + w_r \). Let \( T_w \) be an MRCT with respect to \( w \) and \( T_{w'} \) be that with respect to \( w' \). By Theorem 7, with probability at least \( 1 - 1/(2|V|) \), \( G \) is strongly min-unique with respect to \( w' \). PARALLEL runs FIND-PATH(\( G, w', s, t \)) for an arbitrary pair \( s, t \in V \) outputting a tree \( T \) with

\[
 c_{w'}(T') < \left( \frac{4}{3} + \frac{\epsilon}{2} \right) \cdot c_{w'}(\hat{T}_w')
\]

by invoking Lemma 15 with a sufficiently large constant \( r \) such that \( 8/(9r + 12) < \epsilon/2 \).

We shall prove that

\[
c_w(T') \leq \left( \frac{4}{3} + \epsilon \right) \cdot c_w(\hat{T}_w),
\]

which is true by Lemma 16 for sufficiently large \(|V|\).

\[\square\]

**Sketch of proof of Theorem 20.** We omit the simple case where \( w \) is identically zero and assume without loss of generality that \( \min_{e \in E, w(e) \neq 0} w(e) \geq 1 \). Let \( G_0 \) be the subgraph of \( G \) formed by the zero-weight edges of \( G \). PARALLEL-SROCT tests whether \( G_0 \) is a connected spanning subgraph of \( G \) in logarithmic space [Rei05] and if so, outputs a spanning tree of \( G_0 \) by calling, say, PARALLEL(\( G_0, 0 \)) where 0 denotes the identically zero function.

Below we assume that \( G_0 \) is disconnected. The random input to PARALLEL-SROCT determines an edge-weight function \( w_r : E \rightarrow \mathbb{R}^+_0 \) where for each \( e \in E \), \( w_r(e) \) is independently and randomly chosen from the uniform distribution over \( \{1/|V|^10, \ldots, |V|^6/|V|^{10}\} \). Let \( w' = w + w_r \). Note that \( \max_{e \in E} w_r(e) \leq 1/|V|^4 \).

By Theorem 7, \( G \) is strongly min-unique with respect to \( w' \) with high probability. PARALLEL-SROCT uses FIND-PATH to determine if \( G \) is strongly min-unique with respect to \( w' \) and outputs “fail” if it is not. Below we assume that \( G \) is strongly min-unique with respect to \( w' \) and outputs \( T \) rooted at \( x \) which is tested if \( x \in V \) and 4-bit encoding \( T \) is output. The final step in establishing the
approximation ratio goes by showing that for every spanning tree $T$ of $G$,

$$c_w^{(s)}(T) \geq \max_{v \in V} r(v)$$

by the disconnectedness of $G_0$ and $\min_{e \in E, w(e) \neq 0} w(e) \geq 1$, whereas

$$c_w^{(s)}(T) \leq \max_{u,v \in V} (r(u) + r(v)) \frac{|V|^3}{2} \frac{1}{|V|^4}$$

by Fact 3. The fact that $UL \subseteq NL \subseteq NC^2$ completes the proof.

Sketch of proof of Theorem 22. We omit the simple case when $w$ is identically zero and assume that $\min_{e \in E, w(e) \neq 0} w(e) \geq 1$. The case where the zero-weight edges of $G$ form a connected spanning subgraph of $G$ is dealt with as in the proof of Theorem 20, so we may assume that it is not the case. The random input to WEIGHTED-2-MRCT determines an edge-weight function $w_r : E \to \mathbb{R}_0^+$ where for each $e \in E$, $w_r(e)$ is independently and randomly chosen from the uniform distribution over $\{1/|V|^{10}, \ldots, |V|^{6}/|V|^{10}\}$. Let $w' = w + w_r$. Note that $\max_{e \in E} w_r(e) \leq 1/|V|^4$. WEIGHTED-2-MRCT detects whether $G$ is strongly min-unique with respect to $w'$ by running FIND-PATH and outputs “fail” if it is not, which occurs with a small probability by Theorem 7. Now, assume that $G$ is strongly min-unique with respect to $w'$. The sets $Z_1^{w'}, Z_2^{w'}$ in Theorem 21 where $d_G(\cdot)$ is measured with respect to $w'$ are computable in unambiguous logarithmic space by Theorem 8. For each $v \in V$, let $P_{v,s_1}^{(w')} \in SP_G(v, s_1)$ and $P_{v,s_2}^{(w')} \in SP_G(v, s_2)$ be the unique shortest paths with respect to $w'$. By Theorem 8, $T_1 = \bigcup_{v \in Z_1^{w'}} P_{v,s_1}^{(w')}$ and $T_2 = \bigcup_{v \in Z_2^{w'}} P_{v,s_2}^{(w')}$ are unambiguously computable in logarithmic space and they are trees by the strong min-uniqueness of $G$ with respect to $w'$. The unique shortest path $Q^{w'} = (q_0 = s_1, \ldots, q_j, q_{j+1}, \ldots, s_2) \in SP_G(s_1, s_2)$ with respect to $w'$ is also unambiguously computable by running FIND-PATH($G, w', s_1, s_2$), so is its first vertex $q_{j+1}$ outside of $Z_1^{w'}$. Theorem 21 then implies that a tree $T$ satisfying

$$c_{w'}^{(2)}(T) \leq 2c_{w'}^{(2)}(W-2-MRCT_{w'}(G))$$

can be output in unambiguous logarithmic space. The final step in establishing the approximation ratio is to show that for every spanning tree $T$ of $G$,

$$c_{w}^{(2)}(T, s_1, s_2, \lambda) \geq \lambda + 1$$

(6)

whereas

$$c_{w}^{(2)}(T, s_1, s_2, \lambda) \leq \lambda \frac{|V|^2}{|V|^4} + \frac{|V|^2}{|V|^4}$$

The fact that $UL \subseteq NL \subseteq NC^2$ completes the proof.

References

[Bar98] Y. Bartal, On approximating arbitrary metrics by tree metrics, Proceedings of the 30th Annual ACM Symposium on Theory of Computing, 1998, pp. 161–169.
[BFW73] D. E. Boyce, A. Farhi, and R. Weischedel, *Optimal network problem: A branch-and-bound algorithm*, Environment and Planning 5 (1973), 519–533.

[BLP94] V. Bafna, E. L. Lawler, and P. Pevzner, *Approximations algorithms for multiple sequence alignment*, Proceedings of the 5th Combinatorial Pattern Matching Conference, vol. 807, 1994, pp. 43–53.

[CCGG98] M. Charikar, C. Chekuri, A. Goel, and S. Guha, *Rounding via trees: Deterministic approximation algorithms for group Steiner trees and k-median*, Proceedings of the 30th Annual ACM Symposium on Theory of Computing, 1998, pp. 114–123.

[CLRS01] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*, MIT Press, 2001.

[DF79] R. Dionne and M. Florian, *Exact and approximate algorithm for optimal network design*, Networks 9 (1979), no. 1, 37–60.

[FD87] D. Feng and R. Doolittle, *Progressive sequence alignment as a prerequisite to correct phylogenetic trees*, Journal of Molecular Evolution 25 (1987), 351–360.

[Fis01] Fischer, *The art of uninformed decisions: A primer to property testing*, Bulletin of the European Association for Theoretical Computer Science 75 (2001).

[FLS02] M. Fischetti, G. Lancia, and P. Serafini, *Exact algorithms for minimum routing cost trees*, Networks 39 (2002), 161–173.

[FRT03] J. Fakcharoenphol, S. Rao, and K. Talwar, *A tight bound on approximating arbitrary metrics by tree metrics*, Proceedings of the 35th Annual ACM Symposium on Theory of Computing, 2003, pp. 448–455.

[GJ79] M. R. Garey and D. S. Johnson, *Computers and intractability: A guide to the theory of NP-completeness*, Freeman, 1979.

[Gus93] D. Gusfield, *Efficient methods for multiple sequence alignment with guaranteed error bounds*, Bulletin of Mathematical Biology 55 (1993).

[GW96] A. Gál and A. Wigderson, *Boolean vs. arithmetic complexity classes: randomized reductions*, Random Structures and Algorithms 9 (1996), 99–111.

[Hoa73] H. H. Hoang, *A computational approach to the selection of an optimal network*, Management Science 19 (1973), no. 5, 488–498.

[Hu74] T. C. Hu, *Optimum communication spanning trees*, SIAM Journal on Computing 3 (1974), 188–195.
[Imm88] N. Immerman, *Nondeterministic space is closed under complementation*, SIAM Journal on Computing (1988), 935–938.

[JLK78] D. S. Johnson, J. K. Lenstra, and A. H. G. Rinnooy Kan, *The complexity of the network design problem*, Networks 8 (1978), 279–285.

[Pap94] C. H. Papadimitriou, *Computational complexity*, Addison-Wesley, 1994.

[Pev92] P. Pevzner, *Multiple alignment, communication cost, and graph matching*, SIAM Journal on Applied Mathematics 52 (1992), 1763–1779.

[RA00] K. Reinhardt and E. Allender, *Making nondeterminism unambiguous*, SIAM Journal on Computing 29 (2000), no. 4, 1118–1131.

[Rei05] O. Reingold, *Undirected ST-connectivity in log-space*, Proceedings of the 37th Annual ACM Symposium on Theory of Computing, 2005, pp. 376–385.

[Sip05] M. Sipser, *Introduction to the theory of computation*, 2nd ed., Course Technology, 2005.

[Sze88] R. Szelepesényi, *The method of forced enumeration for nondeterministic automata*, Acta Informatica 26 (1988), 279–284.

[WC04] B. Y. Wu and K.-M. Chao, *Spanning trees and optimization problems*, Chapman & Hall/CRC Press, 2004.

[WCT00a] B. Y. Wu, K.-M. Chao, and C. Y. Tang, *Approximation algorithms for some optimum communication spanning tree problems*, Discrete Applied Mathematics 102 (2000), 245–266.

[WCT00b] ———, *Approximation algorithms for the shortest total path length spanning tree problem*, Discrete Applied Mathematics 105 (2000), 273–289.

[WCT00c] ———, *A polynomial time approximation scheme for optimal product-requirement communication spanning trees*, Journal of Algorithms 36 (2000), 182–204.

[Wes01] D. B. West, *Introduction to graph theory*, 2nd ed., Prentice-Hall, 2001.

[Wig94] A. Wigderson, $\mathcal{NL}/\text{poly} \subseteq \oplus \mathcal{L}/\text{poly}$, Proceedings of the 9th IEEE Structure in Complexity Conference, 1994, pp. 59–62.

[WLB+00] B. Y. Wu, G. Lancia, V. Bafna, K.-M. Chao, R. Ravi, and C. Y. Tang, *A polynomial time approximation scheme for minimum routing cost spanning trees*, SIAM Journal on Computing 29 (2000), 761–778.
[Won80] R. T. Wong, *Worst-case analysis of network design problem heuristics*, SIAM Journal on Matrix Analysis and Applications 1 (1980), 51–63.

[Wu02] B. Y. Wu, *A polynomial time approximation scheme for the two-source minimum routing cost spanning trees*, Journal of Algorithms 44 (2002), 359–378.