Fast Partial Distance Estimation and Applications

Christoph Lenzen
MPI for Informatics
Saarbrücken, Germany
clenzen@mpi-inf.mpg.de

Boaz Patt-Shamir[*]
Tel Aviv University
Tel Aviv, Israel
boaz@tau.ac.il

ABSTRACT

We study approximate distributed solutions to the weighted *all-pairs-shortest-paths* (APSP) problem in the CONGEST model. We obtain the following results.

• A deterministic $(1+\varepsilon)$-approximation to APSP with running time $\mathcal{O}(\varepsilon^{-2} n \log n)$ rounds. The best previously known algorithm was randomized and slower by a $\Theta(\log n)$ factor.

In many cases, routing schemes involve relabeling, i.e., assigning new names to nodes and that are used in distance and routing queries. It is known that relabeling is necessary to achieve running times of $o(n/\log n)$. In the relabeling model, we obtain the following results.

• A randomized $\mathcal{O}(k)$-approximation to APSP, for any integer $k > 1$, running in $\tilde{\mathcal{O}}(n^{1/2+1/k} + D)$ rounds, where D is the hop diameter of the network. This algorithm simplifies the best previously known result and reduces its approximation ratio from $\mathcal{O}(k \log k)$ to $\mathcal{O}(k)$. Also, the new algorithm uses $\mathcal{O}(\log n)$-bit labels, which is asymptotically optimal.

• A randomized $\mathcal{O}(k)$-approximation to APSP, for any integer $k > 1$, running in time $\tilde{\mathcal{O}}((nD)^{1/2} \cdot n^{1/k} + D)$ and producing *compact routing tables* of size $\tilde{\mathcal{O}}(n^{1/k})$. The node labels consist of $\mathcal{O}(k \log n)$ bits. This improves on the approximation ratio of $\Theta(k^2)$ for tables of that size achieved by the best previously known algorithm, which terminates faster, in $\tilde{\mathcal{O}}(n^{1/2+1/k} + D)$ rounds.

In addition, we improve on the time complexity of the best known deterministic algorithm for distributed approximate Steiner forest.

Categories and Subject Descriptors

F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems

[*]Supported in part by the Israel Science Foundation (grant No. 1444/14) and by a grant from Israel Ministry of Science and Technology.

Keywords

CONGEST model, weighted all-pairs shortest paths, routing table construction, Steiner forests, source detection

1. INTRODUCTION

To allow a network to be useful, it must facilitate routing messages between nodes. By the very nature of networks, this computation must be distributed, i.e., there must be a distributed algorithm that computes the local data structures that support routing at network junctions (i.e., routing tables at nodes). A trivial distributed algorithm for this purpose is to collect the entire topology at a single location, apply a centralized algorithm, and distribute the result via the network. This simplistic approach is costly, in particular if the available bandwidth is limited. To study the distributed time complexity of routing table computation, we use the CONGEST model, i.e., we assume that in an n-node network, each link can carry only $\mathcal{O}(\log n)$ bits in each time unit.

In this work, we consider networks modeled by weighted undirected graphs, where the edge weights represent some abstract link cost, e.g., latency. As to the task routing, we note that it is a generic problem with many variants. The specific problems we focus on are the following.

• *Distance Estimation*: How fast can each node obtain an estimate of its distance to each other node, and how good is that estimate?

• *All-Pairs Shortest Paths:* How fast can we construct local data structures so that when given a destination node identifier, the node can locally determine the next hop on a path to the destination, and what is the *stretch* of the resulting route w.r.t. the shortest path?

In modern routing systems, it is common practice to assign to nodes labels (identifiers) that contain some routing information. IP addresses, for example, contain a "network" and a "host" parts which allow for hierarchical routing. Thus, the following questions are also of interest to us.

• *Routing Table Construction:* What are the answers to the above questions if we permit *relabeling*, i.e., allow the algorithm to choose (small) *labels* as node identifiers, and require that distance and routing queries refer to nodes using these labels?

• *Compact Routing:* What are the answers to the above questions when the storage space at the nodes (i.e., routing table size) is small?

Some history. Shortest paths are a central object of study since the dawn of the computer era. The Bellman-Ford algorithm [4, 8], although originally developed for centralized optimization, is one of the very few fundamental distributed algorithms. Implemented as RIP, the algorithm was used in the early days of the Internet (when it was still called ARPANET) [19]. Measured in terms of the CONGEST model, a Bellman-Ford all-pairs shortest paths computation in weighted graphs takes $\Theta(n^2)$ time in the worst case, and requires $\Theta(n \log n)$ bits of storage at each node. Another simple solution to the problem is to collect the complete topology at each node (by flooding) and then apply a local single-source shortest paths algorithm, such as Dijkstra's. This solution has time complexity $\Theta(m)$ and storage complexity $\tilde{\Theta}(n)$, where m denotes the number of links in the network.[1] Since it also enjoys improved stability and flexibility, it became the Internet's routing algorithm in its later stages of evolution (see [18]). Standardized as OSPF [20], it also contains provisions for hierarchical routing.

State of the art: Lower bounds. Recently there has been a flurry of new results about routing in the CONGEST model. Below we review some known lower bounds to help placing our results in the context of what is possible. We use D to denote the *hop diameter*, i.e., the diameter of the network when ignoring weights.

- Without relabeling, any polylog-ratio approximation to APSP requires $\tilde{\Omega}(n)$ rounds [21, 22]. This holds also if tables must only enable either distance estimates or routing.
- With node relabeling, any non-trivial approximation to APSP requires $\tilde{\Omega}(\sqrt{n} + D)$ rounds [7]. The bound holds for both routing and distance queries, and even for $D \in \mathcal{O}(\log n)$. (However, if routing may be *stateful*, i.e., routing decisions may depend on the tables of previously visited nodes, no non-trivial lower bound is known; all our routing algorithms are stateless.)
- If the routing table size is $\tilde{\mathcal{O}}(n^{1/k})$, then the approximation ratio of the induced routes is at least $2k - 1$ [1, 25]. (This result does not hold for stateful routing.) For distance approximation, the same bound has been established for the special cases of $k \in \{1, 2, 3, 5\}$, and is conjectured to hold for any k (see [29]).
- Any randomized $(2 - o(1))$-approximation of APSP, and any $(2 - o(1))$-approximation of the weighted diameter requires $\tilde{\Omega}(n)$ time in the worst case [12]. In the *unweighted* case, $\tilde{\Omega}(n)$ time is required to f-approximate the diameter for $f < 3/2$ [9].

Upper bounds: Our results vs. previous work. We now list our new results (which are all upper bounds), and compare them to the best previously known bounds.

- For any $\varepsilon > 0$, we give a *deterministic* $(1 + \varepsilon)$-approximation to APSP that runs in $\mathcal{O}(\varepsilon^{-2} n \log n)$ rounds. The best known previous result, due to Nanongkai [21], achieves the same approximation ratio within $\mathcal{O}(\varepsilon^{-2} n \log^2 n)$ rounds with high probability—Nanongkai's algorithm is randomized. We note that independently and concurrently to our work, Holzer and Pinsker [12] derived the same algorithm and result for the Broadcast Congested Clique model, in which in each round, each node posts a single $\mathcal{O}(\log n)$-bit message which is delivered to all other nodes.
- Given $k \in \mathbb{N}$, we can compute a randomized $(6k - 1 + o(1))$-approximation to APSP in time $\tilde{\mathcal{O}}(n^{1/2+1/(4k)} + D)$. The algorithm succeeds with high probability (cf. Section 2), as do all our randomized algorithms. This simplifies our previous work [22] and reduces the approximation ratio from $\mathcal{O}(k \log k)$ to $\mathcal{O}(k)$. Also, the new algorithm relabels nodes with labels of $\mathcal{O}(\log n)$ bits, whereas the previous one required $\mathcal{O}(\log n \log k)$-bit labels.
- For any $k \in \mathbb{N}$, we can compute a randomized $(4k - 3 + o(1))$-approximation to APSP running in $\tilde{\mathcal{O}}(\min\{(nD)^{1/2} \cdot n^{1/k}, n^{2/3+2/(3k)}\} + D)$ rounds with tables of size $\tilde{\mathcal{O}}(n^{1/k})$. This improves over the stretch of $\mathcal{O}(k^2)$ in [22], at the cost of increasing the running time (from $\tilde{\mathcal{O}}(n^{1/2+1/k} + D)$). We point out, however, that the proposed algorithm is the first to achieve an asymptotically optimal trade-off between table size and stretch in time $\tilde{o}(n)$ for all $k > 2$ and graphs of diameter $D \in \tilde{o}(n)$.
- Using partial distance estimation, we also improve on the running time of the best known algorithm for distributed Steiner forest construction. A precise statement is provided in Section 4.3; let us just say here that in the worst case, the previous algorithm may give rise to a trivial $\tilde{\mathcal{O}}(n^2)$ time complexity, while the running time of the new algorithm is always bounded by $\tilde{\mathcal{O}}(n^{3/2})$.

Technical Summary. Our key algorithmic tool is a generalization of the (S, h, σ)-detection problem, defined as follows [16].[2] Given a graph with a distinguished set of *source nodes* S, the task is for each node to find the distances to its closest $\sigma \in \mathbb{N}$ sources within $h \in \mathbb{N}$ hops (cf. Definition 2.1). In [16] it is shown that this task can be solved in $h + \sigma$ rounds on *unweighted* graphs. The main new ingredient in all our results is an algorithm that, within a comparable running time, produces an *approximate* solution to (S, h, σ)-detection in *weighted* graphs. We call this version *partial distance estimation*, abbreviated PDE.[3]

Weighted graphs present significant difficulty, because the number of hops in a shortest (by weight) path between two nodes may be a factor of $\Theta(n)$ larger than the minimal number of hops on any path connecting the same two nodes (a weighted clique may demonstrate this phenomenon). Therefore, naïvely finding the absolute closest σ sources (w.r.t. weighted distance) within h hops may require $\Omega(n)$ rounds in the worst case, for any h and σ. One may circumvent this difficulty by replacing the underlying graph metric by *h-hop distances*, which for $v, w \in V$ is defined as the minimum weight of all v-w paths that consist of at most h hops. The collection of h-hop distances does not constitute a metric, but one can solve the (S, h, σ)-detection problem under h-hop distances in time σh using techniques similar to those used in the unweighted case [22].

Unfortunately, as illustrated in Figure 1, this time complexity is optimal in the worst case. To avoid this bottleneck, Nanongkai [21] uses a rounding technique that had previously been employed in the centralized setting [30], solving the problem to within a $(1 + \varepsilon)$-factor by essentially reducing the weighted instance to $\mathcal{O}(\log n/\varepsilon)$ unweighted instances and solving each instance using breadth-first-search. To avoid collisions, independent random delays are applied in [21] to the starting times of these instances. The result is, w.h.p., $(1 + \varepsilon)$-approximate distances to all sources in $\mathcal{O}(\varepsilon^{-2}(h + |S|) \log^2 n)$ rounds. We replace this part of

[1]Throughout this paper, we use $\tilde{\mathcal{O}}$-notation, which hides poly-logarithmic factors. See Section 2.

[2]In [16], the third parameter is called k. We use σ because here, k denotes the parameter controlling the trade-off between approximation ratio and table size.

[3]We note that Henziger et al. already employed our PDE algorithm for deterministic $(1+o(1))$-approximation of single-source shortest paths in almost optimal time [11].

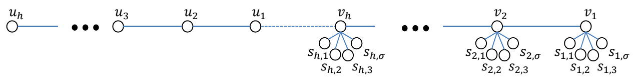

Figure 1: *A graph where $(S, h+1, \sigma)$-detection cannot be solved in $o(h\sigma)$ rounds. Edge weights are $4ih$ for edges $\{v_i, s_{i,j}\}$ for all $i \in \{1, \ldots, h\}$ and $j \in \{1, \ldots, \sigma\}$, and 1 (i.e., negligible) for all other edges. Node u_i, $i \in \{1, \ldots, h\}$, needs to learn about all nodes $s_{i,j}$ and distances $\mathrm{wd}_{h+1}(u_i, s_{i,j})$, where $j \in \{1, \ldots, \sigma\}$. Hence all this information must traverse the dashed edge $\{u_1, v_h\}$. The example can be modified to attach the same source set to each v_h. Varying distances, then still $\sigma h = |S|h$ values must be communicated over the dashed edge. Hence, the special case $|S| = \sigma$ is not easier.*

the algorithm with the deterministic source detection algorithm from [16], obtaining a deterministic algorithm that runs in $\mathcal{O}(\varepsilon^{-2}(h+|S|)\log n)$ rounds. This, using $S = V$ and $h = \sigma = n$, yields the immediate corollary of a deterministic $(1 + o(1))$-approximation to APSP.

For our other results, we abandon the special case of $S = V$ and define (see Definition 2.2) a $(1+\varepsilon)$-approximate version of the (S, h, σ)-detection problem which we call *partial distance estimation (PDE)*. The crucial insight is that by combining Nanongkai's and Zwick's rounding scheme with the algorithm from [16], PDE can be solved within $\mathcal{O}((h + \sigma)\log n/\varepsilon^2)$ rounds, so that no node sends more than $\mathcal{O}(\sigma^2)$ messages. Exploiting these properties carefully, we obtain our other results.

Further ingredients. Our compact routing schemes can be viewed as distributed constructions of the routing hierarchies of Thorup and Zwick [28]. These make use of efficient tree labeling schemes presented in the same paper, which allow for a distributed implementation in time $\tilde{\mathcal{O}}(h)$ in trees of depth h if relabeling is permitted. For compact routing table construction, we continue the Thorup-Zwick construction by simulating the partial distance estimation algorithm on the *skeleton graph* [22], broadcasting all messages via a BFS tree. This avoids the quadratic stretch incurred by the approach in [22] due to approximating distances in the skeleton graph using a *spanner* [24], which is constructed by simulating the Baswana-Sen algorithm [3]. If compact tables are not required, the partial distance estimation algorithm enables to collapse the Thorup-Zwick hierarchy of the lower levels into a single step, giving constant approximation ratio. This shaves off an $\mathcal{O}(\log k)$-factor from [22].

Paper organization. In Section 2 we define the model and problems. In Section 3 we give an algorithm for partial distance estimation. In Section 4 we present applications to the tasks of APSP, compact routing table construction, and Steiner forest construction. Due to lack of space, many details are omitted from this extended abstract; we refer to the full paper for further details [14].

2. MODEL AND PROBLEMS

Computational Model. We follow the CONGEST model as described by [23]. The distributed system is represented by a simple, connected weighted graph $G = (V, E, W)$, where V is the set of nodes, E is the set of edges, and $W : E \to \mathbb{N}$ is the edge weight function. As a convention, we use n to denote the number of nodes. We assume that all edge weights are bounded by some polynomial in n, and that each node $v \in V$ has a unique identifier of $\mathcal{O}(\log n)$ bits (we use v to denote both the node and its identifier).

Initially, nodes are aware only of their neighbors; input values (if any) are assumed to be fed by the environment before the first round. Throughout this paper, we assume that node v is given the weight of each edge $\{v, w\} \in E$ as input. Output values, which are computed at the end of the final round, are placed in special output-registers. In each round, each edge can carry a message of B bits for some given parameter B of the model; we assume that $B \in \Theta(\log n)$ throughout this paper.

Graph-Theoretic Concepts. Fix a weighted undirected graph $G = (V, E, W)$. A *path* p connecting $v, w \in V$ is a sequence of nodes $\langle v = v_0, \ldots, v_k = w\rangle$ such that for all $0 \le i < k$, $\{v_i, v_{i+1}\}$ is an edge in G. Let $\mathrm{paths}(v, w)$ denote the set of all paths connecting nodes v and w. We use the following unweighted concepts.

- The *hop-length* of a path p, denoted $\ell(p)$, is the number of edges in it.
- A path p_0 between v and w is a *shortest unweighted path* if its hop-length $\ell(p_0)$ is minimum among all $p \in \mathrm{paths}(v, w)$.
- The *hop distance* $\mathrm{hd} : V \times V \to \mathbb{N}_0$ is defined as the hop-length of a shortest unweighted path, $\mathrm{hd}(v, w) \overset{\mathrm{def}}{=} \min\{\ell(p) \mid p \in \mathrm{paths}(v, w)\}$.
- The *hop-diameter* of G is $D \overset{\mathrm{def}}{=} \max_{v,w \in V}\{\mathrm{hd}(v, w)\}$.

We use the following weighted concepts.

- The *weight* of a path p, denoted $W(p)$, is its total edge weight, i.e., $W(p) \overset{\mathrm{def}}{=} \sum_{i=1}^{\ell(p)} W(v_{i-1}, v_i)$.
- A path p_0 between v and u is a *shortest weighted path* if $W(p_0) = \min\{W(p) \mid p \in \mathrm{paths}(v, w)\}$.
- The *weighted distance* $\mathrm{wd} : V \times V \to \mathbb{N}$ is $\mathrm{wd}(v, u) \overset{\mathrm{def}}{=} \min\{W(p) \mid p \in \mathrm{paths}(v, u)\}$.
- The *weighted diameter* of G is $\mathrm{WD} \overset{\mathrm{def}}{=} \max\{\mathrm{wd}(v, u) \mid v, u \in V\}$.

Finally, we define the notion of the *shortest paths distance*.

- If $p_0 \in \mathrm{paths}(v, w)$ is a shortest weighted path and $\ell(p_0) = \min\{\ell(p) \mid W(p) = \mathrm{wd}(v, w)\}$, then p_0 is a *min-hop shortest path*. The *shortest path distance* of v and w in this case is $h_{v,w} \overset{\mathrm{def}}{=} \ell(p_0)$.
- $\mathrm{SPD} \overset{\mathrm{def}}{=} \max\{h_{v,w} \mid v, w \in V\}$ is the *shortest path diameter* of G.

Routing. In the *routing table construction* problem (abbreviated RTC), the output at each node v consists of (i) a unique *label* $\lambda(v)$ and (ii) a function "next_v" that takes a destination label λ and produces a neighbor of v, such that given the label $\lambda(w)$ of any node w, and starting from any node, following the "next" pointers leads to w. Formally, the requirement is as follows. Given a start node v and a destination label $\lambda(w)$, let $v_0 = v$ and define $v_{i+1} = \mathrm{next}_{v_i}(\lambda(w))$ for $i \ge 0$. Then $v_i = w$ for some i.

The performance of a solution is measured by its *stretch*. A route has stretch $\rho \ge 1$ if its total weight is at most ρ times the weighted distance between its endpoints, and a solution to RTC has stretch ρ if all its induced routes have stretch at most ρ.

Distance Approximation. The *distance approximation* problem is closely related to the routing problem. Again, each node v outputs a label $\lambda(v)$, but now, v needs to construct a function $\mathrm{dist}_v : \lambda(V) \to \mathbb{R}^+$ (the table) such that for all $w \in V$ it holds that $\mathrm{dist}_v(\lambda(w)) \geq \mathrm{wd}(v,w)$. The stretch between v and w is $\mathrm{dist}_v(\lambda(w))/\mathrm{wd}(v,w)$, and the solution has stretch $\rho \geq 1$ if $\max_{v,w\in V}\{\mathrm{dist}_v(\lambda(w))/\mathrm{wd}(v,w)\} = \rho$.

Partial Distance Estimation. The basic problem we attack in this paper is partial distance estimation, which generalizes the source detection problem. Let us start by defining the simpler variant.

Given a set of nodes $S \subseteq V$ and a parameter $h \in \mathbb{N}$, $L_v^{(h)}$ denotes the list obtained by ordering $\{(\mathrm{wd}(v,w),w) \mid w \in S \wedge h_{v,w} \leq h\}$ lexicographically in ascending order. Formally, we say that $(\mathrm{wd}(v,w),w) < (\mathrm{wd}(v,u),u)$ if either $\mathrm{wd}(v,w) < \mathrm{wd}(v,u)$ or $(\mathrm{wd}(v,w) = \mathrm{wd}(v,u) \wedge w < u)$.

Definition 2.1 ((S,h,σ)-detection). *The input consists of a set of* sources $S \subseteq V$ *and parameters $h, \sigma \in \mathbb{N}$. Each node is assumed to know h, σ, and whether it is in S or not. The goal is to compute at each node $v \in V$ the list L_v of the top σ entries in $L_v^{(h)}$, or the complete $L_v^{(h)}$ if $|L_v^{(h)}| \leq \sigma$.*

Relaxing this by allowing approximation to within $(1+\varepsilon)$, we arrive at the following definition.

Definition 2.2 (Partial Distance Estimation). *Given $S \subseteq V$, $h, \sigma \in \mathbb{N}$, and $\varepsilon > 0$,* $(1+\varepsilon)$-approximate (S,h,σ)-estimation *is defined as follows. Determine a* distance function $\mathrm{wd}' : V \times S \to \mathbb{N} \cup \{\infty\}$ *satisfying*

- $\forall v \in V, s \in S : \mathrm{wd}'(v,s) \geq \mathrm{wd}(v,s)$ *and*
- *if $h_{v,s} \leq h$, then $\mathrm{wd}'(v,s) \leq (1+\varepsilon)\,\mathrm{wd}(v,s)$.*

For each $v \in V$, sort the set $\{(\mathrm{wd}'(v,s),s) \mid s \in S\}$ in ascending lexicographical order. The output L_v at each node v are the (at most) top σ elements with $\mathrm{wd}'(v,s) < \infty$.

Note that setting $\varepsilon = 0$ and choosing wd' as h-hop distances results in an exact weighted version of the source detection problem. Specializing further to unweighted graphs, h-hop distances just become hop distances to nodes within h hops.

General Concepts. We extensively use "soft" asymptotic notation that ignores polylogarithmic factors. Formally, we say that $g(n) \in \tilde{\mathcal{O}}(f(n))$ if and only if there exists a constant $c \in \mathbb{R}_0^+$ such that $f(n) \leq g(n)\log^c n$ for all but finitely many values of $n \in \mathbb{N}$. We define $\tilde{o}(\cdot)$, $\tilde{\Omega}(\cdot)$ and $\tilde{\Theta}(\cdot)$ similarly.

When we say that a certain event occurs "with high probability" (abbreviated "w.h.p."), we mean that the probability of the event not occurring can be set to be less than $1/n^c$ for any desired constant c. Using the union bound, this definition implies that any polynomial number of events that occur w.h.p. also jointly occur w.h.p. We make frequent use of this fact throughout the paper.

3. FROM WEIGHTED TO UNWEIGHTED

Fix $0 < \varepsilon \in \mathcal{O}(1)$. Following Nanongkai [21] and others [5, 13, 17, 26], we reduce PDE to $\mathcal{O}(\log_{1+\varepsilon} \mathrm{WD})$ instances of the unweighted problem as follows. Let $i_{\max} = \lceil \log_{1+\varepsilon} w_{\max} \rceil$, where $w_{\max}$ is the largest edge weight in G. Note that by assumption that edge weights are polynomial in n, $i_{\max} \in \mathcal{O}(\varepsilon^{-1}\log n)$. Clearly $i_{\max}$ can be determined in $\mathcal{O}(D)$ rounds.

For $i \in \{0,\dots,i_{\max}\}$, let $b(i) = (1+\varepsilon)^i$, and define $W_i : E \to b(i)\cdot\mathbb{N}$ by $W_i(e) \overset{\mathrm{def}}{=} b(i)\lceil W(e)/b(i)\rceil$, i.e., by rounding up edge weights to integer multiples of $(1+\varepsilon)^i$. Denote by wd_i the resulting distance function, i.e., the distance function of the graph (V,E,W_i). Clearly $\mathrm{wd}_i(v,w) \geq \mathrm{wd}(v,w)$ for any $v, w \in V$. Regarding an upper bound, we have the following property.

Lemma 3.1 (adapted from [21]). *For all $v, w \in V$ let $i_{v,w} \overset{\mathrm{def}}{=} \max\left\{0, \left\lfloor \log_{1+\varepsilon}\left(\varepsilon\frac{\mathrm{wd}(v,w)}{h_{v,w}}\right)\right\rfloor\right\}$. Then*

$$\mathrm{wd}_{i_{v,w}}(v,w) < (1+\varepsilon)\,\mathrm{wd}(v,w) \in \mathcal{O}\left(\varepsilon^{-1}b(i_{v,w})h_{v,w}\right).$$

Proof. If $i_{v,w} = 0$, then $b_0 = 1$, $\mathrm{wd}_0 = \mathrm{wd}$ and the claim is clear. Assume now that $i_{v,w} > 0$. By definition of $i_{v,w}$, $\varepsilon\frac{\mathrm{wd}(v,w)}{(1+\varepsilon)h_{v,w}} < b_{i_{vw}} \leq \frac{\varepsilon\,\mathrm{wd}(v,w)}{h_{v,w}}$ and hence

$$\mathrm{wd}_{i_{v,w}}(v,w) < \mathrm{wd}(v,w) + b(i_{v,w})h_{v,w} \leq (1+\varepsilon)\,\mathrm{wd}(v,w).$$

To see the second bound, note that by definition of $i_{v,w}$ and $b(i_{v,w})$, $\mathrm{wd}(v,w) \leq \varepsilon^{-1}(1+\varepsilon)b(i_{v,w})h_{v,w}$. Due to the previous inequality and the constraint that $\varepsilon \in \mathcal{O}(1)$, the claim follows. □

Next, let G_i be the *unweighted* graph obtained by replacing each edge e in (V,E,W_i) by a path of $W_i(e)/b(i)$ unweighted edges. Let $\mathrm{hd}_i(v,w)$ denote the distance (minimal number of hops) between v and w in G_i. The previous lemma implies that in $G_{i_{v,w}}$, the resulting hop distance between v and w is not too large.

Corollary 3.2. $\forall v,w \in V$, $\mathrm{hd}_{i_{v,w}}(v,w) \in \mathcal{O}(h_{v,w}/\varepsilon)$.

Therefore, an efficient algorithm for unweighted source detection can be used to solve partial distance estimation at the cost of a small increase in running time.

Theorem 3.3. *Given a deterministic algorithm $\mathcal{A}$ for unweighted (S,h,σ)-detection that runs in $R(h,\sigma)$ rounds, $(1+\varepsilon)$-approximate (S,h,σ)-estimation can be solved within $\mathcal{O}(\log_{1+\varepsilon} n \cdot R(h',\sigma) + D)$ rounds, for some $h' \in \mathcal{O}(h/\varepsilon)$.*

Proof Sketch. Consider the following algorithm for PDE.

1. Let h' be an upper bound on $\mathrm{hd}_{i_{v,s}}(v,s)$ for all $v \in V$ and $s \in S$ with $h_{v,s} \leq h$. By Corollary 3.2, $h' \in \mathcal{O}(h/\varepsilon)$.

2. For all $i \in \{0,\dots,i_{\max}\}$, solve (S,h',σ)-detection on G_i by $\mathcal{A}$. Communication over an edge e in G simulates communication over a path in G_i by introducing a delay of $\mathrm{wd}_i(e)$ steps. Denote by $L_{v,i}$ the computed list.

3. For $s \in S$, define

$$\tilde{\mathrm{wd}}(v,s) \overset{\mathrm{def}}{=} \inf\{\mathrm{hd}_i(v,s)b(i) \mid (\mathrm{hd}_i(v,s),s) \in L_{v,i}\}.$$

Each node v outputs the list L_v consisting of the (up to) first σ elements of the set $\{(\tilde{\mathrm{wd}}(v,s),s) \mid \tilde{\mathrm{wd}}(v,s) < \infty\}$, with respect to ascending lexicographical order. This concludes the description of the algorithm.

Clearly, the resulting running time is as stated. The approximation guarantee follows from Lemma 3.1. □

Applying Theorem 3.3 with the source detection algorithm from [16], we obtain the following.

Corollary 3.4. *For $0 < \varepsilon \in \mathcal{O}(1)$, $(1+\varepsilon)$-approximate (S,h,σ)-estimation can be solved in $\mathcal{O}((h+\sigma)\varepsilon^{-2}\log n + D)$ rounds. Tables of size $\mathcal{O}(\sigma\log n)$ for routing with stretch $1+\varepsilon$ from each $v \in V$ to the (up to) σ detected nodes can be constructed in the same time. Moreover, nodes only broadcast (i.e., send the same message to all neighbors), and each node broadcasts in at most $\mathcal{O}(\sigma^2\varepsilon^{-1}\log n)$ rounds.*

4. APPLICATIONS

We now apply Corollary 3.4 to a few distributed tasks. We improve on (i) the best known results for the running time required to compute small-stretch routes with and without node relabeling, (ii) the stretch we can achieve within a given running time bound, (iii) routing table size, and (iv) the time required for Steiner forest construction.

4.1 Almost Exact APSP With and Without Node Relabeling

First we state our results for distributed computation of all-pairs $(1+\varepsilon)$-approximate shortest paths. The result follows simply by applying Corollary 3.4 with all nodes as sources and $h = \sigma = n$. As $h_{v,w} < n$ for all $v, w \in V$, $\mathrm{wd}'(v,w) \le (1+\varepsilon)\,\mathrm{wd}(v,w) < \infty$ (cf. Definition 2.2). The returned lists thus contain entries for all $n = \sigma$ nodes.

Theorem 4.1. *$(1+\varepsilon)$-approximate APSP can be solved deterministically in $\mathcal{O}(\varepsilon^{-2} n \log n)$ rounds.*

We note that Theorem 4.1 improves on the best known result for computing approximate shortest paths in the CONGEST model [21] in two ways: first, it is deterministic, and second, the running time is reduced by a logarithmic factor.

When computing routing tables, node relabeling is usually allowed. In the remainder of this subsection we use Corollary 3.4 to improve upon the best known previous result to compute routing tables when node relabeling is allowed [22]. Specifically, we prove the following result.

Theorem 4.2. *For any $k \in \mathbb{N}$, routing table construction with stretch $6k-1+o(1)$ and labels of size $\mathcal{O}(\log n)$ can be solved in $\tilde{\mathcal{O}}(n^{1/2+1/(4k)} + D)$ rounds.*

The idea is to modify the algorithm of [22]. In [22], for any given integer $0 < k \le \log n$, the algorithm computes (w.h.p.) in $\tilde{\mathcal{O}}(n^{1/2\cdot(1+1/k)} + D)$ rounds node labels of size $\mathcal{O}(\log n \log k)$ and routes with stretch $\mathcal{O}(k \log k)$. We reduce both the stretch and node label size by a $\log k$ factor without changing the running time. We start with a brief review of the algorithm from [22].

1: Skeleton. Sample nodes independently with probability $\tilde{\Theta}(1/\sqrt{n})$, forming the *skeleton set* S.

2: Skeleton Spanner. Construct and make known to all nodes an α-spanner of the *skeleton graph* (S, E_S, W_S). Here, $\{s,t\} \in E_S$ if $\mathrm{hd}(s,t) \le h$ for a certain $h \in \tilde{\Theta}(\sqrt{n})$, and $W_S(\{s,t\})$ is the minimum weight of an s-t path of at most h hops. It is shown that w.h.p., distances in the skeleton graph are identical to distances in the original graph.

3: Short Range. For each $v \in V$, let s_v be the skeleton node closest to v. For each node v, compute distance and routing tables with stretch β to all nodes $w \in V$ with $(\mathrm{wd}(v,w), w) \le (\mathrm{wd}(v,s_v), s_v)$ and from s_v to v (we use the tree routing scheme of [28]).

4: Long Range. If $(\mathrm{wd}(v,w), w) > (\mathrm{wd}(v,s_v), s_v)$, then the v-w route is obtained by concatenating the short-range route from v to s_v, the route from s_v to s_w in the skeleton spanner (whose edges represent paths of the same weight in G), and the short-range route from s_w to w.

The induced routes have stretch $\mathcal{O}(\alpha\beta)$. The same holds for distance estimation.

To facilitate routing, the label of each node v contains the following components: for the long range, the identity of the closest skeleton node s_v, its distance $\mathrm{wd}(v,s_v)$, and v's tree routing label for the tree rooted at s_v;[4] and, of course, whatever is needed for the short-range scheme.

Spanner construction can be employed as black box, giving stretch $\alpha \in \Theta(k)$ within $\tilde{\mathcal{O}}(n^{1/2+1/k} + D)$ time. Also, it is known how to construct labels for tree routing of size $(1+o(1))\log n$ in time $\tilde{\mathcal{O}}(h)$ in trees of depth h [27, 28].

We follow the general structure of [22] by implementing the short range part so that $\beta \in \mathcal{O}(1)$ and the shortest-paths trees are not too deep. To this end, we apply Corollary 3.4 with $h = \sigma \in \tilde{\Theta}(\sqrt{n})$ and source set V. Note that the error due to approximation is not limited to inaccurate distance estimations: we may also consider nodes to fall under the long-range scheme that should be treated by the short-range scheme and vice versa. However, we can bound the effect of such errors on the approximation ratio as follows.

Lemma 4.3. *Sample each node into S with independent probability p and solve $(1+\varepsilon)$-approximate (V,h,σ)-estimation with $\min\{h,\sigma\} \ge c \log n/p$, where c is a sufficiently large constant. Then w.h.p., the following holds for all $v, w \in V$, where $s'_v = \operatorname{argmin}\{(\mathrm{wd}'(v,s), s) \mid s \in S\}$.*
(1) $(\mathrm{wd}'(v,w), w) \le (\mathrm{wd}'(v,s'_v), s'_v)$
$\Rightarrow [\mathrm{wd}'(v,w) \le (1+\varepsilon)\,\mathrm{wd}(v,w)] \wedge [(\mathrm{wd}'(v,w), w) \in L_v]$
(2) $(\mathrm{wd}(v,w), w) \le (\mathrm{wd}(v,s_v), s_v)$
$\Rightarrow \mathrm{wd}'(v,w) \le (1+\varepsilon)\,\mathrm{wd}(v,w)$
(3) $(\mathrm{wd}(v,w), w) > (\mathrm{wd}(v,s_v), s_v)$
$\Rightarrow \mathrm{wd}'(v,s_v) \le (1+\varepsilon)\,\mathrm{wd}(v,w)$
(4) $(\mathrm{wd}'(v,w), w) > (\mathrm{wd}'(v,s'_v), s'_v)$
$\Rightarrow \mathrm{wd}'(v,s'_v) \le (1+\varepsilon)\,\mathrm{wd}(v,w)$

Proof. Fix $v \in V$ and order $\{(\mathrm{wd}'(v,w), w) \,|\, w \in V\}$ in ascending lexicographic order. Suppose $s'_v \in S$ is the i^{th} element of the resulting list. Then, since s'_v minimizes $(\mathrm{wd}'(v,s), s)$ among nodes $s \in S$,

$$\Pr[i \ge \min\{h,\sigma\}] \le (1-p)^{\min\{h,\sigma\}} \in e^{-\Theta(c \log n)} = n^{-\Theta(c)}.$$

When c is a sufficiently large constant, this implies that $i < h$ w.h.p., and thus $h_{v,w} < i < \min\{h,\sigma\}$ for all w with $(\mathrm{wd}'(v,w), w) \le (\mathrm{wd}'(v,s'_v), s'_v)$. By the properties of (V,h,σ)-estimation, it follows that, w.h.p., $\mathrm{wd}'(v,w) \le (1+\varepsilon)\,\mathrm{wd}(v,w)$ and $(\mathrm{wd}'(v,w), w) \in L_v$ for all such w.

To show (2), we perform the same calculation for the list $\{(\mathrm{wd}(v,w), w) \,|\, w \in V\}$; the element from S minimizing $(\mathrm{wd}(v,s), s)$ is s_v. For (3), we apply the (2) to s_v, deducing that w.h.p.,

$$\mathrm{wd}'(v,s_v) \le (1+\varepsilon)\,\mathrm{wd}(v,s_v) \le (1+\varepsilon)\,\mathrm{wd}(v,w)$$

For (4), note that if $\mathrm{wd}'(v,w) \le (1+\varepsilon)\,\mathrm{wd}(v,w)$ then

$$\mathrm{wd}'(v,s'_v) \le \mathrm{wd}'(v,w) \le (1+\varepsilon)\,\mathrm{wd}(v,w)\ .$$

Otherwise, (2) shows that $\mathrm{wd}(v,w) \ge \mathrm{wd}(v,s_v)$ w.h.p., implying that

$$\begin{aligned}\mathrm{wd}'(v,s'_v) \le \mathrm{wd}'(v,s_v) &\le (1+\varepsilon)\,\mathrm{wd}(v,s_v)\\ &\le (1+\varepsilon)\,\mathrm{wd}(v,w). \quad \square\end{aligned}$$

We use s'_v where s_v was used in the original scheme. By Lemma 4.3 and Corollary 3.4, this achieves $\beta \in 1+o(1)$ stretch within the desired time bound for $p \approx 1/\sqrt{n}$. However, since possibly $s'_v \ne s_v$, we must show that the resulting approximation ratio is still $\mathcal{O}(\alpha)$; another problem is that the $|S|$ trees induced by the *approximately* shortest paths

[4]We ignore the low-probability event that $S = \emptyset$.

from each v to s'_v might overlap. The following two lemmas address these issues, as well as the depth of the trees.

Lemma 4.4. *Sample each node into S with independent probability p and solve $(1+\varepsilon)$-approximate (S,h,σ)-estimation with $h = c\log n/p$, where c is a sufficiently large constant. Denote by wd'_S the associated distance function, and by wd' and L_v the distance function and output of $v \in V$, respectively, of a solution to $(1+\varepsilon)$-approximate (V,h,h)-estimation. If for $v, w \in V$ it holds that $(\mathrm{wd}'(v,w), w) \notin L_v$, then w.h.p. there exist $s_0, \ldots, s_{j_0} = s'_w \in S$ such that*

- $\mathrm{wd}'(w, s'_w) \in (2+\mathcal{O}(\varepsilon))\,\mathrm{wd}_S(v,w)$, *and*
- $\mathrm{wd}'_S(v,s_0)+\sum_{j=1}^{j_0}\mathrm{wd}'_S(s_{j-1},s_j) \in (3+\mathcal{O}(\varepsilon))\,\mathrm{wd}(v,w)$.

Proof. By (1) and (4) of Lemma 4.3, $(\mathrm{wd}'(v,w),w) \notin L_v$ implies that $\mathrm{wd}'(v,s'_v) \le (1+\varepsilon)\,\mathrm{wd}(v,w)$ w.h.p. Hence,

$$\mathrm{wd}(v,s_v) \le \mathrm{wd}(v,s'_v) \le \mathrm{wd}'(v,s'_v) \le (1+\varepsilon)\,\mathrm{wd}(v,w)\ .$$

By the triangle inequality, it follows that w.h.p.,

$$\begin{aligned}\mathrm{wd}(w,s_w) \le \mathrm{wd}(w,s_v) &\le \mathrm{wd}(v,w)+\mathrm{wd}(v,s_v)\\ &\le (2+\varepsilon)\,\mathrm{wd}(v,w)\end{aligned}$$

Applying (2) of Lemma 4.3 to w and s_w, we obtain

$$\begin{aligned}\mathrm{wd}(w,s'_w) \le \mathrm{wd}'(w,s'_w) \le \mathrm{wd}'(w,s_w) &\le (1+\varepsilon)\,\mathrm{wd}(w,s_w)\\ &\le 2(1+\varepsilon)^2\,\mathrm{wd}(v,w)\,,\end{aligned}$$

and, from (1) of Lemma 4.3

$$\mathrm{wd}'(w,s'_w) \le (1+\varepsilon)\,\mathrm{wd}(w,s'_w) \le 2(1+\varepsilon)^3\,\mathrm{wd}(v,w)\ ,$$

and the first part of the lemma follows (recall that $\varepsilon \in \mathcal{O}(1)$). Moreover, we have that

$$\mathrm{wd}(v,s'_w) \le \mathrm{wd}(v,w)+\mathrm{wd}(w,s'_w) \le 3(1+\varepsilon)^2\,\mathrm{wd}(v,w)\ .$$

Consider a shortest path from v to s'_w, and denote the sampled nodes that are encountered when traversing it from v to s'_w by $s_0,\ldots,s_{j_0} \in S$; in particular, $s_{j_0} = s'_w$. By the same calculation as for Lemma 4.3, w.h.p. any two consecutive sampled nodes are no more than h hops apart. As the path is a shortest path from v to s'_w, the subpaths from s_{j-1} to s_j, $j \in \{1,\ldots,j_0\}$, and from v to s_0 are also shortest paths. Therefore, $h_{v,s_0} \le h$ and, for each j, $h_{s_{j-1},s_j} \le h$. We conclude that

$$\begin{aligned}&\mathrm{wd}'_S(v,s_0)+\sum_{j=1}^{j_0}\mathrm{wd}'_S(s_{j-1},s_j)\\ &\quad\le (1+\varepsilon)\left(\mathrm{wd}(v,s_0)+\sum_{j=1}^{j_0}\mathrm{wd}(s_{j-1},s_j)\right)\\ &\quad= (1+\varepsilon)\,\mathrm{wd}(v,s'_v) \ \le\ 3(1+\varepsilon)^3\,\mathrm{wd}(v,w)\ ,\end{aligned}$$

and the second part of the lemma follows. □

Lemma 4.5. *Sample each node into S with independent probability p and solve $(1+\varepsilon)$-approximate (V,h,σ)-estimation with $h = c\log n/p$, where c is a sufficiently large constant. For $s \in S$, denote by T_s the tree induced by the routing paths from v to s for all $v \in V$ with $s'_v = s$. The depth of T_s is bounded by $\mathcal{O}(h\log n/\varepsilon^2)$, and each node participates in at most $\mathcal{O}(\log n/\varepsilon)$ different trees.*

Proof. Recall that routing from v to s'_v is based on the routing tables $L_{v,i}$ determined by the unweighted source detection instances on G_i, $i \in \{0,\ldots,i_{\max}\}$. The induced shortest-paths trees in G_i have depth at most $h' \in \mathcal{O}(h/\varepsilon)$, and they cannot overlap. By construction, the respective paths in G cannot have more hops. However, it is possible that when routing from v to s'_v, some node on the way knows of a shorter path to s'_v due to a source detection instance on G_j, $j \neq i$, and therefore "switches" to the shortest-path tree in G_j. Because $\mathrm{wd}_j(v,w) \ge \mathrm{wd}_i(v,w)$ for all $v,w \in V$ and $j \ge i$, we may however w.l.o.g. assume that the index i such that routing decisions are made according to $L_{v,i}$ is decreasing on each routing path from some node v to s'_v. Thus, the total hop count of the path is bounded by $\mathcal{O}(i_{\max}h') \subseteq \mathcal{O}(h\log n/\varepsilon^2)$. Consequently, the depth of each T_s is bounded by this value.

Concerning the number of trees, observe that if some node v decides that the next routing hop to s'_v is its neighbor u, it does so because s'_v minimizes the hop distance from v to s'_v in G_i, according to its list $L_{v,i}$. As there are $i_{\max}+1 \in \mathcal{O}(\log n/\varepsilon)$ different lists $L_{v,i}$, this is also a bound on the number of different trees v may participate in. □

We can now prove Theorem 4.2.

Proof of Theorem 4.2. Construct S by sampling nodes independently with probability $p := n^{-1/2-1/(4k)}$. W.h.p., $|S| \in \Theta(n^{1/2-1/(4k)})$. Using Corollary 3.4, we solve $(1+\varepsilon)$-approximate (V,h,σ)-estimation with $h = \sigma = c\log n/p$, for $c \in \mathcal{O}(1)$ sufficiently large, and, say, $\varepsilon = 1/\log n$. This takes $\tilde{\mathcal{O}}(1/p) = \tilde{\mathcal{O}}(n^{-1/2-1/(4k)})$ rounds and enables each node $v \in V$ to route to all nodes $w \in V$ with $(\mathrm{wd}'(v,w),w) \in L_v$ along a path of weight at most $\mathrm{wd}'(v,w)$. By Lemma 4.3, this enables each $v,w \in V$ with $(\mathrm{wd}'(v,w),w) \le (\mathrm{wd}'(v,s'_v),s'_v)$ to determine that this condition is satisfied and route from v to w with stretch $(1+\varepsilon)$.

In case that $(\mathrm{wd}'(v,w),w) \notin L_v$, we invoke Corollary 3.4 once more. This time we solve $(1+\varepsilon)$-approximate $(S,h,|S|)$-detection. W.h.p., $\tilde{\mathcal{O}}(n^{-1/2-1/(4k)})$ rounds suffice. Let wd'_S denote the corresponding distance function. Lemma 4.4 shows that there are $s_0,\ldots,s_{j_0} = s'_w \in S$ so that $\mathrm{wd}'(s'_w,w) \in (2+\mathcal{O}(\varepsilon))\,\mathrm{wd}(v,w)$ and $\mathrm{wd}'_S(v,s_0)+\sum_{j=1}^{j_0}\mathrm{wd}'_S(s_{j-1},s_j) \in (3+\mathcal{O}(\varepsilon))\,\mathrm{wd}(v,w)$. If we can route from v to s'_w incurring an additional stretch factor of $2k-1$ and from s'_w to w over a path of weight $\mathrm{wd}'(s'_w,w)$, the total stretch will be

$$(2+\mathcal{O}(\varepsilon))+(2k-1)(3+\mathcal{O}(\varepsilon)) \in 6k-1+\mathcal{O}(\varepsilon) \subset 6k-1+o(1),$$

i.e., the routing scheme satisfies the claimed stretch bound.

Concerning routing from s'_w to w, we employ the algorithm from [28] that terminates in $\tilde{\mathcal{O}}(h)$ rounds in trees of depth h. By Lemma 4.5, this can be done in $\tilde{\mathcal{O}}(h) = \tilde{\mathcal{O}}(n^{-1/2-1/(4k)})$ rounds: since each node is a member in $\mathcal{O}(\log n)$ trees, by time-multiplexing we can simulate a round for all trees in $\mathcal{O}(\log n)$ rounds. We add the computed $(1+o(1))\log n$-bit label to the label of w, inducing an s'_w-w route of weight at most $\mathrm{wd}'(w,s'_w)$.

To route from v to s'_w, consider the graph on node set S with edge set $\{\{s,t\} \mid \mathrm{wd}'_S(s,t) < \infty\}$, where the edge weights are given by wd'_S. For this graph, each node $s \in S$ knows its incident edges and their weights. Using the simulation of the Baswana-Sen algorithm [3] given in [22], we can construct and make known to all nodes a $2k-1$ spanner[5] of this graph in $\tilde{\mathcal{O}}\left(|S|^{1+1/k}+D\right) \subset \tilde{\mathcal{O}}\left(n^{1/2+1/(4k)}+D\right)$

[5]I.e., a subgraph in which distances increase by at most a factor $2k-1$.

rounds. Using this knowledge, the fact that v is aware of $\mathrm{wd}'_S(v, s_0)$, and the routing tables from the second application of Corollary 3.4, w.h.p. we can route with the desired stretch from v to s'_w based on the identifier of s'_w, which we add to the label of v. This completes the proof of the stretch bound. Checking the individual bounds we picked up along the way, we see that the label size is $\mathcal{O}(\log n)$ and the running time is $\tilde{\mathcal{O}}(n^{1/2+1/(4k)} + D)$ w.h.p. $\square$

4.2 Compact Routing on Graphs of Small Diameter

We now show how to reduce routing tables size when computing routing tables distributedly. The idea in the algorithm is to construct an (approximate) Thorup-Zwick routing hierarchy [28]. Our approach is efficient if D is small.

Using exact distances, the construction would look as follows. Let $K_0 \overset{\text{def}}{=} \{0, \dots, k-1\}$.

1. For each node $v \in V$, choose its *level* independently by a geometric distribution, i.e., the probability to have level at least $l \in K_0$ is $p_l = n^{-l/k}$. Denote the set of nodes of level at least l by S_l; trivially $S_0 = V$.

2. For each node v and level $l \in K_0 \setminus \{0\}$, determine the node $s_l(v) \in S_l$ closest to v and the set $S_{l-1}(v) \subseteq S_{l-1}$ of nodes closer to v than $s_l(v)$ (ties broken by node identifiers); for convenience, let $s_0(v) \overset{\text{def}}{=} v$ and $S_{k-1}(v) \overset{\text{def}}{=} S_{k-1}$.

3. Determine tables and labels for routing and distance approximation (i) from v to all nodes in $S_l(v)$ for all $l \in K_0$, and (ii) from $s_l(v)$ to v, where $l \in K_0 \setminus \{0\}$. The final label of v is obtained by concatenating its individual labels and the labels for routing from $s_l(v)$, $l \in K_0 \setminus \{0\}$.

In our implementation, we replace exact distances by $(1+\varepsilon)$-approximate distances for sufficiently small ε. Henceforth, we assume that the sets $S'_l(v)$ and nodes $s'_l(v)$ are defined as above, but with respect to wd'_l, the distance function corresponding to the instance of partial distance estimation we solve for level $l \in K_0$. First we bound the effect of the approximate distances on stretch. This is done by a repeated application of the argument of Lemma 4.4.

LEMMA 4.6. *For $l \in K_0 \setminus \{0\}$, denote by wd'_l the distance function corresponding to a $(1+\varepsilon)$-approximate solution to (S_l, h_l, σ_l)-estimation, where $h_l = \sigma_l = c \log n / p_l$ for a sufficiently large constant c. Suppose $v, w \in V$ and $\ell \in K_0 \setminus \{0\}$ is minimal so that $s'_\ell(w) \in S'_\ell(v)$. Then w.h.p.,*

$$\mathrm{wd}(v, s'_\ell(w)) + \mathrm{wd}(s'_\ell(w), w) \leq (1+\varepsilon)^{4\ell}(4\ell+1)\,\mathrm{wd}(v, w).$$

Lemma 4.6 shows that for $\varepsilon \in o(1/k)$, routing from v to w via $s'_\ell(w) \in S'_\ell(v)$ for minimal ℓ achieves stretch $4k-3+o(1)$. It remains to construct the hierarchy efficiently. We start with a general algorithm.

LEMMA 4.7. *For each level $l \in K_0$, we can determine w.h.p. for all nodes v the set $S'_l(v)$ and the respective distance and routing information in $\tilde{\mathcal{O}}(\varepsilon^{-2} n^{(l+1)/k})$ rounds, where tables have $\mathcal{O}(n^{1/k} \log^2 n/\varepsilon)$ bits. Within this time, we can also determine labels of $(1+o(1))\log n$ bits and tables of $\mathcal{O}(\log^2 n/\varepsilon)$ bits at each node for routing from $s'_l(v)$ to v.*

PROOF. For a sufficiently large constant c, we perform $(1+\varepsilon)$-approximate (S_l, h_{l+1}, σ)-estimation with parameters $h_{l+1} = cn^{(l+1)/k} \log n$ and $\sigma = cn^{1/k} \log n$. For $l < k-1$, the probability that $(\mathrm{wd}'(v, s'_{l+1}(v)), s'_{l+1}(v))$ has index $i \geq \sigma$ if we sort $\{(\mathrm{wd}'(v, s), s) \mid s \in S_l\}$ in increasing order is $(1 - p_l/p_{l+1})^\sigma \in n^{-\Omega(c)}$. The probability that $(\mathrm{wd}'(v, s'_{l+1}(v)), s'_{l+1}(v))$ has index $j \geq h_{l+1}$ if we order $\{(\mathrm{wd}'(v, w), w) \mid w \in V\}$ ascendingly is $(1 - 1/p_{l+1})^{h_{l+1}} \in n^{-\Omega(c)}$. By appending a bit to messages indicating whether $s \in S_l$ is also in S_{l+1}, we can thus use Corollary 3.4 to show that, w.h.p., we obtain suitable tables for routing from $v \in V$ to $S_l(v)$ and $s_{l+1}(v)$ within the stated time bound. If $l = k-1$, we have that $h_{l+1} > n$ and $|S_l| = |S_{k-1}| \leq \sigma$ w.h.p.; in this case, Corollary 3.4 shows that the construction can be performed as well.

Regarding the second part of the statement, observe that analogously to Lemma 4.5, the routing trees rooted at each node $s_{l+1} \in S_{l+1}$ have depth $\mathcal{O}(h_{l+1} \log n/\varepsilon^2)$ and each node participates in at most $\mathcal{O}(\log n/\varepsilon)$ of them. Thus, we can apply the construction from [28] to obtain labels (and tables) of size $(1+o(1))\log n$ for tree routing on each of the trees in $\tilde{\mathcal{O}}(h_{l+1}/\varepsilon^2) \subseteq \tilde{\mathcal{O}}(\varepsilon^{-2} n^{(l+1)/k})$ rounds. As each node participates in $\mathcal{O}(\log n/\varepsilon)$ trees, the table size for this routing information is $\mathcal{O}(\log^2 n/\varepsilon)$. $\square$

We can now state a useful result for small SPD.

THEOREM 4.8. *In the CONGEST model, a routing scheme guaranteeing stretch $4k-3+o(1)$ using tables of size $\tilde{\mathcal{O}}(n^{1/k})$ and node labels of size $\tilde{\mathcal{O}}(1)$ can be computed in $\tilde{\mathcal{O}}(SPD + n^{1/k})$ rounds for any $k \geq 1$.*

Unfortunately, the strategy of Theorem 4.8 can be applied only if an upper bound on SPD is known (and the running time depends on that bound), unlike the algorithm of running time $\tilde{\mathcal{O}}(\mathrm{SPD} \cdot n^{1/k})$ from [6].[6] On the other hand, applying Lemma 4.7 to all levels (without modifying h) results in running time $\tilde{\mathcal{O}}(n)$. In the remainder of this subsection, we explain how to improve on Theorem 4.8 by "short-circuiting" the higher levels of the hierarchy. This approach yields better results when the hop diameter is small.

The construction is as follows. Let $l_0 < k-1$ be some level to be determined later. We truncate the hierarchy at level l_0 by constructing a skeleton graph as follows.

DEFINITION 4.9 (l_0 SKELETON). *Let $h_{l_0} \overset{\text{def}}{=} cn^{l_0/k} \log n$ for some sufficiently large constant c. The* skeleton graph *on level l_0 is $G(l_0) = (S_{l_0}, E_{l_0}, \mathrm{wd})$, where $\{s, t\} \in E_{l_0}$ if and only if $h_{s,t} \leq h_{l_0}$.*

The l_0 skeleton graph preserves the original skeleton distances, as the following lemma states.

LEMMA 4.10. *For any $\varepsilon > 0$, $h, \sigma \in \mathbb{N}$, and $S \subseteq S_{l_0}$, denote by $\mathrm{wd}_{S_{l_0}}$ the distance function resulting from solving $(1+\varepsilon)$-approximate $(S_{l_0}, h_{l_0}, |S_{l_0}|)$-estimation and by wd_S the distance function resulting from solving $(1+\varepsilon)$-approximate $(S, ch \log n, \sigma)$-estimation on $G(l_0)$, where c is a sufficiently large constant. Then w.h.p.,*

$$\mathrm{wd}'(v, s) \overset{\text{def}}{=} \min\{\mathrm{wd}'_{S_{l_0}}(v, t) + \mathrm{wd}'_S(t, s) \mid t \in S_{l_0}\}$$

is a suitable distance function for $(1+\varepsilon)$-approximate $(S, h \cdot h_{l_0}, \sigma)$-estimation on G.

[6]In [6], the algorithm only handles distance queries and assumes that also the table of the destination can be accessed (i.e., the labels are identical to the tables). Both assumptions can be removed to achieve the same properties as our solution within $\tilde{\mathcal{O}}(\mathrm{SPD} \cdot n^{1/k})$ rounds.

Proof. By the triangle inequality, for any $v \in V$, $t \in S_{l_0}$, and $s \in S$,

$$\mathrm{wd}(v,s) \le \mathrm{wd}(v,t) + \mathrm{wd}(t,s) \le \mathrm{wd}'_{S_{l_0}}(v,t) + \mathrm{wd}'_S(t,s).$$

Suppose $h_{v,s} \le h \cdot h_{l_0}$ for some $v \in V$ and $s \in S$. The expected number of nodes in S_{l_0} on a shortest path from v to s of $h_{v,s}$ hops is $p_{l_0} h_{v,s} \in \mathcal{O}(h \log n)$. By Chernoff's bound, this number is smaller than $ch \log n$ w.h.p., as c is sufficiently large. Another application of Chernoff's bound shows that the maximum hop distance between nodes from S_{l_0} on the path is bounded by h_{l_0} w.h.p.

Denoting by $t_{v,s} \in S_{l_0}$ the first sampled node on the path, the above shows that the following properties hold w.h.p.

- $\mathrm{wd}(v,s) = \mathrm{wd}(v,t_{v,s}) + \mathrm{wd}(t_{v,s},s)$,
- $\mathrm{wd}_{G(l_0)}(t_{v,s},s) = \mathrm{wd}(t_{v,s},s)$, where $\mathrm{wd}_{G(l_0)}$ denotes the weighted distance in $G(l_0)$,
- $\mathrm{wd}'_{S_{l_0}}(v,t_{v,s}) \le (1+\varepsilon)\,\mathrm{wd}(v,t_{v,s})$, and
- $\mathrm{wd}'_S(t_{v,s},s) \le (1+\varepsilon)\,\mathrm{wd}_{G(l_0)}(t_{v,s},s)$.

Overall, this yields

$$\begin{aligned}\mathrm{wd}'(v,s) &= \min_{t\in S_{l_0}} \{\mathrm{wd}'_{S_{l_0}}(v,t) + \mathrm{wd}'_S(t,s)\}\\ &\le \mathrm{wd}'_{S_{l_0}}(v,t_{v,s}) + \mathrm{wd}'_S(t_{v,s},s)\\ &\le (1+\varepsilon)(\mathrm{wd}(v,t_{v,s}) + \mathrm{wd}_{G(l_0)}(t_{v,s},s))\\ &= (1+\varepsilon)(\mathrm{wd}(v,t_{v,s}) + \mathrm{wd}(t_{v,s},s))\\ &= (1+\varepsilon)\,\mathrm{wd}(v,s).\end{aligned}$$

□

Corollary 4.11. *If in the construction of Lemma 4.10 we replace $G(l_0)$ by the graph $\tilde{G}(l_0)$ constructed by solving $(1+\varepsilon)$-approximate $(S_{l_0}, h_{l_0}, |S_{l_0}|)$-estimation and assigning weight $\mathrm{wd}'_{S_{l_0}}(s,t)$ to edge $\{s,t\}$, the resulting function wd' is a suitable distance function for $(1+\varepsilon)^2$-approximate $(S, h\cdot h_{l_0}, \sigma)$-estimation.*

Next, we address the "truncated" levels. The general idea is to simulate the construction on the (approximate) skeleton graph given by Corollary 4.11, where communication is pipelined over a global BFS tree. Since nodes broadcast only $\tilde{\mathcal{O}}(\sigma^2)$ times in a call to our PDE algorithm (by Corollary 3.4), the total amount of communication does not become too large. However, each simulated round of the algorithm may incur an additive delay of $\mathcal{O}(D)$ (the depth of the BFS tree), which is reflected in the running time bound.

Lemma 4.12. *For any integer $l_0 \ge k/2+1$, we can construct level $l \ge l_0$ of the routing hierarchy in $\tilde{\mathcal{O}}(\varepsilon^{-2}(n^{l_0/k} + n^{(k-l_0)/k}D))$ rounds w.h.p., where the tables and labels are of size $\tilde{\mathcal{O}}(n^{1/k}/\varepsilon)$ and $\tilde{\mathcal{O}}(\varepsilon^{-1})$, respectively.*

Proof. Recall that $\varepsilon \in \mathcal{O}(1)$. We choose $\varepsilon' \in \Theta(\varepsilon)$ such that $(1+\varepsilon')^2 = (1+\varepsilon)$. We solve $(1+\varepsilon')$-approximate $(S_{l_0}, h_{l_0}, |S_{l_0}|)$-estimation using Corollary 3.4, w.h.p. in time

$$\begin{aligned}\tilde{\mathcal{O}}(\varepsilon^{-2}(h_{l_0} + |S_{l_0}|) + D) &= \tilde{\mathcal{O}}\left(\varepsilon^{-2}\left(n^{l_0/k} + n^{(k-l_0)/k}\right) + D\right)\\ &\subseteq \tilde{\mathcal{O}}\left(\varepsilon^{-2} n^{l_0/k} + D\right) .\end{aligned}$$

To apply Corollary 4.11, we simulate, for $h = h_{l+1}/h_{l_0}$ and a sufficiently large constant c, $(1+\varepsilon')$-approximate estimation on $\tilde{G}(l_0)$ with parameters $(S_l, c\,h\log n, c\,n^{1/k}\log n)$, in a way such that *all* nodes will learn the output of *all* nodes in S_{l_0}. As in Lemma 4.7, a bit indicating whether a source is in S_{l+1} is added to messages if $l < k-1$.

Before we explain how to do this, let us show how this allows the construction of level l of the routing hierarchy. From the collected information, w.h.p. nodes can locally compute the distance function wd' from Corollary 4.11 for the σ closest nodes in S_l w.r.t. wd' and, as in Lemma 4.7, derive their table for routing from v to S'_l and $s'_l(v)$.

To enable tree routing from $s'_l(v)$ to v, split the tree rooted at $s'_l(v)$ into the unique maximal subtrees rooted at $s \in S_{l_0}$ that contain no internal nodes from S_{l_0} (i.e., all such nodes are either the root or leaves). By Lemma 4.5, these subtrees have depth at most $\tilde{\mathcal{O}}(h_{l_0}/\varepsilon^2)$. We use separate labeling schemes for the (globally known) tree on $\tilde{G}(l_0)$ that describes the connections between nodes in S_{l_0} in the routing tree rooted at $s'_l(v)$ and the subtrees rooted at each $s \in S_{l_0}$. The former can be computed locally. The latter can be labeled in time $\tilde{\mathcal{O}}(\varepsilon^{-3}h_{l_0})$, provided that each node participates in $\tilde{\mathcal{O}}(\varepsilon^{-1})$ different trees only. Analogously to Lemma 4.5, this holds true because each routing decision must correspond to one of the $\mathcal{O}(\log n/\varepsilon)$ top entries of the routing tables (either for routing in G to some node in S_{l_0} or in $\tilde{G}(l_0)$). This approach requires each node in the tree to store two labels of size $(1+o(1))\log n$. Routing can now be executed by determining the next node from S_{l_0} to visit on the path from $s'_l(v)$ to v (if there still is one) and then use the label for the current subtree to find the next routing hop.

It remains to discuss how to solve $(1+\varepsilon')$-approximate $(S_l, h, c\,n^{1/k}\log n)$-estimation on $\tilde{G}(l_0)$ quickly. Recall that each node in S_{l_0} knows its neighbors and the weights of incident edges from the solution of $(1+\varepsilon')$-approximate $(S_{l_0}, h_{l_0}, |S_{l_0}|)$-estimation computed earlier. We simulate the algorithm given by Corollary 3.4, exploiting the fact that each node broadcasts in only $\tilde{\mathcal{O}}(n^{2/k})$ rounds in total. For each simulated round $i \in \{1,\dots,h'+\sigma\}$ and all of the $\mathcal{O}(\log n/\varepsilon)$ instances of the unweighted algorithm, we pipeline the communication over a BFS tree, which takes $\mathcal{O}(M_i + D)$ rounds in G, where M_i is the number of messages broadcasted by nodes in $\tilde{G}(l_0)$ in simulated round i; this time bound includes $\mathcal{O}(D)$ rounds for global synchronization of when the next simulated round starts. Therefore, the total number of communication rounds in G is

$$\begin{aligned}\sum_{i=1}^{h'+\sigma} \mathcal{O}(M_i + D) &\subseteq \tilde{\mathcal{O}}(\varepsilon^{-1}(\sigma^2|S_{l_0}| + (h'+\sigma)D))\\ &\subseteq \tilde{\mathcal{O}}(\varepsilon^{-2}(n^{2/k}\cdot n^{(k-l_0)/k} + n^{(l-l_0+1)/k}D))\\ &\subseteq \tilde{\mathcal{O}}(\varepsilon^{-2}(n^{l_0/k} + n^{(k-l_0)/k}D))\end{aligned}$$

w.h.p., as $|S_{l_0}| \in \tilde{\mathcal{O}}(n^{(k-l_0)/k})$ w.h.p. The bounds on table and label size follow from Lemma 4.7 and the above discussion of the tree labeling scheme. □

We can now put all the pieces together to obtain the following result.

Theorem 4.13. *Suppose we are given $k \in \mathbb{N}$ and some integer $k/2+1 \le l_0 \le k$. Then tables of size $\tilde{\mathcal{O}}(n^{1/k})$ and labels of size $\mathcal{O}(k\log n)$ facilitating routing and distance approximation with stretch $4k-3+o(1)$ can be constructed in $\tilde{\mathcal{O}}(n^{l_0/k} + n^{(k-l_0)/k}D)$ rounds w.h.p.*

We can pick an appropriate value for l_0 depending on D. If the running time is worse than about $n^{2/3}$, we handle the higher levels simply by making $\tilde{G}(l_0)$ known to all nodes and solving locally.

Corollary 4.14. *In the* CONGEST *model, $\tilde{\mathcal{O}}(n^{1/k})$-bit tables and $\tilde{\mathcal{O}}(1)$-bit labels for routing with stretch $4k-3+o(1)$ can be computed in $\tilde{\mathcal{O}}\left(D+\min\left\{(Dn)^{\frac{1}{2}}n^{\frac{1}{k}}, n^{\frac{2}{3}+\frac{2}{3k}}\right\}\right)$ rounds, for any $k \in \mathbb{N}$.*

4.3 A Faster $(2+\varepsilon)$-approximation of Steiner Forests

The distributed Steiner forest problem (abbreviated SF henceforth) is defined as follows.

Definition 4.15 (Distributed Steiner Forest).

Input: *At each node v, $\lambda(v) \in \Lambda \cup \{\bot\}$, where Λ is the set of* component identifiers *s.t.* $\bot \notin \Lambda$.

Output: *An edge set $F \subseteq E$ such that $\forall u, v \in V$, if $\lambda(u) = \lambda(v) \neq \bot$ then u and v are connected by F.*

Goal: *Minimize $W(F) = \sum_{e \in F} W(e)$.*

The set of nodes v with $\lambda(v) \neq \bot$ is called *terminals* and is denoted by T. As customary, we denote $t \overset{\text{def}}{=} |T|$ and $k \overset{\text{def}}{=} |\Lambda|$. For $\lambda \in \Lambda$, define *input component* $C_\lambda \subseteq T$ to be the set of all terminals v with $\lambda(v) = \lambda$. The problem specializes to minimum spanning tree (MST) by letting all nodes be terminals in a single input component, and to shortest s-t path by letting s and t be the only terminals, in the same input component. However, general SF is NP-hard [10].

In the CONGEST model, any randomized approximation algorithm for SF requires $\tilde{\Omega}(k+\min\{\sqrt{n}, \mathrm{SPD}\}+D)$ rounds, and it is known how to get a randomized $\mathcal{O}(\log n)$-approximation within that time (up to a polylog n factor) [15]. The best known deterministic algorithm finds a $(2+\varepsilon)$-approximation in $\tilde{\mathcal{O}}(\mathrm{SPD}\,k+\sqrt{\min\{\mathrm{SPD}\,t, n\}})$ rounds, for any $\varepsilon \in 1/\operatorname{polylog} n$. Note the large gap between the lower and upper bounds on deterministic construction. In this paper, using our improved PDE result, we reduce this gap as stated in the following theorem.

Theorem 4.16. *If SPD is known, a $(2+o(1))$-approximate Steiner forest can be deterministically constructed in $\tilde{\mathcal{O}}(\sqrt{\min\{D,k\}}(D+k)+SPD+\sqrt{\min\{SPD\,t, n\}})$ rounds.*

While the time bound of the algorithm in [15] may be as high as $\tilde{\mathcal{O}}(n^2)$ if both SPD and k are $\tilde{\Theta}(n)$, the time complexity in Theorem 4.16 is never more than $\tilde{\mathcal{O}}(n^{3/2})$ (even if we use n as an upper bound on SPD).

The algorithm suggested in Theorem 4.16 builds on the algorithm of [15], which is already quite involved. We only give a very high level overview of the new ideas we use here. More details can be found in the full version of the paper [14].

Approximate Distances. The deterministic algorithm in [15] (which follows the approach of [2]) works in any metric space. In particular, distorting distances in the input graph G by a factor of $1+\varepsilon'$ degrades the approximation ratio by at most a factor of $1+\varepsilon'$. In total, our algorithm computes approximate distances $\mathcal{O}(\log n)$ times, so choosing $\varepsilon' \in 1/\operatorname{polylog} n$ sufficiently small, we can guarantee that the approximation ratio grows only by a multiplicative $1+o(1)$ factor. Note that with this choice of ε', the running time of PDE (cf. Corollary 3.4) is $\tilde{\mathcal{O}}(h+\sigma+D)$, and each node broadcasts in no more than $\tilde{\mathcal{O}}(\sigma^2)$ rounds of the algorithm.

Fast Deterministic Algorithm. The main idea of the algorithm is to let *moats*, rooted at terminals, grow in uniform speed until they touch each other, at which point they merge. A moat becomes inactive and stops growing when it contains only whole input components, at which point the whole moat is logically contracted into a single node (by assigning zero weight to all its edges). To construct the required forest, the sequence of merges is traced back, and edges along the paths connecting the moat centers are added if they don't close cycles. The problem boils down to determining the order of merges. We do that using a graph defined as follows.

1. For each inactive moat M, identify all its nodes with a single source s_M.
2. Identify each terminal in an active moat with a single source s.
3. Compute for each node $(1+\varepsilon')$-approximate distances wd' to all sources.

There are at most k sources: up to $k-1$ sources for inactive moats and s. Thus, applying Corollary 3.4 with $h = \mathrm{SPD}$ and $\sigma = k$, we can complete this computation in $\tilde{\mathcal{O}}(\mathrm{SPD}+k)$ rounds. Using these distances we define a multigraph H_{moat} with nodes $s \cup \{s_M \mid M \text{ is an inactive moat}\}$ and there is an edge $\{s_1, s_2\}$ of weight w if there is a node v associated with s_1 with $\mathrm{wd}'(v, s_2) = w$. An inactive moat M becomes active after time equal to the distance between s and s_M in this graph—provided that no moats become inactive in the meantime. This can be computed at a single node, and the results distributed back, in $\mathcal{O}(Dk)$ time. In fact, we can do better by simulating the PDE algorithm on H_{moat}. The resulting running time depends on $\mathrm{SPD}(H_{\text{moat}})$, and is better than $\mathcal{O}(Dk)$ if $\mathrm{SPD}(H_{\text{moat}}) \ll k$.

Corollary 4.17. *For any $0 < \delta \in 1/\operatorname{polylog} n$, we can solve $(1+\delta)$-approximate single-source shortest paths with source s on H within $\tilde{\mathcal{O}}(D \cdot SPD(H_{moat})+k)$ communication rounds of G. This requires knowledge of $SPD(H_{moat})$.*

Using the shortcut technique of [21], $\mathrm{SPD}(H_{\text{moat}})$ can be reduced to at most $k/\sqrt{kD/(D+k)}$ before applying Corollary 4.17, yielding the following result.

Corollary 4.18. *For any $0 < \delta \in 1/\operatorname{polylog} n$, we can solve $(1+\delta)$-approximate single-source shortest paths with source s on H in $\tilde{\mathcal{O}}(\sqrt{\min\{D,k\}}(D+k))$ communication rounds of G.*

Thus we can reduce the leading term in the complexity of the algorithm from $\mathrm{SPD}\cdot k$ to $\sqrt{\min\{D,k\}}(D+k)$ and arrive at Theorem 4.16.

Leveraging Randomization. Alternatively, we observe that we can "reduce" SPD by sampling nodes with independent probability $1/\sqrt{n}$ and making them into "dummy" terminals, each of them with their unique input component label. We obtain the following result.

Corollary 4.19. *A $(2+o(1))$-approximate Steiner forest can be computed in $\tilde{\mathcal{O}}(\sqrt{\min\{D,k\}}(D+k)+\sqrt{n})$ rounds w.h.p.*

Acknowledgements. The authors thank Stephan Holzer for providing us with a preprint of [12] and Danupon Nanongkai for useful comments.

5. REFERENCES

[1] I. Abraham, C. Gavoille, D. Malkhi, N. Nisan, and M. Thorup. Compact name-independent routing with minimum stretch. *ACM Trans. Algorithms*, 4(3):37:1–37:12, 2008.

[2] A. Agrawal, P. Klein, and R. Ravi. When trees collide: An approximation algorithm for the generalized Steiner tree problem on networks. *SIAM Journal of Computing*, 24:440–456, 1995.

[3] S. Baswana and S. Sen. A simple and linear time randomized algorithm for computing sparse spanners in weighted graphs. *Random Structures and Algorithms*, 30(4):532–563, 2007.

[4] R. E. Bellman. On a routing problem. *Quart. Appl. Math.*, 16:87–90, 1958.

[5] A. Bernstein. Maintaining shortest paths under deletions in weighted directed graphs: [extended abstract]. In *Symp. Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 725–734, 2013.

[6] A. Das Sarma, M. Dinitz, and G. Pandurangan. Efficient computation of distance sketches in distributed networks. In *Proc. 24th ACM Symp. on Parallelism in Algorithms and Architectures*, 2012.

[7] A. Das Sarma, S. Holzer, L. Kor, A. Korman, D. Nanongkai, G. Pandurangan, D. Peleg, and R. Wattenhofer. Distributed Verification and Hardness of Distributed Approximation. *SIAM Journal on Computing*, 41(5):1235–1265, 2012.

[8] L. R. Ford. Network flow theory. Technical Report P-923, The Rand Corp., 1956.

[9] S. Frischknecht, S. Holzer, and R. Wattenhofer. Networks cannot compute their diameter in sublinear time. In *Proc. 23rd ACM-SIAM Symp. on Discrete Algorithms*, pages 1150–1162, 2012.

[10] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP -Completeness.* W.H. Freeman and Company, San Francisco, 1979.

[11] M. Henzinger, S. Krinninger, and D. Nanongkai. An Almost-Tight Distributed Algorithm for Computing Single-Source Shortest Paths. *CoRR*, abs/1504.07056, 2015.

[12] S. Holzer and N. Pinsker. Approximation of Distances and Shortest Paths in the Broadcast Congest Clique. *CoRR*, abs/1412.3445, 2014.

[13] P. N. Klein and S. Subramanian. A fully dynamic approximation scheme for shortest paths in planar graphs. *Algorithmica*, 22:235–249, 1998.

[14] C. Lenzen and B. Patt-Shamir. Fast Partial Distance Estimation and Applications. *CoRR*, abs/1412.7922, 2014.

[15] C. Lenzen and B. Patt-Shamir. Improved distributed steiner forest construction. In *Proc. 32nd ACM Symp. on Principles of Distributed Computing*, pages 262–271, 2014.

[16] C. Lenzen and D. Peleg. Efficient distributed source detection with limited bandwidth. In *Proc. 32nd ACM Symp. on Principles of Distributed Computing*, 2013.

[17] A. Madry. Faster approximation schemes for fractional multicommodity flow problems via dynamic graph algorithms. In *Proc. 42nd ACM Symp. on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, pages 121–130, 2010.

[18] J. McQuillan, I. Richer, and E. Rosen. The new routing algorithm for the ARPANET. *IEEE Trans. Communication*, 28(5):711–719, May 1980.

[19] J. M. McQuillan and D. C. Walden. The ARPANET design decisions. *Networks*, 1, 1977.

[20] J. Moy. OSPF ver. 2, April 1998. Internet RFC 2328.

[21] D. Nanongkai. Distributed Approximation Algorithms for Weighted Shortest Paths. In *Proc. 46th Symp. on Theory of Computing (STOC)*, pages 565–573, 2014.

[22] B. Patt-Shamir and C. Lenzen. Fast Routing Table Construction Using Small Messages [Extended Abstract]. In *Proc. 45th Symposium on the Theory of Computing (STOC)*, 2013. Full version at http://arxiv.org/abs/1210.5774.

[23] D. Peleg. *Distributed Computing: A Locality-Sensitive Approach.* SIAM, Philadelphia, PA, 2000.

[24] D. Peleg and A. A. Schäffer. Graph spanners. *J. Graph Theory*, 13(1):99–116, 1989.

[25] D. Peleg and E. Upfal. A trade-off between space and efficiency for routing tables. *Journal of the ACM*, 36(3):510–530, 1989.

[26] P. Raghavan and C. D. Thompson. Provably good routing in graphs: Regular arrays. In *Proc. 17th Ann. ACM Symp. on Theory of Computing*, STOC '85, pages 79–87, New York, NY, USA, 1985. ACM.

[27] N. Santoro and R. Khatib. Labelling and Implicit Routing in Networks. *Computer J.*, 28:5–8, 1985.

[28] M. Thorup and U. Zwick. Compact routing schemes. In *Proc. 13th ACM Symp. on Parallel Algorithms and Architectures*, 2001.

[29] U. Zwick. Exact and approximate distances in graphs – a survey. In *Proc. 9th European Symp. on Algorithms*, pages 33–48, 2001.

[30] U. Zwick. All pairs shortest paths using bridging sets and rectangular matrix multiplication. *Journal of the ACM*, 49(3):289–317, 2002.