Abstract—This paper presents new deterministic and distributed low-diameter decomposition algorithms for weighted graphs. In particular, we show that if one can efficiently compute approximate distances in a parallel or a distributed setting, one can also efficiently compute low-diameter decompositions. This consequently implies solutions to many fundamental distance based problems using a polylogarithmic number of approximate distance computations.

Our low-diameter decomposition generalizes and extends the line of work starting from [RG20] to weighted graphs in a very model-independent manner. Moreover, our clustering results have additional useful properties, including strong-diameter guarantees, separation properties, restricting cluster centers to specified terminals, and more. Applications include:

- The first near-linear work and polylogarithmic depth randomized and deterministic parallel algorithm for low-stretch spanning trees (LSST) with polylogarithmic stretch. Previously, the best parallel LSST algorithm required $m \cdot n^{o(1)}$ work and $n^{o(1)}$ depth and was inherently randomized. No deterministic LSST algorithm with truly sub-quadratic work and sub-linear depth was known.

- The first near-linear work and polylogarithmic depth deterministic algorithm for computing an $\ell_1$-embedding into polylogarithmic dimensional space with polylogarithmic distortion. The best prior deterministic algorithms for $\ell_1$-embeddings either require large polynomial work or are inherently sequential.

Even when we apply our techniques to the classical problem of computing a ball-carving with strong-diameter $O(\log^2 n)$ in an unweighted graph, our new clustering algorithm still leads to an improvement in round complexity from $O(\log^{10} n)$ rounds [CG21] to $O(\log^4 n)$.

I. INTRODUCTION

This paper\(^1\) (see [REGH22] for its full version) gives deterministic parallel & distributed algorithms for low-diameter clusterings in weighted graphs. The main message of this paper is that once you can deterministically and efficiently compute $(1+1/\poly(\log n))$-approximate distances in undirected graphs in your favorite parallel/distributed model, you can also deterministically and efficiently solve various clustering problems with $\poly \log(n)$ approximate distance computations. Since low-diameter clusterings are very basic objects and approximate distances can efficiently and deterministically be computed in various parallel and distributed models, our clustering results directly imply efficient deterministic algorithms for various problems.

In the literature, a multitude of different clustering problems are defined – you may have encountered buzzwords like low-diameter clusterings, sparse covers, network decompositions, etc. – most of which are tightly related in one way or another. To give an example of a problem that we consider in this paper, suppose you are given a parameter $D$ and you want to partition the vertex set of an input graph $G$ into clusters of diameter $\tilde{O}(D)^2$ such that every edge $e$ is cut, that is, connecting different clusters, with probability at most $\ell(e)/D$. In a deterministic variant of the problem, we instead want the number of edges cut to be at most $1$.

\(^1\)The author ordering was randomized using https://www.aeaweb.org/journals/policies/random-author-order/generator. It is requested that citations of this work list the authors separated by \textcircled{r} instead of commas: Elkin r Rozhono r Haeupler r Grunau.

\(^2\)The $O$-notation hides polylogarithmic factors in the number of vertices.
\[\sum_{e \in E(G)} \ell(e)/D.\] This clustering problem is usually known as a low-diameter clustering problem. Another problem we consider is that of computing a \(D\)-separated clustering: there, we are supposed to cluster each node with probability at least 1/2 (or at least half of the nodes if the algorithm is deterministic) in clusters such that each cluster has diameter \(\bar{O}(D)\) and any pair of clusters has distance at least \(D\).

Our main clustering result solves a very general clustering problem that essentially generalizes both examples above. The algorithm deterministically reduces the clustering problem to \(\text{poly}(\log n)\) approximate distance computations in a parallel/distributed manner. The clustering comes with several additional useful properties. We produce strong-diameter clusters; on the other hand, some results in the literature only give a so-called weak-diameter guarantee where every two nodes of the cluster are close in the original graph but the cluster itself may be even disconnected. Moreover, it can handle several generalizations which are crucial for some applications such as the low-stretch spanning tree problem. Most notably, our clustering result generalizes to the case when a set of terminals is given as part of the input and each final cluster should contain at least one terminal.

A. Main Results

While we think of our general clustering result as the main result of this paper, it is not necessary to state it in this extended abstract in full generality. Instead, we start by discussing its following corollary (see Figure 1). The following type of a clustering result is needed in known approaches to compute low-stretch spanning trees.

**Theorem I.1.** Let \(G\) be a weighted graph. We are given a set of terminals \(Q \subseteq V(G)\) and a parameter \(R > 0\) such that for every \(v \in V(G)\) we have \(d(Q, v) \leq R\). Also, a precision parameter \(0 < \varepsilon < 1\) is given. There is a deterministic distributed and parallel algorithm outputting a partition \(C\) of the vertices into clusters and a subset of terminals \(Q' \subseteq Q\) with the following properties:

1) Each cluster \(C \in C\) contains exactly one terminal \(q \in Q'\). Moreover, for any \(v \in C\) we have \(d_{G[C]}(q, v) \leq (1 + \varepsilon)R\).

2) For the set \(E^{\text{bad}}\) of edges connecting different clusters of \(C\) we have
\[|E^{\text{bad}}| = \bar{O} \left( \frac{1}{\varepsilon R} \right) \cdot \sum_{e \in E(G)} \ell(e).\]

The PRAM variant of the algorithm has work \(\bar{O}(m)\) and depth \(O(1)\). The CONGEST variant of the algorithm runs in \(\bar{O}(\sqrt{n} + \text{HopDiam}(G))\) rounds.

Our result above is in fact quite model-independent as we essentially reduce the problem to \(\text{poly}(\log n)\) \((1 + 1/\text{poly}(\log n))\)-approximate distance computations. The final complexities then follow from the recent work of \([\text{RGH}+22]\): the authors give efficient deterministic parallel and distributed approximate shortest path algorithms in PRAM and CONGEST.

a) Low-Stretch Spanning Trees: As a straightforward corollary of the clustering result in Theorem I.1, we obtain an efficient deterministic parallel and distributed algorithm for computing low-stretch spanning trees. Low-stretch spanning trees were introduced in a seminal paper by Alon et al. \([\text{AKPW}95]\), where they were shown useful for the online \(k\)-server problem. The algorithm of \([\text{AKPW}95]\) constructed spanning trees with average stretch \(\exp(\sqrt{\log n \log \log n})\). In a subsequent work Bar- Tal \([\text{Bar96, Bar98}]\) and Fakcharoenphol et al. \([\text{FRT04}]\) showed that one can get logarithmic stretch if one allows the trees to use edges that are not present in the original graph. In \([\text{EEST08}]\) it was shown that the original problem of low-stretch spanning trees admits a solution with polylogarithmic stretch. That bound was later improved to a nearly-logarithmic bound in \([\text{ABN}07]\). These constructions have important applications to the framework of spectral sparsification \([\text{ST04}]\).

In the distributed setting the problem was studied in \([\text{BEGL19}]\). However, the latter algorithm relies on the computation of exact distances. Our approach, on the other hand, only relies on approximate distance computations that, unlike exact distances, can be computed with near-optimal parallel and distributed complexity \([\text{RGH}+22]\). Hence, we are able to present the first distributed and parallel algorithm for this problem that provides polylogarithmic stretch, polylogarithmic depth and near-linear work.

**Theorem I.2** (Deterministic Low-Stretch Spanning Tree). Let \(G\) be a weighted graph. Each edge \(e\) has
moreover a nonnegative importance \( \mu(e) \). There exists a deterministic parallel and distributed algorithm which outputs a spanning tree \( T \) of \( G \) such that

\[
\sum_{e = \{u,v\} \in E(G)} \mu(e) d_T(u,v) = \tilde{O}\left(\sum_{e = \{u,v\} \in E(G)} \mu(e) d_G(u,v)\right).
\]

The PRAM variant of the algorithm has work \( \tilde{O}(m) \) and depth \( \tilde{O}(1) \). The CONGEST variant of the algorithm runs in \( \tilde{O}(\sqrt{n} + \text{HopDiam}(G)) \) rounds.

Note that plugging in \( \mu(e) := \frac{\mu'(e)}{\ell(e)} \) into (I.1) and using \( d_G(u,v) \leq \ell(e) \), we also get the following similar guarantee of

\[
\sum_{e = \{u,v\} \in E(G)} \mu'(e) \cdot \frac{d_T(u,v)}{\ell(e)} = \tilde{O}\left(\sum_{e = \{u,v\} \in E(G)} \mu'(e)\right).
\]

The stretch is optimal up to polylogarithmic factors.

b) \( \ell_1 \) Embedding: Embeddings of networks in low dimensional spaces like \( \ell_1 \)-space are a basic tool with a number of applications. For example, the parallel randomized approximate shortest path algorithm of [Li20] uses \( \ell_1 \)-embeddings as a crucial subroutine. By using our clustering results, we can use an approach similar to the one from [Bar21] to obtain an efficient deterministic parallel and distributed algorithm for \( \ell_1 \)-embedding.

Theorem 1.3 \( (\ell_1 \text{-Embedding}) \). Let \( G \) be a weighted graph. There exists a deterministic parallel and distributed algorithm which computes an embedding in \( \tilde{O}(1) \)-dimensional \( \ell_1 \)-space with distortion \( \tilde{O}(1) \). The PRAM variant of the algorithm has work \( \tilde{O}(m) \) and depth \( \tilde{O}(1) \). The CONGEST variant of the algorithm runs in \( \tilde{O}(\sqrt{n} + \text{HopDiam}(G)) \) rounds.

c) Other Applications: Since low-diameter clusterings are an important subroutine for numerous problems, there are many other more standard applications for problems like (h-hop) Steiner trees or Steiner forests, deterministic variants of tree embeddings, problems in network design, etc. [BEL20, HHZ21] We do not discuss these applications here due to space constraints. We also note that the distributed round complexities of our algorithms are almost-universally-optimal. We refer the interested reader to [GHZ20, HWZ21, RHG22] for more details regarding the notion of universal optimality.

B. Previous Work and Barriers

We will now discuss two different lines of research that study low-diameter clusterings and mention some limits of known techniques that we need to overcome.

a) Building Network Decompositions: One line of research [AGLP89, PS92, LS93, GHK18, GKM17, RG20, GGR21, CG21] is motivated by the desire to understand the deterministic distributed complexity of various fundamental symmetry breaking problems such as maximal independent set and \((\Delta + 1)\)-coloring. In the randomized world, there are classical and efficient distributed algorithms solving these problems, the first and most prominent one being Luby’s algorithm [ABI86, Lub86] from the 1980s running in \( O(\log n) \) rounds. Since then, the question whether these problems also admit an efficient deterministic algorithm running in \( poly(\log n) \) rounds was open until recently [RG20].

A general way to solve problems like maximal independent set and \((\Delta + 1)\)-coloring is by first constructing a certain type of clustering of an unweighted graph known as network decomposition [GHK18, GKM17]. A \((C,D)\)-network decomposition is a decomposition of an unweighted graph into \( C \) clusterings: each clustering is a collection of non-adjacent clusters of diameter \( D \). A network decomposition with parameters \( C, D = O(\log n) \) exist and can efficiently be computed in the CONGEST model if one allows randomization [LS93]. However, until recently the best known deterministic algorithms for network decomposition [AGLP89, PS92] needed \( n^{o(1)} \) rounds and provided a decomposition with parameters \( C, D = n^{o(1)} \). Only in a recent breakthrough, [RG20] gave a deterministic algorithm running in \( O(\log^2 n) \) CONGEST rounds and outputting a network decomposition with parameters \( C = O(\log n), D = O(\log^3 n) \).

This result was subsequently improved by [GGR21]: their algorithm runs in \( O(\log^3 n) \) rounds with parameters \( C = O(\log n), D = O(\log^2 n) \). However, both of these discussed results offer only a so-called weak-diameter guarantee. Recall that this means that every cluster has the property that any two nodes of it have distance at most \( D \) in the original graph. However, the cluster can even be disconnected.

The more appealing strong-diameter guarantee, matching the state-of-the-art weak-diameter guarantee of [GGR21], was later achieved by [CG21]. However, their algorithm needs \( O(\log^{11} n) \) CONGEST rounds.

Despite the exciting recent progress, many questions are still open: Can we get faster algorithms with better guarantees? Can the algorithms output \( D \)-separated strong-diameter clusters for \( D > 2 \)? Can we get algorithms that handle terminals (cf. Theorem 1.1)? In this work we introduce techniques that help us make some progress on these questions.

b) Tree Embeddings and Low-Stretch Spanning Trees: A very fruitful line of research started with the seminal
papers of [AKPW95, Bar96] and others. The authors were interested in approximating metric spaces by simpler metric spaces. In particular, Bartal [Bar96] showed that distances in any metric space can be probabilistically approximated with polylogarithmic distortion by a carefully chosen distribution over trees. The proof is constructive and based on low-diameter decompositions. Results of this type are known as probabilistic tree embeddings. In [AKPW95] showed that the shortest path metric of a weighted graph $G$ can even be approximated by the shortest path metric on a spanning tree of $G$ sampled from a carefully chosen distribution. A tree sampled from such a distribution is known as a low-stretch spanning tree.

Probabilistic tree embeddings and low-stretch spanning trees are an especially useful tool and have found numerous applications in areas such as approximation algorithms, online algorithms, and network design problems [BHEY05, HHZ21]. Importantly, most of the constructions of these objects are based on low-diameter clusterings.

Many of the randomized low-diameter clustering type problems can elegantly be solved in a very parallel/distributed manner using an algorithmic idea introduced in [MPX13]. We will now sketch their algorithm and then explain why new ideas are needed for our results. Consider as an example the randomized version of the low-diameter clustering problem with terminals. That is, consider the problem from Theorem 1.1, but instead of the deterministic guarantee (2) on the total number of edges cut, we require that a given edge is cut with probability $\tilde{O}\left(\frac{\ell(e)}{q}\right)$.

One way to solve the problem is as follows: every terminal samples a value from an exponential distribution with mean $\frac{\varepsilon R}{q \Theta(\log n)}$. This value is the head start of the respective terminal. Next, we compute a shortest path forest from all the terminals taking the head starts into account. Note that with high probability, the head start of each terminal is at most $\varepsilon R$ and therefore each node $v$ gets clustered to a terminal of distance at most $d(Q,v) + \varepsilon R$.

To analyze the probability of an edge $e$ being cut, let $u$ be one of the endpoints of $e$. If, taking the head starts into account, the closest terminal is more than $2\ell(e)$ closer to $u$ compared to the second closest terminal, then a simple calculation shows that $e$ is not cut. Therefore, using the memoryless property of the exponential distribution, one can show that $e$ gets cut with probability at most

$$\frac{\ell(e)}{\varepsilon R/\Theta(\log n)} = \tilde{O}\left(\frac{\ell(e)}{\varepsilon R}\right).$$

Unfortunately, this simple and elegant algorithm critically relies on exact distances: if one replaces the exact distance computation with an approximate distance computation with additive error $d_{\text{error}}$, then a given edge of length $\ell(e)$ can be cut with probability $\frac{\ell(e) + d_{\text{error}}}{\varepsilon R/\Theta(\log n)}$, which is insufficient for short edges. The left part of Figure 2 illustrates this problem.

The high-level reason why the algorithm fails with approximate distances is that first the randomness is fixed and only then the approximate distances are computed. One way to solve this issue could be to first compute approximate distances $\tilde{d}(q,.)$ from each terminal $q$ separately, then sampling a random head start $h_s q$ for each terminal $q$, followed by clustering each node $v$ to the terminal $q$ minimizing $d(q,v) - h_s q$. Even with this approach, an edge might be cut with a too large probability. Moreover, it is no longer possible to obtain a strong-diameter guarantee, as illustrated in the right part of Figure 2. Also, note that it is not clear how to efficiently compute weak-diameter clusterings with this approach as one has to perform one separate distance computation from each terminal.

In a recent work, [BEL20] managed to obtain an efficient low-diameter clustering algorithm using $\Theta(\log n)$ approximate distance computations. However, their algorithm has three disadvantages compared to our result: (1) it is randomized, (2) it only gives a weak-diameter guarantee and (3) their result is less general; for example it is not obvious how to extend their algorithm to the setting with terminals.

C. Our Techniques and Contributions

We give a clean interface for various distributed clustering routines in weighted graphs that allows to give results in different models (distributed and parallel).
a) Simple Deterministic Strong-Diameter Network Decomposition in CONGEST: In the previous section, we mentioned that the state-of-the-art strong-diameter network decomposition algorithm of [CG21] runs in $O(\log^{11} n)$ CONGEST rounds and produces clusters with diameter $D = O(\log^2 n)$.

Our first result improves upon their algorithm by giving an algorithm with the same guarantees running in $O(\log^5 n)$ CONGEST rounds.

**Theorem I.4.** There is a deterministic CONGEST algorithm computing a network decomposition with $C = O(\log n)$ clusterings such that each cluster has strong-diameter $O(\log^2 n)$. The algorithm runs in $O(\log^5 n)$ CONGEST rounds.

Note that the round complexity of our algorithm matches the complexity of the weak-diameter algorithm of [GGR21]. This is because both our result and the result of [CG21] use the weak-diameter algorithm of [GGR21] as a subroutine.

We prove Theorem I.4 in the full version of the paper and the technical overview of our approach is deferred to the full version of the paper. Here, we only note that on a high-level our algorithm can be seen as a derandomization of the randomized algorithm of [MPX13]. That is, instead of clusters, the algorithm operates with nodes and assigns “head starts” to them in a careful manner.

b) Simple Blurry Ball Growing Procedure: The blurry ball growing problem is defined as follows: given a set $S$ and distance parameter $D$, we want to find a superset $S^{\text{sup}} \supseteq S$ such that the following holds. First, for any $v \in S^{\text{sup}}$ we have $d_G(S^{\text{sup}}, S, v) \leq D$, that is, the set $S$ “does not grow too much”. On the other hand, in the randomized variant of the problem we ask for each edge $e$ to be cut by $S^{\text{sup}}$ with probability $O(\ell(e)/D)$, while in the deterministic variant of the problem we ask for the total number of edges cut to be at most $O(\sum_{e \in E(G)} \ell(e)/D)$.

Here is a simple application of this problem: suppose that we want to solve the low-diameter clustering problem where each edge needs to be cut with probability $\ell(e)/D$ and clusters should have diameter $\tilde{O}(D)$. Assume we can solve the separated clustering problem, that is, we can construct a clustering $C$ such that the clusters are $D$-separated and their diameter is $\tilde{O}(D)$. To solve the former problem, we can simply solve the blurry ball growing problem with $S = \bigcup_{C \in C} C$ and $D_{\text{blurry}} = D/3$. This way, we “enlarge” the clusters of $C$ only by a nonsignificant amount, while achieving the edge cutting guarantee.

The blurry ball growing problem was defined and its randomized variant was solved in [BEL20, Theorem 3.1]. Since blurry ball growing is a useful subroutine in our main clustering result, we generalize their result by giving an efficient algorithm solving the deterministic variant. Furthermore, we believe that our approach to solving that problem is simpler: we require the approximate distance oracle to be $(1 + 1/\log n)$-approximate instead of $\left(1 + \left(\frac{\log \log n}{\log n}\right)^2\right)$-approximate.

**Theorem I.5.** Given a weighted graph $G$, a subset of its nodes $S$ and a parameter $D > 0$, there is a deterministic algorithm computing a superset $S^{\text{sup}} \supseteq S$ such that $\max_{v \in S^{\text{sup}}} d_G(S^{\text{sup}}, S, v) \leq D$, and moreover,

$$\sum_{e \in E(G) \cap (S^{\text{sup}} \times (V(G) \setminus S^{\text{sup}}))} \ell(e) = O\left(\sum_{e \in E(G)} \ell(e)/D\right).$$

The algorithm uses $O(\log D)$ calls to an $(1 + 1/\log D)$-approximate distance oracle.

Our deterministic algorithm is a standard derandomization of the following simple randomized algorithm solving the randomized variant of the problem. The randomized algorithm is based on a simple binary search idea: in each step we flip a fair coin and decide whether or not we “enlarge” the current set $S_i$ by adding to it all nodes of distance at most roughly $D/2^i$. We start with $S_0 = S$ and the final set $S_{\log_2 D} = S^{\text{sup}}$. Hence, we need $O(\log D)$ invocations of the approximate distance oracle. We prove a more general version of Theorem I.5 in the full version of the paper and give more intuition about our approach in the full version of the paper.

c) Main Contribution: A General Clustering Result:

We will now state a special case of our main clustering result. The clustering problem that we solve generalizes the already introduced low-diameter clustering problem that asks for a partition of the vertex set into clusters such that only a small amount of edges is cut. In our more general clustering problem we are also given a set of terminals $Q \subseteq V(G)$ as input. Moreover, we are given a parameter $R$ such that $Q$ is $R$-ruling. Each cluster of the final output clustering has to contain at least one terminal. Moreover, one of these terminals should $(1 + \varepsilon)\cdot R$-rule its cluster.

We note that in order to get the classical low-diameter clustering with parameter $D$ as an output of our general result, it suffices to set $Q = V(G)$, $R = D$ and $\varepsilon = 1/2$.

A more general version of Theorem I.1 is proven in the full version of the paper. The intuition behind the algorithm is explained in the full version of the paper. Here, we note that the algorithm combines the clustering idea of the algorithm from Theorem I.4 and uses as a subroutine the blurry ball growing algorithm from Theorem I.5.
Another corollary of our general clustering result is the following theorem.

**Theorem I.6.** [A corollary of the main clustering result proven in the full version of the paper] We are given an input weighted graph $G$, a distance parameter $D$ and each node $v \in V(G)$ has a preferred radius $r(v) > 0$.

There is a deterministic distributed algorithm constructing a partition $C$ of $G$ that splits $V(G)$ into two sets $V^{\text{good}} \sqcup V^{\text{bad}}$ such that

1) Each cluster $C \in C$ has diameter $\tilde{O}(D)$.
2) For every node $v \in V^{\text{good}}$ such that $v$ is in a cluster $C$, we have $B_G(v, r(v)) \subseteq C$.
3) For the set $V^{\text{bad}}$ of nodes we have

$$\sum_{v \in V^{\text{bad}}} r(v) = \frac{1}{2D} \sum_{v \in V(G)} r(v).$$

The algorithm needs $\tilde{O}(1)$ calls to an $(1 + 1/poly\log n)$-approximate distance oracle.

One reason why we consider each node to have a preferred radius is that it allows us to deduce Theorem I.1 from our general theorem by considering the subdivided graph where each edge is split by adding a node “in the middle of it”, with a preferred radius of $\ell(e)$.

Let us now compare the clustering of Theorem I.6 with the $D$-separated clustering that we already introduced. Recall that in the $D$-separated clustering problem, we ask for clusters with radius $O(D)$ and require the clusters to be $D$-separated. Moreover, only half of the nodes should be unclustered.

In our clustering, we can choose $r(v) = D$ for all nodes $v \in V(G)$, we again get clusters of diameter $O(D)$ and only half of the nodes are bad. The difference with the $D$-separated clustering is that we cluster all the nodes, but we require the good nodes to be “$D$-padded”.

This is a slightly weaker guarantee then requiring the clusters to be $D$-separated: we can take any solution of the $D$-separated problem, and enlarge each cluster by adding all nodes that are at most $D/3$ away from it. We mark all original nodes of the clusters as good and all the new nodes as bad. Moreover, each remaining unclustered node forms its own cluster and is marked as bad. This way, we solve the special case of Theorem I.6 with the padding parameter $D/3$. We do not know of an application of $D$-separated clustering where the slightly weaker $D$-padded clustering of Theorem I.6 does not suffice. However, we also use a different technique to solve the $D$-separated problem.

**Theorem I.7.** We are given a weighted graph $G$ and a separation parameter $D > 0$. There is a deterministic algorithm that outputs a clustering $C$ of $D$-separated clusters of diameter $\tilde{O}(D)$ such that at least $n/2$ nodes are clustered.

The algorithm needs $\tilde{O}(1)$ calls to an $(1 + 1/poly\log n)$-approximate distance oracle computing approximate shortest paths from a given set up to distance $O(D)$.

The algorithm is based on the ideas of the weak-diameter network decomposition result of [RG20] and the strong-diameter network decomposition of [CG21].

Since shortest paths up to distance $D$ can be computed in unweighted graphs by breadth first search, we get as a corollary that we can compute a separated strong-diameter network decomposition in unweighted graphs. No $O(D)$-round deterministic CONGEST algorithm for separated strong-diameter network decomposition was known.

**Corollary I.8.** [D-separated strong-diameter network decomposition] We are given an unweighted graph $G$ and a separation parameter $D > 0$. There is a deterministic algorithm that outputs $O^{\log n}$ clusterings $\{C_1, \ldots, C_{O^{\log n}}\}$ such that

1) Each node $u \in V(G)$ is contained in at least one clustering $C_i$.
2) Each clustering $C_i$ consists of $D$-separated clusters of diameter $\tilde{O}(D)$.

The algorithm needs $\tilde{O}(D)$ CONGEST rounds.

**REFERENCES**

[ABI86] Noga Alon, Laszlo Babai, and Alon Itai. A fast and simple randomized parallel algorithm for the maximal independent set problem. *Journal of Algorithms*, 7(4):567–583, 1986.

[ABN07] Ittai Abraham, Yair Bartal, and Ofer Neiman. Embedding metrics into ultrametrics and graphs into spanning trees with constant average distortion. In Nikhil Bansal, Kirk Pruhs, and Clifford Stein, editors, *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2007*, New Orleans, Louisiana, USA, January 7-9, 2007, pages 502–511. SIAM, 2007.

[AGLP89] Baruch Awerbuch, Andrew V. Goldberg, Michael Luby, and Serge A. Plotkin. Network decomposition and locality in distributed computation. In *Proc. 30th IEEE*
[AKPW95] Noga Alon, Richard M Karp, David Peleg, and Douglas West. A graph-theoretic game and its application to the k-server problem. *SIAM Journal on Computing*, 24(1):78–100, 1995.

[Bar96] Yair Bartal. Probabilistic approximation of metric spaces and its algorithmic applications. In *Proceedings of 37th Conference on Foundations of Computer Science*, pages 184–193. IEEE, 1996.

[Bar98] Yair Bartal. On approximating arbitrary metrics by tree metrics. In Jeffrey Scott Vitter, editor, *Proceedings of the Thirtieth Annual ACM Symposium on the Theory of Computing*, Dallas, Texas, USA, May 23-26, 1998, pages 161–168. ACM, 1998.

[Bar21] Yair Bartal. Advances in metric ramsey theory and its applications, 2021.

[BEGL19] Ruben Becker, Yuval Emek, Mohsen Ghaffari, and Christoph Lenzen. Distributed algorithms for low stretch spanning trees. In *33rd International Symposium on Distributed Computing (DISC 2019)*, volume 146, page 4. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2019.

[BEL20] Ruben Becker, Yuval Emek, and Christoph Lenzen. Low diameter graph decompositions by approximate distance computation. In *ITCS*, 2020.

[BEY05] Allan Borodin and Ran El-Yaniv. *Online computation and competitive analysis*. Cambridge university press, 2005.

[CG21] Yi-Jun Chang and Mohsen Ghaffari. Strong-diameter network decomposition. In *Proceedings of the 2021 ACM Symposium on Principles of Distributed Computing*, PODC’21, page 273–281. New York, NY, USA, 2021. Association for Computing Machinery.

[EEST08] Michael Elkin, Yuval Emek, Daniel A Spielman, and Shang-Hua Teng. Lower-stretch spanning trees. *SIAM Journal on Computing*, 38(2):608–628, 2008.

[FRT04] Jittat Fakcharoenphol, Satish Rao, and Kunal Talwar. A tight bound on approximating arbitrary metrics by tree metrics. *J. Comput. Syst. Sci.*, 69(3):485–497, 2004.

[GGR21] Mohsen Ghaffari, Christoph Grunau, and Václav Rozhoň. Improved deterministic network decomposition. In *Proc. of the 32nd ACM-SIAM Symp. on Discrete Algorithms (SODA)*, page 2904–2923, USA, 2021. Society for Industrial and Applied Mathematics.

[GHK18] Mohsen Ghaffari, David Harris, and Fabian Kuhn. On derandomizing local distributed algorithms. In *Proc. Foundations of Computer Science (FOCS)*, pages 662–673, 2018.

[GHZ20] Mohsen Ghaffari, Bernhard Haeupler, and Goran Zuzic. Hop-constrained oblivious routing, 2020.

[GKM17] Mohsen Ghaffari, Fabian Kuhn, and Yannic Maus. On the complexity of local distributed graph problems. In *Proc. 49th ACM Symp. on Theory of Computing (STOC)*, pages 784–797, 2017.

[HHZ21] Bernhard Haeupler, D Ellis Hershkowitz, and Goran Zuzic. Tree embeddings for hop-constrained network design. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 356–369, 2021.

[HWZ21] Bernhard Haeupler, David Wajc, and Goran Zuzic. Universally-optimal distributed algorithms for known topologies. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 1166–1179, 2021.

[Li20] Jason Li. Faster parallel algorithm for approximate shortest path. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 308–321, 2020.

[LS93] Nati Linial and Michael Saks. Low diameter graph decompositions. *Combinatorica*, 13(4):441–454, 1993.

[Lub86] Michael Luby. A simple parallel algorithm for the maximal independent set problem. *SIAM Journal on Computing*, 15:1036–1053, 1986.

[MPX13] Gary L Miller, Richard Peng, and Shen Chen Xu. Parallel graph decompositions using random shifts. In *Proceedings of the twenty-fifth annual ACM*
symposium on Parallelism in algorithms and architectures, pages 196–203, 2013.

[PS92] Alessandro Panconesi and Aravind Srinivasan. Improved distributed algorithms for coloring and network decomposition problems. In Proc. 24th ACM Symp. on Theory of Computing (STOC), pages 581–592, 1992.

[REGH22] Václav Rozhoň, Michael Elkin, Christoph Grunau, and Bernhard Haeupler. Deterministic low-diameter decompositions for weighted graphs and distributed and parallel applications. 2022.

[RG20] Václav Rozhoň and Mohsen Ghaffari. Polylogarithmic-time deterministic network decomposition and distributed derandomization. In Proc. Symposium on Theory of Computation (STOC), 2020.

[RGH+22] Václav Rozhoň, Christoph Grunau, Bernhard Haeupler, Goran Zuzic, and Jason Li. Undirected (1+ ε)-shortest paths via minor-aggregates: near-optimal deterministic parallel and distributed algorithms. In Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing, pages 478–487, 2022.

[RHG22] Harald Raecke, Bernhard Haeupler, and Mohsen Ghaffari. Hop-constrained expander decompositions, oblivious routing, and distributed universal optimality, 2022.

[ST04] Daniel A. Spielman and Shang-Hua Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In László Babai, editor, Proceedings of the 36th Annual ACM Symposium on Theory of Computing, Chicago, IL, USA, June 13-16, 2004, pages 81–90. ACM, 2004.