Distributed Strong Diameter Network Decomposition

Michael Elkin*1 and Ofer Neiman†1

1Department of Computer Science, Ben-Gurion University of the Negev, Beer-Sheva, Israel.
Email: {elkinm,neimano}@cs.bgu.ac.il

Abstract

For a pair of positive parameters \( D, \chi \), a partition \( P \) of the vertex set \( V \) of an \( n \)-vertex graph \( G = (V,E) \) into disjoint clusters of diameter at most \( D \) each is called a \((D,\chi)\) network decomposition, if the supergraph \( G(P) \), obtained by contracting each of the clusters of \( P \), can be properly \( \chi \)-colored. The decomposition \( P \) is said to be strong (resp., weak) if each of the clusters has strong (resp., weak) diameter at most \( D \), i.e., if for every cluster \( C \in P \) and every two vertices \( u,v \in C \), the distance between them in the induced graph \( G(C) \) of \( C \) (resp., in \( G \)) is at most \( D \).

Network decomposition is a powerful construct, very useful in distributed computing and beyond. It was introduced by Awerbuch et. al. [AGLP89] in the end of the 1980’s. These authors showed that strong \((2^{O(\sqrt{\log n \log \log n})},2^{O(\sqrt{\log n \log \log n})})\) network decompositions can be computed in \( 2^{O(\sqrt{\log n \log \log n})} \) distributed time. Their result was improved at the beginning of nineties by Panconesi and Srinivasan [PS92], who showed that \( 2^{O(\sqrt{\log n \log \log n})} \) in all the three expressions can be replaced by \( 2^{O(\sqrt{\log n})} \). Around the same time Linial and Saks [LS93] devised an ingenious randomized algorithm that constructs weak \((O(\log n),O(\log n))\) network decompositions in \( O(\log^2 n) \) time. Awerbuch et. al. [ABCP96] devised a randomized algorithm that builds a strong \((O(\log n),O(\log n))\) network decomposition in \( O(\log^4 n) \) time, using very large messages and heavy local computations. It was however open till now if strong network decompositions with both parameters \( 2^{\omega(\sqrt{\log n})} \) can be constructed in distributed \( 2^{\omega(\sqrt{\log n})} \) time using short messages, or if a result of [LS93] can be strengthened to provide a strong \((O(\log n),O(\log n))\) network decomposition within \( O(\log^2 n) \) time (even using large messages).

In this paper we answer these long-standing open questions in the affirmative, and show that strong \((O(\log n),O(\log n))\) network decompositions can be computed within \( O(\log^2 n) \) time. We also present a tradeoff between parameters of our network decomposition. Our work is inspired by and relies on the “shifted shortest path approach”, due to Blelloch et. al. [BGK+14], and Miller et. al. [MPX13]. These authors developed this approach for PRAM algorithms for padded partitions. We adapt their approach to network decompositions in the distributed model of computation.

1 Introduction

1.1 Definitions and Motivation

Consider an unweighted undirected \( n \)-vertex graph \( G = (V,E) \), and suppose that it models a communication network. Each vertex hosts a processor with a distinct identity number from the range \( \{1,\ldots,n\} \), and these processors communicate with one another via the edges of \( G \) in synchronous rounds. If messages are...
of unbounded length, the model is called \textit{LOCAL}. If each message has size $O(1)$ words (or, equivalently, $O(\log n)$ bits), then the model is called \textit{CONGEST}. The running time of an algorithm in this model is the number of rounds of distributed communication.

In the coloring problem one wishes to compute a proper coloring $\varphi$ of $G$ that employs a small number of colors. A coloring $\varphi$ is said to be \textit{proper} if for every edge $(u, v) \in E$, we have $\varphi(u) \neq \varphi(v)$. In a seminal paper [AGLP89], Awerbuch et. al. introduced a generalization of vertex coloring, in which one can cluster vertices of $G$ into clusters of small diameter. A partition $\mathcal{P}$ of $G$ into disjoint clusters induces a supergraph $\mathcal{G}(\mathcal{P}) = (\mathcal{P}, \mathcal{E})$, where

$$\mathcal{E} = \{(C, C') \mid C, C' \in \mathcal{P}, C \neq C', \exists (v, v') \in E \cap (C \times C')\}.$$ 

A partition $\mathcal{P}$ is called a \textit{strong} (respectively, \textit{weak}) \textit{network decomposition} of $G$ with parameters $D$ and $\chi$, or shortly, $(D, \chi)$ \textit{network decomposition}, if all clusters of $\mathcal{P}$ have strong (resp., weak) diameter at most $D$, and the supergraph $\mathcal{G}(\mathcal{P})$ can be properly colored with at most $\chi$ colors. Note that an ordinary proper $\chi$-coloring can be viewed as a $(0, \chi)$ network decomposition.

The \textit{strong} (respectively, \textit{weak}) \textit{diameter} of a cluster $C$ is defined by $\text{Diam}(C) = \max_{v,v' \in C} d_{G[C]}(v, v')$ (resp., $\text{WeakDiam}(C) = \max_{v,v' \in C} d_G(v, v')$). The notation $d_G$ (respectively, $d_{G[C]}$) denotes the distance function in $G$ (resp., in the induced subgraph $G(C)$ of $C$). The strong (resp., weak) diameter of a partition $\mathcal{P}$ is the maximum strong (resp., weak) diameter of its clusters.

Network decomposition is a very powerful construct in distributed computing. The original motivation of [AGLP89] was symmetry breaking problems, such as maximal independent set, maximal matching and $(\Delta + 1)$-vertex-coloring, where $\Delta$ is the maximum degree of the input graph. Given a $(D, \chi)$ network decomposition $\mathcal{P}$ along with a $\chi$-coloring of the induced supergraph $\mathcal{G}(\mathcal{P})$, each of these problems can be solved within $O(D \cdot \chi)$ time. This is done by solving them in parallel on each of the clusters of color class 1, then extending the solution to each of the clusters of color class 2, etc. Since clusters within each color class are at least 2 apart one from another, computations within the same color class can be conducted in parallel. Moreover, since the maximum clusters’ diameter is bounded by $D$, one can perform each of these $\chi$ phases within $O(D)$ time by a naive algorithm. (The naive algorithm collects the entire cluster’s topology into a central vertex, solves the problem locally, and disseminates the solution to all vertices of the given cluster.)

Later additional applications of network decompositions were discovered. Dubhashi et. al. [DMP+05] used network decompositions for computing sparse spanners and linear-size skeletons. Barenboim et. al. [Bar12, BEG15] devised distributed approximation algorithm for the graph coloring and minimum dominating set problems, which employ network decompositions. Network decompositions are also closely related to \textit{neighborhood covers}, which are used extensively for routing [AP92] and synchronization [Awe85, APPS92]. The relationship between neighborhood covers and network decompositions was explored in [ABCP96]. The latter authors [ABCP94] also devised an NC PRAM algorithm for computing network decompositions. Barenboim et. al. [BEG15] have also showed that network decompositions can be used to build low-intersecting partitions, which are, in turn, used for computing universal Steiner trees [BDR+12].

To summarize, network decompositions have numerous applications in distributed computing and beyond. They also constitute a very appealing combinatorial construct, well worth studying on its own right.

\subsection*{1.2 Previous and Our Results}

Awerbuch et. al. [AGLP89] devised a deterministic algorithm with running time $2^{O(\sqrt{\log n \log \log n})}$, that computes a strong $(2^{O(\sqrt{\log n \log \log n})}, 2^{O(\sqrt{\log n \log \log n})})$ network decomposition. This result was improved by Panconesi and Srinivasan [PS92], whose algorithm has running time $2^{O(\sqrt{\log n})}$, and both parameters of
Decompositions”, Linial and Saks [LS93] conducted a systematic investigation of network decompositions. They showed that for any \( k \leq \log n \), every \( n \)-vertex graph admits a strong \((2k - 2, 2n^{1/k} \log n)\) network decomposition, and for any \( \lambda \leq \log n \), it admits a strong \((2n^{1/\lambda} \log n, \lambda)\) network decomposition, and that these bounds are nearly tight. They have also devised a randomized distributed algorithm for computing weak network decompositions in expected time \( O(k \cdot n^{1/k} \cdot \log n) \), with essentially the same parameters. In particular, and most notably, for \( k = \log n \), their algorithm produces a weak \((O(\log n), O(\log n))\) network decomposition in \( O(\log^2 n) \) time.

The algorithms of [AGLP89] and [LS93] work in the CONGEST model. Awerbuch et. al. [ABCP96] devised a distributed reduction from strong network decompositions to weak ones. Their reduction introduces an overhead of \( O(\log^2 n) \) in the running time. Also, it employs very large messages, and heavy local computations. By composing the algorithm of [LS93] with the reduction of [ABCP96] one obtains a randomized algorithm in the LOCAL model that computes an \((O(\log n), O(\log n))\) network decomposition in \( O(\log^4 n) \) time.

Remarkably, quarter a century after the SODA’91 publication of Linial and Saks’ paper, their algorithm is still the only algorithm whose running time is at most polylogarithmic in \( n \), and which produces a network decomposition with both parameters being at most polylogarithmic in \( n \), in the CONGEST model.

In this paper we resolve this long-standing open question in the affirmative. We devise a randomized algorithm with running time \( O(\log^2 n) \) that computes a strong \((O(\log n), O(\log n))\) network decomposition in the CONGEST model. Moreover, similarly to Linial and Saks [LS93], we can also trade between the parameters. Specifically, for any \( k \leq \log n \), our randomized algorithm has running time \( O(n^{1/k} \cdot k^2) \) and computes a strong \((2k - 2, O(k \cdot n^{1/k}))\) network decomposition. In the other regime, for any \( \lambda \leq \log n \), in time \( O(\lambda \cdot n^{1/\lambda} \cdot \log n) \) we compute a strong \((O(n^{1/\lambda} \log n), \lambda)\) network decomposition. Note that the number of colors and running time are slightly better than those of [LS93] in the first regime. As in [LS93], all messages sent in our algorithm consist of \( O(1) \) words. Note, however, that even for the LOCAL model, our result improves the previous state-of-the-art running time \( O(\log^4 n) \) of [ABCP96] to \( O(\log^2 n) \).

The main technique that made our result possible is the “shifted shortest path approach”, due to Blelloch et. al. [BGK+14], and Miller et. al. [MPX13]. These authors developed this approach for computing padded partitions in the PRAM model. Specifically, Miller et. al. [MPX13] devised a PRAM algorithm for computing a strong padded partition, i.e., a partition with strong diameter at most \( O(\log n)/\beta \), for a parameter \( \beta \leq 1/2 \), and such that the fraction of edges that cross between different clusters of the partition is at most \( \beta \).

It is known that padded partitions are related to network decompositions. This relationship was exploited by Bartal [Bar96], who showed that the approach of Linial and Saks [LS93] for constructing network decompositions can be used to build padded partitions. In this work we exploit this relationship in the opposite direction, and show that Miller’s et. al. [MPX13] approach for constructing padded partitions can be used for building network decompositions. Our algorithm is similar in spirit to the algorithm of [LS93], in which every vertex \( v \) samples a radius \( r_v \) from a geometric (or exponential, in our case) distribution, and broadcasts this to its \( r_v \)-neighborhood. The main difference is in determining the clusters: While in [LS93] a vertex \( x \) decides to join a cluster centered at \( v \) if \( v \) has the minimal ID among broadcasts that reached \( x \), and furthermore \( r_v \) is strictly larger than the distance \( d(x, v) \) (this is the distance in the current graph). In our algorithm, we do not use IDs, we let \( x \) compare the shifted random variables \( r_v - d(x, v) \) for all vertices \( v \) whose broadcast reached it, and decide according to the difference between the largest and the second largest values. This idea is inspired by [MPX13], who use a similar comparison in the analysis of their algorithm for padded partitions. However, the fact that this algorithm yields a strong diameter is somewhat

3
more involved in our setting.

1.3 Related Work

Barenboim et. al. [BEG15] devised a randomized constant time algorithm for constructing strong \((O(1), n^c)\) network decompositions, for an arbitrarily small constant \(\epsilon > 0\). Kutten et. al. [KNPR14] extended the algorithm of Linial and Saks [LS93] for constructing network decompositions to hypergraphs. A long line of research developed network decompositions for graphs of bounded growth, see, e.g., [GV07, KMW05, SW08]. Miller et. al. [MPVX15] built upon the ideas of [MPX13] to devise efficient parallel algorithms for constructing spanners and hopsets.

2 Distributed Algorithm for Strong Diameter Network Decomposition

Here we prove our main result. For a more accessible presentation, we first show a simpler version, and improve the number of colors in Section 2.1.

**Theorem 1.** For any unweighted graph \(G = (V, E)\) on \(n\) vertices, and parameters \(1 \leq k \leq \ln n, 3 < c, \ln(cn)\) network decomposition of \(G\). The number of rounds required is \(k(cn)^{1/k} \cdot \ln(cn)\), and each message consists of \(O(1)\) words.

Note that taking \(c = 2^k\) does not affect the number of blocks and rounds by more than a constant factor.

Following [LS93], we form the partition by carving blocks. A block \(W \subseteq V\) is a set of vertices, and the connected components of \(G(W)\) are clusters. Clearly, these clusters form an independent set in \(G(P)\), and thus can be colored with a single color. So the chromatic number of \(G(P)\) is bounded by the number of blocks our algorithm generates.

**Construction** The algorithm is a subtle modification of the [LS93] algorithm, inspired by the recent methods of [MPX13]. Let \(\beta = \ln(cn)/k\). The algorithm consists of phases \(t = 1, 2, \ldots, \lambda\), for \(\lambda = (cn)^{1/k} \cdot \ln(cn)\). Let \(G_1 = G\). In each phase \(t\) we carve a block \(W_t\) out of the current graph \(G_t\), and let \(G_{t+1} = G_t \setminus W_t\).

To implement the \(t\)-th phase, every vertex \(v \in V(G_t)\) chooses independently in parallel a value \(r_v^{(t)}\) (we shall omit the superscript whenever it is clear from context), by sampling from the exponential distribution with parameter \(\beta\), denoted \(\text{EXP}(\beta)\), which has density

\[
f(x) = \begin{cases} \beta \cdot e^{-\beta x} & \text{for } x \geq 0 \\ 0 & \text{otherwise.} \end{cases}
\]

For \(v \in V\), let \(E_v\) be the event that at some phase \(t\), \(r_v^{(t)} \geq k + 1\). We will later prove the following lemma.

**Lemma 1.** With probability at least \(1 - 2/c\), none of the events \(E_v\) hold.

Every vertex \(v\) will broadcast the value \(r_v\) to every vertex of \(G_t\) within distance \(R_v := |r_v|\) from it. Note that assuming **Lemma 1**, \(R_v \leq k\). Each vertex \(y\) in \(G_t\) records the values of \(r_v\) for vertices \(v\) whose broadcast reached \(y\), and also the distances in \(G_t\) to these vertices. Then \(y\) orders these vertices \(v_1, \ldots, v_s\) in non-increasing order according to \(m_i = r_{v_i} - d_{G_t}(y, v_i)\). We declare that \(y\) is added to \(W_t\) iff \(m_1 - m_2 > 1\).
Observe that all $m_i$ are nonnegative, since $y$ will hear the broadcast of $v_i$ only if $d_{G_t}(y, v_i) \leq R_v$, the latter is at most $r_{v_i}$. If $s = 1$, i.e. there is no second broadcast that reached $y$, define $m_2 = 0$ (observe $m_1$ is well defined as $y$ also broadcasts). If indeed $y$ joins $W_t$, then we say that $y$ chose the center $v_1$.

We begin by analyzing the strong diameter of the blocks.

**Observation 2.** If $y$ chose $v_1$ as a center at phase $t$, then $d_{G_t}(v_1, y) < r_{v_1} - 1$.

**Proof.** If $d_{G_t}(v_1, y) \geq r_{v_1} - 1$, then $m_1 \leq 1$, which implies that $m_1 - m_2 \leq 1$, contradicting the fact that $y$ is added to $W_t$. □

**Claim 3.** If a vertex $y \in V(G_t)$ chose $v$ at phase $t$, then every vertex $x$ on a shortest path from $v$ to $y$ in $G_t$ must have chosen $v$ at phase $t$ as well.

**Proof.** Since $d_{G_t}(v, x) \leq d_{G_t}(v, y)$, the broadcast of $v$ at phase $t$ must have reached $x$ as well, so $x$ records the value $m = r_v - d_{G_t}(x, v)$. Seeking contradiction, assume $x$ did not choose $v$, then there exists $v'$ for which $x$ records the value $m' = r_{v'} - d_{G_t}(x, v')$ with $m' \geq m - 1$ (if there is no such $v'$, then $x$ would have joined $W_t$ with $v$ as center). In particular,

$$d_{G_t}(x, v') \leq r_{v'} - r_v + d_{G_t}(x, v) + 1. \tag{1}$$

It follows that

$$d_{G_t}(y, v') \leq d_{G_t}(y, x) + d_{G_t}(x, v') \leq d_{G_t}(y, x) + r_{v'} - r_v + d_{G_t}(x, v) + 1 \tag{1}$$

$$= (d_{G_t}(y, v) - r_v + 1) + r_{v'} \tag{2}$$

$$< r_{v'} \tag{3}$$

where the last inequality uses Observation 2. Thus $d_{G_t}(y, v') \leq R_{v'}$, so the broadcast of $v'$ will reach $y$, and $y$ will record a corresponding value of

$$r_{v'} - d_{G_t}(y, v') \geq r_v - d_{G_t}(y, v) - 1, \tag{2}$$

that is, it is within 1 of the value $y$ stored for $v$, which contradicts the fact that $y$ chose $v$. □

**Lemma 4.** For every $1 \leq t \leq \lambda$, the block $W_t$ has strong diameter at most $2k - 2$.

**Proof.** Fix any cluster $C$ which is a connected component of $G(W_t)$. We first argue that if all vertices in $C$ chose the same center $v$, then its strong diameter is at most $2k - 2$. To see this, note that by Observation 2 all vertices $y \in C$ are within $r_v - 1$ distance from $v$, since the graph is unweighted, this is at most $R_v - 1 \leq k - 1$ (assuming the event of Lemma 1 holds). By Claim 3, every vertex on a shortest path from $v$ to $y$ (in $G_t$) is also included in $C$, so the strong diameter is at most $2k - 2$.

Consider now the case that there are two vertices $y, z \in C$ that chose different centers $v, u$. We will show that this assumption must lead to a contradiction. Note we may assume that $y, z$ are adjacent, since for any two non-adjacent $y', z'$ who chose different centers, we can simply walk on the path in $C$ (which is connected) from $y'$ to $z'$ until we find adjacent vertices with a center change occurring. W.l.o.g assume $y$ is the vertex which recorded the larger value, that is,

$$r_v - d_{G_t}(y, v) \geq r_u - d_{G_t}(z, u). \tag{4}$$
By the triangle inequality and Observation 2 we see that $d_{G_t}(z,v) \leq d_{G_t}(y,v) + 1 < r_v$, which implies $d_{G_t}(z,v) \leq R_v$, so that the broadcast of $v$ will reach $z$. The value $z$ obtains from $v$ is

$$r_v - d_{G_t}(z,v) \geq r_v - (d_{G_t}(y,v) + 1) \geq r_u - d_{G_t}(z,u) - 1,$$

which contradicts the assumption that $z$ chose $u$. 

We next show that $\lambda$ phases suffice to exhaust the graph, which gives this bound on the number of blocks. To this end, we use the following result from [MPX13, Lemma 4.4] on the order statistics of shifted exponential random variables.

**Lemma 5** ([MPX13]). Let $d_1 \leq \ldots \leq d_q$ be arbitrary values and let $\delta_1, \ldots, \delta_q$ be independent random variables picked from $\mathcal{E}(\beta)$. Then the probability that the largest and the second largest values of $\delta_j - d_j$ are within 1 of each other is at most $1 - e^{-\beta}$. \(^1\)

We use this result to prove the following:

**Claim 6.** For any $y \in V$, and $1 \leq t' \leq \lambda$,

$$\Pr[y \in G_{t'+1}] \leq (1 - (cn)^{-1/k})^{t'}.$$  

**Proof.** Fix any $1 \leq t \leq t'$, and any possible graph $G_t$ such that $y \in V(G_t)$. Let $v_1, \ldots, v_q$ be the vertices of $G_t$ that are in the same connected component of $G_t$ with $y$. Let $d_j = d_{G_t}(v_j, y)$, and $\delta_j = r_{v_j}$ (where each $r_{v_j}$ is sampled independently from $\mathcal{E}(\beta)$). Recall that $y \in W_t$ iff the maximum value among $\delta_j - d_j$ is larger than the second largest by more than 1 (additively). Applying Lemma 5, we conclude that the probability a vertex $y \in V(G_t)$ joins $W_t$ is at least $e^{-\beta} = (cn)^{-1/k}$ (this holds even in the event that no other broadcast reached $y$, by definition of $\mathcal{E}(\beta)$). Since this bound holds regardless of the outcome of previous phases,

$$\Pr[y \in G_{t'+1}] = \Pr \left[ \bigcap_{t=1}^{t'} \{ y \notin W_t \} \right] = \prod_{t=1}^{t'} \Pr[y \notin W_t \mid y \notin W_1, \ldots, y \notin W_{t-1}] \leq (1 - (cn)^{-1/k})^{t'}$$

**Corollary 7.** With probability at least $1 - 1/c$, $G_{\lambda+1}$ is empty.

**Proof.** Using Claim 6 with $t' = \lambda = (cn)^{1/k} \cdot \ln(cn)$, we see that the probability a vertex $y$ did not join any block is at most $(1 - (cn)^{-1/k})^\lambda \leq 1/(cn)$. Applying the union bound on the $n$ vertices, we get that with probability $1 - 1/c$, within $\lambda$ phases the graph is indeed exhausted. 

We are now ready to prove Lemma 1.

---

\(^1\)We state here a special case of their result. The assertion in [MPX13] gives the bound $O(\beta)$, but their proof in fact yields the stronger bound given here.
Proof of Lemma 1. Fix any \( v \in V \). Since each \( r_v \) is sampled independently from \( \mathcal{E} \mathcal{X} \mathcal{P}(\beta) \), we have for any \( 1 \leq t \leq \lambda, \Pr[r_v^{(t)} \geq k + 1] = e^{-\beta(k+1)} \). By using Claim 6 with \( t' = i \cdot (cn)^{1/k} \) (for some \( 0 \leq i \leq \ln(cn) \)), we obtain \( \Pr[v \in G'_{v+1}] \leq e^{-i} \). Now,

\[
\Pr[E_v] \\
\leq \sum_{t=1}^{\lambda} \Pr[r_v^{(t)} \geq k + 1 \mid v \in G_t] \cdot \Pr[v \in G_t] \\
\leq \sum_{i=0}^{\ln(cn)} \sum_{t=1}^{(cn)^{1/k}} \Pr[r_v^{(i \cdot (cn)^{1/k} + t)} \geq k + 1 \mid v \in G'_{i \cdot (cn)^{1/k} + t}] \\
\cdot \Pr[v \in G'_{i \cdot (cn)^{1/k} + t}] \\
\leq \sum_{i=0}^{\ln(cn)} e^{-i} \cdot \sum_{t=1}^{(cn)^{1/k}} e^{-\beta(k+1)} \\
= \sum_{i=0}^{\ln(cn)} e^{-i} \cdot (cn)^{1/k} \cdot (cn)^{-1 - 1/k} \\
\leq 2/(cn)
\]

The lemma follows from a union bound over the \( n \) vertices. \( \square \)

We conclude by analyzing the running time and messages size. Note that there are \( \lambda = (cn)^{1/k} \cdot \ln(cn) \) phases, and each phase requires \( k \) rounds (assuming Lemma 1), so the total number of rounds is as promised. We claim that our algorithm can in fact be implemented efficiently also in the CONGEST model, where messages must be of size at most \( O(\log n) \) words. This follows since at every round, every vertex can sort the values \( m_i \) it has so far, and send to its neighbors only the top two from its list. This is because the values \( [m_i] \) determine the remaining range the message of \( v_i \) needs to be forwarded to, and clustering decisions are based only on the largest two values, so the third and onward values in \( v_i \)'s list will not be used by any other vertex.

### 2.1 Improved Number of Blocks

Here we show how to improve the bound on the number of colors to \( O(k \cdot n^{1/k}) \), and prove the following.

**Theorem 2.** For any unweighted graph \( G = (V, E) \) on \( n \) vertices, and parameters \( 1 \leq k \leq \ln n, 5 < c \), a variant of our randomized distributed algorithm computes, with probability at least \( 1 - 5/c \), a strong \( (2k - 2, 4k(cn)^{1/k}) \) network decomposition of \( G \). The number of rounds required is \( O(k^2(cn)^{1/k}) \), and each message consists of \( O(1) \) words.

The main difference from the previous construction is that the parameter \( \beta \) of the exponential distribution will change at certain points. There will be \( \ln n \) stages, each stage consists of a certain number of phases in which we use the same value of \( \beta \). The first stage lasts \( s_0 = 2(cn)^{1/k} \) phases in which we use \( \beta_0 = \ln(cn)/k \). The next stage lasts \( s_1 = 2(cn/e)^{1/k} \) phases, in which we use \( \beta_1 = \ln(en/e)/k \). In general, the \( i \)-th stage lasts \( s_i = 2(cn/e^i)^{1/k} \) phases, and we use \( \beta_i = \ln(cn/e^i)/k \) in these phases. For \( 0 \leq i \leq \ln n \), denote by \( J_i \) the set of phases in the \( i \)-th stage, that is, \( J_i = \{ \sum_{j=0}^{i-1} s_j + 1, \ldots, \sum_{j=0}^{i} s_j \} \).
The total number of phases, which bounds the number of colors needed, is thus

\[
\sum_{i=0}^{\ln n} s_i = 2 \sum_{i=0}^{\ln n} (cn/e^i)^{1/k} \leq 2(cn)^{1/k} \sum_{i=0}^{\infty} e^{-i/k} \leq 4k(cn)^{1/k}.
\]

The strong diameter bound of Lemma 4 holds regardless of which \( \beta \) we use, as long as an analogue of Lemma 1 holds. Decreasing the parameter \( \beta \) of the exponential distribution increases the probability that a vertex joins a block (so we need less blocks). However, the radius of blocks tend to increase as \( \beta \) gets smaller. The following claim implies that the graph is exhausted with high probability.

**Claim 8.** For any vertex \( y \in V \), \( 0 \leq i \leq \ln n \), and \( t \in J_i \),

\[
\Pr[y \in G_t] \leq e^{-2i}.
\]  

(5)

**Proof.** In order to be included in \( G_t \), \( y \) must not be selected to a block in any phase of any of the stages \( 0, 1, \ldots, i-1 \). By Lemma 5, the probability that \( y \) did not join a block in a certain phase of stage \( j \) is at most \((1 - e^{-\beta_j})\) (even conditioning on anything that happened in previous phases), thus the probability it survived until stage \( i \) is at most

\[
\prod_{j=0}^{i-1} (1 - e^{-\beta_j})^{s_j} = \prod_{j=0}^{i-1} \left( 1 - \left( \frac{cn}{e^j} \right)^{-1/k} \right)^2 cn e^{j+1/k} \cdot e^{-2i} = e^{-2i}.
\]

The claim implies (by the union bound), that with probability at least \( 1 - 1/n \), there are no remaining vertices after stage \( \ln n \). It remains to prove an analogue of Lemma 1, and argue that with probability at least \( 1 - 4/c \), none of the events \( E_v \) took place. We calculate,

\[
\Pr[E_v] \leq \sum_{i=0}^{\ln n} \sum_{t \in J_i} \Pr[r_v(t) \geq k+1 \mid v \in G_t] \cdot \Pr[v \in G_t]
\]

\[
\leq \sum_{i=0}^{\ln n} \sum_{t \in J_i} e^{-\beta_i(k+1)} \cdot e^{-2i}
\]

\[
= \sum_{i=0}^{\ln n} 2 \left( \frac{cn}{e^i} \right)^{1/k} \cdot \left( \frac{e^i}{cn} \right)^{1+1/k} \cdot e^{-2i}
\]

\[
= \frac{1}{n} \sum_{i=0}^{\ln n} \frac{2}{e^i} \leq \frac{4}{cn}.
\]

So by the union bound, with probability at least \( 1 - 4/c \), none of events \( E_v \) occurred, as desired.
2.2 High Radius Regime

Note that in Theorem 1 and Theorem 2 the number of blocks is $\Omega(\log n)$ for any choice of $k$. In the regime that $k$, the parameter governing the radius, is larger than $\ln n$, we can get fewer than $\ln n$ blocks. Concretely, by Claim 6 we have that the probability that a vertex $y$ is not in any of the first $\lambda$ blocks is at most $(1 - (cn^{-1/k})^\lambda \leq (\ln(cn)/k)^\lambda$ (here we use the estimate $1 - e^{-x} \leq x$, which is useful when $x$ is small). We would like this probability to be at most $1/cn$, so that the graph will be empty after $\lambda$ phases with probability at least $1 - 1/c$. To this end, we need

$$\lambda = \frac{\ln(cn)}{\ln(k/\ln(cn))}.$$

In other words, if the number of blocks we want is $\lambda$, then we need to take $k = (cn)^{1/\lambda} \cdot \ln(cn)$, exactly the inverse tradeoff of Theorem 1.

**Theorem 3.** For any unweighted graph $G = (V, E)$ on $n$ vertices, and parameters $1 \leq \lambda \leq \ln n$, $c > 3$, our randomized distributed algorithm computes, with probability at least $1 - 3/c$, a strong $(2(cn)^{1/\lambda} \cdot \ln(cn), \lambda)$ network decomposition of $G$. The number of rounds required is $\lambda(cn)^{1/\lambda} \cdot \ln(cn)$, and each message consists of $O(1)$ words.

3 Acknowledgement

We are grateful to Nati Linial for discussions that initiated this work. We would also like to thank an anonymous reviewer of PODC’16 and Mohsen Ghaffari for pointing out to us the result of [ABCP96]. We are particularly grateful to Mohsen for explaining us the algorithm and the proof of [ABCP96].

References


