Locally Computing Edge Orientations

Slobodan Mitrović
University of California, Davis, CA, USA

Ronitt Rubinfeld
Massachusetts Institute of Technology, Cambridge, MA, USA

Mihir Singhal
University of California, Berkeley, CA, USA

Abstract

We consider the question of orienting the edges in a graph G such that every vertex has bounded out-degree. For graphs of arboricity α, there is an orientation in which every vertex has out-degree at most α and, moreover, the best possible maximum out-degree of an orientation is at least $\alpha - 1$. We are thus interested in algorithms that can achieve a maximum out-degree of close to α. A widely studied approach for this problem in the distributed algorithms setting is a "peeling algorithm" that provides an orientation with maximum out-degree $\alpha(2+\epsilon)$ in a logarithmic number of iterations.

We consider this problem in the local computation algorithm (LCA) model, which quickly answers queries of the form "What is the orientation of edge (u, v)?" by probing the input graph. When the peeling algorithm is executed in the LCA setting by applying standard techniques, e.g., the Parnas-Ron paradigm, it requires $\Omega(n)$ probes per query on an n-vertex graph. In the case where G has unbounded degree, we show that any LCA that orients its edges to yield maximum out-degree r must use $\Omega(\sqrt{n}/r)$ probes to G per query in the worst case, even if G is known to be a forest (that is, $\alpha = 1$). We also show several algorithms with sublinear probe complexity when G has unbounded degree. When G is a tree such that the maximum degree Δ of G is bounded, we demonstrate an algorithm that uses $\Delta n^{1-\log_\Delta r + o(1)}$ probes to G per query. To obtain this result, we develop an edge-coloring approach that ultimately yields a graph-shattering-like result. We also use this shattering-like approach to demonstrate an LCA which 4-colors any tree using sublinear probes per query.

2012 ACM Subject Classification Theory of computation → Streaming, sublinear and near linear time algorithms

Keywords and phrases local computation algorithms, edge orientation, tree coloring

Digital Object Identifier 10.4230/LIPIcs.ESA.2024.89

1 Introduction

Orienting graph edges while obeying certain constraints has applications in many computational settings. For instance, low-out-degree orientation has been studied in the dynamic [23, 56, 13, 84, 28], distributed [44, 38, 42, 52, 85, 53], and massively parallel computation [43, 16] settings. The general problem of graph orientation is of significant interest as it serves as an important algorithmic tool for other computational problems. In their celebrated result, when the input is given as a rooted tree where each edge is oriented toward its parent, Cole and Vishkin [29] show how to 3-color a tree in only $O(\log^* n)$ many distributed rounds. The algorithm of [84] employs low-out-degree graph orientation to obtain a dynamic algorithm for graph coloring, and the works [75, 56, 14] apply results from dynamic edge orientation in designing algorithms for matching. To dynamically maintain spanners, the work of [20] develops a method that also relies on graph orientation. The authors of [33] design a local computation algorithm (LCA) for bounded-reachability orientations (a different class of orientations) to develop an efficient LCA for coloring. It is well-known that the low-out-degree orientation and densest subgraph are problems that are dual to each other [27, 7]. Moreover, graph orientation has applications in small subgraph counting and listing [12, 16]. Our work focuses on low-out-degree orientation in the context of LCAs.

1.1 The problem of r-orientation

Given a graph G with n vertices, we consider the problem of orienting its edges such that every vertex has out-degree at most r, for some parameter r. We call such an orientation an *r-orientation*. Letting α denote the arboricity and ρ the pseudo-arboricity of G, the best possible achievable value of r is $r = \rho$, and it is also known that $\rho \leq \alpha \leq \rho + 1$ [79, 87, 19]. However, we are also interested in approximation algorithms where we can achieve some value $r \geq \rho$. Since ρ and α differ by at most one, we will mostly compare r to α instead of ρ since it will make some of our algorithms easier to describe.

A line of work [44, 38, 42, 85, 52, 53] studied this problem in the distributed LOCAL model in which, under different conditions on $\epsilon\alpha$, they demonstrate algorithms that use $O(\operatorname{poly}\log n)$ rounds to achieve a value of $r = \alpha(1+\epsilon)$. In particular, Su and Vu [85] provide such an orientation in $\tilde{O}(\log^2 n/\epsilon^2)$ rounds, while Harris, Su, and Vu [53] improve the dependence on $1/\epsilon$ from quadratic to linear at the expense of an additional $\log n$ factor in the round complexity.

We consider this problem in the local computation algorithm (LCA) model (fully defined in Section 2.1), in which the algorithm must be able to orient any input edge such that many copies of the algorithm will, with no interaction between them except for a shared random string, produce a consistent orientation. To the best of our knowledge, the low-out-degree orientation problem has not been previously studied in the LCA setting.

A common method for obtaining LCAs from distributed algorithms is the Parnas-Ron paradigm [77]. As we will discuss further in Section 1.3, in the regime we consider, the Parnas-Ron paradigm does not give any nontrivial sublinear-time algorithm.

1.2 Our results

As our first result, we show that an LCA that finds an r-orientation (even when r depends on n) requires at least $\Omega(n^{1/2}/r)$ probes per query, even if the algorithm is randomized and the input graph is a forest (with $\alpha = 1$).

▶ **Theorem 1** (Rephrasing of Theorem 6)**.** *For any parameter r, any LCA randomized algorithm that yields an r-orientation with probability at least 0.9 must use at least $\Omega(n^{1/2}/r)$ probes per query in the worst case.*

In fact, this lower bound holds against an LCA model with relatively strong queries: we allow algorithms to make adjacency-list, adjacency-matrix, and degree probes.

We then show upper bounds for the problem of r-orientation in arboricity-α graphs of unbounded degree, with different algorithms for different regimes of r. These upper bounds are also polynomial in n, though there is some separation between them and the lower bound of $\Omega(n^{1/2}/r)$.

▶ **Theorem 2** (Rephrasing of Propositions 3.1–3.3)**.** *Suppose there is a parameter r, and an input graph G of arboricity α. Then, if $r \geq 10(\alpha^2 n)^{1/3}$, there is a randomized LCA that can r-orient G with at most $\tilde{O}(\max\{\alpha n/r^2, 1\})$ probes per query. Moreover, if $\alpha = 1$ (i.e., G is a forest), then there is also an LCA which can do so for any r, with at most $\tilde{O}(n/r)$ probes per query.*

For example, in the case of a forest ($\alpha = 1$) and when $r = 10n^{1/3}$, this result gives an LCA for r-orientation with probe complexity $\tilde{O}(n^{1/3})$. In contrast, the lower bound given by Theorem 1 is $\Omega(n^{1/6})$.

Finally, we consider the bounded-degree case, where G has maximum degree Δ, in the specific case where $\alpha = 1$ (that is, G is a forest). In this case, any orientation achieves $r \le \Delta$, so the meaningful case to consider is where $r < \Delta$. We show a sublinear algorithm in the case where $r = \Delta^{1-\Omega(1)}$:

▶ **Theorem 3** (Rephrasing of Theorem 7)**.** *Let r be a parameter and G be an input forest with maximum degree Δ. Then, there is a randomized LCA that r-orients G, with at most $\Delta n^{1-\log_\Delta r + o(1)}$ probes per query.*

To prove this theorem, we use a concentration bound on the size of connected components of a random subgraph of a bounded-degree forest. We also use this to derive an LCA for 4-coloring any tree (or forest) with bounded degree in sublinear time. Specifically, we show the following:

▶ **Theorem 4.** *Let G be an input forest with maximum degree Δ, where Δ is a constant. Then, there is a randomized LCA that 4-colors G with query complexity $O(n^{1-\beta})$, where $\beta > 0$ is a constant depending on Δ.*

1.2.1 Future directions

It remains open whether similar upper bounds for r-orientation can be obtained for other families of graphs. We are optimistic that ideas akin to those we develop for Theorem 3 might yield new upper bounds for minor-free graphs. This is because minor-free graphs do not expand, which intuitively is required for the coloring approach we design to prove Theorem 3. We also believe that useful tools in tackling this question include LCAs for minor-free partitioning oracles [54, 67, 65]. It would be quite interesting to obtain upper bounds for bounded arboricity graphs that go beyond our result Theorem 2 or to bring the lower bound of Theorem 1 up to match the result of Theorem 2 in the case of a forest. We expect that techniques we develop to prove Theorem 3 do not transfer to bounded arboricity graphs, as there are even expanders with arboricity 3.

1.3 Related work

Orientation and Parnas-Ron paradigm. The Parnas-Ron paradigm [77] is an important tool for converting distributed algorithms for use in sublinear-time models. In particular, if there is a T-round distributed local algorithm $\mathcal{A}$, then applying the Parnas-Ron paradigm is equivalent to collecting the T-hop neighborhood of a vertex v and using that neighborhood to obtain the output of $\mathcal{A}$ for v, thus getting an LCA with query complexity $O(\Delta^T)$.

There is a classical peeling algorithm that, given a graph of arboricity at most α, finds a $((2+\epsilon)\alpha)$-orientation in $O(\log n)$ parallel or LOCAL rounds, for a constant $\epsilon > 0$. In each step, this algorithm considers all the vertices with degree at most $(2+\epsilon)\alpha$. All the edges incident to these vertices are oriented outward, and those vertices are then removed from the graph. It is easy to show that this algorithm takes $O(\log n)$ such steps. By following the Parnas-Ron paradigm, in graphs with maximum degree Δ, this peeling algorithm can be simulated with $\Delta^{O(\log n)}$ LCA probes, which yields a trivial upper-bound for any $\Delta > 1$ – this upper-bound amounts to gathering the entire graph. If one aims for a more relaxed approximation, e.g., a $\sqrt{\Delta}$-orientation, then the aforementioned peeling algorithm takes $\log_{\sqrt{\Delta}} n$ many steps. Nevertheless, even in this case the Parnas-Ron approach yields $\Delta^{\log_{\sqrt{\Delta}} n} = n^2$ LCA probe complexity. Moreover, suppose one aims for any LCA probe complexity that is sublinear in n. In that case, the peeling algorithm provides only the guarantee of Δ-approximate orientation, which is trivial to obtain by simply orienting every edge arbitrarily.

Orientation and LLL. Several works study the LCA and the closely related VOLUME complexity of the distributed Lovász local lemma (LLL) [21, 37, 30, 22]. Despite significant progress in understanding the randomized LCA complexity of sinkless orientation and LLL, those techniques do not seem to apply to the problem we study. In particular, the LLL applies in cases where a random orientation will succeed at most vertices (as is the case with sinkless orientation). However, for the problem of r-orientation, a random orientation is very unlikely to be close to correct.

LCA sparsification. One recent development in the LCA model is a graph sparsification technique which is based on considering only a carefully selected subset of the neighbors of a vertex v in order to update the state of v. This approach has led to new advances for problems such as approximate matching [45, 62], maximal independent set [45, 41], set cover [50], and graph coloring [25]. However, it is unclear how to apply this technique in the context of r-orientation.

LCA shattering. Our approach for bounded-degree forests is reminiscent of the graph-shattering technique, which has been very influential in designing efficient LCA algorithms for a number of problems, including approximate matching [68, 9, 45], maximal independent set [81, 3, 40, 9, 45], graph coloring [9, 25] and LLL algorithms [2, 73, 37, 26]. The ideas in this line of work can be traced back to Beck's analysis of the algorithmic LLL [10]. The general idea in these results is to first execute an algorithm that finds only a partial solution, e.g., finds an independent set that is not necessarily maximal. For properly designed algorithms, it can be shown that such an approach "shatters" the graph into relatively small connected components of interest, e.g., after removing the found independent set and its neighbors from the graph each connected component has only $\mathrm{poly}(\Delta)$ vertices. These $\mathrm{poly}(\Delta)$ vertices in a single connected component can be then processed with $\mathrm{poly}(\Delta)$ LCA probes with a simple graph traversal.

We remark that our approach departs from this general scheme in that that we do not design an algorithm that finds a partial solution first. Rather, we immediately shatter the input tree into more manageable components.

Other related work. Some prior work has also studied the complexity of LCA algorithms in unbounded-degree graphs. For instance, [33] develop an LCA algorithm for $O(\Delta^2 \log \Delta)$-coloring a graph with maximum degree $\Delta = o(\sqrt{n})$. Another work [68] designs two algorithms for graphs that do not have constant degree: the first approach is for maximal independent set which has a quasi-polynomial probe complexity in the maximum degree Δ, while the second one is for approximate maximum matchings which uses $\mathrm{poly}(\Delta, \log n)$ probes per query. Yet another line of work [78, 5] develops an LCA algorithm to construct an $O(k^2)$-spanner with $\tilde{O}(n^{1+1/k})$ edges with probe complexity $n^{2/3-\Omega_k(1)}\,\mathrm{poly}(\Delta)$.

Several works study the complexity of LLL and related graph problems in LCA, VOLUME, LOCAL, and locally checkable labeling (LCL) models [48, 21, 57, 80, 22, 51, 8]. We note that our lower-bound applies to the LCA setting which implies, as we discuss in Section 2, that it also applies in the VOLUME model. The authors of [22] show that *deterministically* coloring a tree with maximum degree $\mathrm{poly}(c)$ with $c \geq 2$ colors, for any fixed constant c, requires $\Theta(n)$

VOLUME probes. Another line of work on the hierarchies of VOLUME complexities [80], shows that there are graph problems whose randomized VOLUME complexity is $\Theta(n^c)$, for any constant $c \in (0,1)$. The celebrated result [77] proves that to approximate a vertex cover within a constant factor multiplicative and ϵn additive error it is needed to use at least the order of the average degree many probes.

2 Preliminaries

2.1 LCA model

Throughout this paper, we use the LCA model of computation, which was originally introduced in [81, 3]. The goal of this model, in short, is to be able to orient any input edge e quickly – in particular, without necessarily computing the orientations of the rest of the graph.

Formally, in this model, the algorithm $\mathcal{A}$ receives, as a query, an edge e of the graph G. It also has probe access to an adjacency matrix of G, an adjacency list of G, and the degrees of the vertices of G. Specifically, for any i, u, v, $\mathcal{A}$ can probe the following: whether $\{u, v\}$ is an edge of the graph, the i-th neighbor of v, or the degree of v. $\mathcal{A}$ is assumed to know the number of vertices and their IDs beforehand. $\mathcal{A}$ also has access to a source of shared randomness, which does not use up probes to access. Then, after making some probes to the input, $\mathcal{A}$ must return an orientation of e. Suppose that for each edge of G, one separate copy of $\mathcal{A}$ were run, with the same source of shared randomness across all copies (but no ability to otherwise communicate). Then we wish that the resulting orientation should satisfy the desired property, e.g., low out-degree for every vertex, with a given (high) probability. We will be concerned with the *probe complexity* of such an algorithm, i.e., the maximum number of probes it can possibly use on any edge. In this paper, we will not be concerned with the time or space complexity of our algorithms. In particular, we will not limit how long the shared random string may be.

Though we have defined the model to have access to adjacency matrix probes, all of the algorithms we describe do not use this kind of probe, instead only using adjacency list and degree probes. Our lower bounds, on the other hand, are still robust even against algorithms that use adjacency matrix probes.

The VOLUME model, introduced in [80], is another computation model which is similar to, but slightly weaker than, the LCA model. Since it is strictly weaker, this means that our lower bounds also carry over to the VOLUME model.

There has been significant interest in designing LCA algorithms for fundamental problems in computer science. Some examples include locally (list)-decodable codes [34, 70, 39, 35, 63, 88, 46, 4, 86, 49, 58, 64, 11], local decompression [74, 82, 36, 47], local reconstruction and filters for monotone and Lipshitz functions [1, 83, 15, 59, 6, 66], and local reconstruction of graph properties [60]. The study of LCAs has been very active in the past few years, with recent results that include constructing maximal independent sets [81, 3, 9, 69, 33, 40, 45, 41], coloring [25], approximate maximum matchings [76, 89, 71, 72, 69, 45, 61], satisfying assignments for k-CNF [81, 3], local computation mechanism design [55], local decompression [31], local reconstruction of graph properties [24], minor-free graph partitioning [54, 67, 65], and local generation of large random objects [32, 18, 17].

2.2 Tools and notation

We state Yao's minimax principle, applied specifically to LCAs in the form that we will use.

▶ **Theorem 5** (Yao's minimax principle). *Suppose there exists a randomized LCA $\mathcal{A}$ with worst-case probe complexity $f(n)$ which solves a problem with probability p. Then, for any distribution $\mathcal{D}$ of inputs, there is a deterministic LCA $\mathcal{A}$ with probe complexity $f(n)$ which with probability p solves the problem on an input drawn from $\mathcal{D}$.*

Given a graph $G = (V, E)$, we use $\deg_G u$ to denote the degree of vertex $u \in V$. When G is clear from the context, we omit the subscript G and write $\deg u$ only.

When we use little-o asymptotic notation, it is assumed to be with respect to the variable n, the number of vertices in the input graph. We will also frequently write inequalities that only hold when n is sufficiently large, implicitly using the fact that our asymptotic statements are usually trivially true when n is bounded. Moreover, we will often omit floor and ceiling signs for clarity.

We use the phrase "with high probability" to mean "with probability at least $1 - n^{-c}$," where $c > 1$ is a constant that is sufficiently large such that all union bounds which are used henceforth will still yield sufficiently small probabilities. This is a slight abuse of notation, since the threshold for c will vary in each usage, but the meaning will be clear from context. We also use the phrase "with very high probability" to mean "with probability at least $1 - f(n)$," where $f(n)$ is smaller than n^{-c} for any constant c, for sufficiently large n.

3 Orientation in unbounded-degree graphs

We first consider the question of r-orientation in general graphs of arboricity at most α, with no further restriction on the graph.

3.1 Lower bound for orientation in unbounded-degree graphs

First, we demonstrate a lower bound that shows that even in the $\alpha = 1$ (i.e., forest) case, we need $\Omega(n^{1/2}/r)$ probes to r-orient a graph. Essentially, the proof constructs a random tree in which it is hard to find a particular star hidden inside the tree, so the edges of the star must be oriented randomly. This lower bound is particularly powerful since it works even for an LCA that can probe both the adjacency lists and the adjacency matrix (as well as the degrees).

▶ **Theorem 6.** *Suppose that there is an LCA which, on an input forest G with n vertices, r-orients edges of G with probability greater than 0.9. Then, the LCA must use $\Omega(n^{1/2}/r)$ probes per query in the worst case.*

Proof. We assume that $r \leq 0.001n^{1/2}$ since otherwise the statement is obvious. Suppose for contradiction that an LCA $\mathcal{A}$ exists which can perform the required orientation, using less than $0.001n^{1/2}/r$ probes per query.

We describe the construction of a random graph G, which we will use as input to $\mathcal{A}$. Define the parameters $s = 24r$ and $t = n^{1/2}/4s$. Note that $t > 10$. We construct G as follows.

First, we have a set A of $|A| = st$ vertices. Each $a \in A$ is associated with a set S_a of $|S_a| = st$ vertices, whose vertices are all connected to a (forming a star). Also, there is a larger set B of $|B| = s^2t - 2s$ vertices; each $b \in B$ has a corresponding set S_b with $|S_b| = t$ vertices, whose vertices are all connected to b. These constitute all the vertices in the graph. The total number of vertices in the graph is then $(st + 1)|A| + (t + 1)|B| < 4s^2t^2 < n$. So far, all the edges we have described are deterministic.

We describe three different types of edges; for ease of reference, we assign each type as a color. Let the deterministic edges we have drawn so far be black. We refer to all other edges as colored edges.

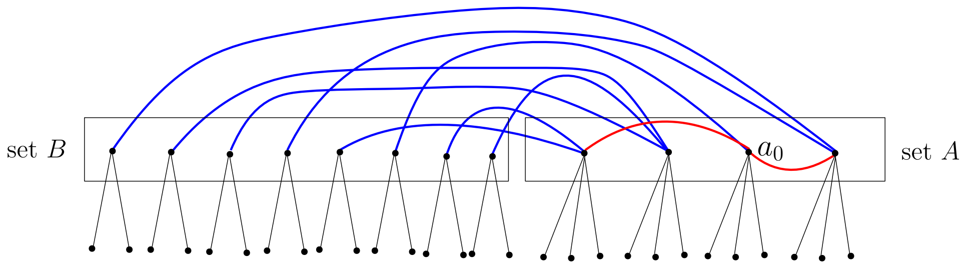

Figure 1 Graph G used to demonstrate lower bound.

Center vertex: Pick a vertex $a_0 \in A$ uniformly at random; we refer to this vertex as the *center*.

Red edges: Pick a set $A' \subseteq A$ of size s not containing a_0 uniformly at random, and for each $a \in A'$, draw a red edge between a and a_0; so, the red edges form a star centered at a_0. These red edges will form the "hidden star" which it will be difficult for the LCA to find.

Blue edges: For each vertex $a \in A$, suppose that a currently has degree k (k must equal one of st, $st + 1$, or $st + s$, since a has st black edges and 0, 1, or s red edges). Then, connect a to $st + s - k$ vertices in B, so that each vertex in B is connected to a single vertex in A. This matching is picked uniformly at random. The size of B is exactly the sum of $st + s - k$ over all vertices a, so every vertex in B can indeed have exactly one blue edge. The purpose of these blue edges is primarily to ensure that the vertices of G have fixed degree, so that degree probes will give no extra information.

Permuted neighbors: The adjacency list for each vertex is also generated as a random permutation of its neighbors.

This completes the description of the graph G. An example rendition of this graph is shown in Figure 1. The red edges form a star centered at a_0; in order to get an r-orientation, many of these edges will have to be oriented away from a_0.

We now outline the remainder of the proof. We will show that the LCA $\mathcal{A}$ cannot keep the out-degree of a_0 below r with probability 0.9; in particular, we show that with probability over 0.1 it must orient at least r of the red edges toward a_0. To prove this, we consider what $\mathcal{A}$ does when it receives a red edge as a query. We will show that, since the black edges constitute most of G, $\mathcal{A}$ will (with some probability) never be able to probe any colored edge at all, and thus must return a deterministic orientation which is independent of which of the endpoints of the queried edge is a_0. It will follow that in expectation, many red edges must be oriented away from a_0, so the expected out-degree of a_0 will be large, which will complete the proof.

By Yao's minimax principle (stated in Theorem 5), we may assume that $\mathcal{A}$ is deterministic; it suffices to show that it is not possible that with probability at least 0.9 (over the randomness of the graph G), the out-degree of a_0 is at most r. Every vertex in G has a fixed degree, regardless of the fact that some of the edges in G are chosen randomly, so we may further assume that $\mathcal{A}$ never makes a degree probe.

We in fact show that for each red edge, upon probing it with $\mathcal{A}$, there is at least a $1/3$ chance it orients toward the center a_0.

Consider the action of $\mathcal{A}$ upon receiving a probe of the red edge (u, v), where $u, v \in A$, and the distribution of the graph G is conditioned on (u, v) being an edge of G (so either u or v is a_0, each with probability $1/2$, and the other is in A'). We claim that with probability at least $1/6$, $\mathcal{A}$ never probes any colored edge (we assume that it does not probe (u, v) in the adjacency matrix since it knows this edge is in G). As a reminder, we are assuming that $\mathcal{A}$ uses less than $0.001n^{1/2}/r$ probes per probe.

Consider the i-th probe that $\mathcal{A}$ makes, assuming that it has not probed any colored edges yet in the previous $i-1$ probes. If the first $i-1$ probes have not revealed any colored edges, then their results are deterministic and fixed, so the i-th probe is also deterministic (since we have assumed that $\mathcal{A}$ is deterministic). We claim that the probability that this fixed probe yields a colored edge is at most $1/t$. Since the probe is deterministic, it must be an adjacency list probe at a fixed index for a fixed vertex, or an adjacency matrix probe for two fixed vertices (every vertex has a fixed degree in G, so a degree probe does not reveal any information). We do casework on each of the possible probe types:

- Adjacency list probe for any vertex in A (including u or v): Each vertex in A has s colored edges and st black edges. Since the adjacency list is permuted randomly, the probability that the probe selects a colored edge is $s/(st+s) < 1/t$.
- Adjacency list probe for any vertex in B: Each vertex in B has 1 colored edge and t black edges, so again the probe selects a colored edge with probability $1/(t+1) < 1/t$.
- Adjacency list probe for a vertex in S_a or S_b for some $a \in A$ or $b \in B$: These vertices have no colored edges, so the probe cannot return a colored edge.
- Adjacency matrix probe between either u or v and another vertex in A: either u or v is a_0 with equal probability, and the probability that a_0 is connected to any other given vertex in A (conditioned on it already being connected to the other one of u and v) is $(s-1)/(st-2)$, so the probability that this edge returns a colored (red) edge is $(s-1)/2(st-2) \leq s/2st < 1/t$.
- Adjacency matrix probe between any vertex $a \in A$ and any vertex $b \in B$: for any fixed $a \in A$, conditioned on the selection of the red edges, the neighbors of a in B are a uniformly random subset of B of some fixed size which is at most s. Therefore, the probability that this probe returns an edge is at most $s/|B| = s/(s^2t-2s) < 2s/s^2t < 1/t$.
- Any other adjacency matrix probe: these cannot reveal any colored edge.

So we have shown that the probability that the first $i-1$ probes do not reveal any colored edges but the i-th probe does is at most $1/t$. It follows that the probability that any probe reveals a colored edge is at most $(0.001n^{1/2}/r)(1/t) < 1/3$.

Therefore, with probability $2/3$, when orienting edge (u, v), conditioned on G containing that edge, $\mathcal{A}$ does not probe any colored edges, and thus returns a deterministic orientation. Since u and v are each a_0 with (conditional) probability $1/2$, this means that $\mathcal{A}$ orients this edge away from a_0 with probability at least $2/3 - 1/2 = 1/6$.

Now, the expected out-degree of a_0 is at least

$$\sum_{u,v\in A} \Pr[(u,v) \in G] \cdot \Pr[(u,v) \text{ oriented away from } a_0 \mid (u,v) \in G]$$

$$\geq \frac{1}{6} \sum_{u,v\in A} \Pr[(u,v) \in G] = \frac{1}{6}\mathbb{E}[\deg a_0] = \frac{s}{6} = 4r.$$

We have shown that the expected out-degree of a_0 is at least $4r$, but what we actually want to show is that it cannot be at most r with probability 0.9. Indeed, the out-degree of a_0 is bounded above by $s = 24r$, so if it were most r with probability at least 0.9, then the expected out-degree of a_0 would be at most $0.9r + 0.1 \cdot 24r < 4r$, which would be a contradiction. Thus we are done. ◀

3.2 Algorithms for orientation in unbounded-degree graphs

With the $\Omega(n^{1/2}/r)$ lower bound in mind that follows from Theorem 6, we demonstrate some algorithms that achieve comparable upper bounds. Specifically, for r-orientation in forests, we can achieve LCA complexities of $O(1)$ when $r \geq n^{1/2}$, $\tilde{O}(n/r^2)$ when $10n^{1/3} \leq r \leq n^{1/2}$, and $\tilde{O}(n/r)$ otherwise.

3.2.1 $O(1)$ algorithm for large r

First, we show the case where $r \geq n^{1/2}$. In this case, the algorithm is very simple: just orient edges toward the higher degree vertex. This also works with graphs of higher arboricity, with a tradeoff in r.

► **Proposition 3.1.** *There exists an LCA that can $(2\alpha n)^{1/2}$-orient an n-vertex α-arboricity graph with probe complexity $O(1)$.*

Proof. Given edge $\{u, v\}$ with $\deg v \geq \deg u$, orient the edge from v to u. (If the degrees are equal, then return an arbitrary orientation.) Suppose for the sake of contradiction that there is a vertex v with out-degree at least $\sqrt{2n\alpha}$. Then, it has at least $\sqrt{2n\alpha}$ neighbors each with a degree at least $\sqrt{2n\alpha}$, so the total degree of the graph is at least $2n\alpha$. But a graph with arboricity α has at most $\alpha(n-1)$ edges, a contradiction. ◄

3.2.2 $\tilde{O}(\alpha n/r^2)$ algorithm for medium r

The following shows that we can use a similar idea for $r \geq 10n^{1/3}$. We can orient all edges toward vertices that have a very large degree since there are very few of these vertices. With vertices of small degrees (less than r), the condition is trivially satisfied, so we can also orient all edges away from these vertices. This leaves the medium-degree vertices, of which there also cannot be too many. We then essentially consider the subgraph induced by these medium vertices and repeat the previous idea, orienting toward the high-degree vertices. Again, this generalizes to arboricity α.

► **Proposition 3.2.** *Suppose a graph G with n vertices and arboricity α is given as input. Then, for all $10(\alpha^2 n)^{1/3} \leq r \leq (\alpha n)^{1/2}$, there is an LCA which, with high probability, r-orients the edges in G with $\tilde{O}(\alpha n/r^2)$ probes per query.*

Proof. We describe the algorithm. Let $s = r/10$. Call a vertex *small* if its degree is at most r, *large* if its degree is at least $\alpha n/s$, and *medium* otherwise.

Suppose the algorithm receives a query of e with endpoints u, v (assume arbitrarily that u has the smaller ID). Then it proceeds as shown in Algorithm 1, splitting into three cases based on the degrees of its endpoints. The probe complexity in Cases 1 and 2 is $O(1)$, and in Case 3 is $\tilde{O}(d/s) = \tilde{O}(\alpha n/r^2)$, as desired. It remains to check correctness of the algorithm.

Correctness. We wish to check that the out-degree of every vertex is at most r with high probability. Small vertices obviously have out-degree at most $s \leq r$, so we only need to check the medium and large vertices.

The total degree of G is less than $2\alpha n$, so there are at most $2s$ large vertices. Thus, large vertices end up with an out-degree of $2s$ at most since edges from large to non-large vertices are oriented toward the large vertex.

It remains to check the medium vertices. Since the total degree of G is $2\alpha n$, there are also at most $2\alpha n/r$ medium vertices.

Algorithm 1 LCA for r-orienting a graph when $r \geq 10(\alpha^2 n)^{1/3}$.

Input : Graph G with n vertices and arboricity α, with probe access
Query edge $e = (u, v)$, where u has lower ID than v

Perform degree probes on u and v.
if *u or v is small* **then**
  `// Case 1`
  Orient e away from that vertex. (Pick arbitrarily if both are small.)
else if *u or v is large* **then**
  `// Case 2`
  Orient e toward that vertex. (Pick arbitrarily if both are large.)
else
  `// Case 3 (`u `and` v `are both medium)`
  Let $d = \deg u$. `// Note that` $r \leq d \leq \alpha n/s$
  Sample $(1000 d \log n)/s$ neighbors of u independently and uniformly at random (using a fixed part of the shared randomness, depending on e).
  Perform a degree probe on each sampled neighbor.
  if *at most a $2s/d$ proportion of the sampled neighbors are medium* **then**
    Orient e away from u.
  else
    Orient e toward u.

Now, consider a medium vertex w. No edges would have been oriented away from w in Case 1, and since there at most $2s$ large vertices, there are at most $2s$ edges oriented away from w in Case 2.

It remains to analyze the number of edges that are oriented away from w in Case 3 (that is, the medium-to-medium edges). Note that by a Chernoff bound, with high probability, Case 3 orients a medium-to-medium edge (u, v) away from u if it has at most s medium neighbors, and toward u if it has at least $3s$ medium neighbors. We then show that there are at most $3s$ edges oriented away from w in Case 3.

If w has at most $3s$ medium neighbors, then this is obvious, so suppose $\deg w > 3s$. Consider the action of the algorithm when orienting an edge e containing w. When we input $e = (u, v)$ into the algorithm, it is possible for either u or v to be w. If $u = w$, then it (with high probability) orients the edge toward w. Otherwise, if $v = w$, then (with high probability) u must be a medium vertex with at least s medium neighbors in order to orient e toward w. Now, the subgraph of G induced by the medium vertices also has arboricity α, and thus has total degree at most $4\alpha^2 n/r$ by the bound on the number of medium vertices. Therefore there are at most $4\alpha^2 n/rs$ medium vertices u with at least s medium neighbors. Therefore, the number of edges oriented away from w in Case 3 is at most $4\alpha^2 n/rs \leq 3s$.

Thus, in total, every medium vertex has at most $5s < r$ edges oriented away from it, completing the proof. ◀

3.2.3 $\tilde{O}(n/r)$ algorithm for all r, if G is a forest

Finally, we consider the case of small r, in the case where G is a forest. Here, we can still orient edges toward very large-degree vertices. Then, we randomly color all the remaining edges with r colors and orient each color class separately (with maximum out-degree 1 within each color class). We show that the color classes are small with high probability, bounding the LCA complexity. Note that this algorithm only applies to arboricity $\alpha = 1$, i.e., the case of a forest.

▶ **Proposition 3.3.** *For all $r \le n^{1/2}$, there exists an LCA which, with high probability, r-orients edges in an n-vertex forest G which uses $\tilde{O}(n/r)$ probes per query.*

Proof. Color each edge randomly with one of $r/5$ colors (using the shared randomness). Say that a vertex is *large* if its degree is at least $5n/r$. Also, define an edge to be large if one of its vertices is large. Then, we proceed as described in Algorithm 2.

Algorithm 2 LCA for r-orienting a forest for any r.

Input : Forest G with n vertices, with probe access
Query edge $e = (u, v)$, where u has lower ID than v

Using the shared randomness, assign an independent and uniformly randomly chosen color from $\{1, \dots, r/5\}$ to every pair of vertices in G (so that every edge computes the same coloring).
Perform degree probes on u and v.
if *u or v is large* **then**
  Orient e toward that vertex. (Pick arbitrarily if both are large.)
else
  Let c be the color of e.
  Perform a depth-first search from each of the endpoints of e to find the connected component of the edge of color c, but stop the search at any large vertex.
  Orient e toward the minimum-ID vertex in the connected component (which is a tree). `// Note that together, the edges in the connected component of color c form an orientation of that connected component with out-degree at most 1.`

Since G has a total degree at most $2n$, there are at most $2r/5$ large vertices, so at most $2r/5$ large edges are oriented away from each vertex. Also, at most one non-large edge of each color is oriented away from each vertex, so in total each vertex has an out-degree less than r.

Thus this algorithm gives a valid orientation. It remains to see that it has low probe complexity. Let u be one of the endpoints of e. It is enough to check that the depth-first search starting at u searches at most $\tilde{O}(n/r)$ edges with a very high probability (by symmetry, the same will hold for the other endpoint of e).

Indeed, let H be the connected component of $G \setminus e$ containing u; consider it as a tree rooted at u. For a vertex $v \neq u$ in H, let X_v be the indicator random variable of whether the edge from v to its parent has color c. Then, in order for the search to have to check the edges from v to its descendants, X_v must equal 1 (and v must also not be large). The search also always checks the edges from u to its descendants (as long as u is not large); there are at most $O(n/r)$ such edges. Therefore, the total number of edges checked is bounded above by the following quantity:

$$O(n/r) + \sum_{\text{non-large } v \in H \setminus u} X_v \deg v.$$

Since $\deg v \le 5n/r$ for each non-large v, the above sum is a sum of random variables bounded in $[0, 5n/r]$. The X_v are all independent Bernoulli random variables with probability $5/r$, and the total degree of all vertices is at most $2n$. The expectation of the sum is then $O(n/r)$. Then, by a Chernoff bound, the sum is $\tilde{O}(n/r)$ with high probability, so we are done. ◀

4 Orientation in bounded-degree forests

Finally, we present an algorithm that r-orients a forest that has maximum degree Δ, using at most $\Delta n^{1-\log_\Delta r+o(1)}$ probes. The basic algorithm colors the edges of the graph with r colors and orients each color separately with a maximum out-degree of 1. Then, the combined out-degree of any vertex is at most r. We analyze the probe complexity by deriving a Chernoff-style concentration bound on the size of any connected component of any color.

The probe complexity bound is meaningful when r is at least polynomial in Δ (though if $r \geq \Delta$ the problem is trivial). For example, if $\Delta = 100$ and $r = 10$, or if $\Delta = \log n$ and $r = \sqrt{\log n}$, then the resulting probe complexity is $n^{1/2+o(1)}$. Even if Δ is polynomial in n, this algorithm can still be nontrivial (though the Δ factor does become important). For example, if $\Delta = n^{0.2}$ and $r = n^{0.1}$, then the complexity is $n^{0.7+o(1)}$.

Specifically, we show the following theorem.

▶ **Theorem 7.** *Given parameters $r \leq \Delta$, there exists an LCA which r-orients an input forest G with n vertices and maximum degree Δ. For any fixed ϵ and with very high probability, this algorithm uses at most $\Delta n^{1-\log_\Delta r+\epsilon}$ probes for every query, for large enough n.*

We show this by showing a concentration bound on the size of connected components in a random subset of a tree:

▶ **Proposition 4.1.** *Fix a constant ϵ. Suppose the edges of a forest G with n vertices and maximum degree Δ are each colored independently with probability $p \geq \Delta^{-1+\epsilon}$. Then, with very high probability (in n), every connected component of colored edges has size at most $n^{1+\log_\Delta p+\epsilon}$.*

Before we prove Proposition 4.1, we show how we can deduce Theorem 7 from it:

Proof of Theorem 7 assuming Proposition 4.1. First, assume that $r \leq \Delta^{1-\epsilon/2}$ (we will handle the other case at the end). Then the algorithm is given in Algorithm 3.

Algorithm 3 LCA for r-orienting a bounded-degree forest.

Input : Forest G with n vertices and maximum degree Δ, with probe access
Query edge $e = (u, v)$, where u has lower ID than v

1. Using the shared randomness, assign an independent and uniformly randomly chosen color from $\{1, \ldots, r\}$ to every pair of vertices in G (so that every edge computes the same coloring).
2. Let c be the color of e.
3. Using adjacency list probes, perform a depth-first search to find the connected component of e within the edges of G with color c.
4. Orient e toward the minimum-ID vertex in the connected component (which must be a tree). `// This ensures that every vertex has at most 1 edge of color` c `oriented away from itself.`

Since each vertex has at most 1 edge of color c oriented away from itself, the total out-degree of any vertex is at most r, as desired.

Using Proposition 4.1 (with $\epsilon/2$ instead of ϵ), with high probability this has size at most $n^{1-\log_\Delta r+\epsilon/2}$ (with a union bound over all colors). Then, the depth-first search (using the adjacency lists) takes at most $\Delta n^{1-\log_\Delta r+\epsilon/2}$ probes since it must probe the entire adjacency list of every vertex in this connected component.

Finally, we handle the case where $\Delta^{1-\epsilon/2} \leq r \leq \Delta$ (if $r \geq D$ then an arbitrary orientation works). By the previous logic, we can still get a $\Delta^{1-\epsilon/2}$-orientation (which is also an r-orientation) with probe complexity Δn^{ϵ}. But we have $\Delta n^{\epsilon} \leq \Delta n^{1-\log_{\Delta} r+\epsilon}$, so we are done. ◀

The remaining details are provided in the full version of this paper.

References

1 N. Ailon, B. Chazelle, S. Comandur, and D. Liu. Property-preserving data reconstruction. *Algorithmica*, 51(2):160–182, 2008.

2 Noga Alon. A parallel algorithmic version of the local lemma. *Random Structures & Algorithms*, 2(4):367–378, 1991.

3 Noga Alon, Ronitt Rubinfeld, Shai Vardi, and Ning Xie. Space-efficient local computation algorithms. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pages 1132–1139. SIAM, 2012.

4 Sanjeev Arora and Madhu Sudan. Improved low-degree testing and its applications. *Combinatorica*, 23(3):365–426, 2003.

5 Rubi Arviv, Lily Chung, Reut Levi, and Edward Pyne. Improved local computation algorithms for constructing spanners. In *APPROX-RANDOM*, volume 275, pages 42:1–42:23, 2023.

6 Pranjal Awasthi, Madhav Jha, Marco Molinaro, and Sofya Raskhodnikova. Limitations of local filters of lipschitz and monotone functions. In *APPROX-RANDOM*, pages 374–386, 2012.

7 Bahman Bahmani, Ashish Goel, and Kamesh Munagala. Efficient primal-dual graph algorithms for mapreduce. In *International Workshop on Algorithms and Models for the Web-Graph*, pages 59–78. Springer, 2014.

8 Alkida Balliu, Sebastian Brandt, and Dennis Olivetti. Distributed lower bounds for ruling sets. *SIAM Journal on Computing*, 51(1):70–115, 2022.

9 Leonid Barenboim, Michael Elkin, Seth Pettie, and Johannes Schneider. The locality of distributed symmetry breaking. *Journal of the ACM (JACM)*, 63(3):1–45, 2016.

10 József Beck. An algorithmic approach to the lovász local lemma. i. *Random Structures & Algorithms*, 2(4):343–365, 1991.

11 Avraham Ben-Aroya, Klim Efremenko, and Amnon Ta-Shma. Local list-decoding with a constant number of queries. Technical Report TR10-047, Electronic Colloquium on Computational Complexity, April 2010.

12 Suman K Bera, Noujan Pashanasangi, and C Seshadhri. Linear time subgraph counting, graph degeneracy, and the chasm at size six. *arXiv preprint arXiv:1911.05896*, 2019.

13 Edvin Berglin and Gerth Stølting Brodal. A simple greedy algorithm for dynamic graph orientation. *Algorithmica*, 82(2):245–259, 2020.

14 Aaron Bernstein and Cliff Stein. Faster fully dynamic matchings with small approximation ratios. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 692–711. SIAM, 2016.

15 Arnab Bhattacharyya, Elena Grigorescu, Madhav Jha, Kyomin Jung, Sofya Raskhodnikova, and David P. Woodruff. Lower bounds for local monotonicity reconstruction from transitive-closure spanners. *SIAM J. Discrete Math.*, 26(2):618–646, 2012.

16 Amartya Shankha Biswas, Talya Eden, Quanquan C Liu, Ronitt Rubinfeld, and Slobodan Mitrović. Massively parallel algorithms for small subgraph counting. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2022)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022.

17 Amartya Shankha Biswas, Edward Pyne, and Ronitt Rubinfeld. Local access to random walks, january 2022. *Innovations in Theoretical Computer Science (ITCS 2022)*, 2022.

18 Amartya Shankha Biswas, Ronitt Rubinfeld, and Anak Yodpinyanee. Local access to huge random objects through partial sampling. In *11th Innovations in Theoretical Computer Science Conference (ITCS 2020)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020.

19 Markus Blumenstock and Frank Fischer. A constructive arboricity approximation scheme. In *International Conference on Current Trends in Theory and Practice of Informatics*, pages 51–63. Springer, 2020.

20 Greg Bodwin and Sebastian Krinninger. Fully dynamic spanners with worst-case update time. *arXiv preprint arXiv:1606.07864*, 2016.

21 Sebastian Brandt, Orr Fischer, Juho Hirvonen, Barbara Keller, Tuomo Lempiäinen, Joel Rybicki, Jukka Suomela, and Jara Uitto. A lower bound for the distributed lovász local lemma. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 479–488, 2016.

22 Sebastian Brandt, Christoph Grunau, and Václav Rozhoň. The randomized local computation complexity of the lovász local lemma. In *Proceedings of the 2021 ACM Symposium on Principles of Distributed Computing*, pages 307–317, 2021.

23 Gerth Stølting Brodal and Rolf Fagerberg. Dynamic representations of sparse graphs. In *Algorithms and Data Structures: 6th International Workshop, WADS'99 Vancouver, Canada, August 11–14, 1999 Proceedings 6*, pages 342–351. Springer, 1999.

24 Andrea Campagna, Alan Guo, and Ronitt Rubinfeld. Local reconstructors and tolerant testers for connectivity and diameter. In *APPROX-RANDOM*, pages 411–424, 2013. doi:10.1007/978-3-642-40328-6_29.

25 Yi-Jun Chang, Manuela Fischer, Mohsen Ghaffari, Jara Uitto, and Yufan Zheng. The complexity of (δ+ 1) coloring in congested clique, massively parallel computation, and centralized local computation. In *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing*, pages 471–480, 2019.

26 Yi-Jun Chang, Qizheng He, Wenzheng Li, Seth Pettie, and Jara Uitto. Distributed edge coloring and a special case of the constructive lovász local lemma. *ACM Transactions on Algorithms (TALG)*, 16(1):1–51, 2019.

27 Moses Charikar. Greedy approximation algorithms for finding dense components in a graph. In *International workshop on approximation algorithms for combinatorial optimization*, pages 84–95. Springer, 2000.

28 Aleksander BG Christiansen, Jacob Holm, Eva Rotenberg, and Carsten Thomassen. On dynamic α+ 1 arboricity decomposition and out-orientation. In *47th International Symposium on Mathematical Foundations of Computer Science (MFCS 2022)*. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022.

29 Richard Cole and Uzi Vishkin. Deterministic coin tossing with applications to optimal parallel list ranking. *Information and Control*, 70(1):32–53, 1986.

30 Andrzej Dorobisz and Jakub Kozik. Local computation algorithms for coloring of uniform hypergraphs. *arXiv preprint arXiv:2103.10990*, 2021.

31 A. Dutta, R. Levi, D. Ron, and R. Rubinfeld. A simple online competitive adaptation of lempel-ziv compression with efficient random access support. In *Proceedings of the Data Compression Conference (DCC)*, pages 113–122, 2013.

32 Guy Even, Reut Levi, Moti Medina, and Adi Rosén. Sublinear random access generators for preferential attachment graphs. In *44th International Colloquium on Automata, Languages, and Programming, ICALP 2017, July 10-14, 2017, Warsaw, Poland*, pages 6:1–6:15, 2017.

33 Guy Even, Moti Medina, and Dana Ron. Deterministic stateless centralized local algorithms for bounded degree graphs. In *Algorithms-ESA 2014: 22th Annual European Symposium, Wroclaw, Poland, September 8-10, 2014. Proceedings 21*, pages 394–405. Springer, 2014.

34 D. Beaver J. Feigenbaum. Hiding instances in multi-oracle queries. In *Proc. 7th Annual STACS conference*, pages 34–48, 1990.

35 J. Feigenbaum and L. Fortnow. Random self-reducibility of complete sets. *SIAM Journal on Computing*, 22:994–1005, 1993.

36 P. Ferragina and R. Venturini. A simple storage scheme for strings achieving entropy bounds. In *ACM-SIAM Symposium on Discrete Algorithms*, pages 690–696, 2007. doi:10.1145/1283383.1283457.

37 Manuela Fischer and Mohsen Ghaffari. Sublogarithmic distributed algorithms for lov\'asz local lemma, and the complexity hierarchy. *arXiv preprint arXiv:1705.04840*, 2017.

38 Manuela Fischer, Mohsen Ghaffari, and Fabian Kuhn. Deterministic distributed edge-coloring via hypergraph maximal matching. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 180–191. IEEE, 2017.

39 P. Gemmell, R. Lipton, R. Rubinfeld, M. Sudan, and A. Wigderson. Self-testing/correcting for polynomials and for approximate functions. In *Proc. 23rd Annual ACM Symposium on the Theory of Computing*, pages 32–42, 1991.

40 Mohsen Ghaffari. An improved distributed algorithm for maximal independent set. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 270–277. SIAM, 2016.

41 Mohsen Ghaffari. Local computation of maximal independent set. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 438–449. IEEE, 2022.

42 Mohsen Ghaffari, David G Harris, and Fabian Kuhn. On derandomizing local distributed algorithms. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 662–673. IEEE, 2018.

43 Mohsen Ghaffari, Silvio Lattanzi, and Slobodan Mitrović. Improved parallel algorithms for density-based network clustering. In *International Conference on Machine Learning*, pages 2201–2210. PMLR, 2019.

44 Mohsen Ghaffari and Hsin-Hao Su. Distributed degree splitting, edge coloring, and orientations. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2505–2523. SIAM, 2017.

45 Mohsen Ghaffari and Jara Uitto. Sparsifying distributed algorithms with ramifications in massively parallel computation and centralized local computation. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1636–1653. SIAM, 2019.

46 Oded Goldreich, Ronitt Rubinfeld, and Madhu Sudan. Learning polynomials with queries: the highly noisy case. *SIAM Journal on Discrete Mathematics*, 13(4):535–570, 2000.

47 R. González and G. Navarro. Statistical encoding of succint data structures. In *Proceedings of CPM*, pages 295–306, 2006.

48 Mika Göös, Juho Hirvonen, Reut Levi, Moti Medina, and Jukka Suomela. Non-local probes do not help with graph problems. *arXiv preprint arXiv:1512.05411*, 2015.

49 Parikshit Gopalan, Adam R. Klivans, and David Zuckerman. List-decoding Reed Muller codes over small fields. In *Proc. 40th Annual ACM Symposium on Theory of Computing*, pages 265–274, 2008.

50 Christoph Grunau, Slobodan Mitrović, Ronitt Rubinfeld, and Ali Vakilian. Improved local computation algorithm for set cover via sparsification. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2993–3011. SIAM, 2020.

51 Christoph Grunau, Václav Rozhoň, and Sebastian Brandt. The landscape of distributed complexities on trees and beyond. In *Proceedings of the 2022 ACM Symposium on Principles of Distributed Computing*, pages 37–47, 2022.

52 David G Harris. Distributed local approximation algorithms for maximum matching in graphs and hypergraphs. In *2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 700–724. IEEE, 2019.

53 David G Harris, Hsin-Hao Su, and Hoa T Vu. On the locality of nash-williams forest decomposition and star-forest decomposition. In *Proceedings of the 2021 ACM Symposium on Principles of Distributed Computing*, pages 295–305, 2021.

54 Avinatan Hassidim, Jonathan A Kelner, Huy N Nguyen, and Krzysztof Onak. Local graph partitions for approximation and testing. In *2009 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 22–31. IEEE, 2009.

55 Avinatan Hassidim, Yishay Mansour, and Shai Vardi. Local computation mechanism design. *ACM Trans. Economics and Comput.*, 4(4):21:1–21:24, 2016.

56 Meng He, Ganggui Tang, and Norbert Zeh. Orienting dynamic graphs, with applications to maximal matchings and adjacency queries. In *Algorithms and Computation: 25th International Symposium, ISAAC 2014, Jeonju, Korea, December 15-17, 2014, Proceedings*, pages 128–140. Springer, 2014.

57 Dan Hefetz, Fabian Kuhn, Yannic Maus, and Angelika Steger. Polynomial lower bound for distributed graph coloring in a weak local model. In *Distributed Computing: 30th International Symposium, DISC 2016, Paris, France, September 27-29, 2016. Proceedings*, pages 99–113. Springer, 2016.

58 Russell Impagliazzo and Avi Wigderson. $P = BPP$ if E requires exponential circuits: Derandomizing the XOR lemma. In *Proc. 29th Annual ACM Symposium on the Theory of Computing*, pages 220–229, 1997.

59 Madhav Jha and Sofya Raskhodnikova. Testing and reconstruction of lipschitz functions with applications to data privacy. *SIAM J. Comput.*, 42(2):700–731, 2013. doi:10.1137/110840741.

60 Satyen Kale, Yuval Peres, and C. Seshadhri. Noise tolerance of expanders and sublinear expansion reconstruction. *SIAM J. Comput.*, 42(1):305–323, 2013. doi:10.1137/110837863.

61 Michael Kapralov, Slobodan Mitrovic, Ashkan Norouzi-Fard, and Jakab Tardos. Space efficient approximation to maximum matching size from uniform edge samples. *CoRR*, abs/1907.05725, 2019.

62 Michael Kapralov, Slobodan Mitrović, Ashkan Norouzi-Fard, and Jakab Tardos. Space efficient approximation to maximum matching size from uniform edge samples. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1753–1772. SIAM, 2020.

63 J. Katz and L. Trevisan. On the efficiency of local decoding procedures for error-correcting codes. In *Proc. 32nd Annual ACM Symposium on the Theory of Computing*, pages 80–86, 2000.

64 Swastik Kopparty and Shubhangi Saraf. Local list-decoding and testing of sparse random linear codes from high-error. Technical Report 115, Electronic Colloquium on Computational Complexity (ECCC), 2009.

65 Akash Kumar, C Seshadhri, and Andrew Stolman. Random walks and forbidden minors III: poly($d\epsilon^{-1}$)-time partition oracles for minor-free graph classes. In *Electron. Colloquium Comput. Complex.*, volume 28, page 8, 2021.

66 Jane Lange, Ronitt Rubinfeld, and Arsen Vasilyan. Properly learning monotone functions via local reconstruction. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2022.

67 Reut Levi and Dana Ron. A quasi-polynomial time partition oracle for graphs with an excluded minor. *ACM Transactions on Algorithms (TALG)*, 11(3):1–13, 2015.

68 Reut Levi, Ronitt Rubinfeld, and Anak Yodpinyanee. Local computation algorithms for graphs of non-constant degrees. In *Proceedings of the 27th ACM symposium on Parallelism in Algorithms and Architectures*, pages 59–61, 2015.

69 Reut Levi, Ronitt Rubinfeld, and Anak Yodpinyanee. Local computation algorithms for graphs of non-constant degrees. *Algorithmica*, 77(4):971–994, 2017.

70 R. Lipton. New directions in testing. In *Proc. DIMACS Workshop on Distributed Computing and Cryptography*, 1989.

71 Y. Mansour, A. Rubinstein, Shai Vardi, and Ning Xie. Converting online algorithms to local computation algorithms. In *Unpublished manuscript*, 2011.

72 Yishay Mansour and Shai Vardi. A local computation approximation scheme to maximum matching. In *APPROX-RANDOM*, pages 260–273, 2013.

73 Michael Molloy and Bruce Reed. Further algorithmic aspects of the local lemma. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 524–529, 1998.

74 S. Muthukrishnan, M. Strauss, and X. Zheng. Workload-optimal histograms on streams. Technical Report 2005-19, DIMACS Technical Report, 2005.

75 Ofer Neiman and Shay Solomon. Simple deterministic algorithms for fully dynamic maximal matching. *ACM Trans. Algorithms*, 12(1), November 2015.

76 H. N. Nguyen and K. Onak. Constant-time approximation algorithms via local improvements. In *Proc. 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 327–336, 2008.

77 Michal Parnas and Dana Ron. Approximating the minimum vertex cover in sublinear time and a connection to distributed algorithms. *Theoretical Computer Science*, 381(1-3):183–196, 2007.

78 Merav Parter, Ronitt Rubinfeld, Ali Vakilian, and Anak Yodpinyanee. Local computation algorithms for spanners. In *ITCS*, volume 124, pages 58:1–58:21, 2019.

79 Jean-Claude Picard and Maurice Queyranne. A network flow solution to some nonlinear 0-1 programming problems, with applications to graph theory. *Networks*, 12(2):141–159, 1982.

80 Will Rosenbaum and Jukka Suomela. Seeing far vs. seeing wide: Volume complexity of local graph problems. In *Proceedings of the 39th Symposium on Principles of Distributed Computing*, pages 89–98, 2020.

81 Ronitt Rubinfeld, Gil Tamir, Shai Vardi, and Ning Xie. Fast local computation algorithms. *arXiv preprint arXiv:1104.1377*, 2011.

82 K. Sadakane and R. Grossi. Squeezing succinct data structures into entropy bounds. In *ACM-SIAM Symposium on Discrete Algorithms*, pages 1230–1239, 2006.

83 M. E. Saks and C. Seshadhri. Local monotonicity reconstruction. *SIAM Journal on Computing*, 39(7):2897–2926, 2010.

84 Shay Solomon and Nicole Wein. Improved dynamic graph coloring. *ACM Transactions on Algorithms (TALG)*, 16(3):1–24, 2020.

85 Hsin-Hao Su and Hoa T Vu. Distributed dense subgraph detection and low outdegree orientation. In *34th International Symposium on Distributed Computing*, 2020.

86 Madhu Sudan, Luca Trevisan, and Salil Vadhan. Pseudorandom generators without the XOR lemma. *Journal of Computer and System Sciences*, 62(2):236–266, 2001.

87 Herbert Hans Westermann. *Efficient algorithms for matroid sums.* University of Colorado at Boulder, 1988.

88 Sergey Yekhanin. Private information retrieval. *Commun. ACM*, 53(4):68–73, 2010.

89 Y. Yoshida, Y. Yamamoto, and H. Ito. An improved constant-time approximation algorithm for maximum matchings. In *Proc. 41st Annual ACM Symposium on the Theory of Computing*, pages 225–234, 2009.