View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Dagstuhl Research Online Publication Server

On the Probe Complexity of Local Computation Algorithms

Uriel Feige[1]
Weizmann Institute of Science, Rehovot, Israel
uriel.feige@weizmann.ac.il

Boaz Patt-Shamir[2]
Tel Aviv University, Tel Aviv, Israel
boaz@tau.ac.il

Shai Vardi[3]
California Institute of Technology, Pasadena, CA, USA
svardi@caltech.edu

Abstract

In the Local Computation Algorithms (LCA) model, the algorithm is asked to compute a part of the output by reading as little as possible from the input. For example, an LCA for coloring a graph is given a vertex name (as a "query"), and it should output the color assigned to that vertex after inquiring about some part of the graph topology using "probes"; all outputs must be consistent with the same coloring. LCAs are useful when the input is huge, and the output as a whole is not needed simultaneously. Most previous work on LCAs was limited to bounded-degree graphs, which seems inevitable because probes are of the form "what vertex is at the other end of edge i of vertex v?". In this work we study LCAs for unbounded-degree graphs. In particular, such LCAs are expected to probe the graph a number of times that is significantly smaller than the maximum, average, or even minimum degree. We show that there are problems that have very efficient LCAs on any graph - specifically, we show that there is an LCA for the weak coloring problem (where a coloring is legal if every vertex has a neighbor with a different color) that uses $\log^* n + O(1)$ probes to reply to any query. As another way of dealing with large degrees, we propose a more powerful type of probe which we call a *strong* probe: given a vertex name, it returns a list of its neighbors. Lower bounds for strong probes are stronger than ones in the edge probe model (which we call *weak* probes). Our main result in this model is that roughly $\Omega(\sqrt{n})$ strong probes are required to compute a maximal matching.

Our findings include interesting separations between closely related problems. For weak probes, we show that while weak 3-coloring can be done with probe complexity $\log^* n + O(1)$, weak 2-coloring has probe complexity $\Omega(\log n / \log\log n)$. For strong probes, our negative result for *maximal* matching is complemented by an LCA for $(1-\epsilon)$-approximate *maximum* matching on regular graphs that uses $O(1)$ strong probes, for any constant $\epsilon > 0$.

2012 ACM Subject Classification Theory of computation → Graph algorithms analysis

Keywords and phrases Local computation algorithms, sublinear algorithms

Digital Object Identifier 10.4230/LIPIcs.ICALP.2018.50

[1] Supported in part by the Israel Science Foundation (grant No. 1388/16). Work partly done in Microsoft Research, Herzeliya, Israel.
[2] Supported in part by the Israel Science Foundation (grant No. 1444/14).
[3] Supported in part by the I-CORE in Algorithms Postdoctoral Fellowship, the Linde Foundation and NSF grants CNS-1254169 and CNS-1518941. Part of the research was carried out when Shai was a postdoctoral researcher at the Weizmann Institute of Science.

© Uriel Feige, Boaz Patt-Shamir, and Shai Vardi;
licensed under Creative Commons License CC-BY
45th International Colloquium on Automata, Languages, and Programming (ICALP 2018).
Editors: Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella;
Article No. 50; pp. 50:1–50:14

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Related Version A full version of the paper can be found at [9], https://arxiv.org/abs/1703.07734.

Acknowledgements We thank Noga Alon for an enlightening discussion and the anonymous reviewers for their useful comments.

1 Introduction

In classical algorithmic models, an algorithm is given an input and is required to compute an output. When dealing with truly massive data, such as the Internet, just reading the entire input may be impractical. The model of *local computation algorithms* (LCAs), as studied by Rubinfeld et al. [33], proposes the following approach. An algorithm in the LCA model is required to produce only a part of the output specified by a "query," and is expected to access only a small part of the input (without any pre-processing) using some simple "probes." For example, an LCA for maximal independent set (MIS) is given a vertex ID as a query, and is expected to return a "yes/no" answer, indicating whether the queried vertex is or is not in the MIS; all replies to queries must be consistent with the same MIS. To do that, the LCA can use probes of the form "which vertex is the ith neighbor of vertex v?", where v and i are the probe arguments. One of the main goals in LCA design is to minimize the number of probes required to produce an answer to a query.

In this paper we consider LCAs for graph problems. Almost exclusively, known LCAs for graph problems are efficient only for graphs of bounded degrees (with the notable exception of [20], which gives LCAs with polylog probe complexity for graphs with polylog degree). This may appear inevitable, because edge probes don't allow sublinear solutions to even learn a complete neighborhood of a linear-degree vertex. Our goal in this paper is to understand what *can* be done efficiently for graphs with unbounded degrees. We give two types of answers. First, we point to problems that admit efficient solutions for any graph. In particular, we give efficient LCAs for weak coloring [29], where the goal is to color vertices so that each vertex has a neighbor of a different color. Weak coloring has played a key role in the study of distributed algorithms e.g., [29, 10], in part due to its applications to resource allocations in distributed settings [25]. Hence it is natural that we study it in the context of LCAs. Moreover, this study has turned out to be worthwhile, because the results were unexpected: in our model there is a separation between weak 2-coloring and weak 3-coloring that was not observed in other models.

As another way of dealing with large degrees, we propose a more powerful probe model, where probing a vertex returns a list of all its neighbors. We call such probes "strong," as opposed to the "weak" edge probes. Lower bounds in this model consider the number of connections the LCA needs to make in order to probe the graph, even if communication along these channels is unbounded, and are stronger than lower bounds in the weak probe model. Strong probes can be thought of as an intermediate model that lies between the weak probe model and the distributed $\mathcal{LOCAL}$ model, and helps clarify the sources of differences between these two extremes. Our main negative result in this model is that approximately $\Omega(\sqrt{n})$ strong probes are required to compute a maximal matching.

Our results include the following. Let *WP* and *SP* stand for weak and strong probes, respectively and let n denote the number of nodes in the underlying graph. We give tight upper and lower bounds for weak 3-coloring. Our algorithm is deterministic and uses weak probes, whereas our lower bound holds also for randomized LCAs that may use strong probes.

▶ **Theorem 1.** *There exists a deterministic WP LCA for weak* 3*-coloring that uses* $\log^* n + O(1)$ *probes.*

▶ **Theorem 2.** *Every (randomized or deterministic) SP LCA for weak* 3*-coloring of the cycle graph requires* $\Omega(\log^* n)$ *probes.*

We describe a deterministic WP LCA for weak 2-coloring, and give a matching lower bound for graphs with maximal degree $d = O\left(\frac{\log n}{\log\log n}\right)$.

▶ **Theorem 3.** *There exists a deterministic WP LCA for weak* 2*-coloring that uses* $\log^* n + 2d_v + O(1)$ *weak probes, where* d_v *is the degree of the queried vertex.*

▶ **Theorem 4.** *Any deterministic WP LCA for weak* 2*-coloring* d*-regular graphs with* $d = O\left(\frac{\log n}{\log\log n}\right)$ *requires at least* $d/2$ *probes.*

We design a randomized LCA for weak 2-coloring, whose probe complexity is independent of the maximal degree and show how it can be implemented in both the strong and weak probe models.

▶ **Theorem 5.** *There exists a randomized WP LCA for weak* 2*-coloring that uses* $\Theta\left(\frac{\log^2 n}{\log\log n}\right)$ *probes, and a randomized SP LCA for weak* 2*-coloring that uses* $\Theta\left(\frac{\log n}{\log\log n}\right)$ *probes.*

We give a lower bound for vertex cover in the strong probe model. Specifically, we show that for high degree graphs, many strong probes are necessary to approximate a minimal vertex cover to any interesting precision.

▶ **Theorem 6.** *For any* $\epsilon < \frac{1}{2}$*, any randomized SP LCA that computes a vertex cover whose size is a* $(\frac{1}{2}n^{1-2\epsilon})$*-approximation to the size of the minimal vertex cover requires at least* ϵn^{ϵ} *probes.*

A corollary of Theorem 6 is the following.

▶ **Corollary 7.** *Any SP LCA for maximal matching on arbitrary graphs requires* $n^{1/2-o(1)}$ *probes.*

We describe an LCA that finds a matching that is a $(1-\epsilon)$-approximation to the maximum matching, for regular graphs, using a constant number of probes.

▶ **Theorem 8.** *There exists an SP LCA that finds a* $(1-\epsilon)$*-approximate maximum matching in expectation on* d*-regular graphs that uses* $\epsilon^{-O(\epsilon^{-2})}$ *probes per query.*

Finally, we show that for graphs with sufficiently high girth and degree, polynomially (in ϵ^{-1}) many strong probes suffice.

▶ **Theorem 9.** *There exists an SP LCA that finds a* $(1-\epsilon)$*-approximate maximum matching in expectation on* d*-regular graphs of girth* g*, with* $d \geq \epsilon^{-1}$ *and* $g \geq \epsilon^{-3}$*, that uses* $O(\epsilon^{-7})$ *probes per query.*

1.1 Overview of Our Techniques

Our main result in this abridged version is the proof of Theorem 6. We construct a family of bipartite graphs in which a large subset of vertices have "almost" the same view at distance 2. Exactly one of these vertices, v_0, needs to be added to the vertex cover; however, there are many vertices for which a small number of strong probes does not suffice in order to

verify that they are not v_0, and hence they must add themselves to the vertex cover. In our construction we use vertex naming schemes based on low degree polynomials to ensure that certain vertices do not share many neighbors. This result gives a separation between the SP model (and hence also the WP model) and the $\mathcal{LOCAL}$ model, as it is possible to compute a maximal matching in the $\mathcal{LOCAL}$ model in $O(\log n)$ rounds w.h.p. [24].

In the full version of the paper [9] we discuss deterministic and randomized methods of sampling a "parent" for each vertex. No matter how the parents are chosen, the resulting graph is a disjoint set of directed subgraphs, each one containing exactly one cycle. The main technical content of here is the analysis of the diameter of these subgraphs, depending on the choice of parent selection scheme.

We then address the problem of weak 3-coloring. Our LCA (the proof of Theorem 1) is based on the following approach. Given an arbitrary graph, the directed subgraphs formed by the parent relation (as above) span all vertices, and each of their connected components has at least two vertices. Hence any weak coloring of every component separately induces a weak coloring of the whole graph. Each component has one cycle, but it turns out that the 3-coloring algorithm for rooted trees of Goldberg, Plotkin and Shannon [13] can be adapted in order to legally 3-color (and hence also weakly 3-color) such a component. When implemented as an LCA, the upper bound of $\log^* n + O(1)$ on the number of probes follows from a similar upper bound on the number of rounds of the (modified) algorithm of [13]. To prove a nearly matching lower bound (Theorem 2) we use a reduction to the lower bound of Naor [28] (extending [21]) for distributed algorithms that legally 6-color a cycle. Adapting lower bounds from the distributed setting to the LCA setting also involves an argument of Göös et al. [15].

We show how to augment the previous algorithm in order to reduce the number of colors to 2, thus proving Theorem 3. It takes only one more probe to transform the weak 3-coloring to a weak 2-coloring that is legal for all vertices except for those vertices that do not serve as parents (which we refer to as *leaves*). The final step involves changing the colors of (some) leaves. In order to determine whether a vertex v is a leaf, we probe all of its neighbors, making the probe complexity linear in the degree of the queried vertex.

One natural method of proving lower bounds for LCAs is by reduction to the distributed $\mathcal{LOCAL}$ model, as was done in [15] (and as we do in the proof of Theorem 2). The relationship between LCAs and distributed algorithms has been studied before (e.g., [6, 31, 32, 33]) – given a distributed algorithm to a problem that takes t rounds, one immediately obtains an LCA that uses $O\left(d^t\right)$ probes (where d is the maximal degree), by probing all nodes within distance t. The inverse reduction doesn't work, as an LCA may probe remote (disconnected) nodes. Consider, for example, the following artificial problem: each vertex has to color itself blue if the node with ID 1 has an odd number of neighbors, and red otherwise. An LCA for this problem needs a single probe, while a distributed algorithm requires time proportional to the graph's diameter. Göös et al. [15] show that for many natural problems, probing remote vertices does not help. But even if we consider only probes to neighbors of discovered vertices, the best lower bound we can hope for using such a reduction is the distributed *time* lower bound: a lower bound of t rounds in the distributed model implies a lower bound of t probes in the LCA model (recall that an upper bound of t rounds in the distributed model implies an upper bound of $O\left(d^t\right)$ probes!). This suggests that we may need new tools to obtain stronger lower bounds. In the proof of Theorem 4, we iteratively construct families of d-regular graphs, where graphs in family F_i (for $i \leq d/2$) contain roughly d^i vertices, and we show that weak 2-coloring of graphs in family F_i requires at least i deterministic weak probes. Every graph G in family F_i is composed of d disjoint copies of graphs $H_1, \ldots, H_d$ from family

F_{i-1} and two auxiliary vertices a_i and b_i. From each graph H_j a single edge is removed, and instead one of its endpoints is connected to a_i and the other to b_i. Intuitively, it is reasonable to expect that a_i cannot decide on a color before it knows the color of at least one of its neighbors. Likewise, it is reasonable to expect that each neighbor, being a member of a graph in F_{i-1}, requires (by an induction hypothesis) $i-1$ probes. The combination of these two non-rigorous claims would imply that a_i requires at least i probes (one to determine the name of one of its neighbors, and $i-1$ probes so as to determine the color of that neighbor). Turning this informal intuition into a rigorous proof is nontrivial, and this is the main content of our proof of Theorem 4.

We design a randomized LCA for weak 2-coloring. A key aspect in our randomized LCA is that each vertex first chooses a random temporary color. This induces a weak 2-coloring for most vertices of the graph: each vertex whose temporary color differs from the temporary color of a neighbor can keep its color. Extending this weak 2-coloring to the remaining vertices is done by associating a parent with each vertex, with the intended goal that the color of the vertex will differ from the color of the parent (determining the color of the parent uses an inductive process). This aspect has several different implementations, leading to different probe complexities. Namely, for an arbitrary parent choice, the number of strong probes is $\Theta(\log n)$. If we implement the arbitrary choice using weak probes, the probe complexity is $\Theta(\log^2 n)$. For a more clever randomized choice of parent, we get that the strong probe complexity is $\Theta\left(\frac{\log n}{\log\log n}\right)$, and the weak probe complexity is $\Theta\left(\frac{\log^2 n}{\log\log n}\right)$. All these bounds on probe complexity hold with high probability.

We note that our results do not prove a separation between the complexities of deterministic and randomized WP LCAs for weak 2-coloring (although we conjecture that there is one), as our lower bound that is linear in the degree is proved only for regular graphs of degree at most $O\left(\frac{\log n}{\log\log n}\right)$, and our upper bound for randomized WP LCAs for weak 2-coloring is $O\left(\frac{\log^2 n}{\log\log n}\right)$.

Randomized LCAs generally use a pseudo-random generator in order to limit the number of bits that they use, while ensuring consistency (e.g., [1, 20, 32]). In order to explore the theoretical limitations of the probe complexity of LCAs, we assume that our randomized LCAs have unbounded access to random bits. Nevertheless, we show that one can implement the randomized WP LCA for weak 2-coloring (the one using the arbitrary parent choice scheme) using a pseudo-random generator with a seed of length $O(\log n)$.

We give an LCA for approximate maximum matching in regular graphs. We first describe an LCA for $(1-\epsilon)$ matching on graphs of degree bounded by d, that uses at most $\left(d+\frac{1}{\epsilon}\right)^{O\left(\epsilon^{-2}\right)}$ probes per query. (Alternatively, if we wish to have a better dependency on ϵ and are willing to have a dependency on n, then an LCA of Even, Medina and Ron [6] has probe complexity $d^{O\left(\frac{1}{\epsilon}\right)}+O\left(\frac{1}{\epsilon^2}\right)\log^* n$.) Our LCA $\mathcal{A}$ is a simple variation on a randomized LCA of Yoshida, Yamamoto and Ito [40]: whereas [40] does not limit the number of probes used by their LCA but instead analyze and provide upper bounds on the expected number of probes used by their LCA (expectation taken both other choice of random edge and randomness of the LCA), we run essentially the same LCA, but with a strict upper bound on the number of probes. This upper bound is a factor of $\frac{d}{\epsilon}$ larger than the expectation. Markov's inequality implies that this gives a $(1-\epsilon)$-approximation to the maximum matching in expectation.

To obtain an LCA for a d-regular graph G, if $d<\frac{1}{\epsilon^2}$ we use LCA $\mathcal{A}$. If $d>\frac{1}{\epsilon^2}$, we sparsify the graph: for some universal constant c (independent of ϵ), each edge remains in the graph with probability $\frac{c}{d\epsilon}$, and then all vertices that still have degree higher than $\frac{2c}{\epsilon}$ are removed from the graph. This results in a graph G' of degree bounded by $\frac{2c}{\epsilon}$. Every matching in G'

is a matching in G. Moreover, we show that the expected size of a maximum matching in G' (expectation taken over choice of random sparsification) is at least $(1 - \frac{\epsilon}{2})$ times the size of the maximum matching in G. Hence it suffices to find a $(1 - \frac{\epsilon}{2})$-approximate maximum matching in the bounded degree graph G', and it will serve as a $(1-\epsilon)$-approximate maximum matching in G. A sufficiently large matching in G' can be found by $\mathcal{A}$, using $\left(\frac{1}{\epsilon}\right)^{O\left(\frac{1}{\epsilon^2}\right)}$ probes to G' per query. To implement $\mathcal{A}$ on G', but using only probes to G, we show that each strong probe to G' can be simulated by $O(1/\epsilon)$ strong probes to G.

We show that we can find a $(1 - \epsilon)$-approximation to the maximum matching for regular graphs with sufficiently high degree and girth using polynomially (in $1/\epsilon$) many probes. Gamarnik and Goldberg [11] show that the randomized greedy algorithm finds a $(1 - \epsilon)$ approximation to the maximum matching on regular graphs with sufficiently high degree and girth. Similarly to the previous result, if the vertex degrees are sufficiently small (say, below $\frac{1}{\epsilon^3}$), we can use an LCA of [40] (an implementation of the randomized greedy algorithm that uses in expectation $O(d)$ probes in graphs of maximum degree d), while placing a strict upper bound on the number of probes (this upper bound is a factor of $\frac{1}{\epsilon^{O(1)}}$ larger than the expectation). If the degrees are large, our approach is once again to sparsify the graph G prior to using the LCA of [40] (modified to have a strict upper bound on the number of probes). Unfortunately, the resulting graph G' is only nearly regular but not actually regular. This requires us to extend the result of [11] from regular graphs to nearly regular graphs. We do so without relying on the proofs of [11], by the following approach. We add imaginary edges to G', making it regular (while maintaining high girth). The LCA now runs on a regular graph, and hence the bounds of [11] apply. The new problem that arises is that the matching that is output by the LCA might contain imaginary edges. However, we prove that the expected fraction of imaginary edges in that matching is similar to their fraction within the input graph (our proof uses both the high girth assumption and the fact that the randomized greedy algorithm is local in nature). This, combined with the fact that the fraction of imaginary edges in the input graph was small (because G' is nearly regular), implies that the imaginary edges can be discarded from the solution without significantly affecting the approximation ratio.

1.2 Related Work

Measures

There are several criteria by which one can measure the performance of LCAs. Rubinfeld et al. [33] focus on the time complexity of LCAs: how long it takes to reply to a query; Alon et al. [1] emphasize the space complexity, in particular, the length of the random seed used (randomized LCAs need a global random seed to ensure consistency). Mansour, Patt-Shamir and Vardi [26] introduce a unified model, that takes into account all four complexity measures: *probe complexity*, *time complexity* (per query) and space complexity, divided into *enduring memory* (in all known LCAs, this includes only the random seed) and *transient space* (the computational space used per query). They show that it is possible to obtain LCAs for which all of these are independent of n for certain problems, such as a $(1 - \epsilon)$ approximation to a maximal acyclic subgraph, using $d^{O(1/\epsilon)}$ probes, where d is the maximal degree of the graph. LCAs that do not use any enduring memory are called *stateless* [6]. Indeed, the deterministic algorithms in this paper are stateless. Another property that is considered desirable in LCAs is *query-obliviousness* [33]: the property that the replies to different queries do not depend on the order in which the queries are given. Again, the LCAs of this paper are all query-oblivious.

LCAs and high-degree graphs

As mentioned above, most known results assume bounded degrees. For example, Mansour et al. [27] describe LCAs for maximal matching and other problems that use polylogarithmic (in the size of the graph; exponential in the degree) time and space when the degrees of the graph are bounded by a constant. Even, Medina and Ron [6], focusing on probe complexity, give deterministic LCAs for MIS, maximal matching and $(d+1)$-coloring for graphs of degree bounded by a constant d, which use $d^{O(d^2)}\log^* n$ probes. Fraigniaud, Heinrich and Kosowski [10] investigate local conflict coloring, a general distributed labeling problem, and use their results to improve the probe complexity of $(d+1)$-vertex coloring to approximately $d^{O(\sqrt{d})}\log^* n$ probes.

Some papers allow for slightly super-constant degrees: Levi, Rubinfeld and Yodpinyanee [20] give LCAs for MIS and maximal matching for graphs of degree $2^{O(\sqrt{\log\log n})}$, using an improvement of Ghaffari [12]. Reingold and Vardi [32] give LCAs for MIS, maximal matching and other problems that apply to graphs that are sampled from some distribution. This limitation allows them to address graphs with higher maximal degree, as long as the average degree is $O(\log\log n)$, and the tail of the distribution is sufficiently light. If we restrict ourselves to LCAs that use polylogarithmic time and space, the approximate maximum matching LCA of [20] accommodates graphs of polylogarithmic degree. Levi, Ron and Rubinfeld [19] describe an LCA that constructs spanners using a number of probes polynomial in d.

Lower bounds

There are few explicit impossibility results LCAs. Göös et al. [15] show that any LCA for MIS requires $\Omega(\log^* n)$ probes, by showing that probing vertices that have not yet been discovered is not useful. This implies that, on a ring, the number of probes that an LCA needs to make is "roughly the same" as the number of rounds required by a distributed LOCAL algorithm, implying that the lower bounds of Linial [21] and Naor [28] hold for LCAs as well. Levi, Ron and Rubinfeld [19] show that an LCA that determines whether an edge belongs to a sparse spanning subgraph requires $\Omega(\sqrt{n})$ probes. Feige, Mansour and Schapire [8], adapting a lower bound from the property testing literature [14], show that approximating the minimum vertex cover in bounded degree bipartite graphs within a ratio of $1+\epsilon$ (for some explicit fixed $\epsilon > 0$) cannot be done with $o(\sqrt{n})$ probes.

Weak coloring

Weak coloring was introduced by Naor and Stockmeyer [29]. They give a LOCAL distributed algorithm that requires $\log^* d + O(1)$ rounds for weak 2-coloring graphs of maximal degree d, assuming all degrees are odd. In contrast, they show that there is no constant time LOCAL algorithm for weak c-coloring all graphs with vertices with even degrees, for constant c. Our deterministic weak 2-coloring LCA (Theorem 3) uses $\log^* n + O(d_v)$ probes, but when cast within the LOCAL model it takes $\log^* n + O(1)$ rounds (independently of d_v).

Additional LCA background

LCAs are not restricted to graphs. Well known examples include *locally-decodable codes* (LDCs) (e.g., [18, 39]) and local reconstruction (e.g., [17, 34]). LDCs are error-correcting codes that allow a single bit of the original message to be decoded with high probability by querying a small number of bits of a codeword. Local reconstruction involves recovering the

value of a function f for a particular input x given oracle access to a closely related function g. LCAs have recently been applied to solving convex problems in a distributed fashion [22]. Traditional methods for solving distributed optimization, such as iterative descent methods (e.g., [23]) and consensus methods (e.g., [4]), require global communication, and any edge failure or lag in the system immediately affects the entire solution, by delaying computation or causing it not to be computed at all; furthermore, if the network changes in a small way, the entire solution needs to be recomputed. If an LCA is used, most of the system remains unaffected by local changes or failures. Hence LCAs can be used to make systems more robust to edge failures, lag, and dynamic changes. LCAs have also been used in the context of mechanism design [16], machine learning [8], and designing distributed algorithms [7]. There are other situations when LCAs may be useful - say we wish to perform some computation on each of the vertices of an MIS of some huge graph. LCAs allow us to be able to begin work on some vertices before the entire MIS is computed, and guarantee that the local replies to the queries will be consistent with the same global solution, that will be available at some point in the future.

LCAs can also be used as subroutines in approximation algorithms e.g., [5, 31, 30, 40]. The goal of such algorithms is to output an approximation to the size of the solution to some combinatorial problem (such as Vertex Cover, Maximum Matching, Minimum Spanning Tree), in time sublinear in the input size. In particular, if one has an LCA whose running time is t that solves some problem, one can obtain an approximation to the size of the solution (with some constant probability) by executing this LCA on a sufficiently large (but constant) number of vertices k chosen uniformly at random, to obtain an approximation algorithm whose running time is kt (see e.g., [30, 36] for more details).

The concept of LCAs is related to but should not be confused with local algorithms as in [2, 3, 35]. The difference is that these local algorithms do not require the output for different probes to satisfy a global consistency property, but rather to satisfy some local criteria. For example, the goal might be for each vertex to output a small dense subgraph that contains it, without requiring two different vertices to agree on whether they share the same dense subgraph or not.

1.3 Paper Outline

Due to space restriction, we include a single result with complete proof in this version. See [9] for the full version of this paper.

2 Preliminaries

We denote the set of integers $\{1, 2, \ldots, n\}$ by $[n]$. All logarithms are base 2. Our input is a simple undirected graph $G = (V, E)$, $|V| = n$, in which every vertex has an ID and all IDs are distinct. For simplicity, we assume that the IDs are taken from the set $[n]$. The neighborhood of a vertex v, denoted $N(v)$, is the set of vertices that share an edge with v: $N(v) = \{u : (u, v) \in E\}$. The *degree* of a vertex v, is $|N(v)|$. The *girth* of G, denoted girth(G) is the length of the shortest cycle in G.

LCAs

Our definition of LCAs is slightly different from previous definitions in that it focuses on probe complexity. We do this so as not to introduce unnecessary notation. See [26, 36] for definitions that also take into account other complexity measures.

▶ **Definition 10** (Probe)**.** We assume that the input graph is represented as a two dimensional n by $d+1$ array, where d is the maximum degree. Rows are labeled from 1 to n by the vertex names. For any v, the cell $(v,0)$ specifies the degree d_v of v, the cell (v,j) for $1 \leq j \leq d_v$ specifies the name of the neighbor connected to v's jth port. Cells (v,j) for $d_v < j \leq d$ contain 0. We define strong and weak probes as follows.

- A *strong probe* (SP) specifies the ID of a vertex v; the reply to the probe is the entire row corresponding to v (namely, the list of all neighbors of v).
- A *weak probe* (WP) specifies a single cell (v,j) and receives its content (namely, only the j^{th} neighbor of v).

We note that knowing that u is v's i^{th} neighbor does not give us information regarding which one of u's ports v is connected to. This property is crucial for the proof of Theorem 4.

▶ **Definition 11** (Local computation algorithm)**.** A deterministic $p(n)$*-probe local computation algorithm* $\mathcal{A}$ for a computational problem is an algorithm that receives an input of size n. Given a query x, $\mathcal{A}$ makes at most $p(n)$ probes to the input in order to reply. $\mathcal{A}$ must be *consistent*; that is, the algorithm's replies to all of the possible queries combine to a single feasible solution to the problem.

A randomized $(p(n), s(n), \delta(n))$*-local computation algorithm* $\mathcal{A}$ differs from a deterministic one in the following aspects. Before receiving its input, it is allowed to write $s(n)$ random bits (referred to as the random *seed*) to memory.[4] Thereafter, it must behave like a deterministic LCA, except that when answering queries it may also read and use the random seed. For any input G, $|G| = n$, the probability (over the choice of random seed) that there exists a query in G for which $\mathcal{A}$ uses more than $p(n)$ probes is at most $\delta(n)$, which is called $\mathcal{A}$'s failure probability.

When LCA $\mathcal{A}$ is given input graph $G = (V, E)$ and queried on vertex $v \in V$, we denote this by $\mathcal{A}(G, v)$. An LCA $\mathcal{A}$ is said to *require* k probes on a graph $G = (V, E)$, if there is at least one query for which $\mathcal{A}$ uses k probes. We say that an LCA $\mathcal{A}$ requires k probes for a family F of graphs, if for some graph $G \in F$, $\mathcal{A}$ requires k probes.

▶ **Definition 12** (Approximation algorithm)**.** Given a maximization problem over graphs and a real number $0 \leq \alpha \leq 1$, a (possibly randomized) α-approximation algorithm $\mathcal{A}$ is guaranteed, for any input graph G, to output a feasible solution whose expected value is at least an α fraction of the value of an optimal solution (in expectation over the random bits used by $\mathcal{A}$).[5]

3 Lower Bound for Vertex Cover

In this section we prove Theorem 6:

▶ **Theorem 6.** *For any $\epsilon < \frac{1}{2}$, any randomized SP LCA that computes a vertex cover whose size is a $(\frac{1}{2}n^{1-2\epsilon})$-approximation to the size of the minimal vertex cover requires at least ϵn^{ϵ} probes.*

In order to prove Theorem 6 , we use the minimax theorem of Yao [38], by showing a lower bound on the expected number of probes required by a deterministic LCA when the input is selected from a certain distribution. To this end we construct, for infinitely many values of n, a family of graphs, parametrized by a constant $k \geq 3$.

[4] In this work, except where explicitly mentioned, we allow the random seed to be unbounded.
[5] The definition of approximation algorithms to minimization problems is analogous, with $\alpha \geq 1$.

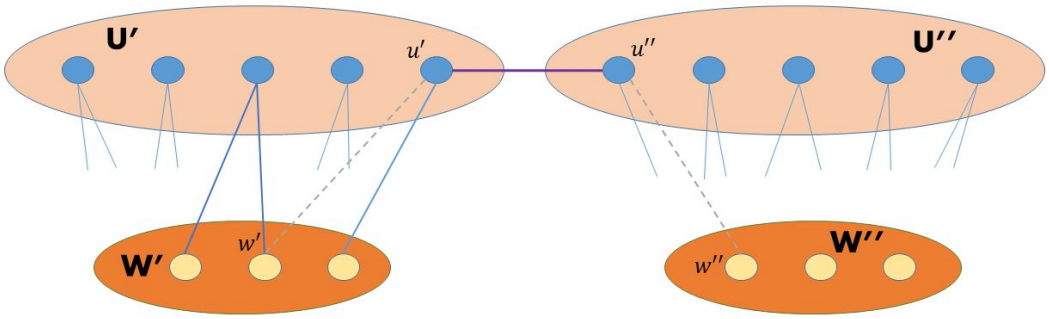

Figure 1 The graph fusion$(\mathbb{G}, (u', w'), (u'', w''))$. The dashed edges $e' = (u', w')$ and $e'' = (u'', v'')$ have been removed and the edge (u', u'') has been added.

Let p be a prime number, and $\mathbb{Z}(p)$ its associated field. Let $G^* = (U^* \cup W^*, E)$ be a bipartite graph, where $|U^*| = p^k$, $|W^*| = p^2$. Label each vertex in U^* by a k-tuple $(a_0, a_1, \ldots, a_{k-1})$, $a_i \in \mathbb{Z}(p), i \in \{0, 1, \ldots, k-1\}$ and the vertices in W^* by a pair (b_0, b_1), $b_i \in \mathbb{Z}(p), i \in \{0, 1\}$. Associate each vertex $u_j = (a_0, a_1, \ldots, a_{k-1}) \in U^*$ with the polynomial $f_j(x) = a_0 + a_1 x + \cdots + a_{k-1}x^{k-1}$. Connect $(b_0, b_1) \in W^*$ to u_j iff $b_1 = f_j(b_0)$.

▶ **Lemma 13.** *For every two different vertices $u_i, u_j \in U^*$, $N(u_i) \cap N(u_j) \leq k - 1$.*

Proof. Consider f_i and f_j, the polynomials associated with u_i and u_j respectively. Let $g(x) = f_i(x) - f_j(x)$. As g is not the zero polynomial, it has at most $k-1$ roots in $\mathbb{Z}(p)$. ◀

Let $\mathbb{G}$ to be a graph consisting of two identical copies of G^*. We denote these two copies by $G' = (U' \cup W', E')$ and $G'' = (U'' \cup W'', E'')$. Let $U = U' \cup U''$; let $W = W'' \cup W''$; let $n = |U \cup W| = 2(p^k + p^2)$.

We define the following operation on $\mathbb{G}$ (see Figure 1). Let e' and e'' be edges such that $e' = (u', w') : u' \in U', w' \in W'$ and $e'' = (u'', w'') : u'' \in U'', w'' \in W''$. Remove e' and e'' from $\mathbb{G}$, and add an edge $e = (u', u'')$. We call this operation fusion$(\mathbb{G}, e', e'')$, and call u and u' the *fusion vertices*. Note that there are p^{k+1} possible choices for e' and p^{k+1} possible choices for e''.

Given a graph $G =$ fusion$(\mathbb{G}, (u', w'), (u'', w''))$, the optimal size of the vertex cover of G is at most $2p^2 + 1$, as W and a fusion vertex constitute a vertex cover.

Note that a vertex can locally detect whether it is in U or in W just by looking at its own degree. However, to detect whether it is one of the fusion vertices, it needs to determine the degrees of its neighbors, which is impossible to do with number of probes significantly smaller than its degree.[6]

We now describe the graph family we use. Let $\mathbb{G} = (U \cup W, E)$ be as above. Let Π be the set of all possible namings of $U \cup W$ by the ID set $[n]$. Let $T = E' \times E''$. Given a naming $\pi \in \Pi$ and a pair of edges $\tau = (e', e'') \in T$, the graph $\mathbb{G}(\tau, \pi)$ is defined as follows:

1. The topology of $\mathbb{G}(\tau, \pi)$ is given by fusion$(\mathbb{G}, e', e'')$.
2. The vertices of $\mathbb{G}(\tau, \pi)$ are named according to π.

The family of graphs we consider is $\mathcal{G}(\Pi, T) = \{\mathbb{G}(\tau, \pi) \mid \tau \in T, \pi \in \Pi\}$. We now analyze the behavior of a given deterministic LCA A with proble complexity less than p/k on a

[6] Compare this with a distributed algorithm in the LOCAL model, for which one round suffices to determine this.

graph chosen uniformly at random from $\mathcal{G}(\Pi, T)$. We first make the following simplification. Suppose that A running on some $\mathbb{G}(\tau, \pi)$ is given v as a query. If A probes w'_τ or w''_τ, it knows that v is neither u'_τ nor u''_τ and hence A need not include v in the VC. For simplicity, we also assume that if A probes u'_τ or u''_τ, it knows not to add v to the VC. Note that this only strengthens A, hence we can make this assumption without loss of generality.

We use the following definition.

▶ **Definition 14** (View)**.** Let $\mathcal{A}$ be a deterministic SP LCA. We denote by $\textsc{View}(\mathcal{A}, G, v)$ the subgraph that $\mathcal{A}$ discovers when queried on vertex v in graph G, i.e., the set of all probed vertices and their neighbors.

▶ **Lemma 15.** *Let $\mathcal{A}$ be an SP LCA with probe complexity less than p/k. Let $G \in \mathcal{G}(\Pi, T)$. Assume that A is queried on G with vertex v. Then there is some vertex $w \in N(v)$ such that w has no neighbors except v in $\textsc{View}(\mathcal{A}, G, v)$. (That is, neither w nor any of its neighbors (except v) is probed in $\mathcal{A}(G, v)$.)*

Proof. $\mathcal{A}$ probes v and, say, a vertices from U and b vertices from W, for some $a, b \in \mathbb{N}$ such that $a + b < p/k$. From Lemma 13, A sees at most $a(k-1)$ vertices from $N(v)$ as a result of probing vertices in $U \setminus \{v\}$. Furthermore, A sees at most b vertices from $N(v)$ as a result of probing vertices in W. As $a(k-1) + b < p = |N(v)|$, at least one vertex in $N(v)$ is only seen once, while probing v. ◀

Consider an input graph $\mathbb{G}(\tau, \pi)$, where $\tau = ((u'_\tau, w'_\tau), (u''_\tau, w''_\tau))$. We use the following notation.

- A_v denotes the event that $\mathcal{A}$ is given v as a query. Note that A_v is independent of π and τ.
- $X_{\tau,\pi,i}$ denotes the event that none of $u'_\tau, w'_\tau, u''_\tau, w''_\tau$ is probed when A is queried on $i \in [n]$.

▶ **Lemma 16.** *Fix π and τ. Let v be a vertex in $U' \setminus \{u'_\tau\}$. If A_v and $X_{\tau,\pi,\pi(v)}$ hold, there exist $\pi_1 \in \Pi, \tau_1 \in T$ such that $\textsc{View}(\mathcal{A}, \mathbb{G}(\pi_1, \tau_1), u'_{\tau_1}) = \textsc{View}(\mathcal{A}, \mathbb{G}(\pi, \tau), v)$.*

Proof. By Lemma 15, there is some vertex $w \in N(v)$ that has no neighbors other than v in $\textsc{View}(\mathcal{A}, G(\pi, \tau), v)$. Let π_1 be identical to π except that v and u'_τ are interchanged. Set $\tau_1 = ((v, w), e''_\tau)$. The lemma follows. ◀

A symmetrical argument holds for $v \in U'' \setminus \{u''_\tau\}$.

▶ **Lemma 17.** *Fix $\mathbb{G}$, and let $i \in [n]$ be the ID of the vertex given to a deterministic SP LCA $\mathcal{A}$ as a query. If the probe complexity of $\mathcal{A}$ is less than p/k, then $\Pr[X_{\tau,\pi,i}] \geq 1 - \frac{1}{kp}$, where the probability is over the choice of π and τ.*

Proof. Fix π. If A probes a vertices in U and b vertices in W, it will hit one of $u'_\tau, w'_\tau, u''_\tau, w''_\tau$ with probability at most $\frac{a}{p^k} + \frac{b}{p^2}$, over the choice of τ. Since $a, b \geq 0$ and $a + b \leq p/k$, the probability is maximized for $a = 0, b = p/k$. As this bound holds for any π, the result follows. ◀

Lemma 16 and Lemma 17 imply that when a deterministic SP LCA is queried on a vertex $u \in U$ from a random graph in $\mathcal{G}(T, \Pi)$, it cannot discern in less than p/k probes whether u is a fusion vertex with probability greater than $\frac{1}{kp}$. Hence the LCA must add vertex u to the VC, because at least one fusion vertex *must* be in the VC. Therefore, the size of the VC that A computes is at least $p^k - O(1)$, whereas the optimal VC has size at most $p^2 + 1$:

▶ **Theorem 18.** *There does not exist a deterministic SP LCA A that computes a VC that is an $(\frac{1}{2}n^{1-2\epsilon})$-approximation to the optimal VC and uses fewer than ϵn^{ϵ} probes with probability greater than $\frac{1}{kp}$ on a graph chosen uniformly at random from $\mathcal{G}(\Pi, T)$.*

Proof. Let $\epsilon = 1/k$. Recall that $n = \Theta(p^k)$. We have shown that if the number of probes is less than $p/k = \Theta(n^{\epsilon} \cdot \epsilon)$, then the approximation ratio is at least $p^k/p^2 - o(1) = n^{1-2/k} - o(1)$. ◀

Applying Yao's principle [38] to Theorem 18 completes the proof of Theorem 6.

▶ **Corollary 19.** *Any SP LCA for maximal matching on arbitrary graphs requires $\Omega(n^{1/2-o(1)})$ probes.*

Proof. It is well known that, given any maximal matching, taking both end vertices of every edge gives a 2-approximation to the VC (e.g., [37]). Therefore, an LCA for maximal matching would immediately give a 2-approximation to the minimal vertex cover. Setting $2 = \Theta(n^{1-2\epsilon})$ in Theorem 6 gives $\epsilon = 1/2$. The result follows. ◀

References

1. Noga Alon, Ronitt Rubinfeld, Shai Vardi, and Ning Xie. Space-efficient local computation algorithms. In *Proc. 22nd ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1132–1139, 2012.
2. Reid Andersen. A local algorithm for finding dense subgraphs. *ACM Trans. Algorithms*, 6(4), 2010.
3. Reid Andersen, Shayan Oveis Gharan, Yuval Peres, and Luca Trevisan. Almost optimal local graph clustering using evolving sets. *J. ACM*, 63(2):15, 2016.
4. Vincent D Blondel, Julien M Hendrickx, Alex Olshevsky, and John N Tsitsiklis. Convergence in multiagent coordination, consensus, and flocking. In *Proceedings of IEEE Conference on Decision and Control*, pages 2996–3000. IEEE, 2005.
5. Bernard Chazelle, Ronitt Rubinfeld, and Luca Trevisan. Approximating the minimum spanning tree weight in sublinear time. *SIAM J. Comput.*, 34(6):1370–1379, 2005.
6. Guy Even, Moti Medina, and Dana Ron. Best of two local models: Local centralized and local distributed algorithms. *CoRR*, abs/1402.3796, 2014. URL: `http://arxiv.org/abs/1402.3796`, `arXiv:1402.3796`.
7. Guy Even, Moti Medina, and Dana Ron. Distributed maximum matching in bounded degree graphs. In *Proceedings of the 2015 International Conference on Distributed Computing and Networking, ICDCN*, pages 18:1–18:10, 2015.
8. Uriel Feige, Yishay Mansour, and Robert E. Schapire. Learning and inference in the presence of corrupted inputs. In *Proceedings of The 28th Conference on Learning Theory, COLT*, pages 637–657, 2015.
9. Uriel Feige, Boaz Patt-Shamir, and Shai Vardi. On the probe complexity of local computation algorithms. *CoRR*, abs/1703.07734, 2017. `arXiv:1703.07734`.
10. Pierre Fraigniaud, Marc Heinrich, and Adrian Kosowski. Local conflict coloring. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS*, pages 625–634, 2016.
11. David Gamarnik and David A. Goldberg. Randomized greedy algorithms for independent sets and matchings in regular graphs: Exact results and finite girth corrections. *Combinatorics, Probability and Computing*, 19:61–85, 1 2010. `doi:10.1017/S0963548309990186`.
12. Mohsen Ghaffari. An improved distributed algorithm for maximal independent set. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016*, pages 270–277, 2016.
13. Andrew V. Goldberg, Serge A. Plotkin, and Gregory E. Shannon. Parallel symmetry-breaking in sparse graphs. *SIAM J. Discret. Math.*, 1(4):434–446, 1988.

14 Oded Goldreich and Dana Ron. Property testing in bounded degree graphs. *Algorithmica*, 32(2):302–343, 2002.

15 Mika Göös, Juho Hirvonen, Reut Levi, Moti Medina, and Jukka Suomela. Non-local probes do not help with many graph problems. *Distributed Computing - 30th International Symposium, DISC*, pages 201–214, 2016.

16 Avinatan Hassidim, Yishay Mansour, and Shai Vardi. Local computation mechanism design. *ACM Trans. Economics and Comput.*, 4(4):21:1–21:24, 2016. doi:10.1145/2956584.

17 M. Jha and S. Raskhodnikova. Testing and reconstruction of Lipschitz functions with applications to data privacy. In *Proc. 52nd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2011.

18 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 (STOC)*, pages 80–86, 2000.

19 Reut Levi, Dana Ron, and Ronitt Rubinfeld. Local algorithms for sparse spanning graphs. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*, pages 826–842, 2014.

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

21 Nathan Linial. Locality in distributed graph algorithms. *SIAM J. Comput.*, 21(1), 1992.

22 Palma London, Niangjun Chen, Shai Vardi, and Adam Wierman. Distributed optimization via local computation algorithms. http://users.cms.caltech.edu/~plondon/loco.pdf, 2017.

23 Steven H Low, Fernando Paganini, and John C Doyle. Internet congestion control. *IEEE control systems*, 22(1):28–43, 2002.

24 Michael Luby. A simple parallel algorithm for the maximal independent set problem. *SIAM J. Comput.*, 15(4):1036–1053, 1986.

25 Nancy A. Lynch. Upper bounds for static resource allocation in a distributed system. *J. Comput. Syst. Sci.*, 23(2):254–278, 1981.

26 Yishay Mansour, Boaz Patt-Shamir, and Shai Vardi. Constant-time local computation algorithms. *Theory of Computing Systems*, pages 1–19, 2017.

27 Yishay Mansour, Aviad Rubinstein, Shai Vardi, and Ning Xie. Converting online algorithms to local computation algorithms. In *Proc. 39th International Colloquium on Automata, Languages and Programming (ICALP)*, pages 653–664, 2012.

28 Moni Naor. A lower bound on probabilistic algorithms for distributive ring coloring. *SIAM J. Discrete Math.*, 4(3):409–412, 1991.

29 Moni Naor and Larry J. Stockmeyer. What can be computed locally? *SIAM J. Comput.*, 24(6):1259–1277, 1995.

30 Huy N. Nguyen and Krzystof Onak. Constant-time approximation algorithms via local improvements. In *Proc. 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 327–336, 2008.

31 M. Parnas and D. Ron. Approximating the minimum vertex cover in sublinear time and a connection to distributed algorithms. *Theoretical Computer Science*, 381(1–3), 2007.

32 Omer Reingold and Shai Vardi. New techniques and tighter bounds for local computation algorithms. *J. Comput. Syst. Sci.*, 82(7):1180–1200, 2016.

33 Ronitt Rubinfeld, Gil Tamir, Shai Vardi, and Ning Xie. Fast local computation algorithms. In *Proc. 2nd Symposium on Innovations in Computer Science (ICS)*, pages 223–238, 2011.

34 Michael E. Saks and C. Seshadhri. Local monotonicity reconstruction. *SIAM J. Comput.*, 39(7):2897–2926, 2010.

35 Daniel A. Spielman and Shang-Hua Teng. A local clustering algorithm for massive graphs and its application to nearly linear time graph partitioning. *SIAM J. Comput.*, 42(1):1–26, 2013.

36 Shai Vardi. *Designing Local Computation Algorithms and Mechanisms*. PhD thesis, Tel Aviv University, Tel Aviv, Israel, 2015.

37 Vijay V. Vazirani. *Approximation Algorithms*. Springer, 2001.

38 Andrew Chi-Chin Yao. Probabilistic computations: Toward a unified measure of complexity. In *Proceedings of the 18th Annual Symposium on Foundations of Computer Science*, FOCS '77, pages 222–227, 1977.

39 Sergey Yekhanin. Locally decodable codes. *Foundations and Trends in Theoretical Computer Science*, 6(3):139–255, 2012.

40 Yuichi Yoshida, Masaki Yamamoto, and Hiro Ito. Improved constant-time approximation algorithms for maximum matchings and other optimization problems. *SIAM J. Comput.*, 41(4):1074–1093, 2012.