Improved Distributed Approximations for Maximum Independent Set

Ken-ichi Kawarabayashi
National Institute of Informatics, Tokyo, Japan
k_keniti@nii.ac.jp

Seri Khoury
University of California, Berkeley, CA, USA
seri_khoury@berkeley.edu

Aaron Schild
University of Washington, Seattle, WA, USA
aschild@uw.edu

Gregory Schwartzman
JAIST, Ishikawa, Japan
gregory.schwartzman@gmail.com

Abstract

We present improved results for approximating maximum-weight independent set (MaxIS) in the CONGEST and LOCAL models of distributed computing. Given an input graph, let n and Δ be the number of nodes and maximum degree, respectively, and let $\mathrm{MIS}(n, \Delta)$ be the running time of finding a *maximal* independent set (MIS) in the CONGEST model. Bar-Yehuda et al. [PODC 2017] showed that there is an algorithm in the CONGEST model that finds a Δ-approximation for MaxIS in $O(\mathrm{MIS}(n, \Delta) \log W)$ rounds, where W is the maximum weight of a node in the graph, which can be as large as $\mathrm{poly}(n)$. Whether their algorithm is deterministic or randomized that succeeds with high probability depends on the MIS algorithm that is used as a black-box. Our results:

1. A deterministic $O(\mathrm{MIS}(n, \Delta)/\epsilon)$-round algorithm that finds a $(1 + \epsilon)\Delta$-approximation for MaxIS in the CONGEST model.
2. A randomized $(\mathrm{poly}(\log \log n)/\epsilon)$-round algorithm that finds, with high probability, a $(1 + \epsilon)\Delta$-approximation for MaxIS in the CONGEST model. That is, by sacrificing only a tiny fraction of the approximation guarantee, we achieve an *exponential* speed-up in the running time over the previous best known result.
3. A randomized $O(\log n \cdot \mathrm{poly}(\log \log n)/\epsilon)$-round algorithm that finds, with high probability, a $8(1 + \epsilon)\alpha$-approximation for MaxIS in the CONGEST model, where α is the arboricity of the graph. For graphs of arboricity $\alpha < \Delta/(8(1 + \epsilon))$, this result improves upon the previous best known result in both the approximation factor and the running time.

One may wonder whether it is possible to approximate MaxIS with high probability in fewer than $\mathrm{poly}(\log \log n)$ rounds. Interestingly, a folklore randomized ranking algorithm by Boppana implies a single round algorithm that gives an expected Δ-approximation in the CONGEST model. However, it is unclear how to convert this algorithm to one that succeeds with high probability without sacrificing a large number of rounds. For unweighted graphs of maximum degree $\Delta \leq n/\log n$, we show a new analysis of the randomized ranking algorithm, which we combine with the local-ratio technique, to provide a $O(1/\epsilon)$-round algorithm in the CONGEST model that, with high probability, finds an independent set of size at least $\frac{n}{(1+\epsilon)(\Delta+1)}$. This result cannot be extended to very high degree graphs, as we show a lower bound of $\Omega(\log^* n)$ rounds for any randomized algorithm that with probability at least $1 - 1/\log n$ finds an independent set of size $\Omega(n/\Delta)$. This lower bound holds even for the LOCAL model. The hard instances that we use to prove our lower bound are graphs of maximum degree $\Delta = \Omega(n/\log^* n)$.

2012 ACM Subject Classification Mathematics of computing → Approximation algorithms; Theory of computation → Distributed computing models

Keywords and phrases Distributed graph algorithms, Approximation algorithms, Lower bounds

34th International Symposium on Distributed Computing (DISC 2020).
Editor: Hagit Attiya; Article No. 35; pp. 35:1–35:16
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Digital Object Identifier 10.4230/LIPIcs.DISC.2020.35

Related Version A full version of the paper is available at https://arxiv.org/abs/1906.11524.

Funding *Ken-ichi Kawarabayashi*: Supported by JST ERATO Kawarabayashi Large Graph Project JPMJER1201 and by JSPS Kakenhi JP18H05291.
Gregory Schwartzman: Supported by JSPS Kakenhi Grant Number JP19K20216 and JP18H052.

1 Introduction and Related Work

In this work we study the problem of approximating Maximum Independent Set (MaxIS) in distributed models. An independent set in a network is a subset of the nodes such that no two nodes in the subset are adjacent. In unweighted graphs, a *maximum* independent set is an independent set of maximum size. In weighted graphs, a *maximum-weight* independent set (MaxIS) is an independent set of maximum total weight, where by total we mean the sum of weights of nodes in the independent set.

The major two models of distributed graph algorithms are the LOCAL and CONGEST models. In the LOCAL model [19], there is a synchronized communication network of n computationally-unbounded nodes, where each node has a unique $O(\log n)$-bit identifier. In each communication round, each node can send an unbounded-size message to each of its neighbors. The task of the nodes is to compute some function of the network (e.g., its diameter, the value of a maximum independent set, etc.), while minimizing the number of communication rounds. The CONGEST model [21] is similar to the LOCAL model, where the only difference is that the message-size is bounded by $O(\log n)$ bits.

The problem of approximating MaxIS has been studied in both the LOCAL and CONGEST models [2, 7, 9, 11, 15, 16, 18]. In unweighted graphs, one can find a Δ-approximation for MaxIS by finding a maximal independent set (MIS). In recent years, our understanding of the complexity of MIS has been substantially improving [6, 12, 13, 23], leading to a recent remarkable breakthrough by Rozhon and Ghaffari [23], where they show a deterministic $\text{poly}(\log n)$-round algorithm for finding an MIS, even in the CONGEST model. This result also implies a randomized algorithm that finds an MIS with high probability in $O(\log \Delta) + \text{poly}(\log \log n)$ rounds, in the CONGEST model[1] [10, 13, 14, 23].

In a weighted graph, an MIS doesn't necessarily constitute a Δ-approximation for MaxIS. For the weighted case, Bar-Yehuda et al. [4] showed a Δ-approximation algorithm in the CONGEST model that takes $O(\text{MIS}(n, \Delta) \cdot \log W)$ rounds, where $\text{MIS}(n, \Delta)$ is the running time for finding an MIS in graphs with n nodes and maximum degree Δ, and W is the maximum weight of a node in the graph (which can be as high as $\text{poly}(n)$). Whether their algorithm is deterministic or randomized that succeeds with high probability depends on the MIS algorithm that is used as a black-box.

In this work we present faster algorithms compared to [4], by paying only a $(1 + \epsilon)$ multiplicative overhead in the approximation factor. Our main result (Theorem 2) is a randomized algorithm that achieves an exponential speed-up compared to [4]. One of the ingredients to prove our main result is an improved algorithm for the deterministic case (Theorem 1). Our results:

[1] We say that an algorithm succeeds with high probability if it succeeds with probability $1 - 1/n^c$ for an arbitrary constant $c > 1$.

▶ **Theorem 1.** *There is an* $O(\text{MIS}(n, \Delta)/\epsilon)$*-round algorithm in the CONGEST model that finds a* $(1+\epsilon)\Delta$*-approximation for maximum-weight independent set. Whether the algorithm is deterministic or randomized, depends on the* MIS *algorithm that is run as a black-box.*

▶ **Theorem 2.** *There is a randomized* $(\text{poly}(\log\log n)/\epsilon)$*-round algorithm in the CONGEST model that finds, with high probability, a* $(1+\epsilon)\Delta$*-approximation for maximum-weight independent set.*

Using the algorithm from Theorem 2, we can also get an improved approximation algorithm for a wide range of arboricity. Let α be the arboricity of the input graph. For graphs of arboricity $\alpha \leq \Delta/(8(1+\epsilon))$, Theorem 3 improves upon [4] in both the running time and approximation factor.

▶ **Theorem 3.** *There is a randomized* $O(\log n \cdot \text{poly}\log\log n/\epsilon)$*-round algorithm in the CONGEST model that finds, with high probability, an* $8(1+\epsilon)\alpha$*-approximation for maximum-weight independent set.*

A discussion on the folklore randomized ranking algorithm. A folklore randomized ranking algorithm by Boppana yields a single round algorithm in the CONGEST model that returns a solution with an *expected* approximation factor $\Delta + 1$.[2] In this ranking algorithm, each node v picks a number r_v uniformly at random in $[0, 1]$. If $r_v > r_u$ for any neighbor u of v, then v joins the independent set. Since every node joins the independent set with probability at least $1/(\Delta + 1)$, the expected weight of the independent set is at least $w(V)/(\Delta + 1)$, where $w(V)$ is the total weight of nodes in the graph. Recently, Boppana et al [8] showed that the ranking algorithm even returns a $(\Delta + 1)/2$-approximation in expectation. However, algorithms that work well in expectation don't necessarily work well with good probability. In fact, for the folklore randomized ranking algorithm, it is not very hard to construct examples in which the *variance* of the solution is very high, in which case the algorithm doesn't return the expected value with high probability. In this work we prove the following stronger hardness result (Theorem 4). Our lower bound in Theorem 4 holds only under the assumption that the nodes don't know the exact value of n, but only a polynomial upper bound on it. We emphasize this because some algorithms in the LOCAL and CONGEST models assume the knowledge of n (our algorithms in this work don't need this assumption).

▶ **Theorem 4.** *If the nodes don't know the exact value of* n*, but only a polynomial upper bound on it, then any algorithm that finds an independent set of size* $\Omega(n/\Delta)$ *in unweighted graphs, with success probability* $p \geq 1 - 1/\log n$ *must spend* $\Omega(\log^* n)$ *rounds, even in the LOCAL model.*

Interestingly, this hardness result applies for graphs of maximum degree $\Delta = \Omega(n/\log^* n)$. One may wonder whether we can extend the lower bound for much smaller maximum degree graphs. We rule out this possibility, with the following theorem. The proof of Theorem 5 relies on a novel idea for analyzing the classical ranking algorithm using *martingales*, and the *local-ratio* technique, on which we elaborate in the technical overview.

▶ **Theorem 5.** *For unweighted graphs of maximum degree* $\Delta \leq n/\log n$*, there is an* $O(1/\epsilon)$*-round algorithm in the CONGEST model that finds, with high probability, an independent set of size at least* $\frac{n}{(1+\epsilon)(\Delta+1)}$.

[2] To the best of our knowledge, this ranking algorithm has first appeared in the book of Alon and Spencer [1] and is due to Boppana (see also the references for this algorithm in [8]).

Further Related Work. Ghaffari et al. [15], showed that there is an algorithm for the LOCAL model that finds a $(1+\epsilon)$-approximation for MaxIS in $O(\text{poly}(\log n/\epsilon))$ rounds, for a constant ϵ. The results in [11, 18] give a lower bound of $\Omega(\log^* n)$ rounds for any deterministic algorithm that returns an independent set of size at least $n/\log^* n$ on a cycle, and a randomized $O(1)$-round algorithm for $O(1)$-approximations in planar graphs, in the LOCAL model. The results by [7, 16] give fast algorithms for approximating MaxIS in unweighted graphs, where the approximation guarantees are only in expectation.

Road-map. In Section 2 we provide a technical overview. Section 3 contains some basic definitions and useful inequalities. The technical heart of the paper starts in Section 4, where we prove our first two results (Theorems 1 and 2). Due to space limitations, we defer the rest of our proofs to the the full version [17].

2 Technical Overview

Results for weighted graphs. Our first two results (Theorems 1 and 2) share a similar proof structure. First, we show that there are fast algorithms for $O(\Delta)$-approximation. Then we use the *local-ratio* technique [3] to prove a general boosting theorem that takes a T-round algorithm for $O(\Delta)$-approximation, and use it as a black-box to output a $(1+\epsilon)\Delta$-approximation in $O(T/\epsilon)$ rounds. An overview of the local-ratio technique and the boosting theorem is provided in Section 2.2. The key ingredient to show a fast $O(\Delta)$-approximation algorithm is a new *weighted sparsification* technique, where we show that it suffices to find an independent set of a good approximation in a sparse subgraph. An overview of the weighted sparsification technique is provided in Section 2.1.

Our improved approximation algorithm for low-arboricity graphs (Theorem 3) uses Theorem 2 as a black-box, where the main technical ingredient is the local-ratio technique. An overview of this algorithm is also provided in Section 2.2.

Results for unweighted graphs. Our upper bound for unweighted graphs of maximum degree $\Delta \leq n/\log n$ (Theorem 5) has a similar two-step structure as the first two results. We first show an $O(\Delta)$-approximation algorithm, and then we use the local-ratio technique to boost the approximation factor. For the $O(\Delta)$-approximation part, we show that running the classical one-round ranking algorithm (that was used by [8]) for c rounds already returns an $O(\Delta)$-approximation for unweighted graphs of maximum degree $\Delta \leq n/\log n$, with probability $\approx 1-1/n^c$. The main technical ingredient for showing this result is a new analysis of the classical ranking algorithm using *martingales*. An overview of this result is provided in Section 2.3. Finally, in Section 2.4, we provide an overview of the lower bound result (Theorem 4).

2.1 Weighted Sparsification for $O(\Delta)$-Approximation

A good way to understand the $O(\Delta)$-approximation algorithm is to first consider the unweighted case. Let $G=(V,E)$ be an unweighted graph. We can find an $O(\Delta)$-approximation for MaxIS in G as follows. First, we sample a sparse subgraph H of G with the following properties. (1) The maximum degree Δ_H of H is small ($O(\log n)$). (2) The ratio between the number of nodes (n_H) and the maximum degree of H is at least as in G, up to a constant multiplicative factor. That is, $n_H/\Delta_H = \Omega(n/\Delta)$. Since any MIS in H has size at least $n_H/\Delta_H = \Omega(n/\Delta)$, it suffices to find an MIS in H, which takes $\text{MIS}(n_H, \Delta_H) \leq \text{MIS}(n, \log n)$ rounds (recall that $\text{MIS}(n, \Delta)$ is the running time of finding an MIS in graphs of n nodes and maximum degree Δ). By the recent breakthrough of Rozhon and Ghaffari [23], $\text{MIS}(n, \log n) = O(\log\log n) + \text{poly}(\log\log n) = \text{poly}\log\log n$ rounds. Furthermore, sampling a subgraph with the aforementioned properties is almost trivial. Each node joins H with probability $\min\{\log n/\Delta, 1\}$, independently. It is not very hard to show, via standard Chernoff (Fact 1) and Union Bound arguments, that H has the desired properties. While this approach is straightforward for the unweighted case, it runs into challenges when trying to apply it for the weighted case, as we explain next.

The challenge in weighted graphs. Perhaps the first thing that comes into mind when trying to extend the sampling technique to weighted graphs is to try to sample a sparse subgraph H with the following properties. (1) The maximum degree $\Delta_H = O(\log n)$. (2) The ratio between the *total weight* in H and the max degree of H is the same as in G, up to a constant multiplicative factor. That is $w(V_H)/\Delta_H = \Omega(w(V)/\Delta)$, where $w(V_H)$ is the total weight of nodes in H and $w(V)$ is the total weight of nodes in G. However, this approach runs into two challenges. The first challenge is that in the weighted case, an MIS doesn't necessarily constitute a Δ-approximation for MaxIS. Therefore, even if we are able to sample a subgraph H with the desired properties, running an MIS algorithm on H might result in an independent set of a very small weight. To overcome this challenge, we show a very simple $\text{MIS}(n, \Delta)$-round algorithm that finds an $O(\Delta)$-approximation. This algorithm runs an MIS algorithm on the subgraph induced by nodes that are relatively heavy, compared to their neighbors. Specifically, a node is considered relatively heavy compared to its neighbors, if it is of weight at least $\Omega(1/\Delta)$-fraction of the sum of weights of its neighbors. It is not very hard to show that this algorithm returns an independent set of total weight $\Omega(w(V)/\Delta)$, where $w(V)$ is the total weight of nodes in the graph. The proof of this argument is provided in Section 4.1.

Furthermore, another challenge is that the same sampling procedure doesn't work for the weighted case. In particular, if we sample each node with probability $p = \min\{(\log n)/\Delta, 1\}$, then *light*-weight nodes will have the same probability of joining H as *heavy*-weight nodes. Intuitively, we need to take the weights into account. For this, we boost the sampling probability of a node v by an additive factor of $w(v)\log n/w(V)$, where $w(v)$ is the weight of v and $w(V)$ is the total weight of nodes in the graph. In order to show that the sampled subgraph has the desired properties, it doesn't suffice to use standard Chernoff and Union-Bound arguments. Instead, we present a more involved analysis that uses Bernstein's inequality (Fact 2). Observe that the nodes don't know the value $w(V)$. Therefore, we define a notion of *weighted degree* of a node, which is the sum of weights of its neighbors. We show that it suffices for a node v to use the maximum weighted degree in its neighborhood, instead of $w(V)$. The full argument is provided in Section 4.2.

2.2 Boosting the Approximation Factor using Local-Ratio

A useful technique for approximation algorithms is the local-ratio technique [3]. In recent years, the local-ratio technique has been found to be very useful for the distributed setting [4,5], and the Δ-approximation algorithm of [4] also uses this technique. In this work we use local-ratio to boost the approximation guarantee for MaxIS. We start with stating the local-ratio theorem for maximization problems. Here, we state it specifically for MaxIS. Given a weighted graph $G_w = (V, E, w)$, where w is a node-weight function $w : V \to \mathbb{R}$, we say that an independent set $I \subseteq V$ is r-approximate with respect to w if it is r-approximate for the optimal solution in G_w.

▶ **Theorem 6** (Theorem 9 in [3]). *Let $G_w = (V, E, w)$ be a weighted graph. Let w_1 and w_2 be two node-weight functions such that $w = w_1 + w_2$. If an independent set I is r-approximate with respect to w_1 and with respect to w_2 then it is r-approximate with respect to w as well.*

Theorem 6 already gives a simple linear-time sequential algorithm for Δ-approximation for MaxIS, as follows. Pick an arbitrary node v of positive weight, push it onto a stack, and reduce the weight of any node in the inclusive neighborhood of v (v and its neighbors) by $w(v)$. Continue recursively on the obtained graph, until there are no nodes of positive weight. When there are no remaining nodes of positive weight, pop out the stack, and construct an independent set I greedily, as follows. For each node v that is popped out from the stack, add v to I, unless it already contains a neighbor of v.

The reason that this simple algorithm gives a Δ-approximation is as follows. Consider the first iteration, when the algorithm picks an arbitrary node v, pushes it onto a stack, and reduces the weight of any node in the inclusive neighborhood of v by $w(v)$. This first iteration implicitly defines two weight functions: the *reduced* weight function w_1, and the *residual* weight function w_2, where $w = w_1 + w_2$. That is, the reduced weigh of a node u in the first step is $w_1(u) = w(v)$ if it belongs to the inclusive neighborhood of v, and $w_1(u) = 0$ otherwise. The residual weight of a node u is the remaining weight $w_2(v) = w(v) - w_1(v)$. To prove that the algorithm returns a Δ-approximation, we can assume by reverse induction that I is a Δ-approximation with respect to the residual weight function w_2. Furthermore, the independent set is constructed in a way such that it must contain at least one node in the inclusive neighborhood of v, where the weight of this node with respect to w_1 is $w(v)$. Since the degree of v is at most Δ, and the value of the optimal solution with respect to w_1 is at most $\Delta w(v)$, it follows that I is also Δ-approximation with respect to the reduced weight function w_1. Hence, by the local-ratio theorem, the independent set is also a Δ-approximation with respect to $w = w_1 + w_2$.

One can extend this idea, and rather than picking a single node in each step, the algorithm can pick an arbitrary independent set I', push all the nodes in I' onto a stack, and perform local weight reductions in the inclusive neighborhood of any node in I'. The algorithm continues recursively on the obtained graph after the weight reductions, until there are no remaining nodes of positive weight. Then, the algorithm constructs an independent set I by popping out the stack and adding nodes in the stack to I greedily. Using a similar local-ratio argument, one can show that this algorithm also returns a Δ-approximation for MaxIS. The idea of picking an independent set rather than a single node in each step was used by [4] to show a Δ-approximation algorithm in $O(\text{MIS}(n, \Delta) \log W)$ rounds.

In this work, we prove a simple yet powerful property about the local-ratio technique. Specifically, we show that the total weight of the independent set I that is constructed in the pop-out stage (with respect to the original input weight function w), is at least the total weight of the nodes in the stack (with respect to the residual weight function at the time they were pushed onto the stack). That is, let S be set of nodes that are pushed onto the stack. For $v \in S$, let w_{i_v} be the residual weight of v at the time it was pushed onto the stack. We prove that $w(I) \geq \sum_{v \in S} w_{i_v}$. We refer to this property as the *stack property*.

The stack property allows us to show a general boosting theorem, as follows. We use the local-ratio algorithm described above, where in each step we pick an independent set I' that is $(c\Delta)$-approximation for MaxIS, for some constant $c > 1$. Hence, intuitively, after $\approx c/\epsilon$ steps, the total weight in the stack should be at least $\frac{OPT(G_w)}{(1+\epsilon)\Delta}$, where $OPT(G_w)$ is the value of an optimal solution in the input graph G_w.

Low-arboricity graphs. Moreover, the stack property allows us to show an improved approximation algorithm for low-arboricity graphs, as follows. In each step, we run a $(1+\epsilon)\Delta$-approximation algorithm on the subgraph induced by the nodes of degree at most 4α, where α is the arboricity of the graph. We push the nodes in the resulting independent set I' onto the stack, and perform local weight reduction in the neighborhoods of the nodes in I'. Then, we delete all the nodes of degree at most 4α, and continue recursively on the resulting graph. Finally, we construct an independent I by popping out the stack greedily. By a standard Markov argument, after $\log n$ push steps, the graph becomes empty. Furthermore, since in each step the algorithm finds a $(1+\epsilon)4\alpha$ approximation in the subgraph induced by the nodes of degree at most 4α, and this independent set is pushed onto the stack, we are able to use the stack property to show that the constructed independent set I is roughly of the same approximation for G_w.

2.3 Analysis of the Ranking Algorithm using Martingales

In this section we provide an overview of our result for unweighted graphs of maximum degree $\Delta \leq n/\log n$ (Theorem 5). First, we find an $O(\Delta)$-approximation, and then we use the boosting theorem to get a $(1+\epsilon)\Delta$-approximation. To find an $O(\Delta)$-approximation, we use the classical ranking algorithm. Recall that in the ranking algorithm, each node v picks a number r_v uniformly at random in $[0,1]$. If $r_v > r_u$ for any neighbor u of v, then v joins the independent set. Let I be the independent set that is returned by the ranking algorithm. The crux of the analysis is in using concentration inequalities to get a high-probability lower bound on the number of nodes in I. However, it is unclear how to make this approach work, as the random variables $X_v = \mathbf{1}_{v\in I}$ are not independent. While these random variables are not independent, one can obtain a weaker result in this direction. Specifically, for graphs of maximum degree at most $n^{1/3}/\text{poly}(\log n)$, one can get a useful bound on the maximum *dependency* among these variables. In particular, one can show that each X_v is dependent on at most $(n^{1/3}/\text{poly}(\log n))^2 = n^{2/3}/\text{poly}(\log n)$ other X_us, which makes it possible to show concentration using the bounded dependence Chernoff bound given in [22]. However, it is unclear how to use this approach for higher degree graphs.

The main idea of our approach is to view the ranking algorithm from a sequential perspective. Instead of picking ranks for the nodes and including a node in I if its rank is higher than that of its neighbors, we draw nodes v from V uniformly at random one at a time and add v to I if it is not adjacent to any previously drawn node. We show that the resulting independent set is identical in distribution to the independent set produced by the ranking algorithm. Note that this is not the same as a sequential greedy algorithm for maximal independent set, which would add v to I if it is not adjacent to any node in I (a weaker condition). The sequential perspective of the ranking algorithm allows us to think about the size of I incrementally. One could directly show concentration if the family of random variables $\{I_t\}_t$ was a martingale. However, this is not the case, as $|I_{t+1}| \geq |I_t|$ so it is not possible for expected increments to be 0. Instead, we create a martingale by shifting the increments so that they have mean 0. More formally, let I_t be the independent set I after the first t nodes have been drawn. Let v_t be the tth node drawn. The random variable

$$Y_t = |I_t| - |I_{t-1}| - \Pr[v_t \in I | I_{t-1}]$$

has mean 0 conditioned on I_{t-1}. Therefore, the Y_ts are increments for the martingale $X_t = \sum_{i=1}^{t} Y_t$. Using Azuma's Inequality, one can show that X_t concentrates around its mean, which is 0. To lower bound the size of the obtained independent set I, one therefore just needs to get a lower bound on the sum of the increment probabilities $\Pr[v_t \in I | I_{t-1}]$.

This can be lower bounded by $1/2$ when $t = o(n/\Delta)$ because when a node is drawn, it eliminates at most Δ other nodes from inclusion into I. But when $t = \Theta(n/\Delta)$, the sum of these probabilities is already $1/2(\Theta(n/\Delta)) = \Theta(n/\Delta)$, so the independent set is already large enough, as desired. The reason that this technique works for $\Delta \leq n/\log n$ is that the success probability is roughly exponential in n/Δ. Hence, by having $\Delta \leq n/\log n$, we get a high probability success, as desired.

2.4 An Overview of the Lower Bound

In oder to prove out lower bound (Theorem 4), we use a *cycle of cliques* graph. We reduce the problem of finding an MIS in a cycle to the problem of finding an independent set of size $\Omega(n/\Delta)$ in a cycle of cliques. We use Naor's lower bound [20] for finding an MIS in a cycle, which holds even against randomized algorithms. We start by stating Naor's lower bound.

▶ **Theorem 7** (Lower bound for the cycle [20]). *Any randomized algorithm in the LOCAL model for finding a maximal independent set that takes fewer than $\frac{1}{2}(\log^* n) - 4$ rounds, succeeds with probability at most $1/2$, even for a cycle of length n.*

A good way of understanding our reduction from Naor's lower bound is to first consider the following failed attempt for deterministic algorithms.

Failed Attempt 1. Let $\mathcal{A}$ be a deterministic algorithm for approximate MaxIS. Suppose that it takes $T(n)$ rounds in graphs of n nodes. We can use $\mathcal{A}$ to find a maximal independent set in a cycle C of n nodes, as follows. We start by running $\mathcal{A}$ on C to produce an independent set I. Since C is a cycle, there is a natural clockwise ordering for the nodes of I. Between any two consecutive nodes of I, there may be nodes along the cycle that are not adjacent to a node in I. We informally call these nodes the "gaps" between consecutive nodes in I. We can obtain a maximal independent set in C by "filling in" the gap between every two consecutive nodes in I with a maximal independent set (sequentially). To bound the runtime of this algorithm, we need to bound the maximum length of a gap.

One can attempt to bound the length of these gaps using what is called an *indistinguishability argument*. From a local perspective, the nodes cannot distinguish between C and a path of length $\omega(T(n))$. Hence, one can show that if there is a gap of length $\omega(T(n))$, then $\mathcal{A}$ doesn't return the required approximation on a path of length $\omega(T(n))$. As a result, filling in the gaps between nodes in I takes $O(T(n))$ rounds. Therefore, by running $\mathcal{A}$ on C and then filling in the gaps sequentially, we get an MIS in $O(T(n))$ rounds. By Linial's lower bound [19], we have that $T(n) = \Omega(\log^* n)$.

However, this argument fails for two reasons. First, this indistinguishability argument does not work. In the LOCAL model, we assume that the ID of each vertex is a number from 1 to $\text{poly}(n)$. However, this is not the case for subpaths of C with length $O(T(n))$, since $\text{poly}(n) >> \text{poly}(T(n))$. Therefore, the approximation guarantee of $\mathcal{A}$ does not need to apply to short subpaths of C, meaning that there may be large gaps in the independent set output by applying $\mathcal{A}$ on C.

Second, our goal is to show a lower bound for randomized algorithms. The main issue when running a randomized algorithm on a cycle is that the maximum length of a gap between two consecutive nodes in the independent set can be larger than $O(T(n))$. This is because randomized algorithms that succeed with high probability can fail with probability $1/\text{poly}(n)$, where n is the number of nodes in the graph. Hence, $\mathcal{A}$ can fail on a path of length $O(T(n))$ with probability $1/\text{poly}(T(n))$ which is non-negligible when $T(n) \ll n$. In particular, since there are $\Omega(|C|/T(n)) = \Omega(n/T(n))$ subpaths of length $O(T(n))$ in C, it is likely that $\mathcal{A}$ fails on at least one of these subpaths. If on the other hand the number of nodes in the $O(T(n))$-radius neighborhood of a node was larger, then one could hope to get around this issue, as it would amplify the "local" success probability in the neighborhood of a node.

Successful Attempt 2. Instead of running $\mathcal{A}$ on C, we run it on a cycle of cliques C_1, which is obtained from C as follows. Each node $v \in C$ is replaced with a clique of size $\approx 2^{|C|}$, denoted by $D(v)$, where every two adjacent cliques are connected by a bi-clique. By running $\mathcal{A}$ on C_1 instead of C, it boosts the success probability of $\mathcal{A}$ in a small-radius neighborhood of any given node. As a result, a small-radius neighborhood of any node in C_1 must contain a node in the independent set. Using the independent set I_1 that was found in C_1, we can map it to an independent set I in C, as follows. Every $v \in C$ joins I if and only if I_1 contains a node in $D(v)$. Due to the approximation guarantee of $\mathcal{A}$ in C_1, we can prove that the maximum distance between two consecutive nodes in I_1 is small and therefore, the maximum length of a gap in I is small. Finally, we can run a greedy sequential MIS algorithm to fill the gap between every two consecutive nodes in I and find an MIS in C. Hence, if we can find an approximate-MaxIS in C_1 in $o(\log^* |C_1|)$ rounds, then we can find an MIS in C in $o(\log^*(2^{|C|})) = o(\log^* |C|)$ rounds, contradicting Naor's lower bound (Theorem 7). An illustration of the reduction with all the steps is provided.

This approach deals with the two issues found in our first reduction attempt, because the size of C_1 is bounded by a polynomial of the size of each clique (the first issue) and the large size of each clique ensures that the algorithm succeeds with high probability (the second issue).

3 Preliminaries

Some of our proofs use the following standard probabilistic tools. An excellent source for the following concentration bounds is the book by Alon and Spencer [1]. These bounds can also be found in many lecture notes about basic tail and concentration bounds.

▶ **Fact 1** (Multiplicative Chernoff Bound). *Let $X_1, ..., X_n$ be independent random variables taking values in $\{0, 1\}$. Let X denote their sum and let $\mu = E[X]$ denote the sum's expected value. Then for any $0 \le \epsilon \le 1$, it holds that:*

$$Pr[|X - \mu| \ge \epsilon\mu] \le 2 \exp\left(-\frac{\epsilon^2}{2+\epsilon}\mu\right)$$

▶ **Fact 2** (Bernstein's Inequality). *Let $X_1, ..., X_n$ be independent random variables such that $\forall i, X_i \le M$. Let X denote their sum and let $\mu = \mathrm{E}[X]$ denote the sum's expected value. Then for any positive t, it holds that:*

$$Pr[|X - \mu| \ge t] \le 2 \exp\left(-\frac{t^2/2}{Mt/3 + \sum_{i=1}^{n} \mathrm{Var}(X_i)}\right)$$

▶ **Fact 3** (One-sided Azuma's Inequality). *Suppose $\{X_i : i = 0, 1, 2, \ldots\}$ is a martingale and that $|X_i - X_{i-1}| \le c_i$ almost surely. Then, for all positive integers N and all positive reals t,*

$$\Pr[X_N - X_0 \le -t] \le \exp\left(-\frac{t^2}{2\sum_{i=1}^{N} c_i^2}\right)$$

Assumptions. In all of our upper and lower bounds, we don't assume that the nodes have any global information. In particular, they don't know n or Δ. The only information that each node has before the algorithm starts is its own identifier, and some polynomial upper bound on n (Since the nodes can send $c \log n$ bits in each round to each of their neighbors, naturally, they know some polynomial upper bound on n).

Some notations. The input graph is denoted by $G_w = (V, E, w)$, where V is the set of nodes, E is the set of edges, and w is the weight function. The reason that we choose to add the weight function in a subscript is that some parts of the analysis deal with graphs that have the same sets of nodes and edges as the input graph, but a different weight function. Hence, such a graph will be denoted by $G_{w'} = (V, E, w')$, to indicate that it is the same as the input graph, but with weight function w' rather than w.

We denote by $N^+(v)$ the inclusive neighborhood of v, which consists of $N(v) \cup \{v\}$, where $N(v)$ is the set of neighbors of v. Furthermore, we denote by $deg(v) = |N(v)|$ the number of neighbors of a node v. Finally, we denote by $w(V')$ the total weight of nodes in $V' \subseteq V$. That is, $w(V') = \sum_{v \in V'} w(v)$.

4 A $(1+\epsilon)\Delta$-Approximation Algorithm

In this section we prove Theorems 1 and 2.

▶ **Theorem 1.** *There is an $O(\mathrm{MIS}(n,\Delta)/\epsilon)$-round algorithm in the CONGEST model that finds a $(1+\epsilon)\Delta$-approximation for maximum-weight independent set. Whether the algorithm is deterministic or randomized, depends on the* MIS *algorithm that is run as a black-box.*

▶ **Theorem 2.** *There is a randomized $(\mathrm{poly}(\log\log n)/\epsilon)$-round algorithm in the CONGEST model that finds, with high probability, a $(1+\epsilon)\Delta$-approximation for maximum-weight independent set.*

Theorems 1 and 2 share a similar proof structure. First, we present algorithms for $O(\Delta)$-approximation in Sections 4.1 and 4.2. Then, by using a general boosting theorem, we get $(1+\epsilon)\Delta$-approximation algorithms.

4.1 An $O(\mathrm{MIS}(n,\Delta))$-Round Algorithm for $O(\Delta)$-Approximation

In this section we show a very simple $O(\mathrm{MIS}(n,\Delta))$-round algorithm that finds an $O(\Delta)$-approximation for MaxIS.

▶ **Theorem 8.** *Given a weighted graph $G_w = (V, E, w)$, there is an $O(\mathrm{MIS}(n,\Delta))$-round algorithm that finds an independent set of weight at least $\frac{w(V)}{4(\Delta+1)}$, in the CONGEST model. Whether the algorithm is deterministic or randomized depends on the* MIS *algorithm that is used as a black-box.*

Algorithm For every $v \in V$, let $\delta(v)$ be the maximum degree of a node in the inclusive neighborhood of v. That is, $\delta(v) = \max\{deg(u) \mid u \in N^+(v)\}$. A node v is called *good* if $w(v) \geq \frac{1}{2(\delta(v)+1)} \sum_{u \in N^+(v)} w(u)$. The algorithm finds a maximal independent set I in the subgraph induced by the set of good nodes. We prove the following lemma.

▶ **Lemma 9.** $w(I) \geq w(V)/4(\Delta+1)$

Proof. Let V_{good} be the set of good nodes, and let $\overline{V} = V \setminus V_{good}$. Observe that,

$$\sum_{v\in\overline{V}} w(v) \le \sum_{v\in\overline{V}} \frac{1}{2(\delta(v)+1)} \sum_{u\in N^+(v)} w(u) \le \sum_{v\in V} \frac{deg(v)+1}{2(deg(v)+1)} w(v) = w(V)/2$$

$$\Rightarrow \sum_{v\in I} w(v) \ge \sum_{v\in I} \frac{1}{2(\delta(v)+1)} \sum_{u\in N^+(v)} w(u) \ge \sum_{v\in I} \frac{1}{2(\Delta+1)} \sum_{u\in N^+(v)\cap V_{good}} w(u)$$

$$\ge \frac{1}{2(\Delta+1)} \sum_{v\in V_{good}} w(v) \ge w(V)/4(\Delta+1)$$

as desired. Since the value of an optimal solution in G_w is at most $w(V)$, the algorithm returns an $O(\Delta)$-approximation for MaxIS. ◀

Success with high probability. Given a graph of n nodes, an algorithm that finds a maximal independent set in the graph with high probability is an algorithm that succeeds with probability at least $1 - 1/n^c$ for some constant $c > 1$. In the algorithm above, the black box can be a randomized algorithm that is run on a subgraph $H = (V_H, E_H)$ of G_w. Since $n_H = |V_H|$ is potentially much smaller than n, one may wonder whether the algorithm above actually succeeds with high probability with respect to n. The main idea is to use an algorithm that is *intended* to work for graphs with n nodes, rather than n_H nodes. We prove the following lemma, whose proof is by a simple padding argument that is deferred to the full version [17].

▶ **Lemma 10.** *Let $\mathcal{A}$ be an $\mathrm{MIS}(n, \Delta)$-round algorithm that finds a maximal independent set with success probability p in a graph of n nodes and maximum degree Δ. Let $H = (V_H, E_H)$ be a graph of $n_H \le n$ nodes with $(c \log n)$-bit identifiers, for some constant c, and let Δ_H be the maximum degree in H. There is an $O(\mathrm{MIS}(n, \Delta_H))$-round algorithm $\mathcal{A}'$ that finds a maximal independent set in H with success probability p.*

4.2 A $\mathrm{poly}(\log\log n)$-Round Algorithm for $O(\Delta)$-Approximation

In this section we show a $\mathrm{poly}(\log\log n)$-round algorithm that finds an $O(\Delta)$-approximation.

▶ **Theorem 11.** *Given a weighted graph $G_w = (V, E, w)$, there is a constant $c > 1$ and a $\mathrm{poly}(\log\log n)$-round algorithm in the CONGEST model that finds, with high probability, an independent set of weight at least $\frac{w(V)}{c\Delta}$.*

Our algorithm has the following two-step structure.

1. First, we sample a sparse subgraph $H_w = (V_H, E_H, w)$ of G_w with the following two properties:
   a. The maximum degree Δ_H of H_w is at most $O(\log n)$.
   b. $w(V_H)/\Delta_H = \Omega(w(V)/\Delta)$. That is, the ratio between the total weight and maximum degree in H_w is at least, up to a constant factor, as in G_w.
2. Then, we use Theorem 8 to find an independent set in H_w of size at least $\frac{w(V_H)}{4(\Delta_H+1)} = \frac{w(V)}{c\Delta}$, for some constant $c > 1$, in $O(\mathrm{MIS}(n, \Delta_H)) = O(\mathrm{MIS}(n, \log n)) = \mathrm{poly}(\log\log n)$ rounds.

The first step: sampling a subgraph with the desired properties. Recall that $w(N(v))$ is the sum of weights of the neighbors of v, which we call the *weighted degree* of v. For each node $v \in V$, let $w_{max}(v) = \max\{w(N(u)) \mid u \in N^+(v)\}$. It is useful to think about $w_{max}(v)$ as the *maximum weighted degree* of a node in the inclusive neighborhood of v. We sample a subgraph $H_w = (V_H, E_H, w)$, as follows. Let $\lambda \geq 1$ be a constant to be chosen later. Recall that $\delta(v)$ is the maximum degree of a node in the inclusive neighborhood of v. Each node $v \in V$ joins V_H with probability

$$p(v) = \lambda \log n \cdot (\frac{1}{\delta(v)} + \frac{w(v)}{w_{max}(v)})$$

Where if $p(v) \geq 1$, then v joins H deterministically. In Lemma 12, we show that the maximum degree of H_w is $\Delta_H = O(\log n)$. In Lemma 16, we show that $w(V_H) = \Omega(\min\{w(V), w(V)\log n/\Delta\})$.

▶ **Lemma 12.** *The maximum degree Δ_H in H_w is $O(\log n)$, with high probability.*

Proof. Let $V^+ = \{v \in V \mid p(v) \geq 1\}$. We show that each node u has at most $O(\log n)$ neighbors in $V^+ \cap V_H$, and at most $O(\log n)$ neighbors in $(V \setminus V^+) \cap V_H$. Let $N_H(v)$ be the set of neighbors of v in H.

1. For every $v \in V$, $|N_H(v) \cap V^+| \leq 2\lambda \log n$: Assume towards a contradiction that there are more than $2\lambda \log n$ nodes in $N_H(v) \cap V^+$. Since each node $v \in V^+$ has $p(v) \geq 1$, it holds that

$$\sum_{u \in N(v) \cap V^+} p(u) \geq \sum_{u \in N_H(v) \cap V^+} p(u) > 2\lambda \log n$$

On the other hand,

$$\sum_{u \in N(v) \cap V^+} p(u) \leq \sum_{u \in N(v)} p(u) = \sum_{u \in N(v)} \lambda \log n \cdot (\frac{1}{\delta(v)} + \frac{w(v)}{w_{max}(v)})$$

Since $deg(v) = |N(v)|$ and $w(N(v)) = \sum_{u \in N(v)} w(u)$ are lower bounds on $\delta(v)$ and $w_{max}(v)$, respectively, we have that

$$\sum_{u \in N(v)} \lambda \log n \cdot (\frac{1}{\delta(u)} + \frac{w(u)}{w_{max}(u)}) \leq \sum_{u \in N(v)} \lambda \log n \cdot (\frac{1}{deg(v)} + \frac{w(u)}{w(N(v))}) = 2\lambda \log n$$

which is a contradiction.

2. $|N_H(v) \cap (V \setminus V^+)| \leq 2\lambda \log n$: Observe that the expected number of neighbors of v in $N_H(v) \cap (V \setminus V^+)$ is

$$\sum_{u \in N(v)} p(u) \leq 2\lambda \log n$$

Since $|N_H(v) \cap (V \setminus V^+)|$ is a sum of independent random variables, one can apply Chernoff's bound (Fact 1) to achieve that this number concentrates around its expectation with high probability.

By applying a standard Union-Bound argument over all the nodes, we conclude that the maximum degree in H_w is $\Delta_H = O(\log n)$ with high probability. ◀

The rest of this section is devoted to the task of proving that $w(V_H) = \Omega(\min\{w(V), w(V)\log n/\Delta\})$. This is proved in Lemma 16. First, we start by proving a slightly weaker lemma, that assumes that for all $v \in V$, $p(v) \leq 1$.

▶ **Lemma 13.** *Assume $p(v) \leq 1$, for all $v \in V$. It holds that $w(V_H) = \Omega(w(V)\log n/\Delta)$, with high probability.*

Main idea of the proof of Lemma 13. Let $w_1 \geq w_2 \geq ... \geq w_n$ be a sorting of the weights of nodes in V in a decreasing order (where ties are broken arbitrarily). Let $V_{high} = \{u \in V \mid w(u) \in \{w_1, ..., w_\Delta\}\}$, and let $V_{low} = V \setminus V_{high} = \{u \in V \mid w(u) \in \{w_{\Delta+1}, ..., w_n\}\}$. That is, V_{high} contains the Δ heaviest nodes, and V_{low} contains all the other nodes. The proof is split into the following two cases that are proven separately in Claims 14 and 15.

1. $w(V_{high}) \geq w(V)/2$: In this case, at least half of the total weight is distributed among high-weight nodes. Intuitively, we need to make sure that we get many of these high-weight nodes. Since the number of high-weight nodes that are sampled is a sum of independent random variables, we are able to use Chernoff's bound to prove that many of them are sampled, with high probability. The full proof for this case is presented in Claim 14.
2. $w(V_{low}) \geq w(V)/2$: In this case, at least half of the total weight is distributed among low-weight nodes. Therefore, it is sufficient to show that $w(V_H) = \Omega(w(V_{low}) \log n/\Delta)$. The key property here is that we can bound the maximum weight of a node in V_{low} by $w(V)/\Delta$. We show how to use this property together with Bernstein's inequality to prove Lemma 13 for this case. The full proof for this case is presented in Claim 15.

▷ Claim 14. Assume that for all $v \in V$, $p(v) \leq 1$. Let $V_{high} = \{u \in V \mid w(u) \in \{w_1, ..., w_\Delta\}\}$. If $w(V_{high}) \geq w(V)/2$, then $w(V_H) = \Omega(w(V) \log n/\Delta)$, with high probability.

Proof. Let $S = \{v \in V_{high} \mid w(v) \geq w(V)/4\Delta\}$. We start by showing that at least a constant fraction of the total weight in G_w is distributed among nodes in S. Let $\overline{S} = V_{high} \setminus S$, we start by showing that $w(\overline{S}) \leq w(V)/4$:

$$w(\overline{S}) \leq \sum_{v \in \overline{S}} w(v) \leq \sum_{v \in \overline{S}} \frac{w(V)}{4\Delta} \leq \frac{w(V)}{4}$$

where the last inequality holds because $|\overline{S}| \leq |V_{high}| = \Delta$. Therefore, $w(S) = w(V_{high} \setminus \overline{S}) = w(V_{high}) - w(V(\overline{S})) \geq w(V)/4$. Next, we show that $|S \cap V_H| = \Omega(\log n)$, by using Chernoff's bound. Let x_v be a $\{0, 1\}$ random variable indicating whether $v \in V_H$, and let $X = \sum_{v \in S} x_v$. We show that the expectation of X is at least $c \log n/4$.

$$\begin{aligned}\mathbb{E}[X] &= \sum_{v \in S} \mathbb{E}[x_v] = \sum_{v \in S} p(v) = \sum_{v \in S} \lambda \log n \cdot (\frac{1}{\delta(v)} + \frac{w(v)}{w_{max}(v)}) \\ &\geq \sum_{v \in S} \frac{w(v)\lambda \log n}{w(V)} \geq \frac{\lambda \log n}{w(V)} \cdot \sum_{v \in S} w(v) = \frac{w(S)\lambda \log n}{w(V)} \geq \frac{\lambda \log n}{4}\end{aligned}$$

Furthermore, sine X is a sum of independent $\{0, 1\}$ random variables with expectation $\Omega(\log n)$, by applying Chernoff's bound (Fact 1), we conclude that there are at least $\Omega(\log n)$ nodes in $S \cap V_H$, with high probability. Since each node in S has weight at least $w(V)/4\Delta$, this implies that the total weight in V_H is $w(V_H) \geq w(S \cap V_H) = \Omega(w(V) \log n/\Delta)$, with high probability, as desired. ◁

▷ Claim 15. Assume that for all $v \in V$, $p(v) \leq 1$. Let $V_{low} = \{v \in V \mid w(v) \in \{w_{\Delta+1}, ..., w_n\}\}$. If $w(V_{low}) \geq w(V)/2$, then $w(V_H \cap V_{low}) = \Omega(w(V) \log n/\Delta)$, with high probability.

Proof. Let x_v be a $\{0, 1\}$ random variable indicating whether $v \in V_H$, let $y_v = x_v \cdot w(v)$, and let $Y = \sum_{v \in V_{low}} y_v$. We prove the following 3 properties:

1. $\mathbb{E}(Y) \geq \frac{w(V)\lambda \log n}{2\Delta}$: this is because

$$\mathbb{E}[Y] = \sum_{v \in V_{low}} p(v) \cdot w(v) = \sum_{v \in V_{low}} \lambda \log n \cdot (\frac{1}{\delta(v)} + \frac{w(v)}{w_{max}(v)}) \cdot w(v)$$
$$\geq \sum_{v \in V_{low}} \frac{w(v)\lambda \log n}{\Delta} = \frac{w(V_{low})\lambda \log n}{\Delta} \geq \frac{w(V)\lambda \log n}{2\Delta}$$

where the last equality holds since $w(V_{low}) \geq w(V)/2$.

2. For any $v \in V_{low}$, it holds that $w(v) \leq w(V)/\Delta$: Recall that $\{w_1, \cdots, w_n\}$ is an ordering of the weight of nodes by a decreasing order. Hence, for any j, it holds that

$$w_j \cdot j \leq \sum_{i=1}^{j} w_j \leq w(V)$$

where the first inequality holds because w_j is the minimum among $\{w_1, ..., w_j\}$. Hence, since each node $v \in V_{low}$ has weigh w_j where $j > \Delta$, we have that $w(v) \leq w(V)/\Delta$ for any $v \in V_{low}$.

3. It holds that $\sum_{v \in V_{low}} \mathbb{E}[y_v^2] \leq w(V) \cdot \mathbb{E}[Y]/\Delta$: First, observe that

$$\sum_{v \in V_{low}} \mathbb{E}[y_v^2] \leq \max\{w(v) \mid v \in V_{low}\} \cdot \sum_{v \in V_{low}} \mathbb{E}[y_v] = \max\{w(v) \mid v \in V_{low}\} \cdot \mathbb{E}[Y]$$
$$\leq \frac{w(V) \cdot \mathbb{E}[Y]}{\Delta}$$

where the last inequality holds by the second property.

By proving these three properties, we have satisfied all the prerequisites of Bernstein's inequality. A direct application of the inequality yields:

$$Pr\Big[|Y - \mathbb{E}[Y]| \geq \mathbb{E}[Y]/2\Big] \leq 2\exp\left(-\frac{\mathbb{E}[Y]^2/8}{M \cdot \mathbb{E}[Y]/6 + \sum_{v \in V_{low}} \mathrm{Var}(y_v)}\right)$$

By the second and third properties, we have that

$$\sum_{v \in V_{low}} \mathrm{Var}(y_v) = \sum_{v \in V_{low}} \mathbb{E}(y_v^2) - \mathbb{E}[y_v]^2 \leq \sum_{v \in V_{low}} \mathbb{E}(y_v^2) \leq \frac{w(V) \cdot \mathrm{E}[Y]}{\Delta}$$
$$\Rightarrow Pr\Big[|Y - \mathbb{E}[Y]| \geq \mathbb{E}[Y]/2\Big] \leq 2\exp\left(-\frac{\mathbb{E}[Y]^2/8}{\frac{w(V)\cdot\mathbb{E}[Y]}{6\Delta} + \frac{w(V)\cdot\mathbb{E}[Y]}{\Delta}}\right) \leq 2\exp\left(-\frac{6\Delta \cdot \mathbb{E}[Y]/8}{7w(V)}\right)$$

Furthermore, by the first property, we have that

$$\mathbb{E}[Y] \geq w(V)\lambda \log n/2\Delta$$
$$\Rightarrow 2\exp\left(-\frac{6\Delta \cdot \mathbb{E}[Y]/8}{7w(V)}\right) \leq 2\exp\left(-\frac{6w(V)\lambda \log n}{56w(V)}\right) = 2\exp\left(-\frac{6\lambda \log n}{56})\right)$$

Finally, choosing $\lambda = 112/6$ implies that:

$$Pr\Big[|Y - \mathbb{E}[Y]| \geq \mathbb{E}[Y]/2\Big] \leq \frac{1}{n^{2\log e}} < \frac{1}{n^2}$$

as desired. Furthermore, we can boost the success probability to $1 - 1/n^k$ for any constant $k > 1$, by setting $\lambda = \frac{112k}{3}$.

◁

Having proved claims 14 and 15, this finishes the proof of Lemma 13. Lemma 13 makes the assumption that $p(v) \leq 1$ for all $v \in V$. We remove this assumption in the proof of the following lemma.

▶ **Lemma 16.** *It holds that $w(V_H) = \Omega(\min\{w(V), w(V)\log n/\Delta\})$, with high probability.*

Proof. Let $V^+ = \{u \in V \mid p(w) \geq 1\}$. The proof is split into two cases:

1. $w(V^+) \geq w(V)/2$: Since all the nodes in V^+ join V_H deterministically, this implies that $w(V_H) \geq w(V^+) \geq w(V)/2$.
2. $w(V^+) < w(V)/2$: This implies that $w(V \setminus V^+) \geq w(V)/2$. Since each node $w \in V \setminus V^+$ has $p(w) < 1$, we can apply Lemma 13 directly on the nodes in $V \setminus V^+$ to conclude that $w(V_H) = \Omega(w(V \setminus V^+)\log n/\Delta) = \Omega(w(V)\log n/\Delta)$, with high probability, as desired. ◀

Now we are ready to finish the proof of Theorem 11.

Proof of Theorem 11. Since both Lemma 12 and 16 above hold with high probability, we can apply another standard Union-Bound argument to conclude that both of them hold with high probability (simultaneously). Hence, by running the algorithm from Section 4.1 on H_w, we get an independent set of weight $\Omega(w(V_H)/\Delta_H) = \Omega(\min\{w(V), w(V)\log n/\Delta\}/\Delta_H) = \Omega(w(V)/\Delta)$, in $\text{MIS}(n, \Delta_H) = \text{MIS}(n, \log n) = \text{poly}(\log\log n)$ rounds, with high probability, as desired. ◀

References

1 Noga Alon and Joel Spencer. *The Probabilistic Method*. John Wiley, 1992.

2 Nir Bachrach, Keren Censor-Hillel, Michal Dory, Yuval Efron, Dean Leitersdorf, and Ami Paz. Hardness of distributed optimization. In Peter Robinson and Faith Ellen, editors, *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing, PODC 2019, Toronto, ON, Canada, July 29 - August 2, 2019*, pages 238–247. ACM, 2019. doi:10.1145/3293611.3331597.

3 Amotz Bar-Noy, Reuven Bar-Yehuda, Ari Freund, Joseph Naor, and Baruch Schieber. A unified approach to approximating resource allocation and scheduling. *Journal of the ACM (JACM)*, 48(5):1069–1090, 2001.

4 Reuven Bar-Yehuda, Keren Censor-Hillel, Mohsen Ghaffari, and Gregory Schwartzman. Distributed approximation of maximum independent set and maximum matching. In *PODC*, pages 165–174. ACM, 2017.

5 Reuven Bar-Yehuda, Keren Censor-Hillel, and Gregory Schwartzman. A distributed $(2 + \epsilon)$-approximation for vertex cover in o(log Δ / ϵ log log Δ) rounds. *J. ACM*, 64(3):23:1–23:11, 2017.

6 Leonid Barenboim, Michael Elkin, Seth Pettie, and Johannes Schneider. The locality of distributed symmetry breaking. *J. ACM*, 63(3):20:1–20:45, 2016. doi:10.1145/2903137.

7 Marijke HL Bodlaender, Magnús M Halldórsson, Christian Konrad, and Fabian Kuhn. Brief announcement: Local independent set approximation. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing, PODC 2016, Chicago, IL, USA, July 25-28, 2016*, To Appear 2016.

8 Ravi B. Boppana, Magnús M. Halldórsson, and Dror Rawitz. Simple and local independent set approximation. In *SIROCCO*, volume 11085 of *Lecture Notes in Computer Science*, pages 88–101. Springer, 2018.

9 Keren Censor-Hillel, Seri Khoury, and Ami Paz. Quadratic and near-quadratic lower bounds for the CONGEST model. In *31st International Symposium on Distributed Computing, DISC 2017, October 16-20, 2017, Vienna, Austria*, pages 10:1–10:16, 2017. doi:10.4230/LIPIcs.DISC.2017.10.

10 Keren Censor-Hillel, Merav Parter, and Gregory Schwartzman. Derandomizing local distributed algorithms under bandwidth restrictions. In *31st International Symposium on Distributed Computing, DISC 2017, October 16-20, 2017, Vienna, Austria*, volume 91 of *LIPIcs*, pages 11:1–11:16. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2017. doi:10.4230/LIPIcs.DISC.2017.11.

11 Andrzej Czygrinow, Michal Hańćkowiak, and Wojciech Wawrzyniak. Fast distributed approximations in planar graphs. In *Distributed Computing*, pages 78–92. Springer, 2008.

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 '16, pages 270–277. SIAM, 2016. URL: http://dl.acm.org/citation.cfm?id=2884435.2884455.

13 Mohsen Ghaffari. Distributed maximal independent set using small messages. In *SODA*, pages 805–820. SIAM, 2019.

14 Mohsen Ghaffari. Personal communication, Feb 2020.

15 Mohsen Ghaffari, Fabian Kuhn, and Yannic Maus. On the complexity of local distributed graph problems. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 784–797, 2017. doi:10.1145/3055399.3055471.

16 Magnús M. Halldórsson and Christian Konrad. Computing large independent sets in a single round. *Distributed Computing*, 31(1):69–82, 2018.

17 Ken-ichi Kawarabayashi, Seri Khoury, Aaron Schild, and Gregory Schwartzman. Improved distributed approximation to maximum independent set. *CoRR*, abs/1906.11524, 2019. arXiv:1906.11524.

18 Christoph Lenzen and Roger Wattenhofer. Leveraging linial's locality limit. In *Distributed Computing*, pages 394–407. Springer, 2008.

19 Nathan Linial. Locality in distributed graph algorithms. *SIAM Journal on Computing*, 21(1):193–201, 1992.

20 Moni Naor. A lower bound on probabilistic algorithms for distributive ring coloring. *SIAM J. Discrete Math.*, 4(3):409–412, 1991. doi:10.1137/0404036.

21 David Peleg. *Distributed Computing: A Locality-sensitive Approach.* Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000.

22 Sriram V. Pemmaraju. Equitable coloring extends chernoff-hoeffding bounds. In Michel X. Goemans, Klaus Jansen, José D. P. Rolim, and Luca Trevisan, editors, *Approximation, Randomization and Combinatorial Optimization: Algorithms and Techniques, 4th International Workshop on Approximation Algorithms for Combinatorial Optimization Problems, APPROX 2001*, volume 2129, pages 285–296. Springer, 2001. doi:10.1007/3-540-44666-4_31.

23 Václav Rozhon and Mohsen Ghaffari. Polylogarithmic-time deterministic network decomposition and distributed derandomization. In Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy, editors, *Proccedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, pages 350–363. ACM, 2020. doi:10.1145/3357713.3384298.