Distributed Graph Coloring in a Few Rounds

Kishore Kothapalli*
Center for Security, Theory, and Algorithmic Research
International Institute of Information Technology
Hyderabad, India 500 032.
kkishore@iiit.ac.in

Sriram Pemmaraju†
Department of Computer Science
The University of Iowa
Iowa City, IA 52242, USA.
sriram@cs.uiowa.edu

ABSTRACT

This paper considers the question of how many colors a distributed graph coloring algorithm would need to use if it had only k rounds available, for any positive integer k. In our main result, we present an algorithm that runs in $O(k)$ rounds for any k bounded below by $\Omega(\log \log n)$ and bounded above by $O(\sqrt{\log n})$, and uses $O(a \cdot n^{1/k})$ colors to color a graph with arboricity a. This result is optimal since the palette size matches the lower bound of Barenboim and Elkin (*PODC 2008*). This result is achieved via the use of several new results developed in this paper on coloring graphs whose edges have been acyclically oriented. For example, suppose that G is an n-vertex, acyclically oriented graph with maximum out-degree Δ_o. We present an algorithm that, for any $k \geq 2 \log \log n$, runs in $O(k)$ rounds on G to produce an (i) $O(\Delta_o)$-coloring when $\Delta_o \in \Omega(\max\{kn^{2/k^2} \log^{1+1/k} n, 2^k\})$ and an (ii) $O(\Delta_o \cdot n^{2/k^2})$-coloring when $\Delta_o \in \Omega(\max\{k \log^{1+1/k} n, 2^k\})$. These results are useful in any setting where it is possible to efficiently compute acyclic orientations of a graph with $\Delta_o \ll \Delta$. We derive non-trivial bounds on the palette size even when $k < 2 \log \log n$.

Our main technical contributions can be summarized as: (i) developing a k-round version of the algorithm of Kothapalli et al. (*IPDPS 2006*) which computes an $O(\Delta)$-coloring of a graph in $O(\sqrt{\log n})$ rounds, and (ii) developing an oriented version of the Brooks-Vizing coloring result of Grable and Panconesi (*SODA 1998*).

Categories and Subject Descriptors

G.2.2 [**Graph Theory**]: [Graph Algorithms]

General Terms

Algorithms, Theory

*Partially supported by the IBM Hybrid Multicore Center of Excellence, IIIT Hyderabad.
†Partially supported by NSF Grant CCF 0915543.

PODC'11, June 6–8, 2011, San Jose, California, USA.

Keywords

Graph coloring, distributed algorithms, symmetry breaking, oriented graphs

1. INTRODUCTION

In this paper we answer the following question: if we have only a few rounds at our disposal, how many colors might a distributed algorithm need for coloring a graph? More specifically, we assume that we have a budget of about k rounds, for k typically much smaller than $\Theta(\log n)$, and our goal is to design distributed algorithms that run in $O(k)$ rounds and use as small a color palette as possible. We expect the number of colors to be an increasing function of graph parameters such as n (number of vertices), and Δ (maximum vertex degree), and a decreasing function of k. Recently this line of investigation, that studies the interplay between the number of rounds k and the quality of the solution computed by a k-round algorithm has been quite fruitful for classical combinatorial optimization problems such as minimum dominating set, facility location, and vertex cover [13, 18, 22]. Distributed algorithms that use very few rounds are important for wireless ad hoc and sensor networks that operate in environments characterized by heavy churn. In such settings, it is important to arrive at a feasible solution very quickly, even if the solution is poor relative to the optimal solution, so that higher-order tasks such as routing and scheduling are minimally affected.

Graph coloring is one of the most widely studied problems in distributed computing (not to mention, graph theory), not only because of its myriad applications to resource allocation and scheduling, but also because graph coloring nicely abstracts fundamental challenges of distributed computing such as contention resolution and symmetry breaking. Distributed graph coloring has a rich history starting with Linial's seminal work [15]. Significant advances continue to be made in the area with two ground breaking papers appearing in PODC 2010, one advancing the state of the art for *randomized* distributed coloring [23] and, the other [2] solving a 20 year old question due to Linial on the existence of a *deterministic* distributed graph coloring algorithm that uses $o(\Delta^2)$ colors, yet runs in $O(\text{polylog}(n))$ rounds. The most well-known distributed graph coloring algorithm is randomized and is based on Luby's maximal independent set (MIS) algorithm [16]. This algorithm yields a $(\Delta + 1)$-coloring in $O(\log n)$ rounds with high probability (w.h.p.). If one wants to use fewer rounds, the algorithm of Kothapalli et al. [11] computes an $O(\Delta)$-coloring in $O(\sqrt{\log n})$ rounds w.h.p.; this was the first evidence that

$O(\Delta)$-coloring is possible in sublogarithmic rounds. If one wants to use even fewer colors, the best results are from the recent work of Schneider and Wattenhofer [23], where randomized algorithms that run in $O(\log\log n)$ rounds, $O(\log^* n)$ rounds, and $O(1)$ rounds are described. See Table 1 for a summary of these results.

Table 1: Fewest colors that distributed coloring algorithms with various running times $k = O(\log n)$ use. Here c is some positive integer constant and $\log^{(c)} n$ is obtained by applying the $\log$ function c times to n.

Running time	Number of colors	Source
$O(\log n)$	$\Delta + 1$	[16]
$O(\sqrt{\log n})$	$O(\Delta)$	[11]
$O(\log\log n)$	$O(\Delta + \log n)$	[23]
$O(\log^* n)$	$O(\Delta + \log^{1+1/\log^* n} n)$	[23]
$O(1)$	$O(\Delta \log^{(c)} n + \log^{1+1/c} n)$	[23]

In our work, we focus on *arboricity*, rather than on the maximum degree Δ, as a parameter for measuring the number of colors used. Let the *density* of a graph $G = (V, E)$, $|V| \geq 2$, be the ratio $\lceil |E|/(|V| - 1) \rceil$. Let the density of a single-vertex graph be 1. The *arboricity* of a graph $G = (V, E)$, denoted $a(G)$, can be defined as $a(G) := \max\{$ density$(G') \mid G'$ is a subgraph of $G\}$. By the celebrated Nash-Williams decomposition theorem [20], the arboricity of a graph is exactly equal to the minimum number of forests that its edge set can be decomposed into. The arboricity of a graph seems to more finely calibrate (as compared to Δ) the number of colors a graph might need. For example, graphs with constant arboricity – planar graphs are a simple example – can be easily colored by a sequential greedy algorithm with constant number of colors, despite having vertices with arbitrarily high degree. In fact, it is not hard to see that for a graph with constant arboricity, there is a distributed $O(1)$-coloring algorithm that runs in $O(\log n)$ rounds (see [1] for a far more general result). The general question we ask is this: if we want to color a graph in a distributed fashion, using $k \leq \log n$ rounds, what is the minimum number of colors we need, expressed as a function of $a(G)$. A specific instance of this question might be: how many colors might we need to color constant-arboricity graphs if we have $k = \sqrt{\log n}$ rounds at our disposal. It is tempting to wonder if an $O(1)$-coloring of constant-arboricity graphs in $O(\sqrt{\log n})$ rounds is possible. Unfortunately, a lower bound due to Linial [15], and extended by Barenboim and Elkin [1], quickly dashes these hopes.

THEOREM 1.1 (**Barenboim and Elkin** [1]). *Given a graph with n vertices and with arboricity a and a parameter $q = O(\sqrt{n}/a^2)$, an $O(a^2 \cdot q)$-coloring requires $\Omega(\frac{\log n}{\log(aq)} + \log^* n)$ time.*

The lower bound can be restated in a language more convenient to us as follows.

THEOREM 1.2. *For any k and $a \leq n^{1/2-1/k}$, coloring an n-vertex arboricity-a graph with $O(a \cdot n^{1/k})$ colors needs $\Omega(k)$ rounds.*

From this lower bound it is immediate that an $O(a)$-coloring for graphs with arboricity a cannot be computed in $o(\log n)$ rounds. In this paper we present algorithms that show that this lower bound can be matched exactly for a wide range of values of k.

1.1 Our Techniques and Results

One of our main technical contribution is in extending the approach of Kothapalli et al. [11] in two directions. This paper [11] starts by assuming that the edges of the given network G have been *oriented*, i.e., each undirected edge $\{u, v\}$ is replaced by the directed edge (u, v) or (v, u). Each vertex starts with a palette of size $O(\Delta)$ and runs a "Luby-like" coloring algorithm in which it only pays attention to color choices made by its out-neighbors. By a "Luby-like" algorithm we mean one that has the following general form: each vertex makes a tentative color at random from its palette; then the vertex makes this choice permanent if none of its neighbors have made the same tentative choice; finally, the vertex updates its color palette by deleting all colors that have been permanently chosen by neighbors in the current round. There are many different variants on this general theme. For example, Grable and Panconesi [9] use a version of this algorithm in which in each round each vertex first decides probabilistically whether to participate in that round or not. For more background on Luby-like algorithms, we refer the reader to [7]. The aforementioned algorithm of Kothapalli et al. [11] can be viewed as yet another variant, one in which each vertex resolves contention for colors only with out-neighbors. Kothapalli et al. [11] show that w.h.p. after $O(\sqrt{\log n})$ rounds the vertices that are still color-free induce a subgraph in which the maximum length of an oriented path is $O(\sqrt{\log n})$. If the initial edge orientation of the input graph is acyclic then the subgraph induced by the color-free vertices can be colored in a deterministic, level-by-level fashion in $O(\sqrt{\log n})$ rounds. An acyclic orientation is easy to obtain, for example, by using vertex IDs and therefore the algorithm of Kothapalli et al. [11] colors an arbitrary graph in $O(\sqrt{\log n})$ rounds using $O(\Delta)$ colors.

In one direction we derive and analyze a version of the Kothapalli et al. algorithm that runs in $O(k)$ rounds, for any k, instead of in $O(\sqrt{\log n})$ rounds. In another direction, we show how this algorithm could use only $O(\Delta_o)$ colors (under certain circumstances) instead of $O(\Delta)$ colors, where Δ_o is an upper bound on the out-degree of vertices in the oriented graph G. Managing with $O(\Delta_o)$ colors requires a fundamental rethinking of the Kothapalli et al. algorithm and more generally "Luby-like" coloring algorithms. This is because a vertex can no longer respond to colors chosen by in-neighbors by deleting these colors from its palette. Such deletions may cause the palette of a vertex to empty out rapidly since the number of in-neighbors of the vertex may be quite large relative to its palette size (which is proportional to its out-degree). We deal with this using several new techniques. First, we allow the construction of an improper coloring, which is then rectified by a "Palette Inflation" technique. In order to keep the number of new colors introduced by Palette Inflation down, we use techniques from the distributed computation of Brooks-Vizing colorings due to Grable and Panconesi [9]. These techniques are designed for sparse, i.e., triangle-free graphs, but we show how despite the presence of triangles, we get the appropriate

behavior due to edge-orientations. In our first main result, we prove the following theorem.

THEOREM 1.3. *Let G be an edge-oriented n-vertex graph with maximum vertex out-degree Δ_o.*

(i) For any positive integer k, G can be colored using $O(k \cdot \Delta_o^{1+1/k} \cdot n^{2/k^2})$ colors in $O(k)$ rounds.

(ii) If $k \geq 2\log\log n$ and $\Delta_o = \Omega(\max\{k \cdot n^{2/k^2} \cdot \log^{1+1/k} n, 2^k\})$, G can be colored using $O(\Delta_o)$ colors in $O(k)$ rounds.

(iii) If $k \geq 2\log\log n$ and $\Delta_o = \Omega(\max\{k \cdot \log^{1+1/k} n, 2^k\})$, G can be colored using $O(\Delta_o \cdot n^{2/k^2})$ colors in $O(k)$ rounds.

(iv) If $k < 2\log\log n$ and $\Delta_o \geq 2^{2^k} \cdot \log n$, then G can be colored using $O\left(n^{2/k^2} \cdot \Delta_o^{1+1/k} \cdot \frac{1}{2^{2^k}}\right)$ colors in $O(k)$ rounds.

Item (i) in the above theorem is our basic result, while items (ii)-(iv) are improvements over (i) for certain ranges of values of k and Δ_o. Since for any edge orientation of a graph $\Delta_o \leq \Delta$, all of the above results hold when Δ_o is replaced by Δ. However, it is worth noting that our result is incomparable to results (e.g., [23]) that use $\Theta(\Delta)$ or more colors. Our result may be relatively weak when $\Delta_o = \Delta$, but it is designed to be particularly useful when $\Delta_o \ll \Delta$. In such situations, our algorithm can rapidly construct colorings with far fewer colors than $\Theta(\Delta)$, something that is not possible with earlier results.

Item (ii) tells us that for "high" out-degree graphs an $O(\Delta_o)$ coloring is possible in very few rounds. It is instructive to consider specific instances of this result for two two extreme values of k. At $k = \sqrt{\log n}$, one can simplify the lower bound on Δ_o to $\Omega(2^{\sqrt{\log n}})$ and therefore this instance of our result is an $O(\Delta_o)$-coloring in $O(\sqrt{\log n})$ rounds when $\Delta_o = \Omega(2^{\sqrt{\log n}})$. At the other end of the spectrum, when $k = 2\log\log n$, Δ_o can be bounded below by $\Omega(n^{c/(\log\log n)^2})$ for some constant $c > 2$ and therefore this instance of our result is an $O(\Delta_o)$-coloring in $O(\log\log n)$ rounds when $\Delta_o = \Omega(n^{c/\log\log n})$. Note that items (ii) and (iii) are identical for $k = \Theta(\sqrt{\log n})$ and this result — $O(\Delta_o)$-coloring in $O(\sqrt{\log n})$ rounds — subsumes the result of Kothapalli et al. [11] for large enough Δ_o and is much stronger whenever $\Delta_o \ll \Delta$.

The above theorem is particularly useful in settings in which "good" edge orientations are easily obtained. One such setting is identified by Barenboim and Elkin [1], who show that the edges of a graph G with arboricity a can be oriented in $O(k)$ rounds so that $\Delta_o = O(a \cdot n^{1/k})$. Our final result is obtained by using the Barenboim-Elkin orientation as a first step followed by the algorithm corresponding to Theorem 1.3.

THEOREM 1.4. *Let G be an n-vertex graph with arboricity a.*

(i) For any k, $2\log\log n \leq k \leq \sqrt{\log n}$, G can be colored with $O(a \cdot n^{1/k})$ colors in $O(k)$ rounds.

(ii) For any $k < 2\log\log n$, G can be colored using $O\left(a^{1+1/k} \cdot n^{\frac{1}{k}+\frac{3}{k^2}} \cdot \frac{1}{2^{2^k}}\right)$ colors in $O(k)$ rounds.

Item (i) in the above theorem provides an upper bound that exactly matches the lower bound in Theorem 1.2, for values of k sandwiched between $\Omega(\log\log n)$ and $O(\sqrt{\log n})$. The orientation of Barenboim and Elkin [1] mentioned above can be obtained via deterministic means and is similar to decomposition techniques used by Czygrinow et al. [3, 4, 6]. As pointed out by Barenboim and Elkin [1], using these decomposition techniques, one can obtain an $O(q \cdot a^2)$-coloring on an n-vertex graph with arboricity a and parameter $q \geq 1$ in $O(\frac{\log n}{\log q} + \log^* n)$ rounds. For $k = \Omega(\log^* n)$ this translates to an $O(a^2 \cdot n^{1/k})$-coloring algorithm in $O(k)$ rounds. For constant values of the arboricity a, this result obtained via a *deterministic* algorithm matches our result (Item (i) of Theorem 1.4) and also matches the lower bound. However, for larger values of a the power of randomization seems to come into play and as the value of a grows, the coloring promised by Theorem 1.4 uses a factor-a fewer colors relative to the result of Barenboim and Elkin [1].

1.2 Related Work

There is a vast body of literature on distributed graph coloring and we do not review this work here, instead referring the reader to "Related Work" sections in [2, 23]. Our work is inspired by two recent trends in distributed algorithms, one that focuses on graph arboricity as a parameter of interest and another that seeks to design k-round algorithms for very small k, with a possible concomittant loss in the quality of the solution.

The work of Barenboim and Elkin [1] on coloring graphs with bounded arboricity has already been discussed. In addition to the coloring problem, this paper also considers the maximal independent set (MIS) problem and presents an algorithm that runs in $O(\frac{\log n}{\log\log n})$ rounds for graphs with constant arboricity. Also worth mentioning is a recent paper by Lenzen and Wattenhofer [14] on distributed approximation algorithms for the minimum dominating set problem on bounded arboricity graphs. The algorithm in this paper computes an $O(a^2)$-approximation for MDS in $O(\log n)$ rounds on graphs with arboricity a. Another example is the work of Czygrinow et al. [5] on distributed maximum matching algorithms on bounded arboricity graphs.

There is recent interest in designing distributed algorithms that run in a fixed number, say k, rounds and compute non-trivial solutions despite having a tiny budget of communication rounds. Kuhn and Wattenhofer [13] seem to have been the first to ask whether one can design a k round distributed algorithm that can compute a non-trivial approximation to MDS even when $k = O(1)$. They answered this question in the affirmative and their approach of using LP relaxation techniques was subsequently generalized and extended to other problems [12, 18]. More recently, Pandit and Pemmaraju [21, 22] have shown that such k-round algorithms for MDS and facility location can also be derived using primal-dual techniques, sometimes yielding better results.

2. PRELIMINARIES

2.1 Model and Notations

Our algorithms run on the standard distributed computing model in which a graph G represents the network, with vertex set $V = V(G)$ representing the computational enti-

ties and edge set $E = E(G)$ representing the communication links. Computation proceeds in synchronous rounds, and in each round, each node can, (i) receive a (possibly) different message from each neighbor, (ii) perform some finite amount of local computation, and (iii) send a (possibly) different message to each neighbor. We assume that all the communication links are undirected and hence bidirectional. Our algorithms will usually assume that the edges of G are *oriented*, even though the underlying network consists of bidirectional links. When the edges of the graph G are oriented, for each $v \in V$, we denote by $d_o(v)$, the *out-degree* of v, i.e., the number of neighbors w of v such that the edge $\{v, w\} \in E$ is oriented from v to w. We call such neighbors as the *out-neighbors* of v. We define $\Delta_o(G) := \max_{v \in V} d_o(v)$.

2.2 The Oriented Graph Coloring Algorithm

Our algorithms and the corresponding analysis use edge-orientation in a critical way. Consider two neighbors v and w such that w is an out-neighbor of v. Consider a Luby-type coloring algorithm in which each node independently and tentatively picks a color from its palette and then coordinates with neighbors to resolve conflicts in color choices. In a round of this algorithm, suppose that both nodes v and w tentatively choose the same color c. After one round of communication, node v detects an out-neighbor with a conflicting color choice and uncolors itself. Node w however, ignores choices made by in-neighbors and can finalize its choice of color c provided it has no out-neighbor that has chosen color c in this round. Thus, the resolution of color conflicts is different from what happens in standard distributed coloring algorithms (see for example [16, 10]) in which both v and w would have remained uncolored. In fact, nodes ignore their in-neighbors to the extent that they do not even update their color palette in response to in-neighbors making a permanent color choice. Specifically, suppose that in some round, node v permanently colors itself c. Color c is not deleted from w's palette and in a subsequent round w may decide to color itself c also, thereby leading to an improper coloring (at least temporarily). Our main idea is to use orientation to quickly produce a partial, improper coloring and then repair it using a separate algorithm. Our algorithm `OrientedRandColor` for vertex coloring oriented graphs is shown in Figure 1. At an arbitrary node u, this algorithm takes two arguments, k, a positive integer specifying the number of rounds the algorithm should run for, and C_u, an initial palette of colors. The algorithm is similar to well-known randomized distributed graph coloring algorithms [17, 10], the main difference being in how color conflicts are resolved. Note that in Steps 3 and 7 node u sends its color choice only to in-neighbors because its out-neighbors are not interested in its actions. Also, in Step 4 node u only considers color choices made by out-neighbors in figuring out whether to color itself permanently. When `OrientedRandColor` terminates, the coloring may be *partial*, i.e., not all nodes may have been assigned a color because k rounds may not suffice to color all vertices in G. Furthermore, as mentioned earlier, the coloring may also be *improper*, i.e., neighbors in G may be assigned the same color.

3. A SLIGHTLY LOOSE ANALYSIS

All edge-orientations we work with are assumed to be *acyclic*. This is mainly for ease of exposition. In subsequent remarks we will point out that the presence of oriented cycles is not a problem, provided the cycles are not small. In Lemmas 3.1 and 3.3 we prove properties of the partial, improper coloring constructed by `OrientedRandColor` and later show how to repair this coloring quickly.

3.1 A Partial Improper Coloring

We first show a useful property of neighbors that are assigned the same color.

LEMMA 3.1. *If `OrientedRandColor` assigns the same color to two neighbors in G, then it does so in different rounds.*

PROOF. Suppose that u and v are neighbors in G with edge $\{u, v\}$ oriented from u to v. Further suppose that u and v are both assigned the same color, say c, by `OrientedRandColor`. Consider the round t in which u was permanently assigned c. Node v could not have been assigned color c in any round earlier than t because that would mean that c is absent from u's color palette at the beginning of round t. Node v could not have been assigned color c in round t, because that would mean that both u and v picked color c tentatively in round t and in such a situation node u would defer to its out-neighbor and remain uncolored. Therefore, node v is assigned a color permanently in a round that comes after round t. □

In Lemma 3.3, we show that w.h.p. there are no long, oriented, uncolored paths after `OrientedRandColor` has terminated. But before we get to this lemma, we need a technical lemma that helps us upper bound the joint probability of certain, possibly dependent, events.

LEMMA 3.2. *Let S be a subset of vertices of G that have remained uncolored in the first $i-1$ rounds of `OrientedRandColor`. Let E_S denote the event that all vertices in S remain uncolored in round i and let E_v denote the event that a vertex $v \in S$ has remained uncolored in round i. Then,*

$$\Pr(E_S) \leq \prod_{v \in S} \Pr(E_v).$$

PROOF. Let $S = \{v_1, v_2, \ldots, v_t\}$. We can write $\Pr(E_S)$ as

$$\Pr(E_S) = \prod_{v_j \in S} \Pr(E_{v_j} | \wedge_{\ell=1}^{j-1} E_{v_\ell}).$$

We will now show that

$$\Pr(E_{v_j} | \wedge_{\ell=1}^{j-1} E_{v_\ell}) \leq \Pr(E_{v_j}), \quad (1)$$

which will yield the inequality claimed in the theorem.

Define a *configuration* as the tentative color choice made in round i by all the vertices in G that did not color themselves in rounds 1 through $i-1$. Fix a configuration C and for any $v \in S$ let $N_o^C(v)$ denote the set of out-neighbors of v that have been assigned the same color as v in the configuration C. The set of all configurations that satisfy $\wedge_{\ell=1}^{j-1} E_{v_\ell}$ can be partitioned into two subsets:

(i) Those configurations C in which for every v_ℓ, $1 \leq \ell \leq j-1$, $N_o^C(v) - N_o(v_j) \neq \emptyset$

(ii) The rest of the configurations, i.e., those configurations C in which for some v_ℓ, $1 \leq \ell \leq j-1$, $N_o^C(v_\ell) \subseteq N_o(v_j)$.

With respect to the configurations of Type (i), the probability of E_{v_j} is equal to its unconditional probability. It

Algorithm `OrientedRandColor`(k, C_u)
Comment: C_u is node u's palette of colors
Begin-Algorithm
1. **for** i := 1 **to** k **do**
2. Node u chooses a color c_u from C_u, uniformly at random.
3. Node u sends c_u to all of its uncolored in-neighbors
4. **if** every color received by node u from an out-neighbor is distinct from c_u, **then**
5. u makes c_u its permanent color. Otherwise node u remains uncolored.
6. **if** u is permanently colored in Step 5 **then**
7. u sends it color choice to all uncolored in-neighbors
8. Node u deletes from C_u all colors assigned permanently to out-neighbors
9. **end-for**
End-Algorithm

Figure 1: Coloring oriented graphs via a Luby-like algorithm.

is possible that in some configurations of Type (ii), two or more out-neighbors of v_j are required to have the same color. Therefore, with respect to configurations of Type (ii), the probability of E_{v_j} is at most the unconditional probability of E_{v_j}. Combining these observations about both types of configurations, we obtain Inequality (1). □

In the rest of the paper we will use Δ_o as a shorthand for $\Delta_o(G)$ whenever G is apparent from the context.

LEMMA 3.3. *Suppose that initially* $|C_u| \geq \Delta_o^{1+1/k} \cdot n^{2/k^2}$ *for all nodes* u. *After* `OrientedRandColor` *has terminated, w.h.p., every oriented uncolored path in* G *has length at most* k.

PROOF. Fix an oriented path $P = (v_1, v_2, \ldots, v_{k+1})$ of length $k+1$. For any integer j, $1 \leq j \leq k+1$ and nonnegative integer i, let $E_{v_j,i}$ denote the event that vertex v_j remains uncolored after the first i rounds and let $E_{P,i}$ denote the event that *all* of the vertices in path P remain uncolored after the first i rounds. We first compute an upper bound on the conditional probability $\Pr(E_{v_j,i}|E_{P,i-1})$.

$E_{P,i-1}$ implies that v_j is not colored at the end of round $i-1$. Suppose that v_j's palette has lost some t colors over the course of the first $i-1$ rounds. This means that the number of uncolored out-neighbors of v_j has shrunk by at least t, i.e., $d_o(v_j) - t$ is an upper bound on the number of uncolored out-neighbors of v_j after the first $i-1$ rounds. Then the probability that v_j is not colored in round i (conditioned on $E_{P,i-1}$) is equal to the probability that it picks the same color as an out-neighbor. And this is at most

$$\frac{d_o(v_j) - t}{|C_{v_j}| - t} \leq \frac{d_o(v_j)}{|C_{v_j}|} \leq \frac{\Delta_o}{\Delta_o^{1+1/k} \cdot n^{2/k^2}} \leq \frac{1}{\Delta_o^{1/k} \cdot n^{2/k^2}}.$$

Let E_P be shorthand for the event $E_{P,k}$. Then,

$$\Pr(E_P) = \prod_{i=1}^{k} \Pr(E_{P,i}|E_{P,i-1}).$$

Using the definition of $E_{P,i}$ and Lemma 3.2, we obtain

$$\begin{aligned} \Pr(E_{P,i}|E_{P,i-1}) &= \Pr(\wedge_{j=1}^{k+1} E_{v_j,i}|E_{P,i-1}) \\ &\leq \prod_{j=1}^{k+1} \Pr(E_{v_j,i}|E_{P,i-1}) \\ &\leq \left(\frac{1}{\Delta_o^{1/k} \cdot n^{2/k^2}}\right)^{k+1}. \end{aligned}$$

Hence,

$$\Pr(E_P) \leq \left(\frac{1}{\Delta_o^{1/k} \cdot n^{2/k^2}}\right)^{k(k+1)} \leq \frac{1}{\Delta_o^{k+1} \cdot n^2}.$$

There are at most $n \cdot \Delta_o^{k+1}$ oriented paths of length $k+1$ and therefore, using the union bound we see that the probability that there exists an uncolored, oriented path of length $k+1$ is at most

$$n \cdot \Delta_o^{k+1} \cdot \frac{1}{\Delta_o^{k+1} \cdot n^2} = \frac{1}{n}.$$

□

3.2 Repairing the Coloring

Even though the coloring constructed by `OrientedRandColor` is partial and improper, the properties established in Lemmas 3.1 and 3.3, allow us to "repair" the coloring in k additional rounds with an accompanying inflation of the palette by factor k.

Palette Inflation.

Let c_u denote the color assigned by `OrientedRandColor` to node u. We recolor each node u with the color (c_u, t_u), where t_u is an integer satisfying $1 \leq t_u \leq k$ that denotes the round in which u was permanently assigned a color by `OrientedRandColor`. Since Lemma 3.1 tells us that any two neighbors which are assigned the same color by `OrientedRandColor` are assigned colors in distinct rounds, we see that neighbors in G now have distinct colors. Furthermore, using the round numbers in this way to disambiguate the colors inflates the palette size by a factor of k.

Deterministic Completion.

Lemma 3.3 allows us to color the nodes not colored by `OrientedRandColor` by using a simple deterministic algorithm. In the each round, all uncolored nodes with no uncolored out-neighbors are assigned an arbitrary color from their palette. Note that this step is well-defined as long as the initial palette size $|C_u|$ is at least $\Delta_o + 1$ for all vertices u. By definition, the set of nodes considered in each round forms an independent set and therefore no coordination is needed while making color choices. Lemma 3.3 tells us that every oriented uncolored path has length at most k. After one round of the Deterministic Completion algorithm, every oriented uncolored path in G has length at most $k-1$. It is easy to see via an inductive argument that after t rounds, $1 \leq t \leq k$, of the Deterministic Completion algorithm, every oriented uncolored path in G has length at most $k-t$. Hence, `OrientedRandColor` augmented by Palette Inflation and followed by Deterministic Completion leads to the following theorem.

THEOREM 3.4. *Let k be any positive integer. Given a graph G with an acyclic orientation with maximum out-degree Δ_o, a $(k\cdot\Delta_o^{1+1/k}\cdot n^{2/k^2})$-coloring of G can be achieved in $2k$ rounds.*

This result appears as item (i) in Theorem 1.3 in the "Introduction."

Remark.

We use the acyclic orientation of G in the Deterministic Completion Step. Such an orientation guarantees that the subgraph induced by uncolored vertices does not contain any oriented cycles and therefore this subgraph can be processed in a level-by-level fashion. It is worth noting that an acyclic orientation is stronger than what we need; it would suffice to have an orientation that did not have any cycles of length k or less.

4. A TIGHTER ANALYSIS

We now use a tighter analysis to improve on Theorem 3.4. We first assume that $k \geq 2\log\log n$ (in Section 4.1) and in this setting we obtain two results: (i) for large enough Δ_o, we shave off the $\Delta^{1/k}\cdot n^{2/k^2}$-term from the number of colors in Theorem 3.4 and show that we can obtain a $k\cdot\Delta_o$-coloring in $2k$ rounds, and (ii) for smaller values of Δ_o, we shave off the $\Delta_o^{1/k}$-term from the number of colors in Theorem 3.4 and obtain a $k\cdot\Delta_o\cdot n^{2/k^2}$-coloring in $2k$ rounds. Subsequently (in Section 4.2) we deal with the case when $k < 2\log\log n$.

4.1 When $k \geq 2\log\log n$

We use "Phase I" to denote the first $k/2$ rounds of the `OrientedRandColor` algorithm and first bound the "effective" maximum out-degree of vertices in G after Phase I in Section 4.1.1. We use "Phase II" to denote the last $k/2$ rounds and analyze this phase separately in Section 4.1.2.

4.1.1 Phase I: Reducing Δ_o Quickly

We define the *effective out-degree* of a vertex v as the number of uncolored out-neighbors of v. In the following analysis, we show that the effective out-degree of every vertex in G decays quite rapidly (at an inverse doubly exponential rate) and this allows us to claim a lot of progress even in $O(\log\log n)$ rounds. This analysis is similar to that used in certain balls-and-bins problems (see Problem 3.4 in [19]) and has been used in other papers [9, 11]. The analysis also requires the use large deviation inequalities due to Grable [8]. We start with the following lemma that shows the progress made by the algorithm in Phase I. The proof of this lemma is similar to the proof of [11, Lemma 4.1].

LEMMA 4.1. *Suppose that $|C_u| \geq 3\Delta_o$ for all $u \in V$. At the end of Phase I, the effective out-degree of every vertex in G is at most $\min\{\log n, \Delta_o\}$ w.h.p.*

PROOF. Let us denote by $d_o^i(v)$ the effective out-degree of v at the start of round i. Let $d_o^i = \max_v d_o^i(v)$ and let C_u^i denote u's palette at the beginning of round i . Then,

$$\begin{aligned} P_u^i &:= \Pr(u \text{ does not get colored in round } i) \\ &\leq \sum_{j=1}^{d_o^i(v)} \frac{1}{|C_u^i|} \\ &\leq \frac{d_o^i}{2\Delta_o}. \end{aligned}$$

Let $N_o^i(v)$ denote the set of uncolored out-neighbors of vertex v at the beginning of round i. Using the above upper bound on P_u^i we obtain

$$\mathrm{Ex}(d_o^{i+1}(v)) = \sum_{u\in N_o^i(v)} P_u^i \leq \frac{(d_o^i)^2}{2\Delta_o}.$$

Using large deviation arguments from [8], it can be shown that $d_o^{i+1}(v)$ exceeds its expectation by more than $\sqrt{cd_o^i\log n}$ (for some constant c) with probability less than $1/n^2$. This means that w.h.p. the d_o^i-values satisfy the following recurrence:

$$d_o^{i+1} \leq \frac{(d_o^i)^2}{2\Delta_o} + \sqrt{cd_o^i\log n}$$

Thus, w.h.p., $d_o^i \leq \frac{d_o^1}{2^{2^i}} = \frac{\Delta_o}{2^{2^i}}$. □

The above lemma guarantees that after Phase I, i.e., the first $k/2$ rounds of `OrientedRandColor`, every vertex has at most $\log n$ uncolored out-neighbors.

4.1.2 Phase II : Breaking Long Uncolored Paths

Suppose that all nodes start Phase II with palettes of size at least $t\cdot\Delta_o$ for a parameter t to be specified later. As in Lemma 3.3, we argue that w.h.p. after Phase II, every oriented path of length at least $(k+1)$ has at least one colored node. However, in the current lemma, the fact that Phase I has caused a significant reduction in the effective out-degree plays a critical role.

LEMMA 4.2. *Suppose that initially $|C_u| \geq t\cdot\Delta_o \geq n^{2/k^2}\cdot\log^{1+1/k} n$ for all nodes u. After `OrientedRandColor` has terminated, w.h.p., every oriented uncolored path in G has length at most k.*

PROOF. This proof is similar to the proof of Lemma 3.3 and so we borrow notation from that proof and only provide a sketch here.

Fix an oriented path $P = (v_1, v_2, \ldots, v_{k+1})$ of length $k+1$. Then,

$$\Pr(E_{v_j,i}|E_{P,i-1}) \leq \frac{d_o(v_j)}{|C_{v_j}|} \leq \frac{\log n}{t\cdot\Delta_o}.$$

Using this and following the calculations in the proof of Lemma 3.3, we see that

$$\Pr(E_P) \leq \left(\frac{\log n}{t\cdot\Delta_o}\right)^{k(k+1)}.$$

There are at most $n\cdot\log^{k+1} n$ oriented paths of length $k+1$ and therefore, using the union bound we see that the probability that there exists an uncolored, oriented path of length $k+1$ is at most

$$n\cdot\log^{k+1} n\cdot\left(\frac{\log n}{t\cdot\Delta_o}\right)^{k(k+1)} \tag{2}$$

For this expression to be at most $1/n$, we require that

$$(t\cdot\Delta_o)^{k(k+1)} \geq n^2\log^{(k+1)^2} n.$$

The requirement in the lemma that $t\cdot\Delta_o \geq n^{2/k^2}\cdot\log^{1+1/k} n$ ensures this. □

4.1.3 Completing the Coloring

Since Phase II requires a palette of size at least $t\Delta_o \geq n^{2/k^2}\cdot\log^{1+1/k} n$, we can distinguish between two cases for Phase II.

- When $\Delta_o \in \Omega(n^{2/k^2} \cdot \log^{1+1/k} n)$, we can choose $t = O(1)$.
- When $\Delta_o \in \Omega(\log^{1+1/k} n)$, we have to choose $t = O(n^{2/k^2})$.

Phase I needs an additional $O(\Delta_o)$ colors. At the end of Phase II, we have a situation that is similar to what was considered by Lemma 3.3. As in the proof of Theorem 3.4, we can use Palette Inflation and Deterministic Coloring to arrive at a proper coloring. Putting everything together, we have the following theorems.

THEOREM 4.3. *Let G be a graph with an acyclic orientation satisfying $\Delta_o \in \Omega(n^{2/k^2} \cdot \log^{1+1/k} n)$. Then G can be properly colored in $2k$ rounds, for any $k \geq 2 \log\log n$ rounds, using $O(k\Delta_o)$ colors.*

THEOREM 4.4. *Let G be a graph with an acyclic orientation satisfying $\Delta_o \in \Omega(\log^{1+1/k} n)$. Then G can be properly colored in $2k$ rounds, for any $k \geq 2 \log\log n$ rounds, using $O(k\Delta_o \cdot n^{2/k^2})$ colors.*

Theorems 4.3 and 4.4 are improvements over Theorem 3.4 for wide ranges of values of Δ_o and k, however they are a factor-k away from what was promised in Theorem 1.3.

4.2 When $k < 2 \log\log n$

When $k < 2 \log\log n$, we can use calculations that are very similar to those above, to prove the following theorem.

THEOREM 4.5. *Given a graph G with an acyclic orientation satisfying $\Delta_o \geq 2^{2^k} \cdot \log n$, the vertices of G can be properly colored w.h.p. in $O(k)$ rounds, for any $k < 2 \log\log n$ rounds, using $O(k \cdot n^{2/k^2} \cdot \Delta_o^{1+1/k} \cdot \frac{1}{2^{2^k}})$ colors.*

PROOF. Our proof for this theorem follows the proof of Theorem 4.3. Since $\Delta_o > 2^{2^k} \cdot \log n$, using the argument of Lemma 4.1 we can show that at the end of $k/2$ rounds of Phase I the effective out-degree of every vertex in G is at most $\Delta_o/2^{2^k}$. During this phase, vertices use a palette of size $3\Delta_0$ as in the Proof of Lemma 4.1. Now, suppose that vertices start Phase II of `OrientedRandColor` with $t \cdot \Delta_o$ colors. Using calculations similar to those in Lemma 4.2, we can see that at the end of Phase II, w.h.p., every uncolored path has length at most k, provided

$$n \cdot \left(\frac{\Delta_o}{2^{2^k}}\right)^{k+1} \cdot \left(\frac{\Delta_o/2^{2^k}}{t\Delta_o}\right)^{k(k+1)} \leq \frac{1}{n} \qquad (3)$$

Setting $t = n^{\frac{2}{k^2}} \Delta_o^{\frac{1}{k}} \cdot \frac{1}{2^{2^k}}$ guarantees that Phase II will complete successfully. Palette Inflation to fix possible improprieties in the coloring causes the extra factor-k to appear in the palette size. Deterministic completion leads to an additional k rounds. The total number of colors required is $O(kt\Delta_o)$ which is $O(k \cdot n^{2/k^2} \cdot \Delta_o^{1+1/k} \cdot \frac{1}{2^{2^k}})$. □

5. AVOIDING PALETTE INFLATION VIA BROOKS-VIZING-TYPE COLORINGS

Applying the Palette Inflation technique from Section 3.2 results in the color palettes being inflated by a factor k for algorithms that run in $O(k)$ rounds. In this section we show how to get rid of this extra factor k in the palette size.

Consider the result in Theorem 4.3 which shows how to color G using $O(k\Delta_o)$ colors in $2k$ rounds, when Δ_o is sufficiently large. One approach to removing the extra factor k from the number of colors and obtaining an $O(\Delta_o)$-coloring is to try and use only $O(\Delta_o/k)$ colors prior to Palette Inflation; then when the palette is inflated by a factor k, we end up with an $O(\Delta_o)$-coloring. In this section, we show how to obtain an $O(\Delta_o/k)$-coloring of G that is improper, but can be rectified via the use of Palette Inflation. Our result employs techniques due to Grable and Panconesi used in the distributed computation of Brooks-Vizing colorings [9] in sparse, i.e., *triangle-free* graphs. Our graphs are not triangle-free, but we can still use certain parts of the Grable-Panconesi approach. In particular, the fact that edges are oriented plays a key role in ensuring that our claims go through despite the presence of triangles in the graph. The main result of Grable and Panconesi [9] is the following.

THEOREM 5.1 (GRABLE AND PANCONESI [9]). *Let G be a Δ-regular triangle-free graph. For any $k = O(\log \Delta)$, G can be colored in $O(k + \frac{\log n}{\log \Delta})$ rounds using $O(\frac{\Delta}{k})$ colors.*

The algorithm that leads to this theorem has two phases: a *dozing phase*, and a *trivial phase*. In both phases the algorithm is essentially similar to Luby's algorithm. Each node starts the dozing phase with the palette $\{1, 2, \ldots, \Delta/k\}$. In each round, each as yet uncolored node chooses to wake up with a probability p and if the node wakes up, it picks a tentative color from amongst the available colors in its palette uniformly at random and independent of the choices of other nodes. Color conflicts are resolved in the usual manner and palettes are updated. The dozing phase runs for $O(k)$ rounds. The key innovation of Grable and Panconesi is to start the wake-up probability p at $1/k$ and then increase it in each round. This ensures that in each round an appropriate fraction of the vertices are competing for the colors in the color palettes. Furthermore, in each round the size of the palette relative to the number of uncolored neighbors of a node grows, allowing p to grow from round to round. The manner in which p increases ensures that p equals 1 in $e \cdot k$ rounds and this brings the algorithm to the trivial phase. For a more detailed description of the algorithm, we refer the reader to [9]. The following lemma [9] describes the progress made during the dozing phase.

LEMMA 5.2 (GRABLE AND PANCONESI [9]). *Under the hypothesis of Theorem 5.1, at the end of the dozing phase involving $e \cdot k$ rounds, the number of uncolored neighbors reduces to $O(\Delta/k)$ w.h.p.*

By modifying the dozing phase suitably for oriented graphs, we show a similar result (see Lemma 5.5) — at the end of $O(k)$ rounds, the number of uncolored *out-neighbors* of every vertex has shrunk to at most Δ_o/k. Once this state is reached, we can use `OrientedRandColor` with a palette size that is a factor $1/k$ smaller.

Let us now trace important steps of the proof of the above lemma from [9]. First define the following quantities: (i) $s_i(u)$ denotes the size of the palette for node u in round i, (ii) $d_i(u, c)$ denotes the number of uncolored neighbors of u which have color c in their palette in round i, and (iii) $D_i(u)$ denotes the number of uncolored neighbors of u at round i

Now consider the following recurrence relations:

$$\begin{aligned} s_{i+1}(u) &= s_i(u) \cdot e^{-1/e}, \\ d_{i+1}(u,c) &= d_i(u)\left(1 - \tfrac{s_i(u)}{d_i(u)}\tfrac{1}{e}\right) \cdot e^{1/e}, \text{ and} \\ D_{i+1}(u) &= D_i(u)\left(1 - \tfrac{s_i(u)}{d_i(u)}\tfrac{1}{e}\right). \end{aligned} \quad (4)$$

Grable and Panconesi [9] show that these recurrence relations respectively describe the behavior of the random variables $s_i(u)$, $d_i(u,c)$, and $D_i(u)$ for all vertices u, rounds i, and colors c. It is first shown that the random variables satisfy these recurrences in expectation and then via the use of concentration inequalities, it is established that the recurrences are also satisfied w.h.p.

To establish the above recurrences, the work of [9] relies on the hypotheses of Lemma 5.2 which require that the graph be (i) triangle-free, and (ii) regular. The triangle-free property of the graph is required for the following reason. To understand the change in the palette size $s_i(u)$ at a node u as it changes from round to round, we need to understand the rate at which colors disappear from u's palette. Fix a round and suppose that at the beginning of the round, u is not yet colored and it has color c in its palette. Let $E_{u,c}$ denote the conditional event (c disappears from u's palette in the round| u does not get colored in the round). Let $N_{u,c}$ be the neighbors of u that are awake in the round and have c in their palette. Then, $\Pr(E_{u,c}) = 1 - \Pr(\forall v \in N_{u,c} : v$ does not get colored c in the round $|u$ does not get colored in the round). Let $F_{v,c}$ denote the event (v does not get colored in the round | u does not get colored in the round). The probability on the right-hand side of the equation for $\Pr(E_{u,c})$ can be expressed as the product $\prod_{v \in N_{u,c}} \Pr(F_{v,c})$ if the events $F_{v,c}$ were independent. And one way to guarantee independence is to ensure that G contains no 3-cycles or 4-cycles (i.e., girth is at least 5). And in fact when the $F_{v,c}$ events are independent, it is not too hard to show that $\Pr(E_{u,c}) \approx 1 - exp(e^{-1})$.

The presence of 4-cycles however does pose a problem since this allows two vertices $v, v' \in N_{u,c}$ to have common neighbors besides u. Grable and Panconesi [9] deal with this problem by showing that the events $F_{v,c}$ are positively correlated and as a result $\Pr[\forall v \in N_{uc} : v$ does not get colored c in the round $|u$ does not get colored in the round] is at least $\prod_{v \in N_{u,c}} \Pr(F_{v,c})$.

As a result, $\Pr(E_{u,c})$ falls relative to its value in the absence of 4-cycles. Also, now (i.e., in the presence of 4-cycles) $\Pr(E_{u,c})$ can be quite different at different vertices u. To fix this, Grable and Panconesi [9] introduce an additional step in their algorithm. Each vertex maintains a complete description of its neighborhood out to distance 2. This allows each vertex u to compute, for each color c in its palette, the natural probability $p_{u,c}$ of decay. This is done before tentative colors are chosen. The round proceeds as before and at the end each color c that was not naturally removed is artificially removed with a certain probability that depends on $p_{u,c}$. This artificially shrinks the palette at each vertex and ensures that the palette sizes remain balanced and decay at the rate specified by the recurrence relation in Equation (4).

In Lemma 5.3 we show that the positive correlation that holds among events $F_{v,c}$ in the absence 3-cycles also holds when we do oriented coloring. Before we state and prove this lemma, we show the Algorithm `DozeOff` in Figure 2, which is the dozing off phase of the Grable-Panconesi algorithm, modified for oriented graphs.

Algorithm `DozeOff`(k, C_u)
Begin-Algorithm
1. $p := 1/k$
2. **for** i := 1 **to** $e \cdot k$ **do**
3. state(u) := `awake` with probability p
4. **if** state(u) =`awake` **then**
5. Node u chooses a tentative color c_u uniformly at random from C_u
6. Node u sends c_u to all of its uncolored in-neighbors
7. **if** every color received by node u from an out-neighbor is distinct from c_u, **then**
8. u makes c_u its permanent color.
9. Otherwise node u remains uncolored.
10. **if** u is permanently colored in Step 8 **then**
11. u sends its color choice to all uncolored in-neighbors
12. Node u deletes from C_u all colors assigned permanently to out-neighbors
13. **end-if**
14. $p := \frac{1}{1/p - 1/e}$
15. **end-if**
16. **end-for**
End-Algorithm

Figure 2: Algorithm `DozeOff`. The argument k satisfies $k = O(\log \Delta_o)$ whereas vertex u's color palette C_u is $\{1, 2, \ldots, \Delta_o/k\}$.

LEMMA 5.3. *Let G be an acyclically oriented graph with out-degree bounded by Δ_o. Fix a round of Algorithm `DozeOff` and consider an uncolored node u and a color c that is available in the palette of u at the beginning of the round. Then,*

$$\Pr(E_{u,c}) \leq 1 - \prod_{i=1}^{k} \Pr(F_{v_i,c})$$

where $N_{u,c} = \{v_1, v_2, \cdots, v_k\}$ denotes the set of uncolored out-neighbors of u that also have the color c in their palette.

PROOF. Consider a round r, an uncolored node u and a color c. We wish to compute the probability that in this round, the color c disappears from the palette of u given that node u does not color. Therefore,

$$\begin{aligned} \Pr(E_{u,c}) &= 1 - \Pr(\forall_{v \in N_{u,c}} v \text{ is not } c\text{-colored in round } r \mid \\ &\qquad u \text{ is not colored in round } r) \\ &= 1 - \textstyle\prod_{i=1}^{k} \Pr(v_i \text{ is not } c\text{-colored in round } r \mid \\ &\qquad u \text{ is not colored, } v_1, \ldots, v_{i-1} \\ &\qquad \text{are not } c\text{-colored in round } r). \end{aligned}$$

Let us denote by C_{v_i} the above conditional event. To compute $\Pr(C_{v_i})$, we first order the v_i's so that for $1 \leq i \leq k$, no out-neighbors of v_i are in the set $\{v_1, v_2, \cdots, v_{i-1}\}$. This is possible as long as the out-neighbors of u do not induce an oriented cycle. The assumption that we are considering an acyclic orientation of G clearly ensures that this is the case. Recall that F_{v_i} is the event that v_i fails to get colored with color c conditioned just on the event that u does not get colored. We now show that the events C_{v_i}, $1 \leq i \leq k$ are positively correlated.

Let us denote by a configuration C the tentative choices of all uncolored neighbors in this round. The set of all configurations $\mathcal{C}$ that satisfy the condition that $u, v_1, v_2, \ldots, v_{i-1}$ are not colored, can be partitioned into two sets $\mathcal{C}_1$ and $\mathcal{C}_2$ as follows. The set $\mathcal{C}_1$ contains all configurations in which

there exists a vertex v_j, $1 \leq j \leq i-1$, so that v_i and v_j share an uncolored out-neighbor w and the tentative choice of w is c. The set $\mathcal{C}_2$ is the remaining set of configurations.

It can be noted that with respect to configurations in $\mathcal{C}_1$, the probability that v_i fails to color with c is 1, as w's tentative choice of c overrides the tentative choice of v_i. In the second set of configurations, the failure of v_i to get colored with c is independent of the events C_{v_i} is conditioned on. Therefore, $\Pr(C_{v_i}) \geq \Pr(F_{v_i})$. Since the above holds for every i, $1 \leq i \leq k$, we have that $\prod_{i=1}^{k} \Pr(C_{v_i}) \geq \prod_{i=1}^{k} \Pr(F_{v_i})$. Due to the above observations, we can say that:

$$\Pr(E_{u,c}) = 1 - \prod_{i=1}^{k} \Pr(E_{v_i}) \leq 1 - \prod_{i=1}^{k} \Pr(F_{v_i}).$$

□

The hypothesis of Lemma 5.2 also requires that the graph be regular. The regularity condition ensures that $d_0(u) = D_0(u) = \Delta$ for all vertices u and this uniform initial condition is critical in ensuring that the recurrence relations shown in Equation (4) hold. In Lemma 5.4, we show how to convert a given oriented graph with maximum out-degree Δ_o to a Δ_o–regular oriented graph, so that the out-degree of *every* vertex is equal to Δ_o. The fact that the in-degrees are not relevant makes this construction relatively easy. The one thing to watch out for is the introduction of oriented cycles; in particular, in the proof of Lemma 5.3 we use the fact that the out-neighbors of a vertex do not induce an oriented cycle. Our construction guarantees this property. In Lemma 5.5, we show the degree reduction result that we obtain via Algorithm `DozeOff`.

LEMMA 5.4. *Let G be an oriented graph with the out-degree bounded by Δ_o. Then, we can construct a graph G' such that G' is regular with each vertex having an out-degree of Δ_o.*

PROOF. Let every vertex of G be a vertex in G'. Every edge of G is added to $E(G')$. In addition, for each vertex $v \in V(G')$ such that $d_o(v) < \Delta_o$, let $b(v) = \Delta_o - d_o(v)$. For each such vertex v, we add additional vertices and edges to G' as follows. Let H_v be a graph with a vertex set that is a union of 4 sets $S_1(v), S_2(v), S_3(v), S_4(v)$ where each such set has Δ_o vertices. Each vertex in $S_i(v)$ is made a neighbor of every vertex in $S_{i+1}(v)$ for $0 < i < 4$. Each such edge with $u \in S_i(v)$ and $v \in S_{i+1}(v)$, the edge uv is oriented from u to v. Each vertex in $S_4(v)$ is also made a neighbor of every vertex in $S_1(v)$ with edges oriented from vertices in $S_4(v)$ to vertices in $S_1(v)$. Further, from the vertex v, $b(v)$ edges are added to from v to *any* $b(v)$ vertices in $S_1(v)$. The graph G' can therefore be defined as $G' = G \bigcup \left(\cup_{v \in V(G)} H_v\right)$. It can be seen that $\max_{v \in V(G')} d_o(v) = \Delta_o$. □

LEMMA 5.5. *Let G be an oriented graph with out-degree bounded above by Δ_o and let $k \in O(\log \Delta_o)$ be a positive integer. There is a distributed algorithm running in $O(k)$ rounds that uses Δ_o/k colors to properly color (some of) the vertices of G such that after the execution of this algorithm every vertex of G has at most Δ_o/k uncolored neighbors w.h.p.*

PROOF. Let us construct the graph G' as described in Lemma 5.4. Then, using the result of Lemma 5.3 and the techniques of [9], it holds that after the execution of the `DozeOff` algorithm on G', every vertex of G' has at most Δ_o/k uncolored neighbors with high probability. Note that Algorithm `DozeOff` uses a palette of size Δ_o/k. Since our network is the graph G and not G', we simulate the execution of Algorithm `DozeOff` on G with each vertex v in G simulating the actions of all vertices in H_v in G'. Note that this is easy since vertices in H_v has edges only amongst themselves or to v. After the simulation of the algorithm, vertices in G simply inherit the color they might have been assigned during the execution of Algorithm `DozeOff` on G'. □

Algorithm `BudgetColor`(G, k)
Begin-Algorithm
1. **for** every node u, $C_u = \{1, 2, \cdots, \Delta_o/k\}$;
2. Call `DozeOff`(k, C_u) for every node u;
3. Call `OrientedRandColor`(k, C_u) at every node u of G

End-Algorithm

Figure 3: Algorithm `BudgetColor`. The argument k is required to satisfy $k = O(\log \Delta_o)$.

We are now ready to show our final result, Theorem 1.3 restated below for convenience.

THEOREM 5.6. *Let G be a graph with an acyclic orientation whose out-degree is bounded by Δ_o. Then,*

(i) If $k \geq 2 \log\log n$ and $\Delta_o = \Omega(\max\{k \cdot n^{2/k^2} \cdot \log^{1+1/k} n, 2^k\})$, G can be colored using $O(\Delta_o)$ colors in $O(k)$ rounds.

(ii) If $k \geq 2 \log\log n$ and $\Delta_o = \Omega(\max\{k \cdot \log^{1+1/k} n, 2^k\})$, G can be colored using $O(\Delta_o \cdot n^{2/k^2})$ colors in $O(k)$ rounds.

(iii) If $k < 2 \log\log n$ and $\Delta_o \geq k \cdot 2^{2^k} \cdot \log n$, then G can be colored using $O\left(\Delta_o^{1+\frac{1}{k}} \cdot n^{2/k^2} \cdot \frac{1}{2^{2^k}}\right)$ colors in $O(k)$ rounds.

PROOF. The algorithm that we use to arrive at the stated theorem is shown in Figure 3. We only discuss (i); (ii) and (iii) are very similar. We first appeal to Lemma 5.5. Notice that since $\Delta = \Omega(2^k)$ is part of the hypothesis of the current lemma, $k = O(\log \Delta_o)$ as required by the hypothesis of Lemma 5.5. The algorithm implied by Lemma 5.5 runs in $O(k)$ rounds, uses Δ_o/k colors, and produces a proper, *partial* coloring of G such that the number of uncolored vertices of every vertex is at most Δ_o/k, w.h.p. We take the subgraph of G induced by the uncolored vertices and apply Theorem 4.3, with the out-degree bound being Δ_o/k rather than Δ_o. At the end of $O(k)$ rounds of the algorithm implied by this theorem, we use an additional $k \cdot \Delta_o/k = \Delta_o$ colors to complete the coloring. □

6. OBTAINING OPTIMAL COLORINGS IN TERMS OF GRAPH ARBORICITY

By combining Theorem 5.6 with a result of Barenboim and Elkin [1] on distributed graph decomposition, we obtain our final result. As mentioned in the introduction, Barenboim and Elkin [1] showed the following result.

THEOREM 6.1. *For a graph G with arboricity $a(G) = a$, and any parameter q, $q > 2$, there is a deterministic, distributed algorithm that partitions the edges of G into at most $(2+q)\cdot a$ forests within $O(\frac{\log n}{\log q})$ rounds. Further, the edges of G can be acyclically oriented so that the out-degree of each vertex is bounded by $O((2+q)\cdot a)$.*

This theorem can be restated in a more convenient form as follows.

THEOREM 6.2. *For a graph G with arboricity $a(G) = a$, and any positive integer $k < \log n$, there is a deterministic, distributed algorithm that constructs an acyclic orientation of G in $O(k)$ rounds so that the out-degree of each vertex is bounded by $O(a \cdot n^{1/k})$.*

PROOF. If we set $k = \frac{\log n}{\log q}$, we obtain that $q = n^{1/k}$. Requiring that $k < \log n$ guarantees that $q > 2$. □

By combining this with Theorem 5.6 we obtain item (ii) of Theorem 1.4 as shown below. Items (i) and (iii) of Theorem 1.4 similarly follow.

COROLLARY 6.3. *Let G be an n-vertex graph with arboricity a. For any k, $2\log\log n \leq k \leq \sqrt{\log n}$, G can be properly colored with $O(a \cdot n^{1/k})$ colors in $O(k)$ rounds.*

PROOF. We first use the Barenboim-Elkin algorithm to obtain an acyclic orientation of G with $\Delta_o(G) \in O(a \cdot n^{1/k})$. To apply Theorem 5.6, we require that $k \geq 2\log\log n$ and also require that $a \cdot n^{1/k} = \Omega(2^k)$ and $a \cdot n^{1/k} = \Omega(k \cdot n^{2/k^2} \cdot \log^{1+1/k} n)$. When $k \leq \sqrt{\log n}$, we have

$$n^{1/k} \geq n^{1/\sqrt{\log n}} \geq c \cdot 2^{\sqrt{\log n}} \geq c \cdot 2^k,$$

for some constant c. This ensures that when $k \leq \sqrt{\log n}$, we have $a \cdot n^{1/k} = \Omega(2^k)$. Also, for some constants c_1, c_2, and c_3, we have

$$k \leq c_1 \cdot n^{1/3k}; n^{2/k^2} \leq c_2 \cdot n^{1/3k}; \log^{1+1/k} n \leq c_3 \cdot n^{1/3k},$$

for all $k \leq \sqrt{\log n}$. This implies that when $k \leq \sqrt{\log n}$, we have that $a \cdot n^{1/k} = \Omega(k \cdot n^{2/k^2} \cdot \log^{1+1/k} n)$ Hence the hypothesis of Theorem 5.6 is satisfied and we obtain an $O(a \cdot n^{1/k})$ coloring of G in $O(k)$ rounds. □

The above corollary restricts k to be in the range $2\log\log n \leq k \leq \sqrt{\log n}$ in order to obtain an optimal coloring. However, for smaller values of k a slightly suboptimal coloring can be obtained via the use of item (iii) in Theorem 5.6.

COROLLARY 6.4. *Let G be an n-vertex graph with arboricity a. For any $k < 2\log\log n$, G can be properly colored with $O(a^{1+1/k} \cdot n^{1/k+3/k^2} \cdot \frac{1}{2^{2^k}})$ colors in $O(k)$ rounds.*

The two corollaries above correspond to Theorem 1.4 in the "Introduction."

7. REFERENCES

[1] BARENBOIM, L., AND ELKIN, M. Sublogarithmic distributed MIS algorithm for sparse graphs using nash-williams decomposition. In *ACM Symp. on Principles of Distributed Computing (PODC)* (2008), pp. 25–34.

[2] BARENBOIM, L., AND ELKIN, M. Deterministic distributed vertex coloring in polylogarithmic time. In *ACM Symp. on Principles of Distributed Computing (PODC)* (2010), pp. 410–419.

[3] CZYGRINOW, A., AND HAŃĆKOWIAK, M. Distributed almost exact approximations for minor-closed families. In *European Symposium on Algorithms* (2006), pp. 244–255.

[4] CZYGRINOW, A., AND HAŃĆKOWIAK, M. Distributed approximation algorithms for weighted problems in minor-closed families. In *COCOON: 13th Annual International Conference on Computing and Combinatorics* (2007), pp. 515–525.

[5] CZYGRINOW, A., HAŃĆKOWIAK, M., AND SZYMANSKA, E. Fast distributed alogrithm for the maximum matching problem in bounded arboricity graphs. In *20th International Symposium on Algorithms and Computation (ISAAC 2009)* (2009), vol. 5878/2009, pp. 668–678.

[6] CZYGRINOW, A., HANCKOWIAK, M., AND WAWRZYNIAK, W. Fast distributed approximations in planar graphs. In *International Symposium on Distributed Computing* (2008), pp. 78–92.

[7] FINOCCHI, I., PANCONESI, A., AND SILVESTRI, R. Experimental analysis of simple, distributed vertex coloring algorithms. In *ACM SODA* (2002), pp. 606–615.

[8] GRABLE, D. A. A large deviation inequality for functions of independent, multi-way choices. *Comb. Probab. Comput. 7*, 1 (1998).

[9] GRABLE, D. A., AND PANCONESI, A. Fast distributed algorithms for brooks-vizing colorings. *J. Algorithms 37*, 1 (2000), 85–120.

[10] JOHANSSON, O. Simpled distributed $\Delta + 1$ coloring of graphs. *Information Processing Letters 70* (1999), 229–232.

[11] KOTHAPALLI, K., SCHEIDELER, C., ONUS, M., AND SCHINDELHAUER, C. Distributed coloring in $O(\sqrt{\log n})$ bit rounds. In *In International Parallel and Distributed Processing Symposium, (IPDPS)* (2006).

[12] KUHN, F., MOSCIBRODA, T., AND WATTENHOFER, R. The price of being near-sighted. In *SODA '06: Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm* (New York, NY, USA, 2006), ACM, pp. 980–989.

[13] KUHN, F., AND WATTENHOFER, R. Constant-time distributed dominating set approximation. *Distributed Computing 17*, 4 (2005), 303–310.

[14] LENZEN, C., AND WATTENHOFER, R. Minimum dominating set approximation in graphs of bounded arboricity. In *International Symposium on Distributed Computing (DISC)* (2010), pp. 510–524.

[15] LINIAL, N. Locality in distributed graph algorithms. *SIAM Journal of Computing 21* (1992), 193–201.

[16] LUBY, M. A simple parallel algorithm for the maximal independent set problem. In *Proc. of ACM Symposium on Theory of Computing* (1985), pp. 1–10.

[17] LUBY, M. Removing randomness in parallel without processor penalty. *Journal of Computer and System Sciences 47*, 2 (1993), 250–286.

[18] MOSCIBRODA, T., AND WATTENHOFER, R. Facility location: distributed approximation. In *Proc. ACM symposium on Principles of distributed computing* (2005), pp. 108–117.

[19] MOTWANI, R., AND RAGHAVAN, P. *Randomized Algorithms*. Cambridge University Press, 1995.

[20] NASH-WILLIAMS, C. Decompositions of finite graphs into forests. *J. London Math 39*, 12 (1964).

[21] PANDIT, S., AND PEMMARAJU, S. Return of the primal-dual: distributed metric facilitylocation. In *Proceedings of the 28th ACM symposium on Principles of distributed computing* (2009), pp. 180–189.

[22] PANDIT, S., AND PEMMARAJU, S. V. Rapid randomized pruning for fast greedy distributed algorithms. In *ACM Symp. on Principles of Distributed Computing (PODC)* (2010), pp. 325–334.

[23] SCHNEIDER, J., AND WATTENHOFER, R. A new technique for distributed symmetry breaking. In *ACM Symp. on Principles of Distributed Computing (PODC)* (2010), pp. 257–266.