Distributed Coloring in $\tilde{O}(\sqrt{\log n})$ Bit Rounds

Kishore Kothapalli *
Department of Computer Science
Johns Hopkins University
3400 N. Charles Street
Baltimore, MD 21218, USA
kishore@cs.jhu.edu

Christian Scheideler
Computer Science Department
Technische Universität München
Boltzmanstr. 3, D-85748
Garching, Germany
scheideler@in.tum.de

Melih Onus
Department of Computer Science
Arizona State University
Tempe, AZ 85287, USA
melih@asu.edu

Christian Schindelhauer †
Heinz Nixdorf Institute and
Computer Science Department
University of Paderborn
33102 Paderborn, Germany
schindel@uni-paderborn.de

Abstract

We consider the well-known vertex coloring problem: given a graph G, find a coloring of the vertices so that no two neighbors in G have the same color. It is trivial to see that every graph of maximum degree Δ can be colored with $\Delta+1$ colors, and distributed algorithms that find a $(\Delta+1)$-coloring in a logarithmic number of communication rounds, with high probability, are known since more than a decade. This is in general the best possible if only a constant number of bits can be sent along every edge in each round. In fact, we show that for the n-node cycle the bit complexity *of the coloring problem is $\Omega(\log n)$. More precisely, if only one bit can be sent along each edge in a round, then* every *distributed coloring algorithm (i.e., algorithms in which every node has the same initial state and initially only knows its own edges) needs at least $\Omega(\log n)$ rounds, with high probability, to color the cycle, for* any *finite number of colors. But what if the edges have orientations, i.e., the endpoints of an edge agree on its orientation (while bits may still flow in both directions)? Does this allow one to provide faster coloring algorithms?*

Interestingly, for the cycle in which all edges have the same orientation, we show that a simple randomized algorithm can achieve a 3-coloring with only $O(\sqrt{\log n})$ rounds of bit transmissions, with high probability (w.h.p.). This result is tight because we also show that the bit complexity of coloring an oriented cycle is $\Omega(\sqrt{\log n})$, with high probability, no matter how many colors are allowed. The 3-coloring algorithm can be easily extended to provide a $(\Delta+1)$-coloring for all graphs of maximum degree Δ in $O(\sqrt{\log n})$ rounds of bit transmissions, w.h.p., if Δ is a constant, the edges are oriented, and the graph does not contain an oriented cycle of length less than $\sqrt{\log n}$. Using more complex algorithms, we show how to obtain an $O(\Delta)$-coloring for arbitrary oriented graphs of maximum degree Δ using essentially $O(\log\Delta+\sqrt{\log n})$ rounds of bit transmissions, w.h.p., provided that the graph does not contain an oriented cycle of length less than $\sqrt{\log n}$.

1. Introduction

A fundamental problem in distributed systems is to compute a proper vertex coloring. The importance of vertex coloring can be seen by observing that many distributed algorithms use such a coloring as a sub-routine in higher-order communication and computation tasks. Examples include scheduling [16], resource allocation [3], and synchronization. Vertex coloring has applications also in wireless networks to determine cluster heads, (see for example [22] and the references therein), routing in wireless networks [16], and in many parallel algorithms [14, 15]. Thus, it is not surprising that this problem has been heavily studied not only in the distributed setting but also in the PRAM model

* Supported by NSF grant CCR-0311121.
† Partially supported by the DFG-Sonderforschungsbereich 376 and by the EU within 6th Framework Programme under contract 001907 Dynamically Evolving, Large Scale Information Systems (DELIS).

of computation starting with Karp and Wigderson [15] and Luby [18].

We consider distributed systems that can be modeled as a graph $G = (V, E)$ with nodes representing the processors and the edges representing the communication links. Given a graph $G = (V, E)$ with maximum degree Δ, the vertex coloring problem is to find a color assignment for the vertices of G so that no two adjacent vertices are given the same color. The minimum number of colors required to properly color a graph is called its *chromatic number*, and is denoted by $\chi(G)$. While it is easy to see that a graph with maximum degree Δ can be colored using at most $\Delta+1$ colors, computing the chromatic number of a graph is NP–hard [9]. Further, $\chi(G)$ cannot be approximated to any reasonable bound in general [6]. Thus, efficient algorithms that color using $\Delta + 1$ colors are of interest.

In the distributed model of computing, communication is an expensive resource and distributed algorithms therefore aim at using as little communication as possible. Distributed algorithms for vertex coloring take the approach of minimizing the number of communication rounds assuming that in each round a reasonable number of bits can be communicated. Deterministic distributed algorithms for $(\Delta + 1)$-coloring that run in a polylogarithmic number of rounds are not known. The best known deterministic algorithm [24] requires $n^{O(1/\sqrt{\log n})}$ rounds where n is the number of vertices. However, randomization can improve the runtime exponentially and in some special cases, such as highly dense graphs, even double exponentially [12]. Randomized distributed algorithms that compute a $(\Delta + 1)$–coloring in $O(\log n)$ rounds, with high probability[1], are known since more than a decade [19, 13]. In this work we show that, interestingly, if the underlying graph G is provided with an orientation on its edges such that the orientation does not induce oriented cycles of length at most $\sqrt{\log n}$, then vertex coloring with $(1 + \epsilon)\Delta$ colors for a constant $\epsilon > 0$, can be obtained by exchanging essentially $O(\log \Delta + \sqrt{\log n})$ bits, with high probability. Thus, we show that having orientations on the edges significantly improves the performance of distributed vertex coloring algorithms.

1.1. Model and Definitions

We model the distributed system as a graph $G = (V, E)$ with V representing the set of computing entities, or processors, and $E \subseteq V \times V$ representing all the available communication links. We assume that all the communication links are undirected and hence bidirectional. All the processors start at the same time and time proceeds in synchronized rounds. We let $n = |V|$. The degree of node u is denoted d_u and by Δ we denote the maximum degree of G, i.e., $\Delta = \max_{u \in V} d_u$. When there is no confusion, d_u will also be used to refer to the number of uncolored neighbors of node u. By N_u we denote the set of neighbors of node u and when there is no confusion, we use N_u to refer to the set of uncolored neighbors of u. We do not require that the nodes in V have unique labels of any kind. For our algorithms to work, it is enough that each node knows a constant factor estimate of the logarithm of the size of the network apart from its own degree and neighbors. When we consider graphs of constant degree, *no* global knowledge is required for our algorithm and it suffices that each node knows its own degree.

Let us denote by $[x]$ the set $\{1, 2, \ldots, x\}$ if $x \in \mathbb{N}$. If $x \in \mathbb{R}^+$, then $[x]$ would be the set $\{1, 2, \ldots, \lceil x \rceil\}$. Given a graph $G = (V, E)$ a vertex coloring is a mapping $c : V \to [C]$ such that if $\{u, v\} \in E$ then $c(u) \neq c(v)$, i.e., no two adjacent vertices receive the same color. Here C denotes the number of colors used in the coloring. We say that a coloring is a *local coloring* if every node u with degree d_u has a color in $[\epsilon d_u]$ when the coloring uses $\epsilon\Delta$ colors. The interest in local coloring arises from the fact that a local coloring has nice implications when using the coloring in scheduling and routing problems [16].

In our model, the measure of efficiency is the number of bits exchanged. We also refer to this as the *bit complexity*. We view each round of the algorithm as consisting of 1 or more *bit rounds*. In each bit round each node can send/receive at most 1 bit from each of its neighbors. We assume that the rounds of the algorithm are synchronized. The bit complexity of algorithm A is then defined as the number of bit rounds required by algorithm A. We note that, since the nodes are synchronized, each round of the algorithm requires as many bit rounds as the maximum number of bit rounds needed by any node in this round. In our model, we do not count local computation performed by the nodes. This is reasonable as in our algorithms nodes perform only simple local computation.

In our model, we assume that the edges in E have an orientation associated with them. That is, for any two neighbors v, w exactly one of the following holds for the edge $\{v, w\}$: $\{v, w\}$ is oriented either $v \to w$ or as $w \to v$. In the former we also call v *superior* to w and vice-versa in the latter. Having orientation on the edges is a property that has not been studied in the context of vertex coloring though it is a natural property since networks usually evolve and for every connection there is usually a node that initiated it. We show that algorithms for symmetry breaking can be greatly improved provided that the underlying graph is oriented. The exact way in which orientation is used for symmetry breaking is explained in Figure 1. As shown, if nodes v and w choose the same color during any round of the algorithm, in the existing algorithms, both nodes remain un-

1 With high probability (w.h.p.) means a probability that is at least $1 - (1/n^c)$ for $c \geq 1$.

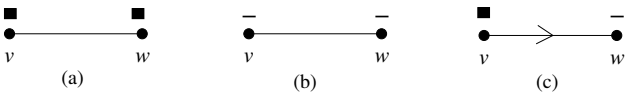

Figure 1. Orientation helps in symmetry breaking. In Figure (a) both v and w choose the same color. In (b), for existing algorithms both remain uncolored whereas in (c), when using orientation, node v may get colored.

colored as in Figure 1(b) and have to try in a later round. With orientation, if the edge $\{v, w\}$ is oriented as $v \rightarrow w$ as shown in Figure 1(c), then node v can retain its choice provided that there is no edge $\{u, v\}$ oriented $u \rightarrow v$ and u also chooses the same color.

One parameter that will be important for our investigations is the length of the shortest cycle in the orientation. We formalize this notion in the following definition.

Definition 1.1 (ℓ–*acyclic Orientation*) *An orientation of the edges of a graph is said to be ℓ–acyclic if the minimum length of any directed cycle induced by the orientation is at least ℓ. Note that this is not the girth of the given graph.*

1.2. Related Work

The problem of vertex coloring in distributed systems has a long and rich history. It is an open problem whether deterministic poly-logarithmic time distributed algorithms exist for the problem of $(\Delta + 1)$-vertex coloring [24]. The best known deterministic algorithm to date is presented in [24] and requires $n^{O(1/\sqrt{\log n})}$ rounds. Following considerations known from the radio broadcasting model [1] the problem cannot be solved at all in a deterministic round model without the use of unique identification numbers. Hence, most of the algorithms presented are randomized algorithms.

Luby [19] and Johansson [13] present parallel algorithms that can be interpreted as distributed algorithms that provide a $(\Delta + 1)$–coloring of a graph G in $O(\log n)$ rounds, with high probability. Recent empirical studies [7] have shown that the constant factors involved are small. Algorithms for vertex coloring are also presented in [10, 14] in the PRAM model of computation.

Cole and Vishkin [4] and Goldberg et. al. [10] have shown that a $(\Delta+1)$–coloring of the cycle graph on n nodes can be achieved in $O(\log^* n)$ communication rounds. This was shown to be optimal in Linial [17] by establishing that 3-coloring an n–node cycle graph cannot be achieved in less than $(\log^* n - 1)/2$ rounds. When arbitrary amount of local computation is allowed [17], De Marco and Pelc [20] show that an $O(\Delta)$ coloring can be achieved in $O(\log^*(n/\Delta))$ rounds improving the results of Linial [17] in this model.

In a related work, Grable and Panconesi [12] present a distributed algorithm in the message passing model for edge coloring that runs in $O((1 + \alpha^{-1}) \log\log n)$ rounds provided that the degree of any node in the graph is $\Omega(n^{\alpha/\log\log n})$ for any $\alpha > 0$. Our analysis of Phase I for arbitrary graphs follows the analysis of Phase II in [12].

Distributed algorithms with the underlying graph equipped with sense of direction have been studied in [25, 8]. Sense of direction is a similar notion to that of orientation on edges. Singh [25] shows that leader election in an n-node complete graph equipped with sense of direction can be performed in a distributed setting via exchange of $O(n)$ messages. In [8], the authors show that having sense of direction reduces the communication complexity of several distributed graph algorithms such as leader election, spanning tree construction, and depth-first traversal.

1.3. Our Results

We start by investigating the bit complexity of distributed vertex coloring algorithms. We first show that the bit complexity of the coloring problem is $\Omega(\log n)$ for an non-oriented n-node cycle graph. That is, any distributed algorithm in which all the nodes start in the same state and know only about n and Δ apart from their neighbors needs $\Omega(\log n)$ rounds with high probability to arrive at a proper coloring using any finite number of colors. We then show that when the edges in the cycle graph are provided with an orientation, then the bit complexity of distributed vertex coloring algorithms is $\Omega(\sqrt{\log n})$, with high probability, when using any finite number of colors. This leads us to the question whether matching upper bounds can be shown for coloring oriented graphs.

We start with the case of constant degree $\sqrt{\log n}$–acyclic oriented graphs and present an algorithm to obtain a $(\Delta + 1)$–coloring with a bit complexity of $O(\sqrt{\log n})$ with high probability. Thus, we show the following theorem.

Theorem 1.2 *Given a $\sqrt{\log n}$–acyclic oriented graph $G = (V, E)$ of maximum degree Δ, if Δ is a constant, a $(\Delta + 1)$–vertex coloring of G can be obtained in $O(\sqrt{\log n})$ bit rounds, with high probability.*

The above theorem directly implies that oriented cycle graphs can be 3–colored in $O(\sqrt{\log n})$ bit rounds. Additionally, for the case of constant degree graphs we can also arrive at a local coloring where the color of every node u is in $[d_u + 1]$.

We then extend our algorithm and analysis to the case of arbitrary $\sqrt{\log n}$–acyclic oriented graphs with maximum

degree Δ. Our main result is a distributed $(1+\epsilon)\Delta$–coloring algorithm for arbitrary $\sqrt{\log n}$–acyclic oriented graphs of maximum degree Δ. Our algorithm has a bit complexity of $O(\log \Delta) + \tilde{O}(\sqrt{\log n})$. By $g(n) = \tilde{O}(f(n))$ we mean $g(n) = O(f(n)\text{polylog}(f(n)))$. Specifically, we prove the following theorem.

Theorem 1.3 *Given a $\sqrt{\log n}$–acyclic oriented graph $G = (V, E)$ of maximum degree Δ, a $(1+\epsilon)\Delta$–vertex coloring of G, for any constant $\epsilon > 0$, can be obtained in $O(\log \Delta) + \tilde{O}(\sqrt{\log n})$ bit rounds, with high probability.*

By further tightening the analysis, we show that the bit complexity can be reduced to $O(\log \Delta + \sqrt{\log n \log\log n})$, with high probability, for $\sqrt{\log n}$–acyclic oriented graphs with $\Delta \geq \log n$.

For the case of arbitrary $\sqrt{\log n}$–acyclic oriented graphs, our algorithm and analysis can be modified easily to get a local coloring such that every node u gets a color in $[(1+\epsilon)d_u]$.

1.4. Summary of our approach

We now provide a brief summary of our basic approach. Our approach has the same flavor as existing distributed vertex coloring algorithms [19, 13]. Given any $\sqrt{\log n}$–acyclic oriented graph $G = (V, E)$ of constant degree Δ, the algorithm for $(\Delta+1)$–coloring proceeds as follows. Communication proceeds in rounds and in each round each yet uncolored node v chooses a color c_v among the available colors in $[\Delta+1]$ uniformly at random. Node v then communicates this color choice to all of its uncolored neighbors. If a node chooses a color that is in conflict with any of the choices of its neighbors, the conflict resolution rule specifies the course of action. In the algorithm of Luby[19], Johansson[13], and most other works, the conflict resolution rule is that uncolored nodes in conflict remain uncolored and have to try again in subsequent rounds. The conflict resolution rule we use is based on the orientation on the edges as explained in Section 1.1. Our algorithm is thus similar to the existing distributed vertex coloring algorithms [19, 13] except for the conflict resolution rule.

In our analysis, after $O(\sqrt{\log n})$ rounds we arrive at the situation where connected components of uncolored nodes only have simple oriented paths of length less than $\sqrt{\log n}$, with high probability. Coupled with the $\sqrt{\log n}$–acyclic orientation, it can be shown that the nodes in each such connected component can be organized into less than $\sqrt{\log n}$ layers. The layering has the property that all the oriented edges are from a node in a lower-numbered layer to a node in a higher numbered layer. This property of the layering guarantees a successful coloring of all remaining uncolored nodes in less than $\sqrt{\log n}$ rounds. This gives us the result for constant degree oriented graphs. (Theorem 1.2). To arrive at the bit complexity for arbitrary graphs, (Theorem 1.3) we need a few additional tricks as our analysis shows.

1.5. Organization of the paper

The rest of the paper is organized as follows. In Section 2 we establish the lower bound results. In Section 3, we present and analyze our algorithm for $(\Delta+1)$–coloring constant degree oriented graphs. This will serve as a base for the $(1+\epsilon)\Delta$–coloring algorithm for arbitrary oriented graphs of maximum degree Δ for any constant $\epsilon > 0$, in Section 4. The paper ends with conclusions in Section 5.

2. Lower Bounds

In this section we establish lower bounds on the bit complexity of finding a proper vertex coloring. Recall that a Las Vegas algorithm is a randomized algorithm that always produces a correct result, with the only variation being its runtime. First, we prove a lower bound for non-oriented graphs, and then we prove a lower bound for oriented graphs. Notice that both bounds hold for any finite number of colors.

Theorem 2.1 *For every Las Vegas algorithm A there is an infinite family of non-oriented graphs $\mathcal{G}$ s.t. A has a bit complexity of at least $\Omega(\log n)$ on $\mathcal{G}$, with high probability, to compute a proper vertex coloring.*

Proof. Consider the cycle of n nodes, and let $S_\ell = (u_\ell, \ldots, u_1, v_1, \ldots, v_\ell)$ be the set of nodes along a path of length 2ℓ of the cycle. Initially, every node in S_ℓ is in the same state s_0, with the only difference that for every $i \in \{1, \ldots, \ell-1\}$, u_i considers its left connection to go to u_{i+1} whereas v_i considers its left connection to go to v_{i+1}. (Notice that the cycle is non-oriented, so we can choose any orientation we want for the individual nodes.) Associated with s_0 is a fixed probability distribution $P_\epsilon = (p_x^\epsilon)_{x\in\{-,0,1\}}$ for sending bit x along the right edge, where "$-$" represents the case that no bit is sent and ϵ represents the empty history. Since P_ϵ has only three probability values, there must be an x_0 with $p_{x_0}^\epsilon \geq 1/3$. Let E_1 be the event that nodes u_1 and v_1 choose that option. Then u_1 and v_1 receive the same information from their right neighbor. Let $P_y = (p_x^y)_{x\in\{-,0,1\}}$ be the probability distribution for sending bit x along the right edge in the second round given that bit y was received from the right edge in the first round. Then P_{x_0} applies to u_1 and v_1. Since P_{x_0} has only three probability values, there must be an x_1 with $p_{x_1}^{x_0} \geq 1/3$. Let E_2 be the event that nodes u_1 and v_1 choose that option. Then u_1 and v_1 again receive the same information from their right neighbor.

Continuing with this argumentation, it follows that there are events $E_1, \ldots, E_\ell$ with E_i having a probability of at least $1/3$ for all i so that u_1 and v_1 have received the same information from their right neighbors. Algorithm A cannot

terminate in this case because in this case the same probability distribution for choosing a color applies to u_1 and v_1, and hence, the probability that u_1 and v_1 choose the same color is non-zero.

When choosing $\ell = \log_3(n/2\log^2 n)$, the probability for $E_1, \ldots, E_\ell$ to occur is at least $\left(\frac{1}{3}\right)^{\log_3(n/2\log^2 n)} = \frac{2\log^2 n}{n}$. Moreover, notice that $E_1, \ldots, E_\ell$ only depend on the nodes in S_ℓ because information can only travel a distance of ℓ edges in ℓ rounds. Hence, we can partition the n-node cycle into $n/2\ell$ many sequences S where each sequence has a probability of at least $\frac{2\log^2 n}{n}$ of running into the events $E_1, \ldots, E_\ell$ that is independent of the other sequences. Thus, the probability that all node sequences can avoid the event sequence $E_1, \ldots, E_\ell$, which is necessary for A to terminate, is at most $\left(1 - \frac{2\log^2 n}{n}\right)^{n/2\ell} \leq 1/n$, which implies that A needs $\Omega(\log n)$ bit-rounds, with high probability, to finish. □

Theorem 2.2 *For every Las Vegas algorithm A there is an infinite family of oriented graphs $\mathcal{G}$ s.t. A has a bit complexity of at least $\Omega(\sqrt{\log n})$ on $\mathcal{G}$, with high probability, to compute a proper vertex coloring.*

Proof. Consider the cycle of n nodes in which all the edges are oriented in the same direction. Let $S_\ell = (u_\ell, \ldots, u_1, v_1, \ldots, v_\ell)$ be the set of nodes along a path of length 2ℓ of the cycle. Initially, every node in S_ℓ is in the same state s_0. Associated with s_0 is a fixed probability distribution $P_0 = (p^0_{x,y})_{x,y\in\{-,0,1\}}$ for sending bit x along the left edge and bit y along the right edge, where "$-$" represents the case that no bit is sent. Since P_0 has only nine probability values, there must be an x_0 and y_0 with $p^0_{x_0,y_0} \geq 1/9$. Let E_1 be the event that all nodes in S_ℓ choose that option. Then all nodes in $S_{\ell-1} = (u_{\ell-1}, \ldots, u_1, v_1, \ldots, v_{\ell-1})$ receive the same information and must therefore be in the same state s_1. Associated with s_1 is a fixed probability distribution $P_1 = (p^1_{x,y})_{x,y\in\{-,0,1\}}$ for sending bit x along the left edge and bit y along the right edge. Since P_1 has only nine probability values, there must be an x_1 and y_1 with $p^1_{x_1,y_1} \geq 1/9$. Let E_2 be the event that all nodes in $S_{\ell-1}$ choose that option. Then all nodes in $S_{\ell-2}$ receive the same information and must therefore be in the same state s_2.

Continuing with this argumentation, it follows that there are events $E_1, \ldots, E_\ell$ with E_i having a probability of at least $(1/9)^{2(\ell-i+1)}$ for all i so that all nodes in $S_{\ell-i}$ are in the same state s_i. Since these nodes are neighbors, algorithm A cannot terminate within ℓ bit exchanges if $E_1, \ldots, E_\ell$ are true because whatever probability distribution A chooses on the colors, the probability that two neighboring nodes choose the same color is non-zero, which would violate the assumption that A is a Las Vegas algorithm.

The probability that $E_1, \ldots, E_\ell$ are true is at least $\left(\frac{1}{9}\right)^{\sum_{i=1}^{\ell} 2(\ell-i+1)} \geq \left(\frac{1}{9}\right)^{\ell^2/2}$ and when choosing $\ell = \sqrt{2\log_9(n/2\log^2 n)}$, this results in a probability of at least $(2\log^2 n)/n$. Moreover, notice that $E_1, \ldots, E_\ell$ only depend on the nodes in S_ℓ because information can only travel a distance of ℓ edges in ℓ rounds. Hence, we can partition the n-node cycle into $n/2\ell$ many sequences S where each sequence has a probability of at least $\frac{2\log^2 n}{n}$ of running into the events $E_1, \ldots, E_\ell$ that is independent of the other sequences. Hence, the probability that all node sequences can avoid the event sequence $E_1, \ldots, E_\ell$, which is necessary for A to terminate, is at most $\left(1 - \frac{2\log^2 n}{n}\right)^{n/2\ell} \leq 1/n$, which implies that A needs $\Omega(\sqrt{\log n})$ bit-rounds, with high probability, to finish. □

Thus, oriented graphs appear to be easier to color than non-oriented graphs. In the next section we show that this is indeed the case by providing a matching upper bound for constant-degree graphs.

3. Upper Bound for Constant Degree Oriented Graphs

In this section we present and analyze the algorithm for $(\Delta + 1)$–coloring constant degree oriented graphs. This demonstrates the efficacy of using orientation in vertex coloring algorithms. We defer the case of arbitrary oriented graphs to Section 4 as it requires more complicated arguments than for constant degree graphs.

The algorithm for vertex coloring constant degree oriented graphs is given in Figure 2. In the algorithm, the parameter C_u refers to the number of colors used in the coloring by node u. Each node executes the algorithm Color-Random until it gets colored.

We analyze algorithm Color-Random for constant degree oriented graphs with a $\sqrt{\log n}$–acyclic orientation and show that algorithm Color-Random can be used to obtain a $(\Delta + 1)$–coloring with a bit complexity of $O(\sqrt{\log n})$. The reduction in the bit complexity from $\Omega(\log n)$ (due to Theorem 2.1) to $O(\sqrt{\log n})$ comes from the fact that once every simple oriented path of length $\sqrt{\log n}$ has at least one colored node, the $\sqrt{\log n}$–acyclic orientation guarantees us connected components of uncolored nodes where each such component only has simple oriented paths of length less than $\sqrt{\log n}$. The $\sqrt{\log n}$-acyclicity of the orientation allows us to finish in a further $\sqrt{\log n}$ rounds.

Theorem 3.1 *Given a $\sqrt{\log n}$–acyclic oriented graph $G = (V, E)$ of maximum degree Δ, if Δ is a constant, a $(\Delta +$*

Algorithm Color-Random(C_u)

While u is not colored do

1. Node u chooses a color c_u from the available colors in $[C_u]$ uniformly at random.
2. Node u communicates its choice c_u, from step 1, to all of its uncolored neighbors that have a lower priority over u, i.e. to nodes v such that $u \rightarrow v$.
3. If node u does not receive a message from any of its neighbors w with $w \rightarrow u$ and $c_w = c_u$, then node u gets colored with color c_u. Otherwise node u remains uncolored.
4. If u is colored during step 3 of the current round, then u informs all of its uncolored neighbors about the color of u.
5. Node u updates the list of available colors acco-rding to colors taken up by u's neighbors.

Figure 2. Coloring constant degree oriented graphs by random choices.

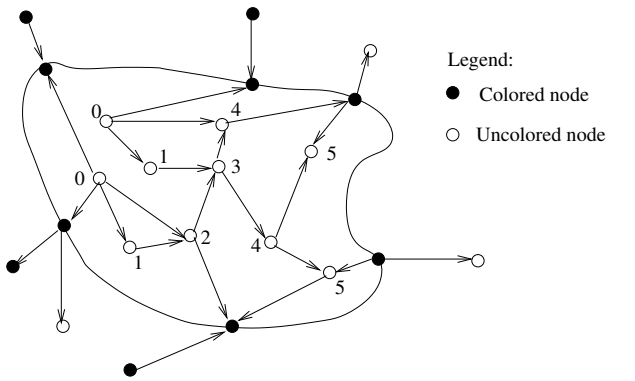

Figure 3. Connected component of uncolored nodes. The number at the uncolored nodes within the connected component gives the layer number they belong to.

1)*–vertex coloring of G can be obtained in $O(\sqrt{\log n})$ bit rounds, with high probability.*

Proof. The analysis below cuts the time into two phases. Phase I ends once every simple oriented path of length $\ell = \sqrt{\log n}$ has at least one colored node, and phase II ends once all nodes are colored. We show that phase I takes at most $r = 4\sqrt{\log n}$ rounds, with high probability. For Phase II, the proof uses the $\sqrt{\log n}$–acyclic orientation to argue that a further $\sqrt{\log n}$ rounds suffice to color all nodes. For simplicity, we set $C_u = 2\Delta$ for every node u, but the analysis works, with minor modifications, for $C_u = \Delta + 1$, as long as Δ is a constant.

Consider any simple oriented path P of length ℓ. For any node $u \in P$ with C'_u remaining colors and d'_u remaining uncolored neighbors, the probability that it chooses a color that is identical to the choice of any of its uncolored neighbors is at most $\sum_{j=1}^{d'_u} 1/C'_u \leq d'_u/(2\Delta - (d_u - d'_u)) \leq 1/2$ as $C'_u = 2\Delta - (d_u - d'_u)$ and $d'_u \leq d_u$.

For any $i \geq 1$, denote by $E_{P,i}$ the event that all nodes in P have a color conflict in round i. Since each node chooses the color independently and uniformly at random, and P is oriented, one can identify a distinct witness for each color conflict so as to upper bound $\Pr[E_{P,i} \mid \cap_{j=0}^{i-1} E_{P,j}]$ as $\Pr[E_{P,i} \mid \cap_{j=0}^{i-1} E_{P,j}] \leq (1/2)^{\ell}$.

Denote by E_P the event that the event $E_{P,i}$ occurs for r consecutive rounds. Then, $\Pr[E_P] = \Pr[\bigcap_{i=1}^{r} E_{P,i}] = \Pi_{i=1}^{r} \Pr[E_{P,i} \mid \cap_{j=1}^{i-1} E_{P,j}] \leq (1/2)^{\ell r}$.

Let E denote the event that for some simple oriented path P the event E_P occurs. The number of simple oriented paths of length ℓ is at most $n\Delta^{\ell}$ by choosing the first vertex from n available choices and choosing each of the next ℓ vertices from the at most Δ available choices. Thus, $\Pr[E] = \Pr[\bigcup_P E_P] \leq n\Delta^{\ell} \Pr[E_P] \leq 1/n^2$, for the above value of r since $\Delta = O(1)$. This completes Phase I of the analysis.

Consider connected components of uncolored nodes. At the end of Phase I, since any simple oriented path of length ℓ has at least one colored node, each such component only has simple oriented paths of length less than ℓ, with high probability. Moreover, the input graph does not have oriented cycles of length less than $\sqrt{\log n}$ which implies that each such component can be organized into less than $\sqrt{\log n}$ layers with oriented edges going only from a node in a lower-numbered layer to a node in a higher numbered layer. This layering can be achieved by the following process. Nodes with no superiors are assigned to layer 0. After removing these nodes, nodes in the rest of the component with no superiors are assigned to layer 1, and so on, until there are no nodes left. Such a procedure terminates in less than $\sqrt{\log n}$ rounds, implying that the layer number that any node belongs to is less than $\sqrt{\log n}$. Otherwise, there must exist either a simple oriented path of length at least $\sqrt{\log n}$ or an oriented cycle of length less than $\sqrt{\log n}$. Both of these conditions result in a contradiction and hence the layering process must terminate in less than $\sqrt{\log n}$ rounds. Figure 3 shows an example along with the assignment of nodes to layers.

Now, in Phase II, during every round the uncolored nodes assigned to the lowest layer number presently get colored as the nodes assigned to the lowest layer can always retain their color choice from Step 1. This implies that Phase II can finish in less than $\sqrt{\log n}$ rounds.

Since in each round each uncolored node has to exchange $O(\log \Delta) = O(1)$ bits, the bit complexity of the algorithm Color-Random is $O(\sqrt{\log n})$. □

Algorithm Color(C_u)
Phase I
1. Set $C_u := c_1\Delta$ for a constant $c_1 \geq 3$.
2. While $d_u \geq c_2 \log n$ for a constant c_2 do
   3. Use Algorithm Color-Random(C_u).
Phase II
4. Set $C_u := \min\{2c_2 \log n, 2d_u\}$.
5. Use Algorithm Color-Random(C_u).

Figure 4. Algorithm for any node u.

We note that the same proof also holds for 3–coloring cycle graphs, with any orientation, with minimal changes. Coupled with the lower bound result in Theorem 2.2, our analysis for the case of constant degree graphs is tight with respect to the bit complexity, up to constant factors. The algorithm and the analysis can be modified easily to achieve a local coloring also.

4. Upper Bound for Arbitrary Oriented Graphs

In this section we describe and analyze our algorithm for vertex coloring an arbitrary $\sqrt{\log n}$–acyclic oriented graph G using $(1+\epsilon)\Delta$ colors for any constant $\epsilon > 0$.

Our algorithm and the analysis in this case requires more tools than that for constant degree graphs while having the same flavor. Theorem 3.1 fails to hold once the degree of the input graph is bounded away from any constant. Graphs below logarithmic degree, but bounded away from a constant, pose additional problems as graphs with degree below a certain threshold are not easily amenable to nice probabilistic bounds. In many papers, for example [12, 23, 5], this problem was overcome by assuming that the number of colors available is $\max\{(1+\epsilon)\Delta, \log n\}$ so that sub-logarithmic degree graphs are colored with $\log n$ colors. We instead take the approach of coloring with $(1+\epsilon)\Delta$ colors as coloring with few colors is more appealing when vertex coloring is used as a sub-routine in other higher order tasks.

To arrive at our result, we proceed in stages. Based on techniques from [12], we first show how to arrive at a bit complexity of $\tilde{O}(\log\Delta + \sqrt{\log n})$. Later, using advanced techniques, we show how to arrive at a bit complexity of $O(\log\Delta) + \tilde{O}(\sqrt{\log n})$. Finally, for graphs with $\Delta \geq \log n$, we show how to arrive at a bit complexity of $O(\log\Delta + \sqrt{\log n \log\log n})$.

Our algorithm for any node u is presented in Figure 4. The parameter C_u denotes the number of colors each vertex u can choose from. Each node runs the algorithm in Figure 4 while it remains uncolored.

We now provide a summary of our analysis of algorithm Color. Our analysis cuts time into two phases. In the first phase we show that for any vertex the number of uncolored neighbors reduces to at most $c_2 \log n$ for a constant c_2, in $O(\log\log n)$ rounds, with high probability. In the second phase we first show that the graph can be decomposed into connected components of uncolored nodes such that each such connected component only has simple oriented paths of length less than $\sqrt{\log n}$, with high probability. The analysis then proceeds to show that all the nodes can be colored in a further $\sqrt{\log n}$ rounds.

In the algorithm and the analysis we also set $C_u = c_1\Delta$ for a constant $c_1 \geq 3$, for every node u, for the sake of simplicity. Using techniques from [12], it is possible to extend the following analysis to use only $(1+\epsilon)\Delta$ colors, for any constant $\epsilon > 0$.

4.1. Analysis for Phase I

In this phase, we show that the number of uncolored neighbors of any node u reduces in a double-exponential fashion, (i.e., in $O(\log\log n)$ rounds) to $c_2 \log n$. This analysis has strong connections to occupancy problems [21, Problem 3.4],[2], and the edge coloring algorithm of [12].

Let $d_u(i), N_u(i), C_u(i)$ refer to the number of uncolored neighbors, the set of uncolored neighbors, and the size of the color palette of node u respectively, at the beginning of round i. Also, let $\hat{d}(i) = \max_u d_u(i)$.

Lemma 4.1 *If $d_u(1) \geq c_2 \log n$ then $d_u(c' \log\log n) \leq c_2 \log n$, with high probability for some constant $c' \geq 1$.*

Proof. The intuition behind the proof is that at the end of every round, the number of remaining uncolored neighbors decreases double-exponentially.

During round i the probability that an uncolored node u fails to get colored can be computed as: $P_u(i) := \Pr[u \text{ does not get colored during round } i] \leq \sum_{j=1}^{d_u(i)} \frac{1}{C_u(i)} \leq \frac{\hat{d}(i)}{\alpha\Delta}$, as, for c_1 sufficiently large, it holds that $C_u(i) \geq \alpha\Delta$ for $\alpha = c_1 - 1$.

The expected number of neighbors of u that are still uncolored after round i is, $\mathrm{E}[d_u(i+1)] = \sum_{v \in N_u(i)} P_v(i) \leq \hat{d}(i)^2/\alpha\Delta$.

Consider the following recurrence relation between $\hat{d}(i+1)$ and $\hat{d}(i)$ for a constant c''.

$$\hat{d}(i+1) \leq \frac{\hat{d}^2(i)}{\alpha\Delta} + \sqrt{c''\hat{d}(i)\log n}. \qquad (1)$$

Using a large deviation bound [11, 12], it can be shown that $d_u(i+1)$ exceeds its expected value by more than $\sqrt{c''\hat{d}(i)\log n}$ with probability less than n^{-2} for some constant c''. Thus, it holds that $d_u(i+1) \leq \hat{d}(i+1)$ w.h.p., for all nodes u. Solving the recurrence relation (cf. [12]) in Equation 1 for a value of i^* such that for any node u, $d_u(i^*) \leq \hat{d}(i^*) \leq c_2 \log n$ results in $i^* = O(\log\log n)$. □

Thus, at the end of $O(\log\log n)$ rounds of the algorithm, the number of uncolored neighbors for every node is at most $c_2 \log n$. This completes Phase I of the analysis.

4.2. Analysis for Phase II

The analysis in this phase consists of two sub-phases. In sub-phase II(a), we argue that along any simple oriented path of length $\sqrt{\log n}$ there exists at least one colored node, with high probability. In the second sub-phase we show that all the remaining uncolored nodes successfully get colored within $\sqrt{\log n}$ rounds. Notice that since the number of uncolored neighbors of any node at the beginning of this phase is at most $c_2 \log n$, nodes can use a color palette of size $\min\{2c_2 \log n, 2d_u\}$ for this phase as shown in the algorithm in Figure 4.

4.2.1. Analysis for Phase II(a)

We now establish the following lemma which shows that every simple oriented path of length $\sqrt{\log n}$ has at least one colored node after $O(\sqrt{\log n})$ rounds, with high probability. Let Δ_* denote the maximum number of uncolored neighbors for any node u. After phase I, it holds that $\Delta_* \leq c_2 \log n$.

Lemma 4.2 *For arbitrary $\sqrt{\log n}$–acyclic oriented graphs G at the end of $O(\sqrt{\log n})$ rounds, any simple oriented path of length $\ell = \sqrt{\log n}$ will have at least one colored node, with high probability. Further, the bit complexity of this phase is $O(\sqrt{\log n}\log\log n)$.*

Proof. The proof of this lemma is similar to the proof of Phase I of Theorem 3.1. Consider any simple oriented path P of length $\ell = \sqrt{\log n}$. Let $E_{P,i}$ denote the event that all the nodes in P are in a color conflict during a given round i. Then, along the lines of Phase I of Theorem 3.1, it holds that $\Pr[E_{P,i} \mid \cap_{j=1}^{i-1} E_{P,j}] \leq (1/2)^{\ell}$. Define E_P to be the event that the event $E_{P,i}$ occurs for $r = 4\sqrt{\log n}$ consecutive rounds. Then, it holds that $\Pr[E_P] = \Pr[\cap_{i=1}^{r} E_{P,i}] = \Pi_{i=1}^{r} \Pr[E_{P,i} \mid \cap_{j=1}^{i-1} E_{P,j}] \leq (1/2)^{\ell r}$.

Let E denote the event that there exists such a path P for which the event E_P occurs. Since the number of simple oriented paths of length P is at most $n\Delta_*^{\ell}$, $\Pr[E] \leq \sum_P \Pr[E_P] \leq n\Delta_*^{\ell}\left(\frac{1}{2}\right)^{r\ell}$. As $r = 4\sqrt{\log n}$ and $\ell = \sqrt{\log n}$ and $\Delta_* \leq c_2 \log n$, the above probability is polynomially small.

The bit complexity of this phase is $O(\sqrt{\log n}\log\log n)$ as each round, each uncolored exchanges $O(\log\log n)$ bits. □

4.2.2. Analysis for Phase II(b)

Consider connected components of uncolored nodes. At the end of Phase II(a), since any simple oriented path of length $\sqrt{\log n}$ has at least one colored node, w.h.p., it holds that each such component has only simple oriented paths of length less than $\sqrt{\log n}$ with high probability. Also, the input graph G does not have oriented cycles of length less than $\sqrt{\log n}$. For Phase II(b), we show the following lemma.

Lemma 4.3 *In Phase II(b), after less than $\sqrt{\log n}$ rounds, all nodes in G are colored properly. Further, the bit complexity of Phase II(b) is $O(\sqrt{\log n}\log\log n)$.*

Proof. The proof of this lemma is similar to that of the proof of Phase II in Theorem 3.1. This phase requires less than $\sqrt{\log n}$ rounds and each node exchanges $O(\log\log n)$ bits during every round of this phase. Thus the bit complexity of this phase is $O(\sqrt{\log n}\log\log n)$. □

From the above discussion, the following theorem holds.

Theorem 4.4 *Given a $\sqrt{\log n}$–acyclic oriented graph $G = (V, E)$ of maximum degree Δ, for any constant $\epsilon > 0$, a $(1+\epsilon)\Delta$–vertex coloring of G can be obtained in $\tilde{O}(\log\Delta + \sqrt{\log n})$ bit rounds, with high probability.*

Proof. Phase I has a bit complexity of $O(\log\Delta\log\log n)$ as for $O(\log\log n)$ rounds, each node exchanges $O(\log\Delta)$ bits. For Phase II, the bit complexity is $O(\sqrt{\log n}\log\log n)$. Adding the bit complexity of both the phases, we arrive at the theorem. □

4.3. Further Improvements

In this section, we show that the bit complexity of Phase I can be reduced to $O(\log\Delta)$ thereby reducing the bit complexity of the algorithm for arbitrary $\sqrt{\log n}$–acyclic oriented graphs to $O(\log\Delta) + \tilde{O}(\sqrt{\log n})$. We then show a tighter analysis for *high* degree graphs to arrive at a bit complexity of $O(\log\Delta + \sqrt{\log n\log\log n})$. By *high* degree graphs, we mean graphs with $\Delta \geq \log n$.

4.3.1. Improvements to the Analysis of Phase I

The tightness of the analysis stems from a gradual reduction in the number of colors during Phase I. This results in savings in the bit complexity of Phase I. To reduce the total number of bits sent by any node during Phase I we proceed as follows. Each node u maintains a threshold, $D_u^{(j)}$, on the number of uncolored neighbors, defined as $D_u^{(1)} = d_u$ and $D_u^{(j)} = \sqrt{D_u^{(j-1)}}$ for $j \geq 2$. Phase I is divided into sub-phases as follows. Let $C_u^{(j)}$ denote the size of the color palette of node u during sub-phase j. For $j \geq 1$, sub-phase j starts when the number of uncolored neighbors of u is at most $D_u^{(j)}$ and at the beginning of sub-phase j node u reduces the size of its color palette so that $C_u^{(j)} = c_1\sqrt{C_u^{(j-1)}}$ with $C_u^{(1)} = c_1\Delta$.

This effectively reduces the number of bits required to be sent in each sub-phase by a factor of 2 but the proof of Lemma 4.1 holds with minimal changes. Thus,

Algorithm PhaseI
1. $D_u := \sqrt{d_u}, C_u := c_1\Delta$.
2. While $d_u \geq c_2 \log n$ do
   3. Run Algorithm Color–Random(C_u).
   4. If $d_u \leq D_u$ then
      5. $D_u := \sqrt{D_u},\ C_u := c_1\sqrt{C_u}$
end-while.

Figure 5. Improved algorithm for Phase I.

in Phase I, it can be seen that over the $O(\log\log n)$ sub-phases the number of bits each node u sends is at most $\sum_{j=1}^{O(\log\log n)} \log C_u^{(j)} \leq \sum_{j=1}^{O(\log\log n)} 2\log c_1 + ((\log\Delta)/2^j) = O(\log\log n + \log\Delta)$. Thus, the bit complexity for Phase I reduces to $O(\log\log n + \log\Delta)$.

The modified algorithm for Phase I for node u is described in Figure 5. Using the tighter analysis for Phase I and Lemmata 4.2–4.3, we arrive at the following theorem.

Theorem 4.5 *Given a $\sqrt{\log n}$–acyclic oriented graph $G = (V, E)$ of maximum degree Δ, a $(1+\epsilon)\Delta$–vertex coloring of G for any constant $\epsilon > 0$, can be obtained in $O(\log\Delta) + \tilde{O}(\sqrt{\log n})$ bit rounds, with high probability.*

4.3.2. Improvements to Phase II

For the case of high degree graphs, we now show how to reduce the bit complexity of Phase II to $O(\sqrt{\log n \log\log n})$. The algorithm for Phase II remains the same as shown in Figure 4. The analysis of Phase II now consists of 3 sub-phases. In sub-phase II(a), we show that the number of uncolored neighbors of any node decreases to $O(\sqrt{\log n}\log\log n)$ after $O(\sqrt{\log n}/\log\log n)$ rounds with high probability. In sub-phase II(b) we then show that every simple oriented path of length $\sqrt{\log n/\log\log n}$ has at least one colored node, with high probability, after $O(\sqrt{\log n/\log\log n})$ rounds. In the final sub-phase, we show that every node can be colored in a further $O(\sqrt{\log n/\log\log n})$ rounds. In this phase, every node can use a color palette of size $2c_2\log n$.

Analysis for Phase II(a)

For sub-phase II(a), we show the following lemma.

Lemma 4.6 *In Phase II(a), in $O(\frac{\sqrt{\log n}}{\log\log n})$ rounds, the number of uncolored neighbors of any node decreases to $\sqrt{\log n}\log\log n$, with high probability. Further, the bit complexity of this sub-phase is $O(\sqrt{\log n})$.*

Proof. Consider any node u. At the end of phase I, it holds that $d_u \leq c_2\log n$, with high probability. Since the number of colors used by u is $2c_2\log n$, it also holds that $\Pr[\text{node } u \text{ fails to get colored in a given round}] \leq 1/2$.

Consider any subset A of the uncolored neighbors of u. Let E_A denote the event that all the nodes in A remain uncolored after $r = \frac{4\sqrt{\log n}}{\log\log n}$ consecutive rounds. Then, it holds that $\Pr[E_A] \leq (1/2)^{r|A|}$ using the orientation and the witnessing scheme of Theorem 3.1. Let $E_{u,s}$ denote the event that for node u, there exists a set of s uncolored neighbors at the end of r rounds. Then, $\Pr[E_{u,s}] = \Pr[\bigcup_{A\subseteq N_u, |A|=s} E_A] \leq \bigcup_{A\subseteq N_u,|A|=s}\Pr[E_A] = \binom{d_u}{s}\left(\frac{1}{2}\right)^{rs}$.

Denote by E_u the event that for node u there exist more than $\sqrt{\log n}\log\log n$ uncolored neighbors. Using Boole's inequality, $\Pr[E_u] \leq \sum_{s=\sqrt{\log n}\log\log n}^{d_u} \Pr[E_{u,s}] \leq \sum_{s=\sqrt{\log n}\log\log n}^{d_u}\binom{d_u}{s}\cdot(1/2)^{rs} \leq 2^{d_u}\cdot(1/2)^{4\log n} \leq \frac{1}{n^3}$ as $rs \geq 4\log n$. Now, denote by E the event that for some node u, the event E_u occurs. Then, $\Pr[E] = \Pr[\bigcup_{u\in V} E_u] \leq 1/n^2$. Thus, the number of uncolored neighbors of any node decreases to $\sqrt{\log n}\log\log n$ with high probability after $4\sqrt{\log n}/\log\log n$ rounds.

During this phase, each uncolored node exchanges $O(\log\log n)$ bits in each round as the palette size is $2c_2\log n$. Thus, the bit complexity of this sub-phase is $O(\sqrt{\log n})$. □

Analysis for Phase II(b)

At the end of sub-phase II(a), it holds that the number of uncolored neighbors of any node u is at most $\sqrt{\log n}\log\log n$. Recall that $\Delta_* = \max_u d_u$. After sub-phase II(a), it holds that $\Delta_* \leq \sqrt{\log n}\log\log n$.

Lemma 4.7 *In Phase II(b), in $16\sqrt{\log n/\log\log n}$ rounds, in every simple oriented path of length $\sqrt{\log n/\log\log n}$ there is at least one node that gets colored, with high probability. Further, the bit complexity of this sub-phase is $O(\sqrt{\log n\log\log n})$.*

Proof. Consider any simple oriented path P of uncolored nodes of length $\ell = \sqrt{\log n/\log\log n}$. Denote by E_P the event that no node in P gets colored in $16\sqrt{\log n/\log\log n}$ rounds. Then, it holds that (cf. Theorem 3.1) $\Pr[E_P] \leq \left(\frac{\sqrt{\log n}\log\log n}{2c_2\log n}\right)^{\ell\cdot 16\sqrt{\log n/\log\log n}} \leq \left(\frac{\log\log n}{2c_2\sqrt{\log n}}\right)^{\frac{16\log n}{\log\log n}} \leq \frac{1}{n^4}$, if n is sufficiently large. In the above, the first inequality holds since the number of uncolored neighbors is $\sqrt{\log n}\log\log n$ and the number of colors that u can choose from is $2c_2\log n$.

Let E denote the event that there exists a simple oriented path P of length ℓ such that for path P, the event E_P occurs. The number of simple oriented paths of length $\ell = \sqrt{\log n/\log\log n}$ is at most $n\cdot\Delta_*^\ell$. Thus, $\Pr[E] = \Pr[\bigcup_P E_P] \leq \sum_{j=1}^{n\Delta_*^\ell} 1/n^4 \leq \Delta_*^\ell/n^3$.

The above probability is polynomially small since $\Delta_* \leq \sqrt{\log n}\log\log n$. Thus, along any simple oriented path of length $\sqrt{\log n/\log\log n}$, at least one node gets colored

with high probability at the end of $16\sqrt{\log n/\log\log n}$ rounds.

The bit complexity of this sub-phase is easily seen to be $O(\sqrt{\log n \log\log n})$ as in each round, each uncolored node exchanges $O(\log\log n)$ bits. □

This completes the analysis for Phase II(b). In Phase II(c), using arguments similar to that of Lemma 4.3, it can be shown that in a further $\sqrt{\log n/\log\log n}$ rounds, every node gets colored, with high probability. The bit complexity of Phase II(c) is $O(\sqrt{\log n \log\log n})$. Putting together everything, we arrive at the following theorem.

Theorem 4.8 *Given a $\sqrt{\log n}$–acyclic oriented graph $G = (V, E)$ of maximum degree $\Delta \geq \log n$, for any constant $\epsilon > 0$, a $(1+\epsilon)\Delta$–vertex coloring of G can be obtained in $O(\log\Delta + \sqrt{\log n \log\log n})$ bit rounds, with high probability.*

Notice that for Theorem 4.8 to hold, the input graph only needs to be $\sqrt{\log n/\log\log n}$–acyclic, but we stated the theorem with the $\sqrt{\log n}$–acyclicity assumption for the sake of consistency.

The following corollary can be easily obtained showing that for the case of dense $\sqrt{\log n}$–acyclic oriented graphs, our result on the bit complexity is close to the worst-case optimal.

Corollary 4.9 *Given an arbitrary $\sqrt{\log n}$–acyclic oriented graph $G = (V, E)$ with $\Delta = \Omega(2^{\sqrt{\log n \log\log n}})$, for any $\epsilon > 0$, a $(1+\epsilon)\Delta$-vertex coloring can be obtained in $O(\log\Delta)$ bit rounds, with high probability.*

5. Conclusions

We presented algorithms for distributed vertex coloring using a simple and natural model. While our results are tight in general, a related question to ask is whether any further conditions on the orientation would result in better bounds or whether certain orientations outperform other orientations. For example, if the orientation or the graph is known to be acyclic, would it be possible to color in fewer bit rounds?

References

[1] L. Barriere, P. Flocchini, P. Fraigniaud, and N. Santoro. Can we elect if we cannot compare? In *ACM SPAA*, pages 324–332, 2003.

[2] M. Bender, M. Farach-Colton, S. He, B. Kuszmaul, and C. Leiserson. Adversarial contention resolution for simple channels. In *ACM SPAA*, pages 325–332, 2005.

[3] M. Choy and A. K. Singh. Efficient fault tolerant algorithms for resource allocation in distributed systems. In *ACM STOC*, pages 169–177, 1992.

[4] R. Cole and U. Vishkin. Deterministic coin tossing and accelerating cascades: micro and macro techniques for designing parallel algorithms. In *ACM STOC*, pages 206–219, 1986.

[5] D. P. Dubhashi and A. Panconesi. Near-optimal distributed edge coloring. In *ESA*, pages 448–459, 1995.

[6] U. Feige and J. Kilian. Zero-knowledge and chromatic number. In *Proc. of the Annual Conference on Computational Complexity*, 1996.

[7] I. Finocchi, A. Panconesi, and R. Silvestri. Experimental analysis of simple, distributed vertex coloring algorithms. In *ACM SODA*, pages 606–615, 2002.

[8] P. Flocchini, B. Mans, and N. Santoro. On the impact of sense of direction on message complexity. *Information Processing Letters*, 63(1):23–31, 1997.

[9] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. 1979.

[10] A. Goldberg, S. Plotkin, and G. Shannon. Parallel symmetry breaking in sparse graphs. In *ACM STOC*, pages 434–446, 1987.

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

[12] D. A. Grable and A. Panconesi. Nearly optimal distributed edge colouring in $O(\log\log n)$ rounds. *Random Structures and Algorithms*, 10(3):385–405, 1997.

[13] O. Johansson. Simpled distributed $\Delta+1$ coloring of graphs. *Information Processing Letters*, 70:229–232, 1999.

[14] M. Karchmer and J. Naor. A fast parallel algorithm to color a graph with Δ colors. *J. Algorithms*, 9:83–91, 1988.

[15] R. Karp and A. Widgerson. A fast parallel algorithm for the maximal independent set problem. *J. ACM*, 32(4):762–773, 1985.

[16] V. Kumar, M. Marathe, S. Parthasarathy, and A. Srinivasan. End-to-end packet-scheduling in wireless ad-hoc networks. In *ACM SODA*, pages 1021–1030, 2004.

[17] N. Linial. Locality in distributed graph algorithms. *SIAM J. of Computing*, 21:193–201, 1992.

[18] M. Luby. A simple parallel algorithm for the maximal independent set problem. In *ACM STOC*, pages 1–10, 1985.

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

[20] G. De Marco and A. Pelc. Fast distributed graph coloring with $O(\Delta)$ colors. In *ACM SODA*, pages 630–635, 2001.

[21] R. Motwani and P. Raghavan. *Randomized Algorithms*. Cambridge University Press, 1995.

[22] M. Onus, A. Richa, K. Kothapalli, and C. Scheideler. Constant density spanners for wireless ad-hoc networks. In *ACM SPAA*, 2005.

[23] A. Panconesi and A. Srinivasan. Fast randomized algorithms for distributed edge coloring. In *ACM PODC*, pages 251–262, 1992.

[24] A. Panconesi and A. Srinivasan. On the complexity of distributed network decomposition. *J. Algorithms*, 20(2):356–374, 1996.

[25] G. Singh. Efficient leader election using sense of direction. *Distributed Computing*, 10(3):159–165, 1997.