SIAM J. COMPUT.
Vol. 43, No. 1, pp. 72–95

DISTRIBUTED $(\Delta + 1)$-COLORING IN LINEAR (IN Δ) TIME*

LEONID BARENBOIM†, MICHAEL ELKIN†, AND FABIAN KUHN‡

Abstract. The distributed $(\Delta + 1)$-coloring problem is one of the most fundamental and well-studied problems in distributed algorithms. Starting with the work of Cole and Vishkin in 1986, a long line of gradually improving algorithms has been published. The state-of-the-art running time, prior to our work, is $O(\Delta \log \Delta + \log^* n)$, due to Kuhn and Wattenhofer [*Proceedings of the* 25*th Annual ACM Symposium on Principles of Distributed Computing*, Denver, CO, 2006, pp. 7–15]. Linial [*Proceedings of the* 28*th Annual IEEE Symposium on Foundation of Computer Science*, Los Angeles, CA, 1987, pp. 331–335] proved a lower bound of $\frac{1}{2} \log^* n$ for the problem, and Szegedy and Vishwanathan [*Proceedings of the* 25*th Annual ACM Symposium on Theory of Computing*, San Diego, CA, 1993, pp. 201–207] provided a heuristic argument that shows that algorithms from a wide family of locally iterative algorithms are unlikely to achieve a running time smaller than $\Theta(\Delta \log \Delta)$. We present a deterministic $(\Delta+1)$-coloring distributed algorithm with running time $O(\Delta) + \frac{1}{2} \log^* n$. We also present a trade-off between the running time and the number of colors, and devise an $O(\lambda \cdot \Delta)$-coloring algorithm, with running time $O(\Delta/\lambda + \log^* n)$, for any parameter $\lambda > 1$. Our algorithm breaks the heuristic barrier of Szegedy and Vishwanathan and achieves running time which is linear in the maximum degree Δ. On the other hand, the conjecture of Szegedy and Vishwanathan may still be true, as our algorithm does not belong to the family of locally iterative algorithms. On the way to this result we study a generalization of the notion of graph coloring, which is called defective coloring [L. Cowen, R. Cowen, and D. Woodall, *J. Graph Theory*, 10 (1986), pp. 187–195]. In an m-defective p-coloring the vertices are colored with p colors so that each vertex has up to m neighbors with the same color. We show that an m-defective p-coloring with reasonably small m and p can be computed very efficiently in the distributed setting. We also develop a technique to employ multiple defective colorings of various subgraphs of the original graph G for computing a $(\Delta + 1)$-coloring of G. We believe that these techniques are of independent interest.

Key words. legal-coloring, defective-coloring, distributed algorithms

AMS subject classifications. 68W15, 05C15

DOI. 10.1137/12088848X

1. Introduction and related work.

In the *message passing model* of distributed computation [35] one is given an undirected n-vertex graph $G = (V, E)$, whose vertices host processors. The vertices have distinct identity numbers in the range[1] $\{1, 2, \ldots, n\}$. Each vertex v can communicate with its *neighbors*, i.e., vertices u such that $(v, u) \in E$. The communication is synchronous, i.e., it occurs in discrete rounds. Messages are sent in the beginning of each round. A message that is sent in a round R arrives at its destination before the next round $R+1$ starts. All vertices start

*Received by the editors August 20, 2012; accepted for publication (in revised form) October 1, 2013; published electronically January 28, 2014. This paper is a compilation of two separate conference papers [5, 22]. These two papers achieved (essentially) the same results by two different techniques. Specifically, the approach of [5] is combinatorial, while the work of [22] employs both combinatorial and algebraic ideas. See also [6] for a preprint version of [5].

http://www.siam.org/journals/sicomp/43-1/88848.html

†Department of Computer Science, Ben-Gurion University of the Negev, Beer-Sheva 84105, Israel (leonidba@cs.bgu.ac.il, elkinm@cs.bgu.ac.il). The research of these authors was supported by the Israeli Academy of Science, grant 483/06. The first author was also supported by the Adams Fellowship Program of the Israel Academy of Sciences and Humanities.

‡Department of Computer Science, University of Freiburg, 79110 Freiburg, Germany (kuhn@cs.uni-freiburg.de).

[1] If the identities are in the larger range $\{1, 2, \ldots, N\}$, for some $N \gg n$, this has only a minor effect on the results discussed in this paper. Specifically, one need only replace the additive term $\frac{1}{2} \log^* n$ by $\frac{1}{2} \log^* N$ in all of the results. Since $\log^*$ is a very slowly growing function, $\log^* N = \log^* n + O(1)$, for, e.g., $N \leq 2^{O(n)}$.

executing an algorithm simultaneously. The *running time* is the number of rounds that elapse from the beginning of an execution of a distributed algorithm until the last vertex terminates. In this model it is assumed that all vertices know the size n and the maximum degree $\Delta(G)$. However, these assumptions often can be eliminated. (We discuss this issue with respect to our algorithms in section 2.)

1.1. Coloring. Let Δ denote the maximum degree of G. Coloring G with $\Delta+1$ or fewer colors so that, for every pair of neighbors u and w, the color of u is different from that of w (henceforth referred to as a $(\Delta+1)$-*coloring*) is one of the most central problems in the area of distributed algorithms. In addition to its theoretical appeal, it is well motivated by many network tasks that are based on a graph coloring subroutine. These tasks include scheduling, resource-allocation, symmetry-breaking, workload balancing, etc. [14, 16, 19, 28].

Distributed coloring has been the focus of intensive research since the mid-eighties. Cole and Vishkin [9] devised a deterministic $O(\log^* n)$-time 3-coloring algorithm for oriented cycles. The algorithm was generalized to a deterministic $2^{O(\Delta)} + O(\log^* n)$-time $(\Delta+1)$-coloring algorithm for general graphs by Goldberg and Plotkin in [15, 36]. Goldberg, Plotkin, and Shannon [16] improved the result of [15] and devised a $(\Delta+1)$-coloring algorithm with running time $O(\Delta^2 + \log^* n)$. They also devised a $(\Delta+1)$-coloring algorithm with running time $O(\Delta \log n)$. (See also [3] for a more explicit version of the algorithm of [16].) Recently, the above bounds were improved by Kuhn and Wattenhofer in [27], where a deterministic $(\Delta+1)$-coloring algorithm that requires $O(\Delta \log \Delta + \log^* n)$ rounds was presented.

In [29, 30] Linial devised an $O(\Delta^2)$-coloring algorithm with running time $\log^* n + O(1)$. Moreover, Linial also proved a lower bound of $\frac{1}{2}\log^* n - O(1)$ for the complexity of the $f(\Delta)$-coloring problem, for any function $f(\cdot)$. Szegedy and Vishwanathan [39] improved the upper bound of [29] and devised an $O(\Delta^2)$-coloring algorithm with running time $\frac{1}{2}\log^* n + O(1)$. (See also [33] for a more explicit construction.) Szegedy and Vishwanathan have also presented a heuristic lower bound for the complexity of $(\Delta+1)$-coloring. They considered a class of algorithms that they called "locally iterative algorithms" and gave a heuristic argument that shows that no locally iterative $(\Delta+1)$-coloring algorithm "is likely to terminate in less than $\Omega(\Delta \log \Delta)$ rounds." Note that most currently known deterministic distributed coloring algorithms belong to the class of locally iterative algorithms. The best-known locally iterative $(\Delta+1)$-coloring algorithms are described in [27] and have time complexities $O(\Delta \log \Delta + \log^* n)$ (deterministic) and $O(\Delta \log \log n)$ (randomized).

The $O(\Delta \log \Delta + \log^* n)$-time algorithm from [27] is the fastest known deterministic algorithm for networks with moderate[2] maximum degree Δ. For dense networks, the best algorithms are based on decompositions of the network into clusters with small diameter. In [3, 34], it is shown that in $2^{O(\sqrt{\log n})}$ rounds, the network graph $G = (V, E)$ can be partitioned into clusters of diameter $2^{O(\sqrt{\log n})}$ so that the clusters can be colored with $2^{O(\sqrt{\log n})}$ colors (adjacent clusters get different colors). This partition can directly be applied to obtain a deterministic $2^{O(\sqrt{\log n})}$-time $(\Delta+1)$-coloring algorithm. We remark, however, that the algorithms of [3, 34] require sending messages of size proportional to the number of edges induced by a cluster, which is $\Theta(|E|)$ in the worst case.

Except for sparse networks, randomized distributed coloring algorithms are more efficient than the best-known deterministic algorithms. Using randomized algorithms

[2]Specifically, for $\Delta = 2^{o(\sqrt{\log n})}$.

Table 1

A concise comparison of previous $(\Delta + 1)$*-coloring algorithms with our algorithm. All listed algorithms, except the algorithm of* [27]*, which requires* $O(\Delta \log \log n)$ *time, are deterministic.*

Running time	Reference	Running Time	Reference
$2^{O(\Delta)} + O(\log^* n)$	Goldberg, Plotkin [15]	$O(\Delta^2) + \frac{1}{2} \log^* n$	Szegedy, Vishwanathan [39]
$O(\Delta^2 + \log^* n)$	Goldberg et al. [16]	$O(\Delta \log \Delta + \log^* n)$	Kuhn, Wattenhofer [27]
$O(\Delta \cdot \log n)$	Goldberg et al. [16]	$O(\Delta \log \log n)$ rand.	Kuhn, Wattenhofer [27]
$O(\Delta^2) + \log^* n$	Linial [29]	$O(\Delta) + \frac{1}{2} \log^* n$	This paper

for the maximal independent set problem (cf. section 1.2) from [1, 31] together with a reduction described in [30], it is possible to compute a $(\Delta + 1)$-coloring in $O(\log n)$ rounds (in expectation and with high probability). An algorithm that achieves the same result directly is described in [20]. However, for $\Delta = o(\log n)$ our deterministic $(\Delta + 1)$-coloring algorithm outperforms all the existing deterministic or randomized algorithms.

In this paper we improve upon the state-of-the-art deterministic $O(\Delta \log \Delta + \log^* n)$ upper bound of [27] on the complexity of the $(\Delta + 1)$-coloring problem and devise a deterministic $(\Delta + 1)$-coloring algorithm with running time $O(\Delta) + \frac{1}{2} \log^* n$. This is the first $(\Delta + 1)$-coloring algorithm with running time linear in Δ. Moreover, our algorithm breaks the heuristic barrier of $\Omega(\Delta \log \Delta)$ due to Szegedy and Vishwanathan [39]. On the other hand, the conjecture of Szegedy and Vishwanathan may still be true, as our algorithm does not belong to the class of locally iterative algorithms. Note also that by the lower bound of Linial [30], the second term $\frac{1}{2} \log^* n$ in the running time of our algorithm cannot be improved. See Table 1 for a concise comparison between previous results and our algorithm.

Also, we generalize our result and devise a trade-off between the running time of the algorithm and the number of colors it employs. Specifically, for a parameter $\lambda > 1$, a variant of our algorithm computes an $O(\lambda \cdot \Delta)$-coloring within $O(\Delta/\lambda + \log^* n)$ time.

1.2. Maximal independent set. A subset $I \subseteq V$ of vertices of the graph G is called a *maximal independent set* (henceforth, MIS) of G if

(1) for every pair $u, w \in V$ of neighbors, either u or w do not belong to I, and

(2) for every vertex $v \in V$, either $v \in I$ or there exists a neighbor $w \in V$ of v that belongs to I.

The MIS problem is closely related to the coloring problem, and, similarly to the latter problem, the MIS problem is one of the most central and intensively studied problems in distributed algorithms [31, 1, 3, 34, 24, 25]. Using a standard reduction, our $(\Delta + 1)$-coloring algorithm directly gives rise to a deterministic distributed algorithm with running time $O(\Delta) + \frac{1}{2} \log^* n$ for computing an MIS on graphs with maximum degree Δ. As for the coloring problem, the previous state of the art for deterministic algorithms depends on Δ. For graphs with moderate degrees, the best-known bound is obtained using the same standard reduction in conjunction with the algorithm of Kuhn and Wattenhofer [27] that requires $O(\Delta \log \Delta + \log^* n)$ time. As for coloring, using network decompositions, it is possible to compute an MIS deterministically in $2^{O(\sqrt{\log n})}$ rounds. The state of the art randomized algorithms for the MIS problem on general graphs are due to Luby [31] and Alon, Babai, and Itai [1] and require $O(\log n)$ time. Hence for graphs with maximum degree $\Delta = o(\log n)$, our (deterministic) algorithm improves the state of the art (randomized or deterministic) for the MIS problem. As for coloring, for graphs with $\Delta = o(2^{c \cdot \sqrt{\log n}})$, our algorithm improves the state of the art with respect to deterministic algorithms.

The distributed MIS and coloring problems have also been studied for various special classes of graphs. Specifically, computing an MIS and colorings on graphs with bounded local independence has recently been studied intensively [26, 23, 37]. In [37], it is shown that the $(\Delta + 1)$-coloring problem and the MIS problem can both be computed in $O(\log^* n)$ time for these graphs. In other recent developments, efficient algorithms for coloring and MIS problems for graphs with small arboricity were devised in [4]. Our results give rise directly to improved algorithms for coloring and computing MIS for graphs of bounded arboricity. Specifically, in [4] it was shown that graphs of arboricity at most a can be $O(a \cdot t)$-colored in time $O(\frac{a}{t} \log n + a \log a)$ for any parameter t, $1 \le t \le a$. As argued in [4], this result implies that in $O(a\sqrt{\log n} + a \log a)$ time one can compute an MIS on graphs with $a = \Omega(\sqrt{\log n})$. Our results in the current paper imply an $O(a \cdot t)$-coloring algorithm with running time $O(\frac{a}{t} \log n + a)$, and an algorithm for computing an MIS on these graphs within $O(a\sqrt{\log n})$ time, for $a = \Omega(\sqrt{\log n})$.

The main technique in [4] is an efficient algorithm for constructing a Nash–Williams decomposition distributively, and all other results there rely on this algorithm. However, as shown in [4], constructing a Nash–Williams decomposition requires $\Omega(\frac{\log n}{\log \log n})$ time. Consequently, one cannot employ Nash–Williams decompositions to achieve a running time of $O(\Delta) + \frac{1}{2} \log^* n$. As discussed below, the algorithms in the present paper rely on different ideas.

1.3. Our techniques. We study a generalized variant of coloring, called *defective coloring*. (It is also known as *improper coloring*.) For a nonnegative integer m and a positive integer χ, an *m-defective χ-coloring* of a graph $G = (V, E)$ is a coloring that employs up to χ colors and satisfies that, for every vertex $v \in V$, there are at most m neighbors of v that are colored by the same color as v. Note that the standard notion of χ-coloring corresponds in this terminology to 0-defective χ-coloring. Defective coloring was introduced by [10] and was extensively studied from a graph-theoretic perspective [2, 17, 11, 13]. As a consequence of a result from the 1960s [32], it is known that for every $k > 0$, every graph G with maximum degree Δ has k-coloring with defect at most $\lfloor \Delta/k \rfloor$. Cowen, Goddard, and Jesurum [11] have also devised efficient *centralized* algorithms for computing defective colorings for various families of graphs. However, to the best of our knowledge, we are the first to develop *distributed* algorithms for computing defective colorings.

We show that m-defective χ-colorings for reasonably small values of m and χ can be efficiently computed in a distributed manner. Also, we demonstrate that defective colorings of various appropriate subgraphs of the input graph G can be combined into a $(\Delta + 1)$-coloring of G.

We present two different methods for efficiently computing defective colorings. The first is a combinatorial method, and the second is a set-theoretic method based on certain algebraic constructions. While the first method does not require computing complex constructions by the vertices, the second method is more efficient in terms of running time. Interestingly, each of the methods can be used independently to devise an algorithm that computes a $(\Delta + 1)$-coloring in time $O(\Delta) + \frac{1}{2} \log^* n$. These two methods for computing defective coloring that we develop in this paper are the first (and up to this date the only known) efficient distributed algorithms for this task. We believe that they constitute a fundamental technical contribution of our paper.

Note that our algorithm does not fall into the framework of locally iterative algorithms. In this framework the algorithm starts with computing an initial coloring that may possibly employ many colors and proceeds iteratively. In each iteration the number of colors is reduced until no further progress can be achieved. Very roughly speaking, our algorithm partitions the graph to many vertex-disjoint subgraphs, computes defective coloring for each of them, and combines them into a unified $(\Delta+1)$-coloring of the original graph. The heuristic barrier of $\Omega(\Delta \log \Delta)$ of Szegedy and Vishwanathan [39] for locally iterative algorithms suggests that this completely different approach that our algorithm employs is necessary for achieving a running time that is linear in Δ for the $(\Delta+1)$-coloring problem.

1.4. Consequent work. Since the preliminary conference versions of the current paper were published in [5, 22], our results and techniques were used in a number of subsequent papers. In particular, Schneider and Wattenhofer devised a randomized algorithm that computes a $(\Delta+1)$-coloring in $O(\log \Delta + \sqrt{\log n})$ rounds in [38]. Barenboim and Elkin [7] devised deterministic algorithms for computing $O(\Delta)$-coloring in $O(\Delta^\epsilon \log n)$ time, and $O(\Delta^{1+\epsilon})$-coloring in $O(\log \Delta \log n)$ time, for an arbitrarily small positive constant ϵ. For graphs with bounded neighborhood independence[3] the latter results were improved in [8]. There the authors devised an $O(\Delta)$-coloring in $O(\Delta^\epsilon + \log^* n)$ time, and an $O(\Delta^{1+\epsilon})$-coloring in $O(\log \Delta + \log^* n)$ time, for this family of graphs.

1.5. Structure of the paper. In section 2 we introduce the notation and terminology used throughout the paper. In section 3 we describe our algorithm for computing defective colorings. Section 3.1 is devoted to our combinatorial method, and section 3.2 is devoted to the set-theoretic method. An alternative set-theoretic method for computing defective colorings is described in Appendix A. In section 4 we employ the algorithm for defective coloring to devise our $(\Delta+1)$-coloring algorithm. This algorithm is then used to obtain the trade-off between the running time and the number of colors. In section 5 we outline a number of possible directions for further research.

2. Preliminaries. For an integer $k \geq 1$, we will frequently make use of the common abbreviation $[k] := \{1, 2, \ldots, k\}$. Further, unless the base value is explicitly specified, log stands for logarithms to base 2. As usual, ln denotes the natural logarithm. For a nonnegative integer i, the *iterative log-functions* $\log^{(i)}(\cdot)$ and $\ln^{(i)}(\cdot)$ are defined as follows. For an integer $n > 0$, $\log^{(0)} n := n$, and $\log^{(i+1)} n := \log(\log^{(i)} n)$, for every $i = 0, 1, 2, \ldots$. (Analogously, $\ln^{(0)} n := n$, and $\ln^{(i+1)} n := \ln(\ln^{(i)} n)$.) Further, $\log^* n$ is defined as $\log^* n := \min\{i \mid \log^{(i)} n \leq 2\}$.

The graph $G' = (V', E')$ is a *subgraph* of $G = (V, E)$, denoted by $G' \subseteq G$, iff $V' \subseteq V$ and $E' \subseteq E$. The set of neighbors of v in G is denoted by $\Gamma(v)$. For a subset $U \subseteq V$, the degree of v with respect to U, denoted $deg(v, U)$, is the number of neighbors of v in U. We also write $deg(v) = deg(v, V)$. The *maximum degree* of a vertex in G, denoted by $\Delta = \Delta(G)$, is defined as $\Delta(G) := \max_{v \in V} deg(v)$.

A coloring $\varphi : V \rightarrow \mathbb{N}$ that satisfies $\varphi(v) \neq \varphi(u)$ for each edge $(u, v) \in E$ is called a *legal coloring.* (It is also known as a *proper coloring.*) For positive integers m and p, a coloring $\varphi' : V \rightarrow \{1, 2, \ldots, p\}$ that satisfies that for every vertex $v \in V$, the number of neighbors u of v with $\varphi'(u) = \varphi'(v)$ is at most m is called an *m-defective p-coloring.*

[3] *Graphs of bounded neighborhood independence* are graphs in which the number of independent vertices in each (1-hop-) neighborhood is bounded by a constant.

We also say that the graph G is *m-defective p-colored* by φ'. The *defect* of a vertex v with respect to φ', denoted by $def_{\varphi'}(v)$, is the number of neighbors u of v with $\varphi'(u) = \varphi'(v)$. The defect of a coloring φ is defined as $def(\varphi) = \max\left\{def_{\varphi}(v) \mid v \in V\right\}$. For convenience, we use the notion m-defective p-coloring more broadly. Specifically, the p colors do not necessarily have to be taken from the range $\{1, 2, \ldots, p\}$. Instead, the p colors can be taken from the range $\{t+1, t+2, \ldots, t+p\}$, for a positive integer parameter t that is known to all vertices. (But each vertex must still have no more than m neighbors that are colored with its color.) To be strict with the above definition, all vertices can subtract t from their colors.

Some of our algorithms use as a black-box a procedure due to Kuhn and Wattenhofer [27]. This procedure accepts as input a graph G with maximum degree Δ, and an initial legal m-coloring, and it produces a $(\Delta + 1)$-coloring of G within time $(\Delta + 1) \cdot \lceil \log(m/(\Delta + 1)) \rceil = O(\Delta \cdot \log(m/\Delta))$. We will refer to this procedure as the *KW iterative procedure*. The KW iterative procedure is used in [27] to devise a $(\Delta + 1)$-coloring algorithm (henceforth, *KW algorithm*) with running time $O(\Delta \log \Delta + \log^* n)$.

In all our algorithms we assume that all vertices know the number of vertices n and the maximum degree Δ of the input graph G before the computation starts. This assumption is required for many coloring algorithms, and, in particular, it is required in the algorithms of Linial [30], Szegedy and Vishwanathan [39], and Kuhn and Wattenhofer [27] that are used in a black-box manner by our algorithms. In a paper subsequent to ours [21] Korman, Sereni, and Viennot showed that these assumptions can be eliminated without compromising the running time or the number of colors of our algorithms by more than a constant factor. In other words, they showed how to extend our algorithm to compute $(\Delta + 1)$-coloring within $O(\Delta + \log^* n)$ time without a priori knowledge of n and Δ. Their technique is quite general and applicable to additional algorithms as well. See [21] for further details. The vertices are also assumed to know their identity numbers (Ids).

Although our computational model allows sending messages of arbitrary size, all algorithms in this paper employ *short* messages, that is, messages with $O(\log n)$ bits each.

3. Defective coloring.

3.1. The combinatorial method.

3.1.1. Procedure Refine. In this section we present an algorithm that produces a defective coloring using a combinatorial method. Many $(\Delta + 1)$-coloring algorithms employ the following standard technique. Whenever a vertex is required to select a color it selects a color that is different from the colors of all its neighbors. Its neighbors select their colors in different rounds. On the other hand, if one is interested in a defective coloring, a vertex can select a color that is used by some of its neighbors. Moreover, some neighbors can perform the selection in the same round. Consequently, the computation can potentially be significantly more efficient than that of a $(\Delta + 1)$-coloring, and we can also aim for a smaller number of colors.

We devise an $O(\Delta/p)$-defective p^2-coloring algorithm. We start by presenting a procedure, called *Refine*, that accepts as input a graph with an m-defective χ-coloring φ, and a parameter p, $1 \le p \le \Delta$, for some integers m, χ, and p, and computes an $(m + \lfloor \Delta/p \rfloor)$-defective p^2-coloring in time $O(\chi)$.

For each vertex v, let $\mathcal{S}(v)$ (resp., $\mathcal{G}(v)$) denote the set of neighbors u of v that have colors smaller (resp., larger) than the color of v, i.e., that satisfy $\varphi(u) < \varphi(v)$

(resp., $\varphi(u) > \varphi(v)$). Procedure Refine computes a new coloring φ'. It proceeds in two stages. In the first stage, each vertex v computes a new color $\psi(v)$ from the range $\{1, 2, \ldots, p\}$ in the following way. Once v receives the color $\psi(u)$ from each of its neighbors u from $\mathcal{S}(v)$, it sets $\psi(v)$ to be the color from $\{1, 2, \ldots, p\}$ that is used by the minimal number of these neighbors, breaking ties arbitrarily. (In other words, v selects a color i such that for every $j = 1, 2, \ldots, p$, it holds that $|\{u \in \mathcal{S}(v) : \psi(u) = i\}| \leq |\{u \in \mathcal{S}(v) : \psi(u) = j\}|$). Then it sends its selection $\psi(v)$ to all its neighbors. In the second stage, each vertex v computes a new color $\Psi(v)$ from the range $\{1, 2, \ldots, p\}$ in a similar way, except that now it considers only neighbors from $\mathcal{G}(v)$. Once v receives the color $\Psi(w)$ from each of its neighbors w from $\mathcal{G}(v)$, it sets $\Psi(v)$ to be the color from $\{1, 2, \ldots, p\}$ that is used by the minimal (with respect to Ψ) number of these neighbors. Then it sends its selection $\Psi(v)$ to all its neighbors.

Once the vertex v has computed both colors $\psi(v)$ and $\Psi(v)$, it sets its final color $\varphi'(v) = (\Psi(v) - 1) \cdot p + \psi(v)$. Intuitively, the color $\varphi'(v)$ can be seen as a pair $(\Psi(v), \psi(v))$. This completes the description of Procedure Refine. Next, we show that the procedure is correct.

LEMMA 3.1. *The coloring φ' produced by Procedure Refine is an $(m + \lfloor \Delta/p \rfloor)$-defective p^2-coloring.*

Proof. First, observe that for each vertex v, it holds that $1 \leq \psi(v), \Psi(v) \leq p$, and thus $1 \leq \varphi'(v) \leq p^2$. It is left to show that for each vertex v, the number of neighbors u of v with $\varphi'(u) = \varphi'(v)$ is at most $m + \lfloor \Delta/p \rfloor$. Each vertex v has at most m neighbors z such that $\varphi(v) = \varphi(z)$. By the pigeonhole principle, the number of neighbors u of v with $\varphi(u) < \varphi(v)$ and $\psi(u) = \psi(v)$ is at most $\lfloor |\mathcal{S}(v)|/p \rfloor$, since v selected $\psi(v)$ to be the color from $\{1, 2, \ldots, p\}$ that is used by the minimal number of its neighbors from $\mathcal{S}(v)$. Similarly, the number of neighbors w of v with $\varphi(w) > \varphi(v)$ and $\Psi(w) = \Psi(v)$ is at most $\lfloor |\mathcal{G}(v)|/p \rfloor$. Observe that for any neighbor u of v, if $\varphi'(u) = \varphi'(v)$, then $\psi(u) = \psi(v)$ and $\Psi(u) = \Psi(v)$. Consequently, the number of neighbors u with $\varphi'(u) = \varphi'(v)$ is at most $m + \lfloor |\mathcal{S}(v)|/p \rfloor + \lfloor |\mathcal{G}(v)|/p \rfloor \leq m + \lfloor deg(v)/p \rfloor \leq m + \lfloor \Delta/p \rfloor$. ∎

The two stages of Procedure Refine can be executed in parallel. Thus, Refine can be executed within χ rounds. This is argued formally in the next lemma.

LEMMA 3.2. *The time complexity of Procedure Refine is χ.*

Proof. Each vertex $v \in V$ can learn the colors $\varphi(u)$ of all its neighbors $u \in \Gamma(v)$ in the first round. We prove by induction on i that after i rounds, $i = 1, 2, \ldots, \chi$, each vertex with $\varphi(v) \leq i$ has selected its color $\psi(v)$. For the base case, consider all the vertices v with $\varphi(v) = 1$. There are no vertices u with $\varphi(u) < 1$, and thus, each vertex v with $\varphi(v) = 1$ selects the color $\psi(v)$ in the first round. Now, assume that after $i-1$ rounds, each vertex with $\varphi(v) \leq i-1$ has selected its color $\psi(v)$. Then, by the induction hypothesis, in round i, for a vertex v with $\varphi(v) = i$, all the neighbors u of v satisfying $\varphi(u) < \varphi(v) = i$ have selected their color $\psi(u)$ in round $i-1$ or earlier. Hence, if v has not selected the color $\psi(v)$ before round i, it necessarily selects it in round i. Therefore, after χ rounds all the vertices in the graph have selected the color $\psi(v)$ and the first stage is completed. Similarly, the second stage is completed after another χ rounds. The computation of $\varphi'(v)$ from $\psi(v)$ and $\Psi(v)$ is performed immediately after the second stage is finished, and it requires no additional communication. Finally, note that the two stages can be executed in parallel in χ rounds. ∎

We summarize this section with the following corollary.

COROLLARY 3.3. *For positive integers χ, m, and p, suppose that Procedure Refine is invoked on a graph G with maximum degree Δ. Suppose also that G is m-defective*

χ-colored. Then the procedure produces an $(m + \lfloor \Delta/p \rfloor)$-defective p^2-coloring of G. It requires at most χ rounds.

3.1.2. Procedure Defective-Color. In this section we devise an algorithm called *Procedure Defective-Color*. The algorithm accepts as input a graph $G = (V, E)$, and two integer parameters p, q such that $1 \le p \le \Delta$, $p^2 < q$, and $q < c' \cdot \Delta^2$, for some positive constant $c' > 0$. It computes from scratch an $O(\frac{\log \Delta}{\log(q/p^2)} \cdot \Delta/p)$-defective p^2-coloring of G in time $O(\log^* n + \frac{\log \Delta}{\log(q/p^2)} \cdot q)$. In particular, if we set $q = \Delta^\epsilon \cdot p^2$ for an arbitrarily small positive constant ϵ, we get an $O(\Delta/p)$-defective p^2-coloring algorithm with running time $O(\log^* n + \Delta^\epsilon \cdot p^2)$, for any p such that $1 \le p < \sqrt{c'} \Delta^{1-\frac{1}{2}\epsilon}$.

The algorithm starts by computing an $O(\Delta^2)$-coloring of the input graph. This coloring φ can be computed in $\log^* n + O(1)$ time from scratch using the algorithm of Linial [30]. Let c be a positive constant such that $c \cdot \Delta^2$ is an upper bound on the number of colors employed. Let $h = \lfloor c \cdot \Delta^2/q \rfloor$. (The constant c' mentioned in the beginning of the section is chosen to be at most c so that $h \ge 1$.) Each vertex v with $1 \le \varphi(v) \le h \cdot q$ joins the set V_j with $j = \lceil \varphi(v)/q \rceil$. Vertices v that satisfy $h \cdot q < \varphi(v) \le c \cdot \Delta^2$ join the set V_h. In other words, the index j of the set V_j to which the vertex v joins is determined by $j = \min\{\lceil \varphi(v)/q \rceil, h\}$. Observe that for every index j, $1 \le j \le h - 1$, the set V_j is colored with at most q colors, and V_h is colored with at most q' colors, where $q \le q' \le 2q$. By definition, for each j, $1 \le j \le h - 1$, V_j is 0-defective q-colored (i.e., $m = 0$, $k = q$) and V_h is 0-defective q'-colored ($m = 0$, $k = q'$). For each j, $1 \le j \le h$, we denote this coloring of V_j by ψ_j. Then, for each graph $G(V_j)$ induced by the vertex set V_j, Procedure Refine is invoked on $G(V_j)$ with the parameter p, in parallel for $j = 1, 2, \ldots, h$. As a result of these invocations, each graph $G(V_j)$ is now $\lfloor \Delta/p \rfloor$-defective p^2-colored. Let φ'_j denote this coloring. Next, each vertex v selects a new color $\varphi''(v)$ by setting $\varphi''(v) = \varphi'_j(v) + (j-1) \cdot p^2$, where j is the index such that $v \in V_j$. The number of colors used by the new coloring φ'' is at most $h \cdot p^2 \le c \cdot \Delta^2 \cdot p^2/q$. It follows that the coloring φ'' is a $\lfloor \Delta/p \rfloor$-defective $(c \cdot \Delta^2 \cdot p^2/q)$-coloring of G.

This process is repeated iteratively. On each iteration the vertex set is partitioned into disjoint subsets V_j such that in each subset the vertices are colored by at most q different colors, except one subset in which the vertices are colored by at most $2q$ colors. Then, in parallel, the coloring of each subset is converted into a p^2-coloring. Consequently, in each iteration the number of colors is reduced by a factor of at least q/p^2. (Except for the last iteration, in which the number of colors is larger than p^2 but might be smaller than q, and it is reduced to p^2.) However, for a vertex v, the number of neighbors of v that are colored by the same color as v, that is, the defect $def_\varphi(v)$ of v, may grow by an additive term of $\lfloor \Delta/p \rfloor$ in each iteration. The process terminates when the entire graph G is colored by at most p^2 colors. (After $\log_{q/p^2} c \cdot \Delta^2$ iterations all vertices know that G is colored by at most p^2 colors.) In each iteration an upper bound χ on the number of currently employed colors is computed. In the last iteration, if $\chi < q$, then all the vertices join the same set V_1, and consequently $V_1 = V$, and Procedure Refine is invoked on the entire graph G. See Figure 1 for an illustration. The pseudocode of the algorithm is provided below.

In what follows we prove the correctness of Procedure Defective-Color. We start by proving the following invariant regarding the variable χ. Let χ_i denote the value of χ at the end of the ith iteration. For technical convenience, we define χ_0 to be the value of χ at the beginning of the first iteration.

LEMMA 3.4. *For $i = 0, 1, 2, \ldots$, after the ith iteration, the number of colors employed by φ is at most χ_i.*

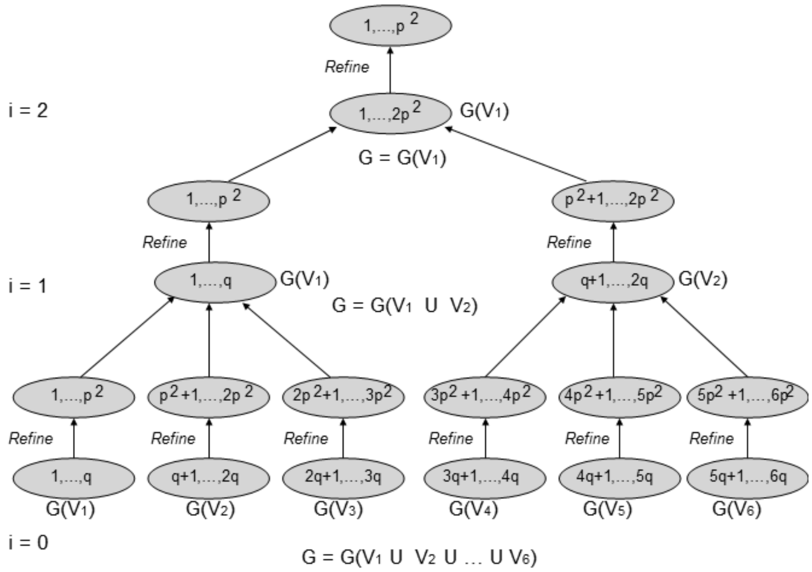

Fig. 1. *An execution of Procedure Defective-Color with the parameters p and q such that $q = 3p^2$ on an initially $(6q)$-colored graph G. Each oval represents a subgraph. The range inside an oval represents the color palette employed by the respective subgraph. For j, $1 \le j \le 6$, the set V_j changes after each iteration and contains all vertices that are currently colored using the palette $\{(j-1)\cdot q+1, (j-1)\cdot q+2, \ldots, j\cdot q\}$.*

Proof. The proof is by induction on i.

Base $(i = 0)$*:* In the first step of Procedure Defective-Color, the graph G is colored using $c \cdot \Delta^2$ colors. Therefore, after 0 iterations, the number of colors employed by φ is at most $\chi_0 = c \cdot \Delta^2$.

Induction step: By the induction hypothesis, after iteration $i-1$, the number of colors employed by φ is at most χ_{i-1}. In iteration i the vertex set V of G is partitioned into $h = \max\{\lfloor \chi_{i-1}/q \rfloor, 1\}$ disjoint subsets V_j, $j = 1, 2, \ldots, h$. Each of these subsets except V_h is colored with at most q colors. The set V_h is colored with at most $2q$ colors. Procedure Refine produces a new coloring in each set V_j such that the number of colors used in the set V_j is at most p^2 for $j = 1, 2, \ldots, h$. Consequently, the number of colors used by φ at the end of iteration i is at most $(\max \lfloor \{\chi_{i-1}/q\rfloor, 1\}) \cdot p^2 = \chi_i$. (See steps 8, 13, and 14 of Algorithm 1.) ∎

By step 14 of Algorithm 1, $\chi_{i+1} \le \max\left\{\chi_i \cdot p^2/q,\ p^2\right\}$ for $i = 0, 1, 2, \ldots,$ and $\chi_0 = c \cdot \Delta^2$. Therefore,

$$\chi_i \le \max\left\{c \cdot \Delta^2 \cdot (p^2/q)^i,\ p^2\right\}. \tag{3.1}$$

Next, we analyze the defect of the coloring produced by Procedure Defective-Color.

Theorem 3.5. *Procedure Defective-Color invoked with parameters p, q, such that $1 \le p \le \Delta$, $p^2 < q < O(\Delta^2)$, computes an $O(\frac{\log \Delta}{\log(q/p^2)} \cdot \Delta/p)$-defective p^2-coloring.*

Proof. We prove by induction on i that after i iterations $\varphi(\cdot)$ is an $(i \cdot \Delta/p)$-defective $(\max\left\{c \cdot \Delta^2 \cdot (p^2/q)^i,\ p^2\right\})$-coloring of G.

Base $(i = 0)$*:* Observe that a 0-defective $(c \cdot \Delta^2)$-coloring is computed in the first

Algorithm 1 Procedure Defective-Color(p, q) (algorithm for a vertex v).

Input: A graph G, and two parameters p, q, such that $p^2 < q$.
Output: $O(\frac{\log \Delta}{\log(q/p^2)} \cdot \Delta/p)$-defective p^2-coloring of G

1: $\varphi :=$ color G with $(c \cdot \Delta^2)$ colors
2: $\chi := c \cdot \Delta^2$ /* the current number of colors */
3: $i = 0$ /* the index of the current iteration */
4: **while** $\chi > p^2$ **do**
5: **if** $\chi < q$ **then**
6: $j := 1$
7: **else**
8: $j := \min\{\lceil \varphi(v)/q \rceil, \lfloor \chi/q \rfloor\}$
9: **end if**
10: set V_j to be the set of v
11: $\psi_j(v) := \varphi(v) - (j-1) \cdot q$ /* $\psi_j(\cdot)$ is an $(i \cdot \lfloor \Delta/p \rfloor)$-defective $(2q)$-coloring of $G(V_j)$ */
12: $\varphi'_j :=$ Refine$(G(V_j), \psi_j, p)$
13: $\varphi(v) := \varphi''(v) := \varphi'_j(v) + (j-1) \cdot p^2$
14: $\chi := (\max\{\lfloor \chi/q \rfloor, 1\}) \cdot p^2$ /* $\varphi(\cdot)$ is an $(i \cdot \lfloor \Delta/p \rfloor)$-defective χ-coloring of G */
15: $i := i + 1$
16: **end while**
17: return φ

step of the algorithm. Therefore, before the beginning of the first iteration, φ is a 0-defective $(c \cdot \Delta^2)$-coloring of G.

Induction step: Let φ be the coloring produced after $i-1$ iterations. By the induction hypothesis, φ is an $((i-1)\cdot\Delta/p)$-defective $(\max\{c \cdot \Delta^2 \cdot (p^2/q)^{i-1},\ p^2\})$-coloring of G. In iteration i, the vertex set V of G is partitioned into $h = \max\{\lfloor \chi_{i-1}/q \rfloor, 1\}$ disjoint subsets V_j. If there is only one subset $V_1 = V$, then $G(V_1) = G$ is colored with at most $2q$ colors. Otherwise, each induced graph $G(V_j)$, $1 \le j < h$, is colored by q different colors. The induced graph $G(V_h)$ is colored by at most $2q$ colors. Therefore, for each j, $1 \le j \le h$, the coloring ψ_j computed in step 11 of the ith iteration is an $((i-1) \cdot \Delta/p)$-defective $(2q)$-coloring of $G(V_j)$. In step 12, Procedure Refine is invoked on $G(V_j)$ with p as input. As a result, an $((i-1) \cdot \Delta/p + \Delta/p)$-defective p^2-coloring φ'_j of $G(V_j)$ is produced. In other words φ'_j is an $(i \cdot \Delta/p)$-defective p^2-coloring of $G(V_j)$, i.e., $\mathit{def}(\varphi'_j) \le i \cdot \Delta/p$. To finish the proof, we next argue that $\mathit{def}(\varphi'')$ is at most $i \cdot \Delta/p$ too.

Consider a vertex v and a neighbor u of v. First, suppose that $v \in V_j$, $u \in V_\ell$ for a pair of distinct indices j,ℓ, $1 \le j < \ell \le h$. Then

$$\varphi''(v) - \varphi''(u) = (\varphi'_j(v) - \varphi'_\ell(u)) + (j-\ell) \cdot p^2 \ge \varphi'_j(v) - \varphi'_\ell(u) + p^2.$$

Since $\varphi'_j(v) - \varphi'_\ell(u) \ge -p^2 + 1$, it follows that $\varphi''(v) \ne \varphi''(u)$. Second, consider a neighbor $w \in V_j$ of v. If $\varphi'_j(v) \ne \varphi'_j(w)$, then also

$$\varphi''(v) = \varphi'_j(v) + (j-1) \cdot p^2 \ne \varphi''(w) = \varphi'_j(w) + (j-1) \cdot p^2.$$

Since $\mathit{def}(\varphi'_j) \le i \cdot \Delta/p$, there are at most $(i \cdot \Delta/p)$ neighbors $w \in V_j$ of v such that $\varphi'_j(w) = \varphi'_j(v)$. Consequently, the coloring $\varphi = \varphi''$ that is produced in step

13 of the ith iteration is an $(i \cdot \Delta/p)$-defective $(\max\{c \cdot \Delta^2 \cdot (p^2/q)^i,\ p^2\})$-coloring of G. This completes the inductive proof. By (3.1) after $\frac{\log(c \cdot \Delta^2)}{\log(q/p^2)}$ iterations, φ is a $(\frac{\log(c \cdot \Delta^2)}{\log(q/p^2)} \cdot \Delta/p)$-defective p^2-coloring of G. ☐

Procedure Defective-Color starts with computing an $O(\Delta^2)$-coloring. The algorithm of Linial [30] computes a $(c \cdot \Delta^2)$-coloring in time $\log^* n + O(1)$. Szegedy and Vishwanathan [39] showed that the coefficient of $\log^* n$ can be improved to 1/2, i.e., they devised an $O(\Delta^2)$-coloring algorithm with time $\frac{1}{2}\log^* n + O(1)$. Henceforth we refer to this algorithm as the *SV algorithm*. The number of iterations performed by Procedure Defective-Color is at most $\log_{q/p^2}(c \cdot \Delta^2) = \frac{\log(c \cdot \Delta^2)}{\log(q/p^2)}$. Each iteration invokes Procedure Refine that requires $O(q)$ time and performs some additional computation that requires $O(1)$ time. The running time of Procedure Defective-Color is given below.

THEOREM 3.6. *Procedure Defective-Color invoked with parameters p,q, such that* $1 \le p \le \Delta$, $p^2 < q < O(\Delta^2)$, *runs in* $T(n) + O(q \cdot \frac{\log \Delta}{\log(q/p^2)})$ *time, where* $T(n)$ *is the time required for computing* $O(\Delta^2)$*-coloring. If the SV algorithm is used for* $O(\Delta^2)$*-coloring, the running time of Procedure Defective-Color is* $O(q \cdot \frac{\log \Delta}{\log(q/p^2)}) + \frac{1}{2}\log^* n$.

3.2. The set-theoretic method. In this section we introduce our set-theoretic method, which allows us to compute defective colorings significantly faster. Specifically, we devise an algorithm for computing an $O(\Delta/p)$-defective p^2-coloring in $\frac{1}{2}\log^* n + O(\log^* \Delta)$ time for any p in the entire range $[1, \Delta]$. Another alternative algorithm that achieves the same result (up to constant factors in the running time) is provided in Appendix A. Both these algorithms are based on the set-theoretic approach, i.e., they extend Linial's algorithm [29] for constructing an $O(\Delta^2)$-coloring (0-defective) in $\log^* n + O(1)$ time. Linial's algorithm itself is based on the work by Erdős, Frankl, and Füredi [12]. The latter authors devised constructions of set systems $\mathcal{F} = \{S^{(1)}, S^{(2)}, \ldots, S^{(n)}\}$, over a ground set $[m]$ (i.e., $S^{(i)} \subseteq [m]$ for each $i \in [n]$), such that no one of these sets is covered by a union of Δ other sets. Here n, m, and Δ are positive integer parameters. In other words, a family $\mathcal{F}$ as above satisfies that for any set $S_0 \in \mathcal{F}$ and Δ sets $S_1, S_2, \ldots, S_\Delta \in \mathcal{F}$, all different from S_0, it holds that $S_0 \nsubseteq \bigcup_{i=1}^{\Delta} S_i$. Such a family is called Δ*-cover-free*. Erdős, Frankl, and Füredi [12] showed that for any n and Δ, $n > \Delta$, one can build a Δ-cover-free family over a ground set of size $m = O(\Delta^2 \cdot \log n)$. The proof of this result is by the probabilistic method. They also provide an algebraic proof of a slightly weaker bound, specifically, $m = O(\Delta^2 \cdot \log^2 n)$. Our proof in this section extends their algebraic proof. We start by outlining the algebraic proof from [12] and explaining how it is used for coloring in Linial's paper [29]. We then proceed to describe our more general construction.

Let X, $|X| = m$, be a ground set, where $m = q^2$ for a prime q. It is convenient to view X as $GF(q) \times GF(q)$. For a positive parameter d, let $\text{Poly}(d, q)$ denote the set of polynomials of degree d over $GF(q)$. For each polynomial $g(\cdot) \in \text{Poly}(d, q)$ let $S_g = \{(a, g(a)) \mid a \in GF(q)\}$ be the set of all points on the graph of the polynomial $g(\cdot)$. Let $\mathcal{F}_{d,q} = \{S_g \mid g(\cdot) \in \text{Poly}(d, q)\}$. Observe that $|S_g| = q$ for every $g(\cdot) \in \text{Poly}(d, q)$. Note that two such distinct sets may intersect in at most d elements. Hence to cover a fixed set S_g, one needs at least q/d other sets S_h from the family $\mathcal{F} = \mathcal{F}_{d,q}$.

Let $\Delta = \lceil q/d \rceil - 1 < q/d$. It follows that $\mathcal{F}$ is a Δ*-cover-free* family. Its cardinality is $|\mathcal{F}| = q^{d+1} = n$. By expressing q and d in terms of n, m, and Δ, we obtain $m \cdot \log^2 m \le 4(\Delta + 1)^2 \log^2 n$. Hence $m \le 4(\Delta + 1)^2 \log^2 n$. (In fact, $m \le 5(\Delta + 1)^2 \cdot \frac{\log^2 n}{\log^2(4(\Delta+1)^2 \log^2 n)}$, but we will not use this estimate.)

Another valuable setting of parameters is $d = 2$. Then we get $n = q^3, m = q^2$, and $\Delta \geq \sqrt{m}/2 - 1$, since $q = \sqrt{m}$. It follows that $m \leq 4(\Delta+1)^2$. In other words, for $n \leq 8(\Delta+1)^3$ one can have a ground set of size $m \leq 4(\Delta+1)^2$. We summarize in the following theorem.

Theorem 3.7 (see [12]). *For any positive integers n, Δ, $n > \Delta$, one can construct a Δ-cover-free family $\mathcal{F}$ with n sets over the ground set $[m]$ for some $m \leq 4(\Delta+1)^2 \log^2 n$. Moreover, if $n \leq 8(\Delta+1)^3$, then $m \leq 4(\Delta+1)^2$.*

Note that for any $\Delta > 0$ and for any $n > \Delta$ (and not only a prime power), we can choose a prime q, $\lfloor(\Delta+1)\log n\rfloor \leq q \leq 2\lfloor(\Delta+1)\log n\rfloor$, and set $d = \lfloor \log n \rfloor$. (Such a prime q exists according to the Bertrand–Chebyshev postulate. See, e.g., Theorem 418 in [18].) Then the family $\mathcal{F}_{d,q}$ is Δ-cover-free. Moreover $|\mathcal{F}_{d,q}| = q^{d+1} > n$. Thus we can select a subset of $\mathcal{F}_{d,q}$ of size n which is a Δ-cover-free family over a ground set $[m]$ of size $m = q^2 \leq 4(\Delta+1)^2 \log^2 n$.

Next, we outline how Theorem 3.7 is used (in [29, 39]) for coloring. Suppose we have an N-coloring φ of a graph G, $\Delta(G) = \Delta$. Let $\mathcal{F} = \{S^{(1)}, \ldots, S^{(N)}\}$ be a Δ-cover-free family of size N, whose existence is guaranteed by Theorem 3.7. For each color $i \in [N]$ we associate a set $S^{(i)} \in \mathcal{F}$ with this color. In one round, vertices learn the colors of their neighbors. Then each vertex v in parallel computes the set $S^{(\varphi(v))} \setminus \bigcup_{u \in \Gamma(v)} S^{(\varphi(u))}$. This set is not empty, by Theorem 3.7, for every vertex $v \in V$. Then each vertex v picks an arbitrary element x from $S^{(\varphi(v))} \setminus \bigcup_{u \in \Gamma(v)} S^{(\varphi(u))}$ and sets its new color $\varphi'(v) = x$. It is easy to verify that φ' is a legal m-coloring of G, $m \leq 4(\Delta+1)^2 \log^2 N$.

We start with a trivial n-coloring φ as our initial coloring. (It assigns the Id of each vertex v as its initial color.) By applying the above recoloring procedure for $\log^* n + O(1)$ times, this coloring is converted into an $O(\Delta^2 \log^2 \Delta)$-coloring ψ for G. For a sufficiently large Δ, it holds that $O(\Delta^2 \log^2 \Delta) \leq 8(\Delta+1)^3$. (If $\Delta = O(1)$, then $O(\Delta^2 \log^2 \Delta) = O(\Delta^2)$.) Hence we next use the second set system from Theorem 3.7 to convert this coloring into a $4(\Delta+1)^2$-coloring of G. The overall running time is $\log^* n + O(1)$.

Next, we extend Theorem 3.7 of [12], and then use it to devise defective colorings. For a set S_0 and Δ other sets $S_1, S_2, \ldots, S_\Delta$, and for a nonnegative integer ρ, we say that $S_1, S_2, \ldots, S_\Delta$ ρ-cover S_0 if every element of S_0 appears in at least ρ of the sets $S_1, S_2, \ldots, S_\Delta$. We say that a family $\mathcal{F}$ is Δ-*union* $(\rho+1)$-*cover-free* if for any set $S_0 \in \mathcal{F}$, and Δ other sets $S_1, S_2, \ldots, S_\Delta \in \mathcal{F}$, the sets $S_1, S_2, \ldots, S_\Delta$ do not $(\rho+1)$-cover S_0. (In other words, S_0 necessarily contains an element x which appears in at most ρ sets among $S_1, S_2, \ldots, S_\Delta$.) Note that for $\rho = 0$, a family is Δ-union $(\rho+1)$-cover-free iff it is Δ-cover-free.

Suppose we have a D-defective N-coloring φ of a graph G with $\Delta(G) = \Delta$, and a Δ-union $(\rho+1)$-cover-free family $\mathcal{F}$ with N sets, over a ground set $[m]$, for some values of D, N, Δ, ρ, and m. Each vertex v can now compute in parallel an element $x \in S_{\varphi(v)}$ that belongs to at most ρ sets $S_{\varphi(u)}$ for $u \in \Gamma(v)$ and $\varphi(u) \neq \varphi(v)$. Such an element exists because $\mathcal{F}$ is Δ-union $(\rho+1)$-cover-free. The vertex v then sets its new color $\varphi'(v) = x$. Among neighbors $u \in \Gamma(v)$ with $\varphi(u) \neq \varphi(v)$, at most ρ of them can select the color x. Also, there are at most D neighbors u with $\varphi(u) = \varphi(v)$. Hence, overall, at most $\rho + \Delta$ neighbors u of v will have $\varphi'(u) = \varphi'(v)$. Therefore, the coloring φ' is a $(\rho+\Delta)$-defective m-coloring. We have proved the following lemma.

Lemma 3.8. *For some values of D, N, Δ, ρ, and m, suppose we are given a D-defective N-coloring φ of a graph G with $\Delta(G) = \Delta$ and a Δ-union $(\rho+1)$-cover-free family $\mathcal{F}$ with N sets over the ground set $[m]$. Then in one round, one can compute a $(\rho+D)$-defective m-coloring for G.*

Consider the family $\mathcal{F} = \mathcal{F}_{q,d}$ that was described above. Each set in this family contains q elements, and any pair of distinct sets intersect in at most d elements. To hit each element of a set $S_0 \in \mathcal{F}$ for $\rho + 1$ times, one needs at least $q \cdot (\rho + 1)/d$ other sets $S_1, S_2, \ldots, S_\Delta$, $\Delta \geq q \cdot (\rho + 1)/d$. Let $\Delta = \lceil q \cdot (\rho + 1)/d \rceil - 1$, $m = q^2$, $n = q^{d+1}$, where n is the number of sets in $\mathcal{F}$. It follows that $q \log q \leq \frac{\delta+1}{\rho+1} \log n$, i.e., $q \leq \frac{\Delta+1}{\rho+1} \log n$, and $m \leq (\frac{\Delta+1}{\rho+1})^2 \log^2 n$. Since q needs to be prime, we may need to adjust the upper bound on q to $q \leq 2 \cdot \frac{\Delta+1}{\rho+1} \log n$, i.e., $m \leq 4 \cdot (\frac{\Delta+1}{\rho+1})^2 \log^2 n$. (This adjustment is equivalent to building a set system for n' sets, where $n' \leq n^2$. Then we have $m \leq (\frac{\Delta+1}{\rho+1})^2 \log^2 n' \leq 4 \cdot (\frac{\Delta+1}{\rho+1})^2 \log^2 n$.) We summarize in the next theorem.

THEOREM 3.9. *For positive integer parameters n, Δ, and ρ such that $n > \Delta > \rho$, we can build a Δ-union $(\rho + 1)$-cover-free set system $\mathcal{F}$ with n sets over the ground set $[m]$ for some $m \leq 4 \cdot (\frac{\Delta+1}{\rho+1})^2 \log^2 n$.*

It is also useful to set the parameter d to 2 in this construction. We obtain $m = q^2$, $n = q^3$, $\Delta \geq q \cdot (\rho+1)/2 - 1$. Hence for $n \leq 8 \cdot (\frac{\Delta+1}{\rho+1})^3$ we get $m \leq 4 \cdot (\frac{\Delta+1}{\rho+1})^2$. Theorem 3.9 and Lemma 3.8 imply the following corollary.

COROLLARY 3.10. *Given a D-defective N-coloring φ, one can get a $(\Delta + \rho)$-defective $(4 \cdot (\frac{\Delta+1}{\rho+1})^2 \log^2 n)$-coloring φ' within one single round. Moreover, if $N \leq 8 \cdot (\frac{\Delta+1}{\rho+1})^3$, then the number of colors in φ' is at most $4 \cdot (\frac{\Delta+1}{\rho+1})^2$.*

Next we show that these constructions can be composed iteratively for obtaining a $\frac{\Delta}{p}$-defective $O(p^2)$-coloring for any parameter $1 \leq p \leq \Delta$. We start by computing an $O(\Delta^2)$-coloring φ in $\frac{1}{2} \log^* n + O(1)$ rounds, using an algorithm of Szegedy and Vishwanathan [39]. For a positive integer i and a number y, let $\log^{2[i]} y$ denote the i-times iterative function $\log^2$ applied to the value y. That is, $\log^{2[0]} y = y$, $\log^{2[1]} y = \log^2 y$, and for every $i \geq 1$, $\log^{2[i+1]} y = \log^2(\log^{2[i]} y)$. Denote also by $\log^{2*} y$ the number i such that $\log^{2[i]} y < 32$. Observe that $\log^{2*} y = O(\log^* y)$. (The threshold value 32 is somewhat arbitrary. Any constant c such that $\log^2 x < x$ for all $x \geq c$ would do.)

We now apply the recoloring procedure of Corollary 3.10 to the $O(\Delta^2)$-coloring φ, iteratively, for $\log^{2*} \Delta + O(1)$ times, with some fixed parameter ρ (to be determined later). As a result we obtain a $(\log^{2*} \Delta + O(1)) \cdot \rho$-defective $O((\frac{\Delta+1}{\rho+1})^2 \log^2(\frac{\Delta+1}{\rho+1}))$-coloring. By applying one single time the second recoloring procedure of Corollary 3.10 we obtain a $(\log^{2*} \Delta + O(1)) \cdot \rho$-defective $O((\frac{\Delta+1}{\rho+1})^2)$-coloring of the original graph G. Since $\log^{2*} \Delta = O(\log^* \Delta)$, it follows that within $O(\log^* \Delta)$ rounds we converted an $O(\Delta^2)$-coloring $\varphi = \varphi_0$ into an $O((\log^* \Delta) \cdot \rho)$-defective $O((\frac{\Delta+1}{\rho+1})^2)$-coloring $\varphi = \varphi_1$ for G.

By rescaling (setting $\rho' = O(\rho \log^* \Delta)$) we get a ρ-defective $O((\frac{\Delta+1}{\rho+1})^2 (\log^* \Delta)^2)$-coloring φ_1 for $\rho \geq \log^* \Delta$. For $\rho < \log^* \Delta$, a legal $O(\Delta^2)$-coloring can also serve as a ρ-defective $O((\frac{\Delta+1}{\rho+1})^2 (\log^* \Delta)^2)$-coloring. Hence the above procedure produces a ρ-defective $O((\frac{\Delta+1}{\rho+1})^2 (\log^* \Delta)^2)$-coloring φ_1 for all values of the parameter ρ. Within additional $\log^{2*}(\log^* \Delta)^2 = O(\log^* \log^* \Delta)$ rounds this coloring is transformed into a ρ-defective $O((\frac{\Delta+1}{\rho+1})^2 (\log^* \log^* \Delta)^2)$-coloring, etc. In overall $O(\log^* \Delta + \log^* \log^* \Delta + \log^* \log^* \log^* \Delta + \cdots) = O(\log^* \Delta)$ time we obtain a ρ-defective $O((\frac{\Delta+1}{\rho+1})^2)$-coloring. Now write $\rho = \frac{\Delta}{p}$ to obtain the ultimate result of this section.

THEOREM 3.11. *For any p, $1 \leq p \leq \Delta$, in total $O(\log^* n)$ time we can compute a $\lfloor \Delta/p \rfloor$-defective $O(p^2)$-coloring of a graph G with $\Delta(G) = \Delta$.*

4. $(\Delta + 1)$-coloring. In this section we employ the techniques and algorithms described in section 3 to devise an efficient $(\Delta + 1)$-coloring algorithm. We begin with describing a technique based on our combinatorial method. As a first step, we devise a $(\Delta + 1)$-coloring algorithm $\mathcal{J}$ with running time $O(\Delta \log\log\Delta) + \log^* n$. Set $p = \log\Delta$, and $q = \Delta^\epsilon$, for an arbitrarily small positive constant ϵ, $0 < \epsilon < 1$. By Theorems 3.5 and 3.6, Procedure Defective-Color invoked with these parameters computes an $O(\Delta/\log\Delta)$-defective $(\log^2\Delta)$-coloring φ in $O(\Delta^\epsilon) + \frac{1}{2}\log^* n$ time. Let V_j denote the set of vertices v with $\varphi(v) = j$ for $j = 1, 2, \ldots, \lfloor\log^2\Delta\rfloor$. Observe that the maximum degree $\Delta_j = \Delta(G(V_j))$ of the graph $G(V_j)$ induced by V_j is at most the defect $\mathit{def}(\varphi)$ of the coloring φ. Thus, $\Delta_j = O(\Delta/\log\Delta)$. Consequently, all graphs $G(V_j)$ can be colored in parallel with $O(\Delta/\log\Delta)$ colors, each using the KW algorithm. The running time of this step is $O(\Delta + \log^* n)$. If we use distinct palettes of size $O(\Delta/\log\Delta)$ for each graph $G(V_j)$, then we get an $O(\log^2\Delta\cdot\Delta/\log\Delta) = O(\Delta\log\Delta)$-coloring of the entire graph G. Next, we use the KW iterative procedure with the parameter $m = O(\Delta\log\Delta)$ to compute a $(\Delta+1)$-coloring from $O(\Delta\log\Delta)$-coloring in time $O(\Delta\cdot\log\frac{m}{\Delta}) = O(\Delta\log\log\Delta)$. The total running time of the above algorithm for computing $(\Delta + 1)$-coloring is $O(\Delta\log\log\Delta + \log^* n)$.

Corollary 4.1. *The algorithm $\mathcal{J}$ computes a $(\Delta + 1)$-coloring in time $O(\Delta\cdot\log\log\Delta + \log^* n)$.*

Corollary 4.1 is already a significant improvement over the previous state-of-the-art running time of $O(\Delta\cdot\log\Delta + \log^* n)$, due to Kuhn and Wattenhofer [27]. In what follows we improve this bound further and devise a $(\Delta + 1)$-coloring algorithm with running time $O(\Delta) + \frac{1}{2}\log^* n$. We do it in two steps. First, we improve it to $O(\Delta\cdot\log^{(k)}\Delta + \log^* n)$ for an arbitrarily large constant integer k. Second, we achieve our ultimate goal of $O(\Delta) + \frac{1}{2}\log^* n$.

Suppose that there exists an algorithm $\mathcal{A}_k$ that computes a $(\Delta + 1)$-coloring in $O(\Delta\log^{(k)}\Delta) + \frac{k}{2}\cdot\log^* n$ time for some integer $k > 0$. We employ this algorithm to devise a more efficient $(\Delta + 1)$-coloring algorithm $\mathcal{A}_{k+1}$. (The pseudocode and the verbal description of $\mathcal{A}_{k+1}$ are provided below.) Specifically, $\mathcal{A}_{k+1}$ has running time $O(\Delta\log^{(k+1)}\Delta) + \frac{(k+1)}{2}\cdot\log^* n$. For an input graph G, invoke Procedure Defective-Color with the parameters $p = \lfloor\log^{(k)}\Delta\rfloor$, $q = \lfloor\Delta^\epsilon\rfloor$, for a constant ϵ, $0 < \epsilon < 1$. We obtain an $O(\Delta/\log^{(k)}\Delta)$-defective $((\log^{(k)}\Delta)^2)$-coloring of G, and the running time of this step is $O(\Delta^\epsilon) + \frac{1}{2}\log^* n$. Let V_j denote the subset of vertices that were assigned the color j. Invoke in parallel the algorithm $\mathcal{A}_k$ on the subgraphs $G(V_j)$, for $j = 1, 2, \ldots, p^2$, using distinct palettes. The resulting coloring of these invocations is a (0-defective) $O(\Delta\log^{(k)}\Delta)$-coloring. Invoke the KW iterative procedure with the parameter $m = O(\Delta\log^{(k)}\Delta)$ to compute a $(\Delta + 1)$-coloring of G in time $O(\Delta\log\frac{m}{\Delta}) = O(\Delta\log^{(k+1)}\Delta)$.

The running time of the algorithm $\mathcal{A}_{k+1}$ consists of the running time of Procedure Defective-Color, which is $O(\Delta^\epsilon) + \frac{1}{2}\log^* n$, the running time of the algorithm $\mathcal{A}_k$ on graphs with maximal degree $\Delta/\log^{(k)}\Delta$, which is $O(\Delta) + \frac{k}{2}\cdot\log^* n$, and the running time of the KW iterative procedure, which is $O(\Delta\log^{(k+1)}\Delta)$. Therefore, the total running time of $\mathcal{A}_{k+1}$ is $O(\Delta\log^{(k+1)}\Delta) + \frac{(k+1)}{2}\cdot\log^* n$.

We summarize this argument with the following theorem.

Theorem 4.2. *For a* constant *arbitrarily large positive integer k, the algorithm $\mathcal{A}_k$ computes a $(\Delta + 1)$-coloring of the input graph in time $O(\Delta\log^{(k)}\Delta + \log^* n)$.*

Next, we turn to our set-theoretic method in order to derive a $(\Delta + 1)$-coloring algorithm with running time $O(\Delta) + \frac{1}{2}\log^* n$. We remark that it is also possible

Algorithm 2 $\mathcal{A}_{k+1}(G)$.

Input: An input graph G
Output: A legal $(\Delta+1)$-coloring of G

$p := \lfloor \log^{(k)} \Delta \rfloor$; $q = \lfloor \Delta^{\epsilon} \rfloor$
compute an $O(\Delta/p)$-defective p^2-coloring of G by invoking Procedure Defective-Color(p,q)
for $j = 1, 2, \ldots, p^2$, in parallel **do**
  compute a legal $O(\Delta/p)$-coloring of the subgraph induced by color j using the algorithm $\mathcal{A}_k$
end for
combine colors to get a legal $O(\Delta \cdot p)$-coloring of G
reduce the number of colors to $\Delta + 1$ using the KW iterative procedure
return computed coloring

to further the combinatorial method so that it will achieve the same running time. Basically, to this end one needs to extend the algorithm $\mathcal{A}_k$ to the range $k \le \log^* \Delta$, avoiding the slack that accumulates due to multiple recursive invocations. The full details and analysis of the extended algorithm based on our combinatorial method can be found in [5]. (One can also find there a more general result of computing a $\lambda \cdot (\Delta+1)$-coloring in $O(\Delta/\lambda + \log^* n)$ time, for any $1 \le \lambda \le \Delta^{1-\epsilon}$, where $\epsilon > 0$ is an arbitrarily small constant. This result is obtained in [5] using the combinatorial method as well. On the other hand, in the current paper we will present the same result using the set-theoretic method. In this case the result is applicable for the entire range $1 \le \lambda \le \Delta$.) Since using both methods yields the same running time, and the algorithm based on the set-theoretic technique is somewhat simpler, we do not elaborate further on the combinatorial method.

We will now discuss how to use our set-theoretic method to obtain an algorithm for $(\Delta+1)$-coloring in $O(\Delta) + \frac{1}{2}\log^* n$ time. In addition to Theorem 3.11, we require the following two results. We have already mentioned these results, but they are explicitly given again in Lemmas 4.3 and 4.4 for convenience.

LEMMA 4.3 (Szegedy and Vishwanathan algorithm [39]). *An $O(\Delta^2)$-coloring can be computed in $\frac{1}{2}\log^* n + O(1)$ rounds.*

Remark. In his seminal paper [29] Linial showed an algorithm with a slightly inferior running time of $\log^* n + O(1)$.

LEMMA 4.4 (Kuhn and Wattenhofer iterative procedure [27]). *Let A and B be integers such that $B > A \ge \Delta + 1$. When starting with a B-coloring, an A-coloring can be computed in $O(\Delta \cdot \log(B/A))$ rounds.*

The details of our $(\Delta+1)$-coloring algorithm are given by Algorithm 3. The core of the algorithm is the procedure Color(D), which is used on a subgraph with maximum degree at most D and computes a $(\mathsf{D}+1)$-coloring of the subgraph. The procedure first partitions the graph into subgraphs of maximum degree at most $d = \lfloor \mathsf{D}/2 \rfloor$ by computing a d-defective C-coloring (see Theorem 3.11), and by using the C colors to partition the graph. (Note that C is a constant.) The procedure Color($\cdot$) is then called recursively for each of the C subgraphs. Note that these C recursive calls can be done in parallel. Every vertex then has a color between 1 and C from the defective coloring and a color between 1 and $d+1$ from the recursive call to Color($\cdot$). Combining the two colors gives a legal $C \cdot (d+1)$-coloring of the graph on which Color(D) was called. Using Lemma 4.4, a $(\mathsf{D}+1)$-coloring can then be computed in $O(\mathsf{D} \cdot \log C) = O(\mathsf{D})$ rounds. The following lemma bounds the time needed to execute procedure Color(D).

Algorithm 3 $(\Delta+1)$-coloring algorithm.

Input: M-coloring of the graph
Output: Call to Color(Δ) returns $(\Delta+1)$-coloring

```
1:  procedure Color(deg):
2:    if D = 1 then
3:      compute 2-coloring in 1 round
4:    else
5:      d := ⌊D/2⌋
6:      compute d-defective C-coloring in time O(log* M)    /* C = O(1) is a
        universal constant */
7:      for all colors c ∈ [C] in parallel do
8:        call Color(d) on subgraph induced by color c
9:      end for
10:     combine colors to get C · (d + 1)-coloring
11:     reduce to (D + 1)-coloring in time O(D · log C)
12:   end if
13:   return computed coloring
14: end procedure
```

LEMMA 4.5. *Let $T(\mathsf{D})$ be the number of rounds needed to execute procedure* Color(D)*. For all $\mathsf{D} \geq 0$, we have*

$$T(\mathsf{D}) = O(\mathsf{D} + \log(\mathsf{D}) \cdot \log^* M).$$

Proof. For $\mathsf{D} = 1$, a 2-coloring can be computed in one round, as the subgraph consists of at most two vertices. We therefore have $T(1) = 1$. For $\mathsf{D} \geq 2$, we first show that $T(\mathsf{D})$ can be computed recursively as follows. There is a positive constant α such that for all $\mathsf{D} > 0$,

$$T(\mathsf{D}) \leq T(\lfloor \mathsf{D}/2 \rfloor) + \alpha \cdot \mathsf{D} + \log^* M. \tag{4.1}$$

Since the number of colors C computed in line 6 is $O(\mathsf{D}/d) = O(1)$ (see Theorem 3.11), the number of rounds needed to reduce the colors in line 10 is $O(\mathsf{D})$. Inequality (4.1) now directly follows because all the recursive calls in line 8 can be executed in parallel. Hence $T(\mathsf{D}) \leq 2\alpha \cdot \mathsf{D} + \log(\mathsf{D}) \cdot \log^* M$, and therefore also the lemma now follows by induction on D. ∎

As a consequence of Lemma 4.5, we can state the main theorems of this section.

THEOREM 4.6. *A legal $(\Delta+1)$-coloring can be computed in $O(\Delta) + \frac{1}{2}\log^* n$ rounds.*

Proof. We start by applying the SV algorithm (Lemma 4.3) and compute an $O(\Delta^2)$-coloring in $\frac{1}{2}\log^* n + O(1)$ rounds. We show how to obtain a $(\Delta+1)$-coloring in time $O(\Delta)$. Because we already have an $O(\Delta^2)$-coloring, we can apply procedure Color(Δ) to the whole graph with this initial coloring and can thus compute a $(\Delta+1)$-coloring in time $O(\Delta + \log(\Delta) \cdot \log^* \Delta) = O(\Delta)$ by Lemma 4.5. Therefore, the total running time is $O(\Delta) + \frac{1}{2}\log^* n$. ∎

THEOREM 4.7. *For every $\lambda \geq 1$, a legal $\lambda \cdot (\Delta+1)$-coloring can be computed in $O(\Delta/\lambda + \log^* n)$ rounds.*

Proof. We prove the theorem by constructing an algorithm that achieves this result. Again, we start by applying the SV algorithm and compute an $O(\Delta^2)$-coloring in $\frac{1}{2}\log^* n + O(1)$ rounds. It now remains to show that based on an $O(\Delta^2)$-coloring, a $\lambda \cdot (\Delta+1)$-coloring can be computed in time $O(\Delta/\lambda)$. Because we can compute a $(\Delta+1)$-coloring in time $O(\Delta)$ and because we already start with an $O(\Delta^2)$-coloring, we can assume without loss of generality that $\alpha \leq \lambda \leq \Delta/\alpha$ for a sufficiently large constant α. By Theorem 3.11 and the assumption that we already have an $O(\Delta^2)$-coloring, there is a positive constant β such that for every integer $d \geq 1$, a d-defective $\lfloor \beta \cdot \Delta^2/(d+1)^2 \rfloor$-coloring can be computed in time $O(\log^* \Delta)$. Let $\mathsf{D} < \Delta$ be the smallest integer such that

$$\beta \cdot \frac{\Delta^2}{(\mathsf{D}+1)^2} \cdot (\mathsf{D}+1) = \beta \cdot \frac{\Delta^2}{\mathsf{D}+1} \leq \lambda \cdot (\Delta+1). \tag{4.2}$$

We define $c = \lfloor \beta \cdot \Delta^2/(\mathsf{D}+1)^2 \rfloor$. If the constant α is chosen sufficiently large, we have $\mathsf{D} \geq 1$ and $\mathsf{D} < \Delta$. Further, by inequality (4.2), we have $\mathsf{D} = \Theta(\Delta/\lambda)$. The algorithm first computes a D-defective c-coloring in time $O(\log^* \Delta)$. For each of the c color classes, in parallel a $(\mathsf{D}+1)$-coloring is computed in time $O(\mathsf{D} + \log^* \Delta) = O(\Delta/\lambda + \log^* \Delta)$. In combination with the c colors from the defective coloring, this allows us to compute a $c \cdot (\mathsf{D}+1) \leq \beta \cdot \frac{\Delta^2}{\mathsf{D}+1} \leq \lambda \cdot (\Delta+1)$-coloring of the original graph. ∎

It is well known [30] that given a $(\Delta+1)$-coloring one can produce an MIS within $\Delta+1$ rounds. Consequently, Theorem 4.7 implies that our algorithm in conjunction with the reduction from [30] produces an MIS in time $O(\Delta) + \frac{1}{2}\log^* n$.

Finally, we remark that Theorem 4.7 implies an improved trade-off for coloring graphs of bounded arboricity, and an improved algorithm for computing an MIS for the latter family of graphs. Specifically, in [4] it was shown (Theorem 5.1) that graphs of arboricity at most a can be $O(a \cdot t)$-colored in time $O(\frac{a}{t}\log n + a\log a)$ for any parameter t, $1 \leq t \leq a$. The algorithm that achieves this trade-off employs the KW algorithm on graphs with maximum degree $O(a)$. This step requires $O(a\log a + \log^* n)$ time. By replacing the invocation of the KW algorithm by an invocation of our new algorithm from Theorem 4.7 we improve the running time of this step to $O(a + \log^* n)$ and the overall running time to $O(\frac{a}{t}\log n + a)$.

In addition, in [4] this trade-off is used to achieve an algorithm for computing an MIS for graphs with arboricity $a = \Omega(\sqrt{\log n})$ in time $O(a\sqrt{\log n} + a\log a)$. (See Theorem 6.4 in [4].) This is done by first computing the $O(a \cdot t)$-coloring, and then converting the $O(a \cdot t)$-coloring into an MIS within additional $O(a \cdot t)$ rounds. By employing our improved trade-off for $O(a \cdot t)$-coloring (in time $O(\frac{a}{t}\log n + a)$), we obtain an overall time of $O(\frac{a}{t}\log n + a \cdot t)$. Finally, we set $t = \sqrt{\log n}$ and obtain the running time of $O(a\sqrt{\log n})$.

COROLLARY 4.8. *For a parameter t, $1 \leq t \leq a$, an algorithm from* [4] *that uses Algorithm* 3 *instead of the KW algorithm as a subroutine computes an $O(a \cdot t)$-coloring for graphs with arboricity at most a. Its running time is $O(\frac{a}{t}\log n + a)$. As a result, we can compute an MIS for graphs with $a = \Omega(\sqrt{\log n})$ in time $O(a\sqrt{\log n})$.*

5. Conclusion. We have presented efficient algorithms for computing defective colorings in the distributed setting. The algorithms are significantly faster than any known algorithm for computing a (legal) $(\Delta+1)$-coloring. As an important application, our technique of employing defective colorings yields an improved deterministic distributed algorithm for $(\Delta+1)$-coloring. However, the technique could also be useful for other symmetry breaking problems. This is an interesting venue for further research. Another important direction for further study concerns the parameters in the defective coloring. Our p-defective q-coloring algorithm is very efficient for certain values of p and q. It is an open question whether yet smaller parameters can be achieved. Although p-defective $\lfloor \Delta/p \rfloor$-coloring can be computed efficiently in the sequential model [32], it is currently not known whether one can compute it in the distributed setting in sublinear in Δ time. If this question is answered in the affirmative, it would immediately imply a distributed $O(\Delta)$-coloring algorithm with time $o(\Delta)$.

Appendix A. An alternative algorithm for defective coloring.

Algorithm 4 Computing a coloring with defect at most d.

Input: color $x \in [M]$, neighbor colors $y_1, \dots, y_\delta \in [M]$, parameter d
Output: a new color

1: search $\alpha \in \mathcal{A}$ such that $\big|\{i \in [\delta] : \varphi_x(\alpha) = \varphi_{y_i}(\alpha)\}\big| \le d$
2: color $:= \big(\alpha, \varphi_x(\alpha)\big)$

In this section we devise an alternative algorithm (to the one presented in Section 3.2) for computing defective colorings in $O(\log^* n)$ time. We believe that it is of independent interest.

We first describe an algorithm to reduce the number of colors of a given defective coloring in a single round. Assume that we start with an M-coloring of G (possibly with a nonzero defect), and that the M colors of this coloring are $1, \dots, M$. Our coloring algorithm generalizes techniques described in [30]. It is based on a mapping from the color set $[M]$ to functions from a finite set $\mathcal{A}$ to a finite set $\mathcal{B}$. The new colors are chosen from the set $\mathcal{A} \times \mathcal{B}$. Let $v \in V$ be a vertex with degree $\delta \le \Delta$ and color $x \in [M]$, and assume that the δ neighbors of v have colors $y_1, \dots, y_\delta \in [M]$. Further, let φ_x be the function $\varphi_x : \mathcal{A} \to \mathcal{B}$ assigned to a color $x \in [M]$. The basic idea is to choose a value $\alpha \in \mathcal{A}$ such that the number of values y_i for which $\varphi_x(\alpha) = \varphi_{y_i}(\alpha)$ is sufficiently small. The details for computing d-defective colorings are given by Algorithm 4. The following lemma is the basis of the analysis of the algorithm.

LEMMA A.1. *Assume that we are given an M-coloring of G with defect at most $d' \le d$. For a value $\kappa > 0$, suppose that the functions φ_x for $x \in [M]$ satisfy that for any two distinct colors $x, y \in [M]$, there are at most κ values $\alpha \in \mathcal{A}$ for which $\varphi_x(\alpha) = \varphi_y(\alpha)$. Suppose also that $|\mathcal{A}| > \kappa \cdot (\Delta - d')/(d - d' + 1)$. Then, Algorithm* 4 *computes an $|\mathcal{A}| \cdot |\mathcal{B}|$-coloring ψ with defect at most d, i.e., def$(\psi) \le d$.*

Proof. We prove the lemma in two steps. We first show that if all vertices choose a color $(\alpha, \beta) \in \mathcal{A} \times \mathcal{B}$ (i.e., if there is an $\alpha \in \mathcal{A}$ that satisfies the condition in line 1), the defect of the computed coloring is at most d. We then show that for every vertex there is a value $\alpha \in \mathcal{A}$ that satisfies the condition in line 1 of Algorithm 4.

Consider a vertex v with color $x \in [M]$ and degree $deg(v) = \delta \le \Delta$ that has neighbors with colors $y_1, \dots, y_\delta \in [M]$. Assume that v chooses a color $\big(\alpha, \varphi_x(\alpha)\big)$. Let u be a neighbor with a color $y \in [M]$ for which $\varphi_y(\alpha) \ne \varphi_x(\alpha)$. Assume that u chooses color (α', β'). If $\alpha = \alpha'$, we have $\varphi_y(\alpha) = \beta' \ne \beta = \varphi_x(\alpha)$. Therefore, either $\alpha \ne \alpha'$ or $\beta \ne \beta'$, and thus u chooses a color different from that of v. Because there are at most d neighbors u' with a color $y' \in [M]$ for which $\varphi_{y'}(\alpha) = \varphi_x(\alpha)$, the defect of the computed coloring is therefore at most d.

It remains to prove that every vertex can choose a color. Because additional neighbors can certainly not increase the number of available colors, we can assume without loss of generality that the degree of v is $\delta = \Delta$. We definitely have $\varphi_x(\alpha) = \varphi_{y_i}(\alpha)$ for all α if $x = y_i$. Let $\ell \le d'$ be the number of values $i \in [\Delta]$ for which this is the case. Let $S = \{i \in [\Delta] : y_i \ne x\}$ be the indices of those neighbors of

v with a different initial color. We need to show that there is an $\alpha \in \mathcal{A}$ such that $|\{i \in S : \varphi_x(\alpha) = \varphi_{y_i}(\alpha)\}| \le d - \ell$. For the sake of contradiction, assume that for every $\alpha \in \mathcal{A}$, there are at least $d - \ell + 1$ values $i \in S$ for which $\varphi_x(\alpha) = \varphi_{y_i}(\alpha)$. Because for every $i \in S$, $\varphi_x(\alpha) = \varphi_{y_i}(\alpha)$ for at most κ values $\alpha \in \mathcal{A}$, we then have

$$(\Delta - \ell)\cdot\kappa \ge \sum_{i\in S}\left|\{\alpha \in \mathcal{A} : \varphi_x(\alpha) = \varphi_{y_i}(\alpha)\}\right| = \sum_{\alpha\in\mathcal{A}}\left|\{i \in S : \varphi_x(\alpha) = \varphi_{y_i}(\alpha)\}\right| \ge |\mathcal{A}|\cdot(d - \ell + 1). \tag{A.1}$$

Because computing a Δ-defective coloring is trivial, we can certainly assume that $d < \Delta$. Inequality (A.1) then implies that

$$|\mathcal{A}| \le \frac{(\Delta - \ell)\cdot\kappa}{d + 1 - \ell} \le \frac{(\Delta - d')\cdot\kappa}{d - d' + 1}.$$

This is a contradiction to the assumption $|\mathcal{A}| > \kappa \cdot (\Delta - d')/(d + 1 - d')$, and thus proves the lemma. ∎

In order to apply Lemma A.1, we need a function $\varphi_x : \mathcal{A} \to \mathcal{B}$ for every $x \in [M]$ such that for any two distinct colors $x, y \in [M]$, there are at most κ values $\alpha \in \mathcal{A}$ for which $\varphi_x(\alpha) = \varphi_y(\alpha)$. To obtain efficient algorithms, we want the values $|\mathcal{B}|$ and κ to be as small as possible. The following lemma proves the existence of such functions.

LEMMA A.2. *Let the set $\mathcal{A}$ be fixed, let $\mathcal{B}$ be a set of cardinality $|\mathcal{B}| \ge |\mathcal{A}|/(2\ln M)$, and let $\kappa = \lfloor 2e \ln M\rfloor$. For $x \in [M]$, there are functions $\varphi_x : \mathcal{A} \to \mathcal{B}$ such that for any two distinct $x, y \in [M]$, $\varphi_x(\alpha) = \varphi_y(\alpha)$ holds for at most κ values $\alpha \in \mathcal{A}$.*

Proof. We prove the lemma by using the probabilistic method. We show that choosing the functions independently and uniformly at random from all functions from $\mathcal{A}$ to a sufficiently large set $\mathcal{B}$ produces functions that satisfy the required conditions with positive probability.

Let us therefore assume that the functions φ_x for $x \in [M]$ are chosen independently and uniformly at random. Let p be the probability that for two distinct colors $x, y \in [M]$ and a value $\alpha \in \mathcal{A}$, we have $\varphi_x(\alpha) = \varphi_y(\alpha)$. Because $\varphi_x(\alpha)$ and $\varphi_y(\alpha)$ are independent, random elements of $\mathcal{B}$, we have $p = 1/|\mathcal{B}|$. Let Z be the number of values $\alpha \in \mathcal{A}$ for which $\varphi_x(\alpha) = \varphi_y(\alpha)$ for two distinct values $x, y \in [M]$. We have $\mathbb{E}[Z] = |\mathcal{A}| \cdot p = |\mathcal{A}|/|\mathcal{B}| \le 2\ln M$. Applying the Chernoff bound gives

$$\mathbb{P}[Z > 2e\ln M] < \left(\frac{e^{e-1}}{e^e}\right)^{2\ln M} = \frac{1}{e^{2\ln M}} = \frac{1}{M^2}.$$

By union bound, we therefore obtain that

$$\mathbb{P}\left[\max_{x\ne y\in[M]}\left|\{\alpha \in \mathcal{A} : \varphi_x(\alpha) = \varphi_y(\alpha)\}\right| > \kappa\right] = \mathbb{P}\left[\max_{x\ne y\in[M]}\left|\{\alpha \in \mathcal{A} : \varphi_x(\alpha) = \varphi_y(\alpha)\}\right| > 2e\ln M\right] < 1. \qquad ∎$$

Combining Lemmas A.1 and A.2 allows us to quantify the progress that can be achieved in a single communication round. Denote $\Upsilon = (\Delta - d')/(d + 1 - d')$.

THEOREM A.3. *Assume that we are given an M-coloring of G with defect at most $d' \le d < \Delta$. There is a constant $C > 0$ such that a $(C \cdot \Upsilon^2 \cdot \ln M)$-coloring ψ with defect at most d (i.e., $def(\psi) \le d$) can be computed in a single communication round.*

Proof. We choose $\kappa = \lfloor 2e \ln M \rfloor$ and $\mathcal{A}$ and $\mathcal{B}$ such that

$$|\mathcal{A}| = \left\lfloor \frac{(\Delta - d')2e\ln M}{d+1-d'} \right\rfloor + 1 > \frac{\kappa(\Delta - d')}{d+1-d'} = \kappa \cdot \Upsilon,$$
$$|\mathcal{B}| = \left\lceil \frac{|\mathcal{A}|}{2\ln M} \right\rceil.$$

By Lemma A.2, there exist functions $\{\varphi_x : x \in [M]\}$, $\varphi_x : \mathcal{A} \to \mathcal{B}$, such that for any two distinct $x, y \in [M]$, $\varphi_x(\alpha) = \varphi_y(\alpha)$ holds for at most κ values of $\alpha \in \mathcal{A}$. Plugging these functions into Lemma A.1 we obtain a d-defective $|\mathcal{A}|\cdot|\mathcal{B}|$-coloring. The number of colors is given by

$$|\mathcal{A}| \cdot |\mathcal{B}| = O\left(\frac{|\mathcal{A}|^2}{\ln M}\right) = O\left(\Upsilon^2 \cdot \ln M\right). \qquad \square$$

Unfortunately, Lemma A.2 only proves the existence of functions φ_x for $x \in [M]$ with the given guarantees. The lemma does not give an explicit way to construct such functions. In the following, we show an explicit algebraic construction that achieves similar guarantees. The same construction has been described as an explicit way to construct families of sets such that no set is contained in the union of k other sets for some parameter k in [12]. Such set systems have been used by Linial to obtain distributed algorithms for the standard coloring problem in [30].

For a prime power q, let $\mathcal{P}(q,\kappa)$ be the set of all $q^{\kappa+1}$ polynomials of degree at most κ in the polynomial ring $\mathbb{F}_q[z]$, where $\mathbb{F}_q$ is the finite field of order q. It is well known that two polynomials of degree at most κ can be equal at no more than κ positions. We can therefore choose the functions φ_x from $\mathcal{P}(q,\kappa)$. The details are given by the following two theorems.

THEOREM A.4. *Assume that we are given an M-coloring of G with defect at most $d' \le d < \Delta/4$. There are explicit functions φ_x for $x \in [M]$ and a constant $\overline{C} > 0$ such that Algorithm* 4 *computes a $\overline{C} \cdot (\Upsilon \log_\Upsilon M)^2$-coloring ψ with defect at most d.*

Proof. Choosing the functions φ_x from $\mathcal{P}(q,\kappa)$ gives $\mathcal{A} = \mathcal{B} = \mathbb{F}_q$. To apply Lemma A.1, we therefore need $q > \kappa \cdot \Upsilon$. Because we need to assign different polynomials to every color $x \in [M]$, we also need $|\mathcal{P}(q,\kappa)| = q^{\kappa+1} \ge M$. We choose

$$\kappa = \lceil \log_\Upsilon M \rceil \quad \text{and} \quad \lfloor \kappa\Upsilon \rfloor + 1 < q \le 2(\lfloor \kappa\Upsilon \rfloor + 1).$$

By the Bertrand–Chebyshev postulate, there must be a prime power q in the given interval. (See, e.g., Theorem 418 in [18].) Choosing the parameters this way guarantees that $q = |\mathcal{A}| > \kappa\Upsilon$. We can certainly assume that $M \ge \Upsilon$, as otherwise the theorem becomes trivial. This implies

$$|\mathcal{P}(q,\kappa)| = q^{\kappa+1} > (\Upsilon \log_\Upsilon M)^{\log_\Upsilon M} = e^{(\ln(M)/\ln(\Upsilon))\cdot(\ln \Upsilon + \ln(\log_\Upsilon M))} \ge e^{\ln M} = M.$$

The resulting coloring ψ employs $|\mathcal{A}| \cdot |\mathcal{B}| = O(\kappa^2 \cdot \Upsilon^2)$. By Lemma A.1, $\mathit{def}(\psi) \le d$. $\square$

Algorithm 4 reduces the number of colors in a single round. One can obtain better defective colorings by iterative applications of the algorithm.

THEOREM A.5. *Assume that we are given an M-coloring of G with defect at most $d' < d < \Delta/4$ such that* $\log(d-d') + \log^* \frac{\Delta - d'}{d-d'} > \log^* M + 2$. *Iteratively applying Algorithm* 4*, an $O((\Delta - d')^2/(d+1-d')^2)$-coloring with defect at most d can be computed in $O(\log^* M)$ rounds.*

Proof. Note that we cannot choose the same value for d throughout the algorithm. If we always use the same value for d in each iterative application of Algorithm 4, we can compute an $O((\Delta - d')^2)$-coloring with defect at most d. (Always choosing the same value d only gives an $O((\Delta - d')^2)$-coloring, because in this case $d + 1 - d' = 1$.) Also, note that an $O((\Delta - d')^2)$-coloring is good enough if $d - d' = O(1)$. We can therefore assume without loss of generality that $d - d'$ is sufficiently large.

We iteratively apply the algorithm T times for an integer $T \geq 1$ that will be determined below. Without loss of generality, we can assume that $d = d' + 2^h$ for some integer $h \geq 0$. For $i \geq 1$, let d_i be the value for d that is used in the ith iterative application. We choose $d_i = d' + (d - d')/2^{T-i}$ and use Algorithm 4 with the polynomial functions described in the proof of Theorem A.4. (It will hold that $T < h$, and so $d_1, d_2, \ldots, d_T$ are all integer numbers.) For convenience, we also define $d_0 := d'$. Further, let M_i be the number of colors after the ith iterative application of Algorithm 4. We choose T to be the smallest positive integer such that

$$\ln^{(T-1)} M \;<\; 32 \cdot \sqrt{\overline{C}} \cdot \frac{\Delta - d'}{d - d'}.$$

(Observe that by the assumption of the theorem,

$$d - d' > 2^{2 + \log^* M - \log^*((\Delta - d')/(d - d'))}.$$

Also,

$$\ln^{(\log^* M - \log^*((\Delta - d')/(d-d')) + 1)} M \leq \log^{(\log^* M - \log^*((\Delta - d')/(d-d')) + 1)} M \leq \frac{\Delta - d'}{d - d'}.$$

Hence, by the minimality of T,

$$T - 1 \leq \log^* M - \log^*((\Delta - d')/(d - d')) + 1.$$

Thus,

$$2^h = d - d' > 2^{2 + \log^* M - \log^*((\Delta - d')/(d - d'))} \geq 2^T,$$

i.e., $h > T$.)

For $i \in \{2, \ldots, T-1\}$, this implies that

$$\begin{aligned} 8\sqrt{\overline{C}} \cdot \frac{\Delta - d'}{d_i + 1 - d_{i-1}} &= 8\sqrt{\overline{C}} \cdot \frac{\Delta - d'}{1 + \frac{d-d'}{2^{T-i}} - \frac{d-d'}{2 \cdot 2^{T-i}}} \\ &\leq 32\sqrt{\overline{C}} \cdot 2^{T-1-i} \cdot \frac{\Delta - d'}{d - d'} \\ &\leq 2^{T-1-i} \cdot \ln^{(T-2)} M \\ &\leq \ln^{(i-1)} M. \end{aligned} \tag{A.2}$$

The inequality on the third line follows from the choice of T. The last inequality because $2 \ln x < x$ for all $x > 0$ and from the definition of T. Next, we use induction to show that for all $i \leq T - 1$, we have

$$M_i \;\leq\; 16 \cdot \overline{C} \cdot \left(\frac{\Delta - d'}{d_i + 1 - d_{i-1}} \cdot \ln^{(i)} M \right)^2. \tag{A.3}$$

For $i = 1$, inequality (A.3) is true by Theorem A.4. For $1 < i \leq T-1$, we get

$$
\begin{aligned}
M_i &\leq \overline{C} \cdot \left(\frac{\Delta - d_{i-1}}{d_i + 1 - d_{i-1}} \cdot \frac{\ln M_{i-1}}{\ln\left(\frac{\Delta - d_{i-1}}{d_i+1-d_{i-1}}\right)} \right)^2 \\
&\leq \overline{C} \cdot \left(\frac{\Delta - d'}{d_i + 1 - d_{i-1}} \cdot \ln M_{i-1} \right)^2 \\
&\leq \overline{C} \cdot \left(\frac{\Delta - d'}{d_i + 1 - d_{i-1}} \quad \cdot \quad \ln \left[16 \cdot \overline{C} \cdot \left(\frac{\Delta - d'}{d_{i-1} + 1 - d_{i-2}} \cdot \ln^{(i-1)} M \right)^2 \right] \right)^2 \\
&\leq \overline{C} \cdot \left(\frac{\Delta - d'}{d_i + 1 - d_{i-1}} \quad \cdot \quad \ln \left[64 \cdot \overline{C} \cdot \left(\frac{\Delta - d'}{d_i + 1 - d_{i-1}} \cdot \ln^{(i-1)} M \right)^2 \right] \right)^2 \\
&\leq \overline{C} \cdot \left(\frac{\Delta - d'}{d_i + 1 - d_{i-1}} \cdot \ln \left((\ln^{(i-1)} M)^4 \right) \right)^2 \\
&= 16 \cdot \overline{C} \cdot \left(\frac{\Delta - d'}{d_i + 1 - d_{i-1}} \cdot \ln^{(i)} M \right)^2 .
\end{aligned}
$$

The first inequality is by Theorem A.4. The second inequality follows because $0 \leq d_{i-1} < d_i \leq \Delta/4$, and thus $\frac{\Delta - d_{i-1}}{d_i+1-d_{i-1}} > e$. The third inequality is by the induction hypothesis. The fourth inequality follows because $\frac{d_i - d_{i-1}}{d_{i-1}-d_{i-2}} = 2$. The fifth inequality follows from inequality (A.2). Applying inequality (A.3) for $i = T-1$ yields

$$
\begin{aligned}
M_{T-1} &\leq 16 \cdot \overline{C} \cdot \left(\frac{\Delta - d'}{d_{T-1} + 1 - d_{T-2}} \cdot \ln^{(T-1)} M \right)^2 \\
&\leq 16 \cdot \overline{C} \cdot \left(4 \cdot \frac{\Delta - d'}{d - d'} \cdot 32\sqrt{\overline{C}} \cdot \frac{\Delta - d'}{d - d'} \right)^2 \\
&= 2^{18} \cdot \left(\sqrt{\overline{C}} \cdot \frac{\Delta - d'}{d - d'} \right)^4 .
\end{aligned}
$$

So after $T - 1 = O(\log^* M)$ iterations the algorithm produces an $O((\frac{\Delta-d'}{d-d'})^4)$-coloring with defect $d_{T-1} = \frac{d+d'}{2}$. Invoke Algorithm 4 one more time (and thus Theorem A.4). Note that in this invocation $\Upsilon = \frac{\Delta - d_{T-1}}{d+1-d_{T-1}} = \frac{2\Delta-(d+d')}{d-d'+2}$. Hence, $\frac{1}{3} \cdot \frac{\Delta-d'}{d-d'} \leq \Upsilon \leq 2 \cdot \frac{\Delta-d'}{d-d'+1}$, and $\log_\Upsilon M_{T-1} = O(1)$. Hence, as a result we obtain an $O((\frac{\Delta-d'}{d-d'+1})^2)$-coloring with defect d. The overall number of iterations is $T = O(\log^* M)$. $\square$

As an initial coloring we use a (legal) $O(\Delta^2)$-coloring which can be computed within $\frac{1}{2}\log^* n + O(1)$ time by algorithms of [30, 39]. With this coloring, $M = C \cdot \Delta^2$ for some universal constant $C > 0$, and $d' = 0$. We next argue that Theorem A.5 is applicable for $32 < d < \Delta/4$. Indeed, if $d \geq \sqrt{\Delta}$, then $\log d + \log^* \frac{\Delta}{d} \geq \frac{1}{2}\log\Delta > \log^*(C \cdot \Delta^2) + 2$ for a sufficiently large Δ. For $32 < d < \sqrt{\Delta}$, it holds that $\log d + \log^* \frac{\Delta}{d} > 5 + \log^* \sqrt{\Delta} \geq 5 + \log^*(\log\Delta) = 4 + \log^*\Delta$. On the other hand, for a sufficiently large Δ, $\log^*(C \cdot \Delta)^2 + 2 \leq \log^*(2^\Delta) + 2 = \log^*\Delta + 3$. Hence $\log d + \log^* \frac{\Delta}{d} > 4 + \log^*\Delta > \log^*(C \cdot \Delta^2) + 2$, and thus the assumption of Theorem A.5 holds.

By Theorem A.5, invoking the algorithm with the initial legal $O(\Delta^2)$-coloring with d in the range $32 < d < \Delta/4$ results in a d-defective $O(\frac{\Delta^2}{d^2})$-coloring. This invocation requires $O(\log^* \Delta)$ time. So a d-defective $O(\frac{\Delta^2}{d^2})$-coloring can be computed in total $O(\log^* n)$ time for d in the above range. For $d \leq 32$, the initial legal $O(\Delta^2)$-coloring is a d-defective $O(\frac{\Delta^2}{d^2})$-coloring as well. On the other hand, for $d \geq \Delta/4$, we can compute a $(\Delta/8)$-defective $O(\Delta^2/(\Delta/8)^2)$-coloring, which is, in particular, a d-defective $O(\Delta^2/d^2)$-coloring as well. To summarize, for every $1 \leq d \leq \Delta$, a d-defective $O(\Delta^2/d^2)$-coloring can be computed in $O(\log^* n)$ rounds.

COROLLARY A.6. *For every p, $1 \leq p \leq \Delta$, a $\lfloor \Delta/p \rfloor$-defective $O(p^2)$-coloring can be computed in $O(\log^* n)$ rounds.*

REFERENCES

[1] N. Alon, L. Babai, and A. Itai, *A fast and simple randomized parallel algorithm for the maximal independent set problem*, J. Algorithms, 7 (1986), pp. 567–583.

[2] J. Andrews and M. Jacobson, *On a generalization of a chromatic number*, Congress. Numer., 47 (1985), pp. 33–48.

[3] B. Awerbuch, A. V. Goldberg, M. Luby, and S. Plotkin, *Network decomposition and locality in distributed computation*, in Proceedings of the 30th Annual IEEE Symposium on Foundations of Computer Science, Research Triangle Park, NC, 1989, pp. 364–369.

[4] L. Barenboim and M. Elkin, *Sublogarithmic distributed MIS algorithm for sparse graphs using Nash–Williams decomposition*, Distrib. Comput., 22 (2010), pp. 363–379.

[5] L. Barenboim and M. Elkin, *Distributed (Delta +1)-coloring in linear (in Delta) time*, in Proceedings of the 41st Annual ACM Symposium on Theory of Computing, Bethesda, MD, 2009, pp. 111–120.

[6] L. Barenboim and M. Elkin, *Distributed (Delta +1)-Coloring in Linear (in Delta) Time*, preprint, http://arXiv.org/abs/0812.1379v2[cs.DC].

[7] L. Barenboim and M. Elkin, *Deterministic distributed vertex coloring in polylogarithmic time*, J. ACM, 58 (2011), 23.

[8] L. Barenboim and M. Elkin, *Distributed deterministic edge coloring using bounded neighborhood independence*, in Proceedings of the 30th Annual ACM Symposium on Principles of Distributed Computing, San Jose, CA, 2011, pp. 129–138.

[9] R. Cole and U. Vishkin, *Deterministic coin tossing with applications to optimal parallel list ranking*, Inform. Control, 70 (1986), pp. 32–53.

[10] L. Cowen, R. Cowen, and D. Woodall, *Defective colorings of graphs in surfaces: Partitions into subgraphs of bounded valence*, J. Graph Theory, 10 (1986), pp.. 187–195.

[11] L. Cowen, W. Goddard, and C. Jesurum, *Coloring with defect*, in Proceedings of the 8th Annual ACM-SIAM Symposium on Discrete Algorithms, New Orleans, LA, 1997, pp. 548–557.

[12] P. Erdős, P. Frankl, and Z. Füredi, *Families of finite sets in which no set is covered by the union of r others*, Israel J. Math., 51 (1985), pp. 79–89.

[13] M. Frick, *A survey of (m, k)-colorings*, in Quo Vadis, Graph Theory?, Ann. Discrete Math. 55, North–Holland, Amsterdam, 1993, pp. 45–57.

[14] S. Gandham, M. Dawande, and R. Prakash, *Link scheduling in sensor networks: Distributed edge coloring revisited*, in Proceedings of the 24th International Conference on Computer Communications (INFOCOM), Vol. 4, Miami, FL, 2005, pp. 2492–2501.

[15] A. Goldberg and S. Plotkin, *Efficient parallel algorithms for $(\Delta + 1)$-coloring and maximal independent set problem*, in Proceedings of the 19th Annual ACM Symposium on Theory of Computing, New York, 1987, pp. 315–324.

[16] A. V. Goldberg, S. A. Plotkin, and G. E. Shannon, *Parallel symmetry-breaking in sparse graphs*, SIAM J. Discrete Math., 1 (1988), pp. 434–446.

[17] F. Harary and K. Jones, *Conditional colorability* II: *Bipartite variations*, Congress. Numer., 50 (1985), pp. 205–218.

[18] G. Hardy and E. Wright, *An Introduction to the Theory of Numbers*, 5th ed., Oxford University Press, Oxford, UK, 1980.

[19] S. H. Hosseini, B. Litow, M. Malkawi, J. McPherson, and K. Vairavan, *Analysis of a graph coloring based distributed load balancing algorithm*, J. Parallel Distrib. Comput., 10 (1990), pp. 160–166.

[20] Ö. Johansson, *Simple distributed $(\Delta + 1)$-coloring of graphs*, Inform. Process. Lett., 70 (1999), pp. 229–232.

[21] A. Korman, J. Sereni, and L. Viennot, *Toward more localized local algorithms: Removing assumptions concerning global knowledge*, in Proceedings of the 30th Annual ACM Symposium on Principles of Distributed Computing, San Jose, CA, 2011, pp. 49–58.

[22] F. Kuhn, *Deterministic distributed algorithms for weak colorings*, in Proceedings of the 21st Annual ACM Symposium on Parallelism in Algorithms and Architectures, Calgary, Alberta, Canada, 2009, pp. 138–144.

[23] F. Kuhn, T. Moscibroda, T. Nieberg, and R. Wattenhofer, *Fast deterministic distributed maximal independent set computation on growth-bounded graphs*, in Proceedings of the 19th International Symposium on Distributed Computing, Kraków, Poland, 2005, pp. 273–287.

[24] F. Kuhn, T. Moscibroda, and R. Wattenhofer, *What cannot be computed locally!*, in Proceedings of the 23rd Annual ACM Symposium on Principles of Distributed Computing, St. John's, Newfoundland, Canada, 2004, pp. 300–309.

[25] F. Kuhn, T. Moscibroda, and R. Wattenhofer, *Local Computation: Lower and Upper Bounds*, preprint, http://arXiv.org/abs/1011.5470, 2010.

[26] F. Kuhn, T. Moscibroda, and R. Wattenhofer, *On the locality of bounded growth*, in Proceedings of the 24th Annual ACM Symposium on Principles of Distributed Computing, Las Vegas, NV, 2005, pp. 60–68.

[27] F. Kuhn and R. Wattenhofer, *On the complexity of distributed graph coloring*, in Proceedings of the 25th Annual ACM Symposium on Principles of Distributed Computing, Denver, CO, 2006, pp. 7–15.

[28] S. Kutten and D. Peleg, *Fast distributed construction of k-dominating sets and applications*, in Proceedings of the 14th Annual ACM Symposium on Principles of Distributed Computing, Ottawa, Ontario, Canada, 1995, pp. 238–249.

[29] N. Linial, *Distributive graph algorithms global solutions from local data*, in Proceedings of the 28th Annual IEEE Symposium on Foundation of Computer Science, Los Angeles, CA, 1987, pp. 331–335.

[30] N. Linial, *Locality in distributed graph algorithms*, SIAM J. Comput., 21 (1992), pp. 193–201.

[31] M. Luby, *A simple parallel algorithm for the maximal independent set problem*, SIAM J. Comput., 15 (1986), pp. 1036–1053.

[32] L. Lovász, *On decompositions of graphs*, Studia Sci. Math. Hungar., 1 (1966), pp. 237–238.

[33] A. Mayer, M. Naor, and L. Stockmeryer, *Local computations on static and dynamic graphs*, in Proceedings of the 3rd Israel Symposium on the Theory of Computing and Systems, Tel Aviv, Israel, 1995, pp. 268–278.

[34] A. Panconesi and A. Srinivasan, *On the complexity of distributed network decomposition*, J. Algorithms, 20 (1995), pp. 581–592.

[35] D. Peleg, *Distributed Computing: A Locality-Sensitive Approach*, Discrete Math. Appl. 5, SIAM, Philadelphia, 2000.

[36] S. Plotkin, *Graph-Theoretic Techniques for Parallel, Distributed, and Sequential Computation*, Doctoral thesis, MIT Cambridge Lab for Computer Science, MIT, Cambridge, MA, 1988.

[37] J. Schneider and R. Wattenhofer, *A log-star distributed maximal independent set algorithm for growth bounded graphs*, in Proceedings of the 27th Annual ACM Symposium on Principles of Distributed Computing, Toronto, Ontario, Canada, 2008, pp. 35–44.

[38] J. Schneider and R. Wattenhofer, *A new technique for distributed symmetry breaking*, in Proceedings of the 29th Annual ACM Symposium on Principles of Distributed Computing, Zurich, Switzerland, 2010, pp. 257–266.

[39] M. Szegedy and S. Vishwanathan, *Locality based graph coloring*, in Proceedings of the 25th Annual ACM Symposium on Theory of Computing, San Diego, CA, 1993, pp. 201–207.