Faster Deterministic Distributed Coloring Through Recursive List Coloring

Fabian Kuhn*

Abstract

We provide novel deterministic distributed vertex coloring algorithms. As our main result, we give a deterministic distributed algorithm to compute a $(\Delta+1)$-coloring of an n-node graph with maximum degree Δ in $2^{O(\sqrt{\log \Delta})} \cdot \log n$ rounds. For graphs with arboricity a, we obtain a deterministic distributed algorithm to compute a $(2+o(1))a$-coloring in time $2^{O(\sqrt{\log a})} \cdot \log^2 n$. Further, for graphs with bounded neighborhood independence, we show that a $(\Delta+1)$-coloring can be computed more efficiently in time $2^{O(\sqrt{\log \Delta})} + O(\log^* n)$. This in particular implies that also a $(2\Delta-1)$-edge coloring can be computed deterministically in $2^{O(\sqrt{\log \Delta})} + O(\log^* n)$ rounds, which improves the best known time bound for small values of Δ. All results even hold for the list coloring variants of the problems. As a consequence, we also obtain an improved deterministic $2^{O(\sqrt{\log \Delta})} \cdot \log^3 n$-round algorithm for Δ-coloring non-complete graphs with maximum degree $\Delta \geq 3$. Most of our algorithms only require messages of $O(\log n)$ bits (including the $(\Delta+1)$-vertex coloring algorithms).

Our main technical contribution is a recursive deterministic distributed list coloring algorithm to solve list coloring problems with lists of size $\Delta^{1+o(1)}$. Given some list coloring problem and an orientation of the edges, we show how to recursively divide the global color space into smaller subspaces, assign one of the subspaces to each node of the graph, and compute a new edge orientation such that for each node, the list size to out-degree ratio degrades at most by a constant factor on each recursion level.

1 Introduction and Related Work

Distributed graph coloring is one of the core problems and probably also the most extensively studied problem in the area of *distributed graph algorithms*. It is one of the classic problems to study the more general fundamental question of how to efficiently break symmetries in a network in a distributed way. The network is modeled as an n-node graph $G=(V,E)$ in which the nodes communicate over the edges and the objective is to compute a proper coloring of G. If the maximum degree of G is Δ, the goal typically is to compute a coloring with $f(\Delta)$ colors for some function f, where the standard task is to color G with $\Delta+1$ colors.[1] The problem is usually studied in a synchronous message passing model known as the LOCAL model [Lin92, Pel00]. Each node $v \in V$ is equipped with a unique $O(\log n)$-bit identifier $\mathrm{ID}(v)$. The nodes operate in synchronous rounds, where in each round each node can send a message to each of its neighbors. Initially, the nodes only know the IDs of their neighbors and at the end, each node needs to know its own color of the computed vertex coloring of G. The time complexity of an algorithm is measured as the number of rounds. In the LOCAL model, the maximum message size is not bounded. The more realistic variant of the model where messages have to be of size at most $O(\log n)$ bits is known as the CONGEST model [Pel00].

Early Work on Distributed Coloring: The work on distributed coloring started more than 30 years ago with work on solving symmetry breaking problems in the parallel setting [ABI86, CV86, GPS88, Lub86] and with a seminal paper by Linial [Lin87], which introduced the LOCAL model for solving graph problems in a distributed setting. Linial in particular showed that a graph can be colored (deterministically) in $O(\log^* n)$ rounds with $O(\Delta^2)$ colors and that $\Omega(\log^* n)$ rounds are necessary even for graph of degree 2.[2] Naor later extended the lower bound to randomized algorithms [Nao91]. Together with a simple single-round color elimination scheme, the $O(\Delta^2)$-coloring algorithm of Linial directly implies a deterministic $O(\Delta^2 + \log^* n)$-time algorithm for $(\Delta+1)$-coloring. With a refined single-round color reduction scheme, the running time of this algorithm can be improved to $O(\Delta \log \Delta + \log^* n)$ [SV93, KW06]. If randomization is allowed, already the early works imply that a $(\Delta+1)$-coloring can always be computed in $O(\log n)$ rounds [ABI86, Lub86, Lin87].

*Dept. of Computer Science, University of Freiburg, Germany, kuhn@cs.uni-freiburg.de.

[1] $\Delta+1$ colors is what can be guaranteed by a simple sequential greedy algorithm. It also is the minimum number of colors necessary such that an arbitrary partial proper coloring can be extended to a proper coloring of all the nodes.

[2] The $\log^* n$ function measures the number of times the $\log(\cdot)$ function has to applied iteratively in order to obtain a constant value when starting with value n.

Exponential Separation Between Randomized and Deterministic Complexity: While there are very efficient (and simple) randomized algorithms for solving $(\Delta+1)$-coloring, the best known deterministic algorithm is almost exponentially slower an requires time $2^{O(\sqrt{\log n})}$ [PS95b]. The algorithm is based on a structure known as a network decomposition, which was introduced by Awerbuch et al. [AGLP89] and decomposes the graph into clusters of small diameter and a coloring of the cluster graph with a small number of colors. The distributed coloring problem is then solved in a brute-force way: One iterates over the cluster colors and for each cluster color, the respective clusters are colored by locally learning the complete cluster topology in time proportional to the cluster diameter. The question of whether there really is an exponential gap between the best randomized and deterministic time complexities for the $(\Delta+1)$-coloring problem (and in fact for many more problems) is one of the most fundamental and long-standing open problems in the area of distributed graph algorithms (see, e.g., [Lin87, BE13, GKM17]). In fact, in their book on distributed graph coloring, Barenboim and Elkin [BE13] explicitly state that *"perhaps the most fundamental open problem in this field is to understand the power and limitations of randomization."*. In the present paper, we make progress on establishing the deterministic complexity of the distributed $(\Delta+1)$-coloring problem. While we do not improve the best time bound for general graphs, we significantly improve the best known bound for a large range of values for Δ. We also give the first $2^{O(\sqrt{\log n})}$-time deterministic $(\Delta+1)$-coloring algorithm that does not depend on computing a network decomposition. In fact, our algorithm even works in the CONGEST model.

Deterministic Distributed Coloring: As we focus on deterministic distributed coloring algorithms, we first discuss this case in more detail. While as a function of n, the $2^{O(\sqrt{\log n})}$-time solution of [PS95b] is still the fastest known deterministic algorithm, there has been extensive work on determining the round complexity of distributed coloring as a function of the maximum degree Δ. The first improvement over the $O(\Delta\log\Delta+\log^* n)$-algorithm mentioned above was obtained independently by Barenboim and Elkin [BE09] and Kuhn [Kuh09], who showed that the problem can be solved in time $O(\Delta+\log^* n)$. The algorithms of [BE09, Kuh09] introduced *distributed defective coloring* as a new technique. A d-defective c-coloring of a graph is a coloring with c colors such that the subgraph induced by each color class has maximum degree at most d. Such a coloring allows to solve the distributed coloring problem in a divide-and-conquer fashion by decomposing the graph into several smaller-degree subgraphs. This basic idea was further developed by Barenboim and Elkin in [BE10], where they showed that for every constant $\varepsilon>0$, it is possible to compute an $O(\Delta)$-coloring in time $O(\Delta^{\varepsilon}\log n)$ and that a $\Delta^{1+\varepsilon}$-coloring can be computed in time $O(\operatorname{poly}\log\Delta\cdot\log n)$. The paper introduced *arbdefective colorings* as a generalization of defective colorings. Such colorings partition the nodes of a graph into subgraphs of small arboricity[3] rather into subgraphs of small degree. Note that while the paper shows that there are very efficient coloring algorithms that use a relatively small number of colors, it does not improve the time required to compute a $(\Delta+1)$-coloring. A further improvement for the $(\Delta+1)$-coloring problem was obtained by Barenboim [Bar15], who improves the time from $O(\Delta+\log^* n)$ to $O(\Delta^{3/4}\log\Delta+\log^* n)$. The key insight of the paper was to consider the more general list coloring problem rather than the ordinary vertex coloring problem. In a C-list coloring problem, every node $v\in V$ has to choose a color from a list L_v of size $|L_v|\geq C$, which is given to v initially. List colorings in particular allow to extend previously computed partial colorings. Considering list colorings thus allows to recursively decompose the graph and to then color the parts sequentially without having to divide the color space among the different parts. The result of [Bar15] was improved to $O(\sqrt{\Delta}\cdot\operatorname{poly}\log\Delta+\log^* n)$ by Fraigniaud, Heinrich, and Kosowski [FHK16], who introduced a generalized version of list coloring, which they could solve more efficiently. Our algorithms build on the techniques developed in all the above works. From a technical point of view, our main contribution is an extension of the recursive algorithm of [BE10] to list colorings. We then combine this list coloring algorithm with the framework developed in [Bar15, FHK16].

We conclude the discussion on existing deterministic distributed coloring by mentioning a few additional notable results. While except for the original $\Omega(\log^* n)$ bound, no lower bound for the LOCAL model has been proven, it was recently shown that in a weak version of the LOCAL model (called the SET−LOCAL model), the $(\Delta+1)$-coloring problem has a lower bound of $\Omega(\Delta^{1/3})$ [HMKS16]. Further, in [BEG18], an alternative and particularly elegant deterministic $O(\Delta+\log^* n)$-time $(\Delta+1)$-coloring algorithm was given. The algorithm reduces the colors iteratively in a round-by-round manner rather than by using defective colorings. The technique also allows to slightly improve the best $(\Delta+1)$-coloring bound of [FHK16] to $O(\sqrt{\Delta\log\Delta}\log^*\Delta+\log^* n)$ [BEG18]. Finally, some authors also considered

[3] The arboricity of a graph G is the minimum number of forests into which the edge set of G can be decomposed.

the problem of coloring the nodes of a graph with Δ instead of $\Delta+1$ colors. Although this problem has a less local nature, it has been shown that up to poly $\log n$ factors, Δ-colorings can essentially be computed as fast as a $(\Delta+1)$-colorings [PS95a, GHKM18].

Randomized Distributed Coloring: There has also been a lot of work on understanding the randomized complexity of the distributed coloring problem [KSOS06, SW10, BEPS12, PS13, HSS16, GHKM18, CLP18]. A particularly important contribution was provided by Barenboim, Elkin, Pettie, and Schneider [BEPS12], who introduced the so-called graph shattering technique to the theory of distributed graph algorithms. The paper shows that a $(\Delta+1)$-coloring can be computed in time $O(\log\Delta)+2^{O(\sqrt{\log\log n})}$. The basic idea of shattering is to first use an efficient randomized algorithm that solves the problem on most of the graph, has a time complexity that often just depends on Δ, and that essentially only leaves unsolved components of size poly $\log n$. In the case of [BEPS12], this randomized part of the algorithm requires $O(\log\Delta)$ rounds. The remaining problems on poly $\log n$-size components are then solved by using the most efficient deterministic algorithm for the problem, i.e., in time $2^{O(\sqrt{\log\log n})}$ in the case of distributed coloring. In fact, in [CKP16], Chang, Kopelowitz, and Pettie showed that this shattering technique is essentially necessary and that the randomized complexity of coloring (and many more graph problems) on graphs of size n is at least the deterministic complexity of the problem on graphs of size $\sqrt{\log n}$. Note that this adds another very strong motivation to understand the deterministic complexity of the distributed coloring problem. In subsequent work, the time bound of the randomized $(\Delta+1)$-coloring problem was improved to $O(\sqrt{\log\Delta})+2^{O(\sqrt{\log\log n})}$ in [HSS16] and finally to $2^{O(\sqrt{\log\log n})}$ in [CLP18]. Note that in [CLP18], the time is dominated by the deterministic time for coloring components of size poly $\log n$. Hence, the current best randomized time complexity of the $(\Delta+1)$-coloring problem can only be improved by at the same time also improving the best deterministic algorithm for the problem.

Distributed Edge Coloring: A particularly extensively studied special case of distributed coloring is the edge coloring problem. The task corresponding to computing a $(\Delta+1)$-vertex coloring is to compute an edge coloring with $2\Delta-1$ colors. It was already realized 20 years ago that the distributed edge coloring problem has a simpler structure than the distributed vertex coloring problem and there have been many contributions towards the randomized [PS97, DGP98, EPS15, CHL+18] and the deterministic [PR01, CHK01, GS17, FGK17, GHK18, GKMU18, Har18, SV19] complexity of the problem. In both case, we now know efficient algorithms for computing $(2\Delta-1)$-edge colorings and (under some conditions in the deterministic case) even for computing $(1+\varepsilon)\Delta$-edge colorings. Note that by Vizing's classic theorem, every graph has an edge coloring with $\Delta+1$ colors [Viz64].

A particularly important step was obtained by Fischer et al. [FGK17], who show that the $(2\Delta-1)$-edge coloring problem can be solved deterministically in $O(\text{poly}\log\Delta\cdot\log n)$ rounds. Previously, the best deterministic algorithm for general graphs had a time complexity of $2^{O(\sqrt{\log n})}$ and as in the case of the best $(\Delta+1)$-vertex coloring algorithm, it was a brute-force algorithm based on first computing a network decomposition [AGLP89, PS95b]. Together with the best known randomized algorithms (which use the shattering technique), the algorithm of [FGK17] also implied that the $(2\Delta-1)$-edge coloring problem has a randomized complexity of poly $\log\log n$. The current best time complexities for computing an $(2\Delta-1)$-edge coloring in general graphs are $\tilde{O}(\log^2\Delta\log n)$ for the deterministic and $\tilde{O}(\log^3\log n)$ for the randomized setting [Har18]. For small values of Δ, the best known complexity is $O(\sqrt{\Delta\log\Delta}\log^*\Delta+\log^* n)$ [FHK16, BEG18], as for computing a $(\Delta+1)$-vertex coloring. As one of our contributions, we improve this last result to $2^{O(\sqrt{\log\Delta})}+O(\log^* n)$.

1.1 Contributions and Main Ideas

We will now describe our contributions and the core ideas that lead to those contributions.

Recursive Distributed List Coloring: As stated above, from a technical point of view, our main contribution is a recursive solution for the list coloring problem. Our algorithm generalizes the algorithm of Barenboim and Elkin in [BE10]. In [BE10], the node set of G is recursively divided into smaller sets and the subgraphs of G induced by these node sets are then colored independently by using disjoint sets of colors for the different parts. The recursive partitioning is done by a so-called arbdefective coloring. Given a graph G with an edge orientation where the out-degree of each node is at most β, a β'-arbdefective coloring of G is a coloring and a new orientation of G such that each node has at most β' out-neighbors (w.r.t. to the new orientation) of its own color. In [BE10], it is shown that in time $O(k^2\log n)$, it is possible to compute an $O(\beta/k)$-arbdefective coloring with k colors. Note that this recursive partitioning of the graphs requires the total number of colors to be sufficiently large and it in particular cannot provide a $(\Delta+1)$-coloring. In each

recursive step, the number of parts grows by a constant factor more than the maximum out-degree is reduced.

When trying to extend the above solution to list coloring, one could try to divide the space of all possible colors into k parts and assign each node to one of the parts. If we could find such a partition such that the color list L_v of each node is roughly divided by a factor k, we could use the same arbdefective colorings as in [BE10] to recursively divide problem. Finding such a partition of the color space is however a problem that does not have a local solution (whether such a partition is good depends on all the lists). We therefore adapt the solution of [BE10] in a more careful way that is specifically designed to work for the list coloring problem. We consider an arbitrary partition of the global color space into k (roughly) equal parts and we partition the nodes by assigning each node to one of the k parts. However, instead of equally reducing the out-degrees, we find a partition and a new orientation such that for every node v, the size of its list gets smaller at roughly the same rate as the out-degree of v (again up to a constant factor). For a formal definition of our generalization of arbdefective colorings, we refer to Definition 2.1. If the initial lists are sufficiently large, this recursive partitioning allows to maintain list coloring problems that are solvable. We formally prove the following main technical result. The construction follows the same basic ideas as the construction in [BE10], carefully adapted to the more general list coloring setting.

THEOREM 1.1. *Let $G = (V, E)$ be a graph where all edges are oriented and for each $v \in V$, let $\beta(v)$ be the out-degree of v. Let $\varepsilon \in (0, 1]$ and $r \geq 1$ be two parameters and assume that each node v is assigned a list L_v of size at least $|L_v| > (2+\varepsilon)^r \beta(v)$ consisting of colors from a color space of size at most C. Then, there is a deterministic $O\big(\frac{r \cdot C^{2/r}}{\varepsilon^3} \cdot \log n\big)$-round algorithm to solve the given list coloring instance. The algorithm requires messages of size $O(\log n + \log C)$.*

Note that Theorem 1.1 directly gives a generalization of the results of [BE10] to list colorings. Specifically, for an arbitrary initial edge orientation, the theorem implies that if C is a most polynomial in Δ, for every constant $\varepsilon > 0$, there are deterministic distributed coloring algorithms to solve list coloring instances with lists of size $c\Delta$ for a sufficiently large constant $c > 1$ in $O(\Delta^\varepsilon \log n)$ rounds and list coloring instances with lists of size $\Delta^{1+\varepsilon}$ in $O(\text{poly}\log\Delta \cdot \log n)$ rounds. The two results are obtained by setting $r = \Theta(1/\varepsilon)$ and $r = \Theta(\varepsilon \log \Delta)$, respectively.

$(\Delta+1)$-List Coloring: By using the framework developed in [Bar15, FHK16], we can use the list coloring algorithm of Theorem 1.1 to efficiently compute list colorings for lists of size $\Delta + 1$. The basic idea is to first use a defective coloring to partition the nodes of G into $\Delta^{O(\varepsilon)}$ parts of maximum degree at most $\Delta^{1-\varepsilon}$. The algorithm then sequentially iterates through the $\Delta^{O(\varepsilon)}$ parts and when processing part i, it colors the nodes of that part so that the coloring is consistent with the coloring computed for the previous parts. Note that when coloring a part in a consistent way with already colored parts, each node v is only allowed to choose a color that has not already been chosen by a neighbor of v. Coloring each part is therefore a list coloring problem, even if the original problem is not a list coloring problem. As long as at least half the neighbors of a node v are uncolored when processing node v, the list of v is of length more than $\Delta/2$. Because each part has maximum degree at most $\Delta^{1-\varepsilon}$, we are thus in a setting where we can apply Theorem 1.1. After going through the $\Delta^{O(\varepsilon)}$ parts, we get a partial list coloring where the maximum degree of the graph induced by the uncolored nodes is at most $\Delta/2$. Iterating $O(\log\Delta)$ times then gives a $(\Delta+1)$-coloring. By setting $\varepsilon = \Theta(1/\sqrt{\log\Delta})$, we obtain our main theorem, which is stated in the following.

THEOREM 1.2. *Let $G = (V, E)$ be an n-node graph with maximum degree Δ and assume that we are given a $(\deg(v)+1)$-list coloring instance with colors from a color space of size $\text{poly}(\Delta)$. Then, there is a deterministic* CONGEST *model algorithm to solve this list coloring problem in time $2^{O(\sqrt{\log\Delta})} \log n$.*

Note that by $(\deg(v)+1)$-list coloring we refer to a list coloring instance where the size of the list of each node v is at least $\deg(v)+1$. The theorem thus in particular implies that the standard $(\Delta+1)$-coloring problem can be solved in time $2^{O(\sqrt{\log\Delta})} \log n$ in the CONGEST model. Theorem 1.2 improves the best known deterministic time complexity of $\min\left\{\tilde{O}(\sqrt{\Delta}) + O(\log^* n), 2^{O(\sqrt{\log n})}\right\}$ for solving $(\Delta+1)$-coloring even in the LOCAL model for all Δ between $\log^{2+\varepsilon} n$ and $n^{o(1)}$. The theorem also extends the range of Δ for with $(\Delta+1)$-coloring can be solved in deterministic $\text{poly}\log n$ time from $\Delta \leq \text{poly}\log n = 2^{O(\log\log n)}$ to $\Delta = 2^{O(\log^2 \log n)}$. It is further notable that our algorithm gives an alternative way of solving $(\Delta+1)$-coloring in deterministic $2^{O(\sqrt{\log n})}$ time. Our algorithm does not depend on computing a network decomposition and solving the clusters in a brute-force way and the algorithm even works in the CONGEST model (i.e., with $O(\log n)$-bit messages).

Bounded Neighborhood Independence: A graph $G = (V, E)$ is said to have neigborhood independence θ if every node $v \in V$ has at most θ independent (i.e., pairwise non-adjacent) neighbors (see also Section 2). In [BE11], Barenboim and Elkin showed that in graphs of bounded neighborhood independence, the recursive coloring algorithm of [BE10] can be improved: Essentially, they show that the multiplicative $O(\log n)$ factor in all the time bounds of [BE10] can be replaced by an additive $O(\log^* n)$ term if the neighborhood independence is $\theta = O(1)$. We show that the same improvement can also be obtained for list colorings, leading to the following theorem.

THEOREM 1.3. *Let $G = (V, E)$ be an n-node graph with maximum degree Δ and neighborhood independence at most $\theta \geq 1$. Assume that we are given a $(\deg(v) + 1)$-list coloring instance with colors from a color space of size* $\text{poly}(\Delta)$. *Then, there is a deterministic* CONGEST *model algorithm to solve this list coloring problem in time $2^{O(\sqrt{\log \theta \cdot \log \Delta})} + O(\log^* n)$.*

The line graph of any graph G has neighborhood independence at most 2 (for each edge $\{u, v\}$ of G, the adjacent edges can be covered by two cliques in the line graph, defined by the edges incident to u and the edges incident to v). Theorem 1.3 thus leads to the following direct corollary for the *distributed edge coloring* problem.

COROLLARY 1.1. *Let $G = (V, E)$ be an n-node graph with maximum degree Δ and assume that we are given a lisnt edge coloring instance with lists from a color space of size at most* $\text{poly}(\Delta)$. *If the list L_e of each edge $e = \{u, v\} \in E$ is of size at least $\deg(u) + \deg(v) - 1$, there is a deterministic* LOCAL *algorithm to solve the given list edge coloring problem in time $2^{O(\sqrt{\log \Delta})} + O(\log^* n)$.*

Note that even though on the line graph, the algorithm of Theorem 1.3 can be implemented in the CONGEST model (i.e., with messages of size $O(\log n)$ bits), it is not immediate whether the same is also true when implementing the algorithm on G. The theorem therefore only implies a LOCAL algorithm for the edge coloring problem.[4] The result in particular improves the time for computing a $(2\Delta - 1)$-edge coloring for $\Delta \leq \log^{2-\varepsilon} n$ from $\tilde{O}(\sqrt{\Delta}) + O(\log^* n)$ to $2^{O(\sqrt{\log \Delta})} + O(\log^* n)$.

Applications: Theorem 1.2 has two immediate applications. If $G = (V, E)$ is a graph with arboricity a, one can use an algorithm from [BE08] to decompose V into $p = O(\log(n)/\varepsilon)$ parts $V_1, V_2, \ldots, V_p$ such that every node $v \in V_i$ $(\forall i)$ has at most $(2 + \varepsilon)a$ neighbors in $\bigcup_{j=i}^{p} V_j$. The time for computing this decomposition is $O(\log(n)/\varepsilon)$ (deterministically in the CONGEST model). By sequentially list coloring the sets $V_1, \ldots, V_p$ in reverse order, we get the following corollary of Theorem 1.2.

COROLLARY 1.2. *Let $G = (V, E)$ be an n-node graph with arboricity a and assume that for some $\varepsilon > 0$, we are given a $\lceil (2 + \varepsilon)a \rceil$-list coloring instance with lists from a color space of size at most* $\text{poly}(\Delta)$. *Then there is a deterministic* CONGEST *model algorithm that solves this list coloring problem in time $\frac{1}{\varepsilon} \cdot 2^{O(\sqrt{\log a})} \cdot \log^2 n$.*

We note that the number of colors is close to optimal as there are graphs with arboricity a and chromatic number $2a$ (see also the more detailed discussion in Section 2).

Finally, we also obtain an improved algorithm for deterministically computing a Δ-coloring of a graph G. By a classic result of Brooks from 1941 [Bro09], it is well-known that every graph G with maximum degree Δ can be colored with Δ colors unless G is an odd cycle or a complete graph. In [GHKM18], it was shown that for $\Delta \geq 3$, in the LOCAL model, the problem of computing a Δ-coloring can be deterministically reduced to $O(\log^2 n)$ instances of computing a $(\deg(v) + 1)$-list coloring problem. This implies the following corollary of Theorem 1.2.

COROLLARY 1.3. *Let $G = (V, E)$ be an n-node graph with maximum degree Δ. If G is not a complete graph, a $\max\{3, \Delta\}$-coloring of G can be computed deterministically in $2^{O(\sqrt{\log \Delta})} \cdot \log^3 n$ rounds in the* LOCAL *model.*

2 Model and Preliminaries

Mathematical Preliminaries: Let $G = (V, E)$ be an undirected graph. For a node $v \in V$, we use $N_G(v) := \{u \in V : \{u, v\} \in E\}$ to denote the set of neighbors and we use $\deg_G(v) := |N(v)|$ to denote the degree of v (if it is clear from the context, we omit the subscript G). The maximum degree of G is denoted by $\Delta(G)$ and the arboricity of G (definition see below) is denoted by $a(G)$. Again, when G is clear from the context, we just use Δ and a instead of $\Delta(G)$ and $a(G)$. The *neighborhood independence* of a graph $G = (V, E)$ is the size of the largest independent set of any subgraph $G[N(v)]$ that is induced by the neighbors $N(v)$ of some node $v \in V$. For any natural number $k \in \mathbb{N}$, we use $[k]$ to denote the set $[k] := \{1, \ldots, k\}$. If not specified otherwise, logarithms are to base 2, i.e., $\log x$ means $\log_2 x$.

[4] We believe that it might be possible to actually also implement the algorithm in the CONGEST model on G.

Communication Model: We assume the standard synchronous communication model in graphs. The network is modeled as an n-node graph $G = (V, E)$, where each node hosts a distributed process (which for simplicity will be identified with the node itself). Time is divided into synchronous rounds. In each round, each node $v \in V$ can send a possibly different message to each of its neighbors in G, receive the messages sent to it by its neighbors in the current round, and perform some arbitrary internal computation. We assume that each node $v \in V$ has a unique $O(\log n)$-bit identifier $\mathrm{ID}(v)$. Depending on how much information can be sent over each edge of G in a single round, one usually distinguishes two variants of this model. In the LOCAL model, the messages can be of arbitrary size and in the CONGEST model, messages have to be of size at most $O(\log n)$ bits. To keep the arguments as simple as possible, we assume that all nodes know the number of nodes n and the maximum degree Δ of G. For algorithms that depend on the arboricity a of G, we also assume that all nodes know a. All our results are relatively straightforward to generalize to the case where the nodes only know a polynomial upper bound on n and linear upper bounds on Δ and a.[5]

Arboricity: The *arboricity* a of a graph $G = (V, E)$ is the minimum number of forests that are needed to cover all edges of E. This implies that the number of edges is at most $a(n-1)$ and more generally, any k-node subgraph has at most $a(k-1)$ edges. The arboricity is thus a measure of the sparsity of G. A classic theorem of Nash-Williams shows that the arboricity is equal to the smallest integer a such that every subgraph $G_S = (V_S, E_S)$ of G has at most $a(|V_S| - 1)$ edges [NW64].

Note that every subgraph of a graph G of arboricity a has a node of degree strictly less than $2a$. By iteratively removing a node of degree at most $2a-1$ and coloring the nodes in the reverse order, we can therefore always find a vertex coloring with at most $2a$ colors. Note that this is tight in general as Nash-William's theorem for example directly implies that the complete graph K_{2a} of size $2a$ has arboricity a. The process of iteratively removing a node of degree at most $2a-1$ until the graph is empty also implies that every graph of arboricity a has an acyclic orientation of the edges such that each node has out-degree at most $2a-1$. If we drop the requirement that the orientation needs to be acyclic, the edges can be oriented such that the out-degree of each node is at most a (by taking an out-degree 1 orientation of each of the a forests into which the edge set E decomposes).

List Coloring: In a *list coloring* instance of a graph $G = (V, E)$, every node $v \in V$ is initially assigned a list L_v of colors and the objective is to assign a color $x_v \in L_v$ such that the assigned colors form a proper vertex coloring of G. A graph G is said to be $c(v)$-list-colorable if any assignment of lists of size $|L_v| \geq c(v)$ for each $v \in V$ has a solution. Note that every graph is $(\deg(v)+1)$-list colorable and every graph of arboricity a is $2a$-list colorable. Some of our algorithms only compute partial colorings. For a given list coloring instance with lists L_v, a partial proper list coloring is an assignment of values $x_v \in \{\perp\} \cup L_v$ to each node such that for any two neighbors u and v, $x_u \neq x_v$ or $x_u = x_v = \perp$ (where an assignment of $\perp$ is interpreted as not assigning a color to a node).

Defective Colorings: For a graph $G = (V, E)$ and two integers $d \geq 0$ and $c \geq 1$, a *d-defective c-coloring* is an assignment of a color $x_v \in [c]$ to each node $v \in V$ such that for every $v \in V$, the number of neighbors $u \in N(v)$ with $x_u = x_v$ is at most d. We also use a version of defective coloring where the defect of a node v can be a function of the degree $\deg(v)$. For any $\lambda \in [0, 1]$ and an integer $c \geq 1$, we define a *λ-relative defective c-coloring* of G as an assignment of colors $x_v \in [c]$ such that each node $v \in V$ has at most $\lambda \cdot \deg(v)$ neighbors of the same color. We will make frequent use of the following distributed defective coloring result, which was proven in [Kuh09, KS18].

THEOREM 2.1. *[Kuh09, KS18] Let $G = (V, E)$ be a graph, $\lambda > 0$ a parameter, and assume that an $O(\mathrm{poly}(\Delta))$-coloring of G is given. Then, there is a deterministic $O(\log^* \Delta)$-time CONGEST algorithm to compute an λ-relative defective $O(1/\lambda^2)$-coloring of G.*

By using an algorithm of Linial [Lin92], it is possible to deterministically compute an $O(\Delta^2)$-coloring of G in $O(\log^* n)$ rounds in the CONGEST model. As all our time complexities are lower bounded by $\Omega(\log^* n)$, we will generally assume that an $O(\Delta^2)$-coloring of the network graph G is given. In [Kuh09], Theorem 2.1 was proven for the standard defective coloring definition. In [KS18], it was observed and formally proven that the algorithm and proof of [Kuh09] directly extends to the relative defective coloring problem defined above. In [BE10], Barenboim and Elkin defined a generalization of defective coloring that decomposes the

[5] If the IDs are from a possibly larger space $1, \ldots, N$, all results still directly hold with an additional additive time cost of $O(\log^* N)$ as long as we are allowed to also increase the maximum message size to $O(\log N)$ (in order to fit a single ID in a message). Further, with standard "exponential guessing" techniques, it should be possible to get rid of the knowledge of Δ and a entirely (see, e.g., [KSV11]).

graph into subgraphs of small arboricity (rather than into subgraphs of small degree). Formally, for integers $\beta \geq 0$ and $c \geq 1$, a *β-arbdefective c-coloring* of G is an assignment of colors $x_v \in [c]$ to the nodes in $v \in V$ and an orientation of the edges of G such that each node $v \in V$ has at most β out-neighbors of the same color.

List Color Space Reduction: Defective and arbdefective colorings are a natural way to decompose a given distributed coloring problem into multiple coloring problems on sparser (and thus easier to solve) instances. Over the last 10 years, the technique has been applied successfully in several distributed coloring algorithms [BE09, Kuh09, BE10, BE11, Bar15, FHK16]. In its most basic variant, the technique divides the color space into several disjoint subspaces and it then in parallel solves the resulting coloring problems for each of the smaller color spaces. The main technical contribution of this paper are generalizations of defective and arbdefective colorings that allow to reduce the color space for list colorings. For our main result, we need a generalization of a decomposition into parts of low arboricity, which is defined next. In Section 4, we will also introduce a similar unoriented notion of list color space reduction. An **oriented list color space reduction** is defined as follows.

Definition 2.1. *Let $G = (V, E)$ be a graph where all edges are oriented. For each $v \in V$, let $\beta(v)$ be the out-degree of v. Assume that we are given a list coloring instance on G with lists $L_v \subseteq \mathcal{C}$ for some (global) color space $\mathcal{C}$. Assume also that we are given a partition $\mathcal{C} = \mathcal{C}_1 \cup \dots \cup \mathcal{C}_p$ of the color space. Further, assume that each node $v \in V$ is assigned an index $i_v \in [p]$ and that we are given a new edge orientation π. For each $v \in V$, we define new lists $L'_v := L_v \cap \mathcal{C}_{i_v}$ and we let $\beta'(v)$ be the number of out-neighbors u for which $L'_u \cap L'_v \neq \emptyset$ (w.r.t. the new orientation π). The assignment i_v together with the edge orientation π is called an* oriented (η, γ)-list color space reduction *for $\eta > 1$ and $\gamma \geq 1$ if*

$$\forall i \in [p] \,:\, |\mathcal{C}_i| \leq \frac{1}{\eta} \cdot |\mathcal{C}| \;\; \text{and} \;\; \forall v \in V \,:\, \frac{|L'_v|}{\beta'(v)} \geq \frac{1}{\gamma} \cdot \frac{|L_v|}{\beta(v)}.$$

Hence, essentially, a list color space reduction allows to reduce the global color space by a factor η while losing a factor of γ in the list size to out-degree ratio. We will therefore try to find color space reductions where η is as large as possible and γ is as small as possible. When starting with a color space of size C and applying $t = O(\log_\eta C)$ (η, γ)-list color space reductions, this will allow to compute a proper coloring as long as the initial list size to degree ratio is larger than γ^t.

3 Distributed List Coloring in General Graphs

In this section, we provide our list coloring algorithms for general graphs. In Section 3.1, we describe the basic recursive algorithm that allows to prove Theorem 1.1. In Section 3.2, we show how to combine the scheme with the framework of [Bar15,FHK16] to obtain Theorem 1.2, our main result.

3.1 Basic Recursive Distributed List Coloring

Our scheme is a generalization of a coloring algorithm by Barenboim and Elkin in [BE10]. The algorithm of [BE10] is based on recursively decomposing a given graph G of arboricity a into $O(k)$ parts of arboricity a/k. While this allows to recursively divide the color space of a standard vertex coloring problem, it does not work for list coloring (it is not clear how to evenly divide the color space such that also the lists of all the nodes are divided in a similar way). We will show that by replacing the arbdefective coloring of [BE10] with an oriented list color reduction, a similar algorithm works and it can also be used in an analogous way recursively. The decomposition of the nodes in [BE10] is based on first computing a so-called H-partition [BE08], a partition of the nodes of G into $h = O(\log(n)/\varepsilon)$ sets $V_1, \dots, V_h$ such that for each $i \in [h]$ and every $v \in V_i$, v has at most $(2+\varepsilon)a$ neighbors in set V_j for $j \geq i$. We define a generalized H-partition as follows.

Definition 3.1. (Generalized H-Partition) *Let $G = (V, E)$ be a graph with a given edge orientation such that the out-degree of each node $v \in V$ is upper bounded by $\beta(v)$. A* generalized H-partition *of G with parameter α and of depth h is a partition of V into sets $V_1, \dots, V_h$ such that for every $i \in [h]$ and every $v \in V_i$, the number of neighbors of v in the set $\bigcup_{j=i}^{h} V_j$ is at most $\alpha \cdot \beta(v)$.*

The following lemma shows that every edge-oriented graph G has a generalized H-partition of logarithmic depth with a constant parameter and that such a partition can also be computed efficiently in the distributed setting. The construction is a natural adaptation of the standard H-partition construction of [BE08].

Lemma 3.1. *Let $G = (V, E)$ be a graph with a given edge orientation such that for each $v \in V$, $\beta(v)$ denotes the out-degree of v. For every $0 < \varepsilon = O(1)$, a generalized H-partition of G of depth $h = O(\log(n)/\varepsilon)$ and with parameter $2 + \varepsilon$ can be computed deterministically in $O(\log(n)/\varepsilon)$ rounds in the* CONGEST *model.*

Proof. We first define the construction of the partition $V_1, \dots, V_h$. In the following, we call the nodes V_i the nodes of level i of the construction. We iteratively

construct the sets level by level. For each $i \in \{0, \dots, h\}$, let $U_i := V \setminus \bigcup_{j=1}^{i} V_i$ be the set of nodes that are not in the first i levels and let $G_i := G[U_i]$ be the (oriented) subgraph of G induced by U_i. We define the set V_i to be the set of nodes $v \in U_{i-1}$ that have degree at most $(2+\varepsilon)\beta(v)$ in G_i. We stop the construction as soon as $U_h = \emptyset$. It is clear from the construction that each node $v \in V_i$ has at most $(2+\varepsilon)\beta(v)$ neighbors in sets $V_i \cup \dots \cup V_h$. It is also straightforward to see that a single level can be constructed in $O(1)$ rounds in the CONGEST model and the whole construction therefore requires $O(h)$ rounds. It therefore remains to show that the construction ends after constructing $O(\log(n)/\varepsilon)$ levels.

We consider one of the graphs $G_i = G[U_i]$ for $i \geq 0$ and for each $v \in U_i$, we define $\beta_{G_i}(v) \leq \beta(v)$ to be the out-degree of v in G_i. We define $W := \{u \in U_i : \deg_{G_i}(u) \leq (2+\varepsilon)\beta_{G_i}(u)\}$ and $\overline{W} := U_i \setminus W$. Note that $W \subseteq V_{i+1}$ (i.e., layer i contains at least all nodes in W). For a set $S \subseteq U_i$, we define $\mathrm{vol}(S) := \sum_{v \in S} \deg_{G_i}(v)$ and $\mathrm{out}(S) := \sum_{v \in S} \beta_{G_i}(v)$. We will show that

$$\mathrm{out}(W) > \varepsilon \cdot \mathrm{out}(\overline{W}). \tag{3.1}$$

?? implies that the set V_{i+1} contains at least an ε-fraction of all the out-degrees of G_i. The sum of the out-degrees in G_{i+1} is at most the sum of the out-degrees of the nodes $\overline{W}$ in G_i and thus **??** implies that with each constructed level, the sum of the out-degrees decreases by at least a factor $1-\varepsilon$. **??** therefore directly implies $h = O(\log(n)/\varepsilon)$ and it thus remains to prove **??**. We have

$$\mathrm{vol}(U_i) = \mathrm{vol}(W) + \mathrm{vol}(\overline{W}) = 2\mathrm{out}(W) + 2\mathrm{out}(\overline{W}).$$

By the definition of W, we further have

$$\mathrm{vol}(\overline{W}) > (2+\varepsilon)\mathrm{out}(\overline{W}).$$

Combining the two inequalities and the fact that for every $S \subseteq V$, $\mathrm{out}(S) \leq \mathrm{vol}(S)$, we have

$$\begin{aligned} \mathrm{out}(W) \leq \mathrm{vol}(W) &= 2\mathrm{out}(W) + 2\mathrm{out}(\overline{W}) - \mathrm{vol}(\overline{W}) \\ &< 2\mathrm{out}(W) - \varepsilon\mathrm{out}(\overline{W}), \end{aligned}$$

and thus **??** holds. ∎

Based on the generalized H-decomposition construction, we can now prove the main technical lemma for general graphs, which proves the existence of efficiently constructible oriented list color space reductions.

LEMMA 3.2. *Let $G = (V, E)$ be an edge-oriented graph, where the out-degree of each node $v \in V$ is denoted by $\beta(v)$. Let us further assume that each node $v \in V$ is assigned a list of colors L_v, where $L_v \subseteq \mathcal{C}$ for some color space $\mathcal{C}$ of size C. For every integer $\eta \in [1, C]$ and every $\varepsilon \in (0, 1]$, there is an oriented $(\eta, 2+\varepsilon)$-list color space reduction that can be computed deterministically in $O\big(\frac{\eta^2}{\varepsilon^3} \cdot \log n\big)$ rounds with messages of size $O(\log n + \log C)$.*

Proof. Let $\mathcal{C} = \mathcal{C}_1 \cup \dots \cup \mathcal{C}_p$ be an arbitrary partition of $\mathcal{C}$ into $p = O(|\mathcal{C}|/\eta)$ parts such that for all $x \in [p]$, $|\mathcal{C}_i| \leq |\mathcal{C}|/\eta$. In order to construct an oriented list color space reduction for the given partition of $\mathcal{C}$, we need to provide an orientation π of the edges and an assignment of values $x_v \in [p]$ to all nodes $v \in V$. We define $L'_v := L_v \cap \mathcal{C}_{x_v}$ as the new color list of v and $\beta'(v)$ to be the number of out-neighbors of v (w.r.t. the new orientation π) for which $x_u = x_v$. To show that we have computed an oriented $(\eta, 2+\varepsilon)$-list color space reduction, we need to show that

$$\beta'(v) \leq (2+\varepsilon) \cdot \frac{|L'_v|}{\alpha_v}, \quad \text{where } \alpha_v = \frac{|L_v|}{\beta(v)}. \tag{3.2}$$

Let $\delta := \varepsilon/2$. As a first step, we compute a generalized H-partition $V = V_1 \cup \dots \cup V_h$ with parameter $2+\delta$ and of depth $h = O(\log(n)/\delta) = O(\log(n)/\varepsilon)$. By using Lemma 3.1, this generalized H-partition can be computed in $O(\log(n)/\varepsilon)$ rounds in the CONGEST model. We then consider the subgraphs $G_i = G[V_i]$ induced by all the levels $i \in [h]$ of the generalized H-partition. Note that each node $v \in V$ has degree at most $(2+\delta)\beta(v)$ in its subgraph G_i. For each of the graphs G_i, we (in parallel) compute a (δ/p)-relative defective coloring. By applying Theorem 2.1, such a coloring with $O(p^2/\delta^2) = O(\eta^2/\varepsilon^2)$ colors can be computed in $O(\log^* n)$ rounds. Based on the generalized H-partition and the defective coloring of each of the levels, we now define a partial orientation π', which we will later extend to obtain the orientation π. Let $F \subseteq E$ be the set of edges $\{u, v\}$ of G such that either u and v are in different levels of the partition or such that if u and v are on the same level i, the defective coloring of G_i assigns u and v different colors. Orientation π' orients all edges in F and it leaves all edges in $E \setminus F$ unoriented. For $\{u, v\} \in F$, if the two nodes are in different levels V_i and V_j for $i < j$, the edge is oriented from V_i to V_j (i.e., from the smaller to the larger level). If u and v are on the same level but have different colors, the edge is oriented from the larger to the smaller color. Note that by the property of the generalized H-partition, every node $v \in V$ has at most $(2+\delta)\beta(v)$ neighbors u for which π' orients v to u. Further by the property of the defective colorings, every node $v \in V$ has at most $\delta/p \cdot \beta(v)$ neighbors to which the edge is not oriented by π'.

We are now ready to assign the color subspaces $\mathcal{C}_1, \dots, \mathcal{C}_p$ to the nodes, i.e., to assign $x_v \in [p]$ to each $v \in V$. We do this in h phases, each consisting of $O(p^2/\delta^2) = O(\eta^2/\varepsilon^2)$ rounds. In the phases, we iterate through the h levels of the partition in reverse order (i.e., we first process the nodes in V_h, then in V_{h-1}, and so on). In the phase when processing level $i \in [h]$, we iterate through the $O(p^2/\delta^2)$ colors of G_i in ascending order. All the nodes on the same level and of the same color are processed in parallel. Note by the construction of π', when processing a node $v \in V$, all out-neighbors u (w.r.t. π') have already been processed.

Let us now focus on the choice of x_v for a particular node $v \in V$. For each $x \in [p]$, we define $\ell_x := |L_v \cap \mathcal{C}_x|$ to be the new list size of v if we set $x_v = x$. We further define b_x as the number of neighbors u of v for which π' orients the edge from v to u and where $x_u = x$. In the end, we will define the orientation π of all edges of G by using orientation π' for all edges in F and by arbitrarily orienting all edges in $E \setminus F$. If we set $x_v = x$, we can therefore upper bound the final number $\beta'(v)$ of out-neighbors u of v (w.r.t. orientation π) for which $x_u = x_v$ by $\beta'(v) \leq b_x + \delta/p \cdot \beta(v)$. In order to guarantee **??**, we thus need to find a value x_v for which $b_{x_v} + \delta/p \cdot \beta(v) \leq (2+\varepsilon)\ell_{x_v}/\alpha_v$, where α_v is defined as in **??**. Such an x_v exists because

$$\sum_{x=1}^{p} \left(b_x + \frac{\delta}{p} \cdot \beta(v) \right) = \sum_{x=1}^{p} b_x + \delta\beta(v) \leq (2+\varepsilon)\beta(v) = \frac{2+\varepsilon}{\alpha_v} \sum_{x=1}^{p} \ell_x.$$

The first inequality follows because $\delta = \varepsilon/2$ and because $\sum_{x=1}^{p} b_i$ is the total number of out-neighbors of v w.r.t. orientation π' and it is thus upper bounded by $(2+\delta)\beta(v)$. The last equation follows because $\sum_{x=1}^{p} \ell_x = |L_v|$ and because $|L_v| = \alpha_v \beta(v)$. It is clear that for all the nodes that are processed in parallel (the nodes of the same color on the same level), the color subspace assignment can be done in a single communication rounds. The claim of the lemma therefore follows. ∎

Note that for $\eta = |\mathcal{C}|$ and sufficiently large lists, the above theorem directly solves the given list coloring instance. We remark that for this special case, the time complexity could be improved to $O(\eta/\varepsilon \cdot \log n)$ by using a more efficient algorithm to compute a proper coloring of each level of the generalized H-partition. We can now prove Theorem 1.1. The following is a restated, slightly more general version of the theorem.

THEOREM 3.1. *Let $G = (V, E)$ be an n-node graph where all edges are oriented and for each $v \in V$, let $\beta(v)$ be the out-degree of v. Let $\varepsilon \in (0, 1]$ and $r \geq 1$ be two parameters and assume that each node v is assigned a list L_v consisting of colors from some color space of size C. Then, there is a deterministic distributed $O\big(\frac{r \cdot C^{2/r}}{\varepsilon^3} \cdot \log n\big)$-round algorithm that computes a proper partial coloring for the given list coloring instance such that every node with $\beta(v) < (2+\varepsilon)^r \cdot |L_v|$ outputs a color. The algorithm uses messages of size $O(\log n + \log C)$.*

Proof. We start with the original list coloring instance and we apply Lemma 3.2 r times to obtain list coloring instances with smaller and smaller color spaces. Each time, we divide the given color space into several disjoint subspaces. For each subspace, we obtain an independent list coloring problem and those independent problems can be then be solved in parallel. For $i \in \{0, \dots, r\}$, let C_i be the maximum size of the color spaces after i list color space reductions (note that $C_0 = C$). For each node, let $L_{i,v}$ be the color list of v in its list coloring instance after i steps and let $\beta_i(v)$ be v's out-degree of the corresponding edge orientation after i steps. We have $L_{0,v} = L_v$ and $\beta_0(v) = \beta(v)$. We set $\eta := \lceil C^{1/r} \rceil$ and we get the list coloring instances after i steps by applying oriented $\big(\min\{\eta, |C_{i-1}|\}, 2+\varepsilon\big)$-list color reductions to the list coloring instances after $i-1$ steps. By induction on i, for each $i \geq 1$, we thus have $C_i \leq \max\{1, C/\eta^i\}$ and $L_{i,v}/\beta_i(v) \geq L_v/\beta(v) \cdot (2+\varepsilon)^{-i}$. After r iterations, we therefore have $C_r = 1$ and therefore all the lists $L_{r,v}$ are also of size 1. That is, each node $v \in V$ has picked exactly one color c_v from its original list L_v. If $L_v/\beta(v) > (2+\varepsilon)^r$, we have $L_{r,v}/\beta_r(v) > 1$ and thus $\beta_r(v) = 0$. Nodes for which $L_v/\beta(v) > (2+\varepsilon)^r$ therefore have no out-neighbor with the same color and if only those nodes keep their colors, no two neighbors get the same color. The total time complexity follows directly from Lemma 3.2 and the fact that we have r iterations. ∎

3.2 Solving Degree + 1 List Coloring

The coloring result of Theorem 1.1 gives some trade-off between the list sizes (or number of colors for standard coloring) and the time complexity. The time complexity depends linearly on $C^{2/r}$ and the list coloring algorithm is thus only efficient if the value of r is sufficiently large. However, the required list size to degree ratio grows exponentially with r and for large r, we therefore need large lists. In order to reduce the number of colors, we use a framework that has been introduced by Barenboim in [Bar15] and later refined by Fraigniaud, Kosowski, and Heinrich in [FHK16].

The basic idea is as follows. In order to increase the list size to degree ratio, we first compute some defective (or arbdefective) coloring to decompose the graph into graphs of smaller maximum (out-)degree. Let us say we decompose the graph into K parts, which reduces the maximum degree by a factor $f(K)$ (where $f(K)$ is at

most linear in K). Unlike in the previous algorithm, we do not divide the lists among the different parts, but we use the same lists for coloring all the parts. If the maximum degree is reduced by a factor of $f(K)$ and we can use the same color lists, the list size to degree ratio improves by a factor $f(K)$ and thus if K is chosen sufficiently large, we can apply Theorem 1.1. However, since now the the different parts are dependent, we have to solve them consecutively and thus the time complexity will grow linearly with the number of parts K. The main technical result is given by the following lemma. In order to also be applicable in Section 4, the lemma is stated more generally than what we need in this section.

LEMMA 3.3. *Let S and T be two functions from the set of undirected graphs to the positive reals such that for every graph G and every subgraph H of G, we have $S(H) \leq S(G)$ and $T(H) \leq T(G)$. Assume that we are given a distributed algorithm $\mathcal{A}$ with the following properties. For list coloring instances in graphs $G = (V, E)$ with maximum degree Δ, $\mathcal{A}$ assigns a color $c_v \in L_v$ to each node $v \in V$ of degree $\deg(v) \geq \Delta/2$ and with a list L_v of size $|L_v| > \deg(v) \cdot S(G)$. Assume further that the running time of $\mathcal{A}$ is at most $T(G)$ if the lists contain colors from a color space of size at most Δ^c for an arbitary, but fixed constant $c > 0$. Then, if an $O(\Delta^2)$-coloring of G is given, any $(\deg(v)+1)$-list coloring instance with colors from a color space of size at most $\mathrm{poly}(\Delta)$ can be solved in time $O\big((S(G)^2 \cdot T(G) + \log^* \Delta) \cdot \log \Delta\big)$. The resulting algorithm is deterministic if $\mathcal{A}$ is deterministic and it works in the* CONGEST *model if $\mathcal{A}$ works in the* CONGEST *model.*

Proof. By using the same basic idea as in [FHK16], we show that in time $O(S(G)^2 \cdot T(G) + \log^* \Delta)$, we can reduce the problem to a $(\deg(v)+1)$-list coloring instance on a graph of maximum degree at most $\Delta/2$. The lemma then follows by repeating $O(\log \Delta)$ times. More formally, we show that given an arbitrary $(\deg(v)+1)$-list coloring instance with lists L_v in a graph $G = (V, E)$ of maximum degree at most Δ, we can compute a partial coloring with the following properties in time $O(S(G)^2 \cdot T(G) + \log^* \Delta)$:

(1) Each node $v \in V$ either outputs a color $c_v \in L_v$ or it outputs $c_v = \perp$.

(2) For any two neighbors u and v, either $c_u \neq c_v$ or $c_u = c_v = \perp$.

(3) The subgraph of G induced by the nodes $v \in V$ with $c_v = \perp$ has maximum degree at most $\Delta/2$.

Note that because the computed coloring is a proper partial coloring (property (2)), the coloring can be extended to a proper coloring of all nodes by solving a $(\deg(v)+1)$-list coloring problem on the subgraph H induced by the nodes v for which $c_v = \perp$. By property (3), this subgraph H has maximum degree at most $\Delta/2$. Note further that by the properties of S an T, we have $S(H) \leq S(G)$ and $T(H) \leq T(G)$ and by repeating $O(\log \Delta)$ times, we thus get a proper coloring of G in time $O\big((S(G)^2 \cdot T(G) + \log^* \Delta) \cdot \log \Delta\big)$ and the claim of the lemma follows.

In order to construct a partial list coloring algorithm that satisfies properties (1)–(3), we first set $\varepsilon = 1/S(G)$ and we use Theorem 2.1 to compute an $(\varepsilon/2)$-relative defective q-coloring of G, where $q = O(1/\varepsilon^2) = O(S(G)^2)$. Since we assume that an initial $O(\Delta^2)$-coloring of G is given, the defective coloring can be deterministically computed in time $O(\log^* \Delta)$ in the CONGEST model. For each $x \in [q]$, let $V_x \subseteq V$ be the set of nodes with color x in the defective coloring and let $G_x := G[V_x]$ be the subgraph of G induced by the nodes with color x. We compute the partial coloring c_v for G with properties (1)–(3) as follows.

We iterate through the q colors of the defective coloring. Initially, we set $c_v = \perp$ for all $v \in V$. When processing color $x \in [q]$, we consider the graph G_x and we assign colors $c_v \neq \perp$ to some nodes $v \in V_x$. For the assignment of colors in G_x, we consider the following list coloring problem. For each node $v \in V_x$, we define the list L'_v as the set of colors from L_v that have not yet been assigned to a neighbor (in a graph $G_{x'}$ for $x' < x$). We use algorithm $\mathcal{A}$ to compute a partial proper coloring of G_x w.r.t. the lists L'_v for $v \in V_x$. We then set the colors c_v accordingly for all $v \in V_x$ to which the partial coloring of G_x assigns a color. The choice of lists L'_v guarantees that the partial coloring computed for G will be proper and thus property (2) is satisfied. Note that if a node $v \in V_x$ has at least $\Delta/2$ uncolored neighbors in G before coloring G_x, the list L'_v is of size $|L'_v| > \Delta/2$ (because we are given a $(\deg(v)+1)$-list coloring instance). As the degree of v in G_x is at most

$$\deg_{G_x}(v) \leq \frac{\varepsilon}{2} \cdot \deg(v) = \frac{1}{S(G)} \cdot \frac{\deg(v)}{2} \leq \frac{1}{S(G)} \cdot \frac{\Delta}{2} < \frac{1}{S(G)} \cdot |L'_v|,$$

algorithm $\mathcal{A}$ is guaranteed to assign a color to node v. This proves property (3) of our partial list coloring algorithm for G. Since the total number of colors q of the defective coloring is $O(S(G)^2)$, we only need to invoke $\mathcal{A}$ $O(S(G)^2)$ times and thus also the required time complexity of $\mathcal{A}$ follows. The proof is completed by observing that the reduction is deterministic and can clearly be implemented with messages of size $O(\log n)$. ∎

We can now prove our main result Theorem 1.2, which shows that any $(\deg(v)+1)$-list coloring instance

can be solved deterministically in $2^{O(\sqrt{\log \Delta})} \cdot \log n$ rounds in the CONGEST model.

Proof. [**Proof of Theorem 1.2**] **We set** $S(G) = 2^{c\sqrt{\log \Delta}}$ **and** $T(G) = 2^{c'\sqrt{\log \Delta}} \cdot \log n$ **for appropriate constants** $c, c' > 0$**. The claim follows directly from Lemma 3.3 if we can show that there is a deterministic** $T(G)$**-round CONGEST model algorithm** $\mathcal{A}$ **with the following properties. In a graph of maximum degree** Δ**,** $\mathcal{A}$ **computes a proper partial list coloring that assigns a color to each node** v **of degree** $\Delta/2$ **and for which there is a list** L_v **of size** $|L_v| \geq \deg(v) \cdot S(G)$**. The existence of** $\mathcal{A}$ **follows directly from Theorem 3.1 by setting** $\varepsilon = 1$ **and** $r = \Theta(\sqrt{\log \Delta})$**.** ∎

4 Bounded Neighborhood Independence

In this section, we give a more efficient algorithm for the $(\deg(v)+1)$-list coloring problem in graphs of bounded neighborhood independence. In the following, we assume that we are given a graph $G = (V, E)$, where the neighborhood independence is bounded by a constant $\theta \geq 1$. Our algorithm is based on the fact that for graphs with bounded neighborhood independence, there are more efficient defective coloring algorithms [BE11] and that defective colorings of such graphs also have stronger properties, as given by the following lemma.

LEMMA 4.1. *Let $\theta \geq 1$ and let $G = (V, E)$ be a graph with neighborhood independence at most θ. Assume that we are given a d-defective coloring of the nodes of G. Then, for every color x of the given defective coloring and every node $v \in V$, the number of neighbors $u \in N(v)$ with color x is at most $\theta(d+1)$.*

Proof. Let $S_x \subseteq N(v)$ be the nodes of $N(v)$ that have color x. Because the given coloring has defect d, the subgraph $G[S_x]$ induced by S_x has degree at most d and can thus be colored with $d+1$ colors. The set S_x can thus be partitioned into at most $d+1$ independent sets of G. Because the neighborhood independence of G is at most θ, $N(v)$ can only contain independent sets of G of size at most θ and we thus have $|S_x| \leq \theta(d+1)$. ∎

COROLLARY 4.1. *Let $\theta \geq 1$ and $\varepsilon \in (0, 1]$ and let $G = (V, E)$ be a graph with neighborhood independence at most θ. If an $O(\Delta^2)$-coloring of G is given, there is a deterministic $O(\log^* \Delta)$-round CONGEST algorithm to compute a coloring of the nodes of G with $O(1/\varepsilon^2)$ colors such that for each $v \in V$ and each of the colors x, there are at most $\max\{\theta, \varepsilon \deg(v)\}$ nodes of color x in $N(v)$.*

Proof. The corollary follows directly from Lemma 4.1 by applying Theorem 2.1 with parameter $\lambda = \max\left\{\frac{1}{2\Delta}, \varepsilon - \frac{1}{\Delta}\right\} \geq \frac{\varepsilon}{3}$. ∎

In [BE11], it was shown that in graphs with bounded neighborhood independence, it is possible to efficiently compute a d-defective $O(\Delta/d)$-coloring. This is then used to decompose a high degree coloring instance into several independent low-degree instances while only losing a constant factor in the total number of colors. In the following, we combine ideas from [BE11] with Corollary 4.1 to obtain such a recursive coloring algorithm for computing list colorings of graphs with bounded neighborhood independence. In this case, we need an unoriented variant of the list color space reduction of Definition 2.1. To make our algorithm work, we have to define a weaker version of list color space reduction, where only nodes of sufficiently large degrees are assigned a new color space.

DEFINITION 4.1. (WEAK LIST COLOR SPACE REDUCTION) *Let $G = (V, E)$ be a graph and assume that we are given a list coloring instance on G with lists $L_v \subseteq \mathcal{C}$ for some color space $\mathcal{C}$ and all nodes $v \in V$. Assume further that we are given a partition $\mathcal{C} = \mathcal{C}_1 \cup \dots \cup \mathcal{C}_p$ of the color space.*

Consider an assignment i_v that assigns each node $v \in V$ either a color subspace $i_v \in [p]$ or a default value $i_v = \perp$. For each $v \in V$ with $i_v \neq \perp$, we define a new list $L'_v := L_v \cap \mathcal{C}_{i_v}$ and we define $\deg'(v)$ to be the number of neighboring nodes u for which $i_u = i_v$.

The assignment of the values i_v is called an weak (η, γ, D)-list color space reduction *for $\eta > 1$, $\gamma \geq 1$, and $D \geq 0$ if $i_v \neq \perp$ for all v for which $\deg(v) > D$ and*

$$\forall i \in [p] : |\mathcal{C}_i| \leq \frac{1}{\eta} \cdot |\mathcal{C}| \quad \text{and} \quad \forall v \in V : i_v \neq \perp \rightarrow \frac{|L'_v|}{\deg'(v)} \geq \frac{1}{\gamma} \cdot \frac{|L_v|}{\deg(v}$$

We next describe the algorithm for obtaining such a weak list color space reduction for graphs with bounded neighborhood independence. Assume that we are given a graph $G = (V, E)$ with maximum degree Δ. Assume further that we are given an $O(\Delta^2)$-coloring of G and that each node $v \in V$ is given a list L_v of colors of some color space $\mathcal{C}$. The algorithm has a parameter $\eta \geq 1$ that specifies the factor by which we reduce the color space. The details of the algorithm are described in the following.

Weak List Color Space Reduction Algorithm:

(1) Let $\mathcal{C} = \mathcal{C}_1 \cup \dots \cup \mathcal{C}_p$ be an arbitrary partition of the color space $\mathcal{C}$ such that $p \leq 2\eta$ and $|\mathcal{C}_i| \leq |\mathcal{C}|/\eta$ for all $i \in [p]$. Note that such a partition is always possible.

(2) At the end of the algorithm, each node $v \in V$ outputs an index $i_v \in \{\perp\} \cup [p]$ indicating which color subspace v chooses (if $i_v \neq \perp$). Assume that initially, i_v is set to $\perp$ for all $v \in V$

(3) Use the algorithm of Corollary 4.1 with parameter $\varepsilon := 1/p$ to compute a defective coloring of G with $q = O(p^2)$ colors in time $O(\log^* \Delta)$.

(4) Iterate through phases $\phi = 0, 1, 2, \ldots, O(\log \Delta)$. Define $\delta_0 := \Delta$ and $\delta_\phi := \lceil \delta_{\phi-1}/2 \rceil$ for $\phi \geq 1$. Stop after phase ϕ, where $\delta_\phi = 1$.

(5) In each phase ϕ, iterate through the q colors x of the defective coloring. When processing color x, all nodes $v \in V$ of color x do the following (in parallel):

If $i_v = \perp$ (i.e., if i_v is not yet set), $\deg(v) > \theta p$, and if there exists an $i^* \in [p]$ such that v has at most δ_ϕ neighbors u with $i_u = i^*$ and $\delta_\phi + \frac{\deg(v)}{p} \leq 3 \cdot |L_v \cap \mathcal{C}_{i^*}| \cdot \frac{\deg(v)}{|L_v|}$, set $i_v := i^*$.

LEMMA 4.2. *Let $G = (V, E)$ be a graph with maximum degree at most Δ and neighborhood independence at most $\theta \geq 1$. Assume that we are given a list coloring instance with lists $L_v \subseteq \mathcal{C}$ for each $v \in V$. For a parameter $\eta > 1$, the above algorithm computes a weak $(\eta, 3\theta, 2\theta\eta)$-list color space reduction in $O(\eta^2 \log \Delta)$ rounds deterministically in the* CONGEST *model.*

Proof. We first show that the algorithm assigns an color subspace $i_v \in [p]$ to each node $v \in V$ with $\deg(v) > \theta p$. For the sake of contradiction, assume that there is a node $v \in V$ with $\deg(v) > \theta p$ to which the algorithm does not assign a color subspace $i_v \in [p]$. Node v in considered in each phase ϕ and it chooses a color subspace in the first phase ϕ in which there is an $i^* \in [p]$ such that v has at most δ_ϕ neighbors u with $i_u = i^*$ and for which

$$\delta_\phi + \frac{\deg(v)}{p} \leq 3 \cdot \frac{\deg(v)}{|L_v|} \cdot |L_v \cap \mathcal{C}_{i^*}|. \tag{4.3}$$

If v does not pick a color space, this condition is not satisfied for any phase ϕ. To prove a contradiction, we consider the assignment to the neighbors $u \in N(v)$ of v at the end of the algorithm. For each $i \in [p]$, let d_i be the number of node $u \in N(v)$ for which $i_u = i$ at the end of the algorithm. Note that clearly $\sum_{i=1}^{p} d_i \leq |N(v)| = \deg(v)$. Further, for each $i \in [p]$, we define $\ell_i := |L_v \cap \mathcal{C}_i|$ to be the new list size of v if v sets $i_v = i$. We have

$$\sum_{i=1}^{p} \left(2d_i + \frac{\deg(v)}{p} \right) \leq 3 \deg(v) = 3 \cdot \frac{\deg(v)}{|L_v|} \cdot |L_v| = 3 \cdot \frac{\deg(v)}{|L_v|} \cdot \sum_{i=1}^{p} \ell_i.$$

Therefore, there exists an i^* for which $2d_{i^*} + \frac{\deg(v)}{p} \leq 3\frac{\deg(v)}{|L_v|} \cdot \ell_{i^*}$. Let ϕ^* be the last phase for which $d_{i^*} \leq \delta_{\phi^*}$. We have $\delta_{\phi^*} \leq 2d_{i^*}$ and therefore $\delta_{\phi^*} + \frac{\deg(v)}{p} \leq 3 \cdot \frac{\deg(v)}{|L_v|} \cdot \ell_{i^*}$. Hence, if i_v is still set to $\perp$ in phase ϕ^* when processing node v, the algorithm can set $i_v = i^*$ in phase ϕ^* (note that the number of neighbors $u \in N(v)$ with $i_u = i^*$ in phase ϕ^* is at most $d_i \leq \delta_{\phi^*}$). This is a contradiction to the assumption that v does not set its i_v to a value in $[p]$ and we have thus shown that at the end of the algorithm, all nodes v with $\deg(v) > \theta p$ have $i_v \in [p]$.

For the remainder of the proof, consider some node v for which the algorithm sets $i_v = i^*$. Note that we have $\deg(v) > \theta p$ as the algorithm would otherwise not set the value $i_v \neq \perp$. Assume that node v sets its value i_v to i^* in phase ϕ^*. Let $S(v)$ be the set of neighbors $u \in N(v)$ of v for which at the end of the algorithm $i_u = i_v = i^*$. We next bound the size of $S(v)$ as

$$|S(v)| \leq \theta \cdot \left(\delta_{\phi^*} + \frac{\deg(v)}{p} \right). \tag{4.4}$$

In order to prove **??**, we consider the subgraph $G[S(v)]$ of G induced by the nodes in $S(v)$ and we show that $G[S(v)]$ can be colored with a small number of colors. Together with the neighborhood independence of G, this implies that $S(v)$ is not too large. To bound the chromatic number of $G[S(v)]$, we compute a low outdegree edge orientation of $G[S(v)]$. Consider two nodes $u, w \in S(v)$ such that $\{u, w\} \in E$. If u and w set $i_u = i_w = i^*$ at the same time, we orient the edge $\{u, w\}$ from the node with the lower ID to the node with the larger ID. Otherwise, we orient the edge from the node that sets its color subspace to $\mathcal{C}_{i^*}$ later to the node that set its color subspace to $\mathcal{C}_{i^*}$ first. Note that the resulting orientation is clearly acyclic.

To bound the outdegree of a node $u \in S(v)$, let us first bound the number of neighbors w of u in $G[S(v)]$ that set their color space to $\mathcal{C}_{i^*}$ before u. Before node v sets $i_v = i^*$, the number of nodes in $S(v)$ is at most d_{ϕ^*} and therefore for all nodes u in $S(v)$ that choose the color subspace before v, the number of nodes $w \in S(v)$ that choose the color subspace before u is bounded by d_{ϕ^*}. Every node $u \in S(v)$ that sets $i_u = i^*$ together with v or after v does this in a phase $\phi \geq \phi^*$. Hence, for such nodes u, when u sets i_u to i^*, it has at most $d_\phi \leq d_{\phi^*}$ neighbors w with $i_w = i^*$. For every node $u \in S(v)$, the number of neighbors $w \in G[S(v)]$ that set $i_w = i^*$ before u sets $i_u = i^*$ is therefore upper bounded by d_{ϕ^*}.

We next bound the number of neighbors w in $G[S(v)]$ of a node $u \in S(v)$ that choose their color subspace at the same time as u. Note that only

nodes that have the same color in the defective coloring computed in step (3) of the algorithm can choose their color subspace together. Note that by Corollary 4.1 (and the choice of $\varepsilon = 1/p$), for each of the colors x of the defective coloring, the number of nodes $u \in N(v)$ with color x is at most $\max\{\theta, \deg(v)/p\} \leq \deg(v)/p$ (because $\deg(v) > \theta p$). For each u in $S(v)$, the number of nodes $w \in S(v) \setminus \{u\}$ that choose their color subspace together with u is therefore upper bounded by $\deg(v)/p - 1$. Consequently, the computed acyclic edge orientation of $G[S(v)]$ has out-degree at most $d_{\phi^*} + \deg(v)/p - 1$. This implies that $G[S(v)]$ has chromatic number at most $d_{\phi^*} + \deg(v)/p$ and thus that the nodes in $S(v)$ can be partitioned into $k \leq d_{\phi^*} + \deg(v)/p$ independent sets (of G). Because the neighborhood independence of G is bounded by θ, each of these independent sets is of size at most θ and **??** therefore follows. It now follows directly by combining **??** with **??** that the algorithm computes a weak $(\eta, 3\theta, 2\theta\eta)$-list color space reduction as claimed. The algorithm is deterministic and it can clearly be implemented in the CONGEST model. The time complexity of the algorithm is dominated by iterating through the $O(\log \Delta)$ phases and the $O(p^2) = O(\eta^2)$ colors of the defective coloring in each phase. ∎

LEMMA 4.3. *Let $G = (V, E)$ be a graph with maximum degree Δ and neighborhood independence at most $\theta \geq 1$. Assume that each node $v \in V$ is assigned a list $L_v \subseteq \mathcal{C}$ of colors from some color space $\mathcal{C}$ of size $|\mathcal{C}| = C$. Assume further that an $O(\Delta^2)$-coloring of G is given. Then, there is a deterministic distributed $O\big(\theta \cdot r \cdot C^{2/r} \log \Delta\big)$-round algorithm that computes a proper partial list coloring such that each node v with a list of size $|L_v| > (3\theta)^{r-1} \cdot \deg(v)$ is assigned a color.*

Proof. We first show that for any graph $G = (V, E)$ and an arbitrary parameter d, it is possible to compute a partial proper list coloring of the nodes $v \in V$ with $\deg(v) \leq d$ in time $O(d + \log^* \Delta)$. The partial list coloring assigns a color $v \in L_v$ to all such nodes with $|L_v| > \deg(v)$. Let V_d be the set of nodes with $\deg(v) \leq d$ and let $G[V_d]$ be the subgraph of G induced by the nodes in V_d. In the graph $G[V_d]$ every node has clearly degree at most d and by (for example) using the algorithm of [BE09, Kuh09] and the initial $O(\Delta^2)$-coloring of G (and thus also of $G[V_d]$), we can therefore compute a $(d+1)$-coloring of $G[V_d]$ in time $O(d + \log^* \Delta)$. We can then iterate through the $d + 1$ colors of this coloring and assign list colors greedily. When doing this, every node v with $|L_v| > \deg(v)$ definitely succeeds in getting a color.

We can now prove the lemma by induction on r. Let us first consider the case $r \leq 2$ as the base of the induction. In this case, we can spend time $O(C + \log^* \Delta)$ and since any node v with $\deg(v) > |L_v|$ clearly has $\deg(v) < C$, for $r \leq 2$, the claim of the lemma directly follow from the above $O(d + \log^* \Delta)$-round algorithm.

To prove the induction step, let us therefore consider some $r > 2$ and assume that the claim of the lemma holds for smaller r. We define $\eta := C^{1/r}$ and we partition the set of nodes V into low-degree nodes V_L and high-degree nodes V_H. We let V_L be the set of node $v \in V$ for which $\deg(v) \leq 2\theta\eta$ and we accordingly define $V_H := V \setminus V_L$. We apply the above weak list color space reduction algorithm to G with parameter η. The algorithm partitions the color space $\mathcal{C}$ into subspaces $\mathcal{C}_1, \ldots, \mathcal{C}_p$ of size at most $C^{(r-1)/r}$. Because the nodes in V_H have degree more than $2\theta\eta$, by Lemma 4.2, all nodes $v \in V_H$ are assigned a subspace $\mathcal{C}_{i_v}$ (for $i_v \in [p]$). Let $L'_v = L_v \cap \mathcal{C}_{i_v}$ be the new list of a node $v \in V_H$ and let $\deg'(v)$ be the degree of v in the subgraph of G induced by all the nodes in V_H that were assigned to the same color subspace $\mathcal{C}_{i_v}$. By Lemma 4.2, we have $|L'_v|/\deg'(v) \geq |L_v|/(3\theta \deg(v))$. Hence, all nodes $v \in V_H$ for which $|L_v| > (3\theta)^{r-1} \cdot \deg(v)$, we have $|L'_v| > (3\theta)^{r-2} \cdot \deg'(v)$. For each of the color subspaces $\mathcal{C}_i$, we can therefore apply the induction hypothesis to list color the subgraphs induced by the nodes $v \in V_H$ that chose $i_v = i$. The list coloring problem on each of these subgraphs has a color space of size at most $C^{(r-1)/r}$. Hence by applying the induction hypothesis for $r - 1$, we can list color each of those subgraphs in time $O((r-1)C^{2/r} \log \Delta)$. All nodes $v \in V_H$ with $|L'_v| > (3\theta)^{r-2} \deg'(v)$ and thus in particular all nodes $v \in V_H$ with $|L_v| > (3\theta)^{r-1} \deg(v)$ are assigned a color.

It therefore just remains to color the nodes $v \in V_S$ for which $|L_v| > (3\theta)^{r-1} \deg(v)$ and thus in particular $|L_v| > \deg(v)$. Note that by definition of V_S, the graph induced by the nodes in V_S has degree at most $2\theta\eta$. Further, for each node $v \in V_S$ with $|L_v| > \deg(v)$, also for the remaining list coloring problem on $G[V_S]$, the list size of v exceeds its degree (for each already colored neighbor in V_H, the degree of v drops by 1 and the list size of v drops by at most 1). Hence, the remaining problem on $G[V_S]$ can be solved in time $O(\theta\eta + \log^* \Delta) = O(\theta C^{1/r} + \log^* \Delta)$ by using the $O(d + \log^* \Delta)$ algorithm described in the first paragraph of this proof. Thus the claim of the lemma follows. ∎

Based on Lemma 4.3, we can now prove Theorem 1.3 the main theorem of this section. The theorem is almost an immediate consequence of Lemmas 3.3 and 4.3.

Proof. [**Proof of Theorem 1.3**] **We start by computing an $O(\Delta^2)$-coloring of G. By using the algorithm of [Lin92], this can be done in time**

$O(\log^* n)$. **Based on this coloring, we can now apply Lemma 4.3. When setting $r = \Theta(\sqrt{\log\theta \cdot \log\Delta})$ and using that $C \leq \text{poly}(\Delta)$, the lemma implies that there is an $O(r\theta C^{2/r}\log\Delta) = 2^{O(\sqrt{\log\theta\cdot\log\Delta})}$-time algorithm that computes partial proper list coloring such that all nodes $v \in V$ with $|L_v| > (3\theta)^{r-1}\cdot\deg(v) = 2^{\Theta(\sqrt{\log\theta\cdot\log\Delta})}\cdot\deg(v)$ are assigned a color. The algorithm thus satisfies the requirements of Lemma 3.3 for $S(G) = T(G) = 2^{O(\sqrt{\log\theta\cdot\log\Delta})}$ and for graphs with neighborhood independence at most θ, we get a $2^{O(\sqrt{\log\theta\cdot\log\Delta})}$-time deterministic distributed algorithm to solve an arbitrary list coloring instance with lists of size $|L_v| > \deg(v)$ for all $v \in V$ and colors from a range of size $\text{poly}(\Delta)$.** □

5 Acknowledgements

I am grateful to various colleagues for several discussions on distributed coloring algorithms that helped shaping my understanding of the problem. I am particularly thankful to Mohsen Ghaffari, Yannic Maus, and Jara Uitto for discussions on the relevance of finding better list coloring algorithms even for relatively large list sizes (such as the ones considered in Section 3.1).

References

[ABI86] N. Alon, L. Babai, and A. Itai. A Fast and Simple Randomized Parallel Algorithm for the Maximal Independent Set Problem. *J. of Algorithms*, 7(4):567–583, 1986.

[AGLP89] B. Awerbuch, A. V. Goldberg, M. Luby, and S. A. Plotkin. Network Decomposition and Locality in Distributed Computation. In *Proc. 30th Symp. on Found. of Computer Science (FOCS)*, pages 364–369, 1989.

[Bar15] L. Barenboim. Deterministic (Δ + 1)-Coloring in Sublinear (in Δ) Time in Static, Dynamic and Faulty Networks. In *Proc. 34th Symp. on Principles of Distributed Computing (PODC)*, pages 345–354, 2015.

[BE08] L. Barenboim and M. Elkin. Sublogarithmic Distributed MIS Algorithm for Sparse Graphs using Nash-Williams Decomposition. In *Proc. 27th ACM Symp. on Principles of Distributed Computing (PODC)*, pages 25–34, 2008.

[BE09] L. Barenboim and M. Elkin. Distributed (Delta+1)-coloring in linear (in Delta) time. In *Proc. 41st ACM Symp. on Theory of Computing (STOC)*, pages 111–120, 2009.

[BE10] L. Barenboim and M. Elkin. Deterministic Distributed Vertex Coloring in Polylogarithmic Time. In *Proc. 29th Symp. on Principles of Distributed Computing (PODC)*, 2010.

[BE11] L. Barenboim and M. Elkin. Distributed deterministic edge coloring using bounded neighborhood independence. In *Proc. 30th ACM Symp. on Principles of Distributed Computing (PODC)*, pages 129–138, 2011.

[BE13] L. Barenboim and M. Elkin. *Distributed Graph Coloring: Fundamentals and Recent Developments*. Morgan & Claypool Publishers, 2013.

[BEG18] L. Barenboim, M. Elkin, and U. Goldenberg. Locally-Iterative Distributed (Δ + 1)-Coloring below Szegedy-Vishwanathan Barrier, and Applications to Self-Stabilization and to Restricted-Bandwidth Models. In *Proc. 37th ACM Symp. on Principles of Distributed Computing (PODC)*, pages 436–446, 2018.

[BEPS12] L. Barenboim, M. Elkin, S. Pettie, and J. Schneider. The Locality of Distributed Symmetry Breaking. In *Proc. 53th Symp. on Foundations of Computer Science (FOCS)*, 2012.

[Bro09] R. L. Brooks. On colouring the nodes of a network. In *Classic papers in combinatorics*, pages 118–121. Springer, 2009.

[CHK01] A. Czygrinow, M. Hańćkowiak, and M. Karoński. Distributed O(Δ log n)-Edge-Coloring Algorithm. In *Proc. 9th Annual European Symposium on Algorithms (ESA)*, pages 345–355, 2001.

[CHL+18] Y. Chang, Q. He, W. Li, S. Pettie, and J. Uitto. The Complexity of Distributed Edge Coloring with Small Palettes. In *Proc. 29th ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 2633–2652, 2018.

[CKP16] Y.-J. Chang, T. Kopelowitz, and S. Pettie. An Exponential Separation Between Randomized and Deterministic Complexity in the LOCAL Model. In *Proc. 47th Symp. on Foundations of Computer Science (FOCS)*, pages 615–624, 2016.

[CLP18] Y.-J. Chang, W. Li, and S. Pettie. An Optimal Distributed (Δ+1)-Coloring Algorithm? In *Proc. 50th ACM Symp. on Theory of Computing (STOC)*, pages 445–456, 2018.

[CV86] R. Cole and U. Vishkin. Deterministic Coin Tossing with Applications to Optimal Parallel List Ranking. *Information and Control*, 70(1):32–53, 1986.

[DGP98] D. Dubhashi, D. A. Grable, and A. Panconesi. Near-optimal, distributed edge colouring via the nibble method. *Theoretical Computer Science*, 203(2):225–251, 1998.

[EPS15] M. Elkin, S. Pettie, and H. Su. (2Δ - l)-Edge-Coloring is Much Easier than Maximal Matching in the Distributed Setting. In *Proc. 26th ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 355–370, 2015.

[FGK17] M. Fischer, M. Ghaffari, and F. Kuhn. Deterministic Distributed Edge-Coloring via Hypergraph Maximal Matching. In *Proc. 58th IEEE Symp. on Foundations of Computer Science (FOCS)*, 2017.

[FHK16] P. Fraigniaud, M. Heinrich, and A. Kosowski. Local Conflict Coloring. In *Proc. 57th Symp. on Foundations of Computer Science (FOCS)*, 2016.

[GHK18] M. Ghaffari, D. G. Harris, and F. Kuhn. On derandomizing local distributed algorithms. In *Proc. 59th Symp. on Foundations of Computer Science (FOCS)*, pages 662–673, 2018.

[GHKM18] M. Ghaffari, J. Hirvonen, F. Kuhn, and

Y. Maus. Improved Distributed Delta-Coloring. In *Proc. 37th ACM Symp. on Principles of Distributed Computing (PODC)*, pages 427–436, 2018.

[GKM17] M. Ghaffari, F. Kuhn, and Y. Maus. On the Complexity of Local Distributed Graph Problems. In *Proc. 49th ACM Symp. on Theory of Computing (STOC)*, pages 784–797, 2017.

[GKMU18] M. Ghaffari, F. Kuhn, Y. Maus, and J. Uitto. Deterministic Distributed Edge-Coloring with Fewer Colors. In *Proc. 50th ACM Symp. on Theory of Computing (STOC)*, pages 418–430, 2018.

[GPS88] A. Goldberg, S. Plotkin, and G. Shannon. Parallel Symmetry-Breaking in Sparse Graphs. *SIAM Journal on Discrete Mathematics*, 1(4):434–446, 1988.

[GS17] M. Ghaffari and H. Su. Distributed Degree Splitting, Edge Coloring, and Orientations. In *Proc. 28th ACM-SIAM Symp. on Discrete Algorithms (SODA)*, pages 2505–2523, 2017.

[Har18] D. G. Harris. Distributed approximation algorithms for maximum matching in graphs and hypergraphs. *CoRR*, abs/1807.07645, 2018.

[HMKS16] D. Hefetz, Y. Maus, F. Kuhn, and A. Steger. A Polynomial Lower Bound for Distributed Graph Coloring in a Weak LOCAL Model. In *Proc. 30th Symp. on Distributed Computing (DISC)*, pages 99–113, 2016.

[HSS16] D. G. Harris, J. Schneider, and H. Su. Distributed $(\Delta + 1)$-coloring in sublogarithmic rounds. In *Proc. 48th ACM Symp. on Theory of Computing (STOC)*, pages 465–478, 2016.

[KS18] K. Kawarabayashi and G. Schwartzman. Adapting Local Sequential Algorithms to the Distributed Setting. In *32nd Symp. on Distributed Computing (DISC)*, pages 35:1–35:17, 2018.

[KSOS06] K. Kothapalli, C. Scheideler, M. Onus, and C. Schindelhauer. Distributed coloring in $\tilde{O}(\sqrt{\log n})$ bit rounds. In *Proc. 20th Int. Parallel and Distributed Processing Symposium (IPDPS)*, 2006.

[KSV11] A. Korman, J. Sereni, and L. Viennot. Toward more localized local algorithms: removing assumptions concerning global knowledge. In *Proc. 30th ACM Symp. on Principles of Distributed (PODC)*, pages 49–58, 2011.

[Kuh09] F. Kuhn. Local Weak Coloring Algorithms and Implications on Deterministic Symmetry Breaking. In *Proc. of 21st ACM Symp. on Parallelism in Algorithms and Architectures (SPAA)*, 2009.

[KW06] F. Kuhn and R. Wattenhofer. On the Complexity of Distributed Graph Coloring. In *Proc. 25th ACM Symp. on Principles of Distributed Computing (PODC)*, pages 7–15, 2006.

[Lin87] N. Linial. Distributive graph algorithms – Global solutions from local data. In *Proc. 28th IEEE Symp. on Foundations of Computer Science (FOCS)*, pages 331–335, 1987.

[Lin92] N. Linial. Locality in Distributed Graph Algorithms. *SIAM Journal on Computing*, 21(1):193–201, 1992.

[Lub86] M. Luby. A Simple Parallel Algorithm for the Maximal Independent Set Problem. *SIAM J. on Computing*, 15:1036–1053, 1986.

[Nao91] M. Naor. A Lower Bound on Probabilistic Algorithms for Distributive Ring Coloring. *SIAM J. on Discrete Math.*, 4(3):409–412, 1991.

[NW64] C. Nash-Williams. Decompositions of finite graphs into forests. *J. of the London Math. Society*, 39:12, 1964.

[Pel00] D. Peleg. *Distributed Computing: A Locality-Sensitive Approach*. SIAM, 2000.

[PR01] A. Panconesi and R. Rizzi. Some simple distributed algorithms for sparse networks. *Distributed Computing*, 14(2):97–100, 2001.

[PS95a] A. Panconesi and A. Srinivasan. The local nature of Δ-coloring and its algorithmic applications. *Combinatorica*, 15(2):255–280, 1995.

[PS95b] A. Panconesi and A. Srinivasan. On the Complexity of Distributed Network Decomposition. *Journal of Algorithms*, 20(2):581–592, 1995.

[PS97] A. Panconesi and A. Srinivasan. Randomized Distributed Edge Coloring via an Extension of the Chernoff–Hoeffding Bounds. *SIAM Journal on Computing*, 26(2):350–368, 1997.

[PS13] S. Pettie and H. Su. Fast Distributed Coloring Algorithms for Triangle-Free Graphs. In *Proc. 40th Int. Coll. on Automata, Languages, and Programming (ICALP)*, pages 681–693, 2013.

[SV93] M. Szegedy and S. Vishwanathan. Locality based graph coloring. In *Proc. 25th ACM Symp. on Theory of Computing (STOC)*, pages 201–207, 1993.

[SV19] H. Su and H. T. Vu. Towards the Locality of Vizing's Theorem. In *Proc. 51st ACM Symp. on Theory of Computing (STOC)*, 2019.

[SW10] J. Schneider and R. Wattenhofer. A new technique for distributed symmetry breaking. In *Proc. 29th ACM Symp. on Principles of Distributed Computing (PODC)*, pages 257–266, 2010.

[Viz64] V. G. Vizing. On an estimate of the chromatic class of a p-graph. *Diskret Analiz*, 3(7):25–30, 1964.