Local Conflict Coloring Revisited: Linial for Lists

Yannic Maus
Technion – Israel Institute of Technology, Haifa, Israel
yannic.maus@cs.technion.ac.il

Tigran Tonoyan
Technion – Israel Institute of Technology, Haifa, Israel
ttonoyan@gmail.com

Abstract

Linial's famous color reduction algorithm reduces a given m-coloring of a graph with maximum degree Δ to a $O(\Delta^2 \log m)$-coloring, in a single round in the LOCAL model. We show a similar result when nodes are restricted to choose their color from a list of allowed colors: given an m-coloring in a directed graph of maximum outdegree β, if every node has a list of size $\Omega(\beta^2(\log \beta + \log\log m + \log\log |\mathcal{C}|))$ from a color space $\mathcal{C}$ then they can select a color in two rounds in the LOCAL model. Moreover, the communication of a node essentially consists of sending its list to the neighbors. This is obtained as part of a framework that also contains Linial's color reduction (with an alternative proof) as a special case. Our result also leads to a *defective list coloring* algorithm. As a corollary, we improve the state-of-the-art *truly local* $(deg+1)$-list coloring algorithm from Barenboim et al. [PODC'18] by slightly reducing the runtime to $O(\sqrt{\Delta \log \Delta}) + \log^* n$ and significantly reducing the message size (from huge to roughly Δ). Our techniques are inspired by the *local conflict coloring* framework of Fraigniaud et al. [FOCS'16].

2012 ACM Subject Classification Theory of computation → Distributed algorithms

Keywords and phrases distributed graph coloring, list coloring, low intersecting set families

Digital Object Identifier 10.4230/LIPIcs.DISC.2020.16

Related Version A full version of the paper is available at `https://arxiv.org/abs/2007.15251`.

Funding This project was supported by the European Union's Horizon 2020 Research and Innovation Programme under grant agreement no. 755839.

1 Introduction

Symmetry breaking problems are a cornerstone of distributed graph algorithms in the LOCAL model.[1] A central question in the area asks: *How fast can these problems be solved in terms of the maximum degree Δ, when the dependence on n is as mild as $O(\log^* n)$?* [12]. That is, we are looking for *truly local* algorithms, with complexity of the form $f(\Delta) + O(\log^* n)$.[2] The $O(\log^* n)$ term is unavoidable due to the seminal lower bound by Linial [41] that, via simple reductions, applies to most typical symmetry breaking problems. $(\Delta+1)$-*vertex coloring* and *maximal independent set (MIS)* are the key symmetry breaking problems. Both can be solved by simple centralized greedy algorithms (in particular they are always solvable), and even more importantly in a distributed context, any partial solution (e.g. a partial coloring) can be extended to a complete solution. The complexity of MIS is settled up to constant factors with $f(\Delta) = \Theta(\Delta)$, by the algorithm from [14] and a recent breakthrough lower bound by

[1] In the LOCAL model [41, 46] a communication network is abstracted as an n-node graph $G = (V, E)$ with unique $O(\log n)$-bit identifiers. Communications happen in synchronous rounds. Per round, each node can send one (unbounded size) message to each of its neighbors. At the end, each node should know its own part of the output, e.g., its own color. In the CONGEST model, there is a limitation of $O(\log n)$ bits per message.

[2] Not all symmetry breaking problems admit such a runtime, e.g., Δ-coloring and sinkless orientation [17].

Balliu et al. [3]. In contrast, the complexity of vertex coloring, despite being among the most studied distributed graph problems [12], remains widely open, with the current best upper bound $f(\Delta) = \tilde{O}(\sqrt{\Delta})$ and lower bound $f(\Delta) = \Omega(1)$. While most work has focused on the $(\Delta+1)$-coloring problem, recent algorithms, e.g., [8, 23, 13, 37], rely on the more general *list coloring problem* as a subroutine, where each vertex v of a graph G has a list $L_v \subseteq \mathcal{C}$ of colors from a colorspace $\mathcal{C}$, and the objective is to compute a proper vertex coloring, but each vertex has to select a color from its list. Again, a natural case is the always solvable $(deg+1)$*-list coloring problem*, where the list of each vertex is larger than its degree. Our paper contributes to the study of list coloring problems. To set the stage for our results, let us start with an overview of truly local coloring algorithms.

In [41], besides the mentioned lower bound, Linial also showed that $O(\Delta^2)$-coloring can be done in $O(\log^* n)$ rounds. Szegedy and Vishwanathan [49] improved the runtime to $\frac{1}{2}\log^* n + O(1)$ rounds, and showed that in additional $O(\Delta \cdot \log \Delta)$ rounds, the $O(\Delta^2)$-coloring could be reduced to $(\Delta+1)$-coloring. The latter result was rediscovered by Kuhn and Wattenhofer [39]. Barenboim, Elkin and Kuhn used *defective coloring* for partitioning the graph into low degree subgraphs and coloring in a divide-and-conquer fashion, and brought the complexity of $(\Delta+1)$-coloring down to $O(\Delta + \log^* n)$ [14]. A simpler algorithm with the same runtime but without using defective coloring was obtained recently in [13]. All of these results also hold for $(deg+1)$-list coloring, since given a $O(\Delta)$-coloring, one can use it as a "schedule" for computing a $(deg+1)$-list coloring in $O(\Delta)$ additional rounds. Meantime, two sub-linear in Δ algorithms were already published [8, 23]. They both used *low outdegree colorings*, or *arb-defective colorings*, introduced by Barenboim and Elkin in [10], for graph partitioning purposes. The basic idea here is similar to the defective coloring approach, with the difference that the graph is partitioned into directed subgraphs with low *maximum outdegree*. In [13] they also improved and simplified the computation of low outdegree colorings, which led to improved runtime and simplification of that component in the mentioned sublinear algorithms. As a result, the currently fastest algorithm in CONGEST needs $O(\Delta^{3/4} + \log^* n)$ rounds ([8]+[13]),[3] while the fastest one in LOCAL needs $O(\sqrt{\Delta \log \Delta} \log^* \Delta + \log^* n)$ rounds ([23]+[13]).

Let us take a better look at the latter result, which is the closest to our paper. The algorithm consists of two main ingredients: (1) an algorithm that partitions a given graph into $p = O(\Delta/\beta)$ subgraphs, each equipped with a $\beta = O(\sqrt{\Delta/\log \Delta})$-outdegree orientation, in $O(p) + \frac{1}{2}\log^* n$ rounds [13], (2) a list coloring subroutine that gives rise to the following [23]:

▶ **Theorem 1.1** ([23]). *In a directed graph with max. degree Δ, max. outdegree β, and an input m-coloring, list coloring with lists L_v of size $|L_v| \geq 10\beta^2 \ln \Delta$ from any color space can be solved in $O(\log^*(\Delta + m))$ rounds in LOCAL.*

The two ingredients are combined to give a $(deg+1)$-list coloring of the input graph G [23]. First, partition G using (1), iterate through the p directed subgraphs, and for each of them, let uncolored nodes refine their lists by removing colors taken by a neighbor, and use (2) to color nodes that still have sufficiently large lists ($\geq 10\beta^2 \ln \Delta$). With a fine grained choice of β it is ensured that every node v with (refined) list size greater than $\Delta/2$ is colored. Thus, after iterating through all p subgraphs, all uncolored nodes have list size at most $\Delta/2$, and because each vertex always has *more colors in its list than uncolored neighbors*, the max degree of the graph induced by uncolored nodes is at most half of that of G. Thus, we can apply the partition-then-list-color paradigm recursively to the subgraph induced by uncolored nodes to complete the coloring. The runtime is dominated by the first level of recursion.

[3] For more colors, i.e., $(1+\varepsilon)\Delta$-coloring, [8] gives runtime $O(\sqrt{\Delta} + \log^* n)$ in CONGEST.

The strength of the partitioning algorithm above is that it is conceptually simple and works with small messages. In contrast, the algorithm from Thm. 1.1 is conceptually complicated and uses gigantic messages. This complication might be due to the generality of the addressed setting as, in fact, [23] studies the more general local conflict coloring problem, and Thm. 1.1 is only a special case. In *local conflict coloring*, each edge of the given graph is equipped with an arbitrary conflict relation between colors and this relation may vary across different edges. This framework is also leveraged to achieve the colorspace size independence of Thm. 1.1 (see the technical overview below). It was not clear prior to our work whether the situation simplifies significantly if one restricts to ordinary list coloring because, even if the input to the algorithm in Thm. 1.1 is a list coloring problem, the intermediate stages of the algorithm fall back to the more general local conflict coloring.

The overarching goal of our paper is providing deeper understanding of the remarkable framework of [23], better connecting it with classic works in the area, and obtaining a simpler list coloring algorithm that also uses smaller messages, by moderately sacrificing generality.

1.1 Our Contribution

Our main result is a simple algorithm that yields the following theorem.

▶ **Theorem 1.2** (Linial for Lists). *In a directed graph with max. degree Δ, max. outdegree β, and an input m-coloring, list coloring with lists L_v from a color space $\mathcal{C}$ and of size $|L_v| \geq l_0 = 4e\beta^2(4\log\beta + \log\log|\mathcal{C}| + \log\log m + 8)$ can be solved in 2 rounds in LOCAL. Each node sends $l_0 + 1$ colors in the first round, and a l_0/β^2-bit message in the second.*

The name "Linial for Lists" stems from the fact that Thm. 1.2 is a "list version" of one of the cornerstones of distributed graph coloring, Linial's color reduction, which says that an m-coloring can be reduced to a $(5\Delta^2\log m)$-coloring in a single round [41]. Moreover, our framework is itself a *natural generalization of Linial's approach of cover-free set families*. Applied to equal lists, it yields an alternative proof of Linial's color reduction, in the form of a *greedy construction of cover-free families* (Linial proved their existence using the probabilistic method [41]; he also used an alternative construction from [21] via polynomials over finite fields, which however yields a weaker color reduction for $m \gg \Delta$) (see Sec. 2.3 and Sec. 5).

Compared with Thm. 1.1, we lose colorspace independence, and our algorithm does not extend to general local conflict coloring (although we use a kind of conflict coloring in the process). In exchange, we eliminate Δ from the bound on the list size, reduce the runtime to exactly 2 rounds, and dramatically reduce message size. This is achieved by a non-trivial paradigm shift in the local conflict coloring framework (see the technical overview below). The runtime cannot be reduced to 1 round, due to a lower bound of [49] (see Sec. 5).

Combining Thm. 1.2 with the partitioning algorithm of [13] as outlined above gives us a $(deg+1)$-list coloring algorithm. Note that any improvement in the list size bound in Thm. 1.2 (with little increase in runtime) would yield a faster $(deg+1)$-list coloring algorithm.

▶ **Theorem 1.3** ($(deg+1)$-List Coloring). *In a graph with max. degree Δ, $(deg+1)$-list coloring with lists $L_v \subseteq \mathcal{C}$ from a color space of size $|\mathcal{C}| = 2^{\mathrm{poly}(\Delta)}$ [4] can be solved in $O\big(\sqrt{\Delta\log\Delta}\big) + \frac{1}{2}\cdot\log^* n$ rounds in LOCAL. Furthermore, each node only needs to broadcast to its neighbors a single non-CONGEST message consisting of a subset of its list.*

[4] We use the notation $\mathrm{poly}(X) = O(X^c)$, for an absolute constant c, and $\tilde{O}(X) = X\cdot\mathrm{poly}(\log X)$.

The bound on the color space size stems from the color space dependence in Theorem 1.2. As discussed in Sec. 5, it is possible to trade color space dependence with runtime in Theorem 1.2, which could improve or suppress the bound in Theorem 1.3. That, however, comes with the cost of having huge messages.

Theorem 1.3 immediately provides the fastest known truly local $(\Delta+1)$-coloring algorithm in LOCAL. Below we list further implications of our framework.

- **CONGEST (see Cor. 4.2):** We obtain an improved $(\Delta+1)$-coloring algorithm in a *low degree* regime in CONGEST. In particular, if $\Delta = \tilde{O}(\log n)$ then $(\Delta+1)$-coloring (more generally, $(deg+1)$-list coloring with colorspace of size $|\mathcal{C}| = \text{poly}(\Delta)$) can be solved in $\tilde{O}(\sqrt{\Delta}) + \frac{1}{2} \cdot \log^* n$ rounds in CONGEST. Generally, if one allows messages of size B, this runtime holds for degree up to $\Delta = \tilde{O}(B)$. On the other hand, if $\Delta = \Omega(\log^{2+\varepsilon} n)$, for an arbitrarily small constant $\varepsilon > 0$, an algorithm from recent work [37] achieves runtime $O(\sqrt{\Delta})$ in CONGEST (if one recasts their dependency on n as a Δ-dependency). Thus, only for the regime of $\Delta \in \Omega(\log^{1+\epsilon} n) \cap O(\log^{2+\epsilon} n)$ we do not have an algorithm with runtime $\tilde{O}(\sqrt{\Delta})$ in CONGEST (with the current best being $O(\Delta^{3/4} + \log^* n)$ due to [8]).
- **Defective list coloring:** Our framework extends to *d-defective list coloring*, that is, list coloring where each node v can have at most d neighbors with the same colors as v: If lists are of size $\Omega\big((\Delta/(d+1))^2 \cdot (\log \Delta + \log\log |\mathcal{C}| + \log\log m)\big)$ we can compute a d-defective list coloring in 2 rounds in LOCAL. The result can be seen as the "list variant" of a defective coloring result in [36]. While we are not aware of an immediate application, defective list coloring with a better "colors vs. defect" tradeoff (d vs. $O(\Delta/d)$) for *line graphs* has recently been used to obtain a *edge-coloring* algorithm with complexity quasi poly $\log \Delta + O(\log^* n)$ [6]. See the full paper [42] for the formal statement and proof.
- **Δ-coloring:** The improvements obtained in Thm. 1.3 also imply respective improvements for several Δ-*coloring* algorithms that use $(\deg+1)$-list coloring as a subroutine [27].

1.2 Technical Overview

At their core, the proofs of Theorems 1.1 and 1.2 are based on three important concepts: *conflict coloring*, *problem amplification* and *0-round solvability*. A *conflict coloring problem* is a list coloring problem where two colors can conflict even if they are not equal. The associated *conflict degree* is the maximum number of conflicts per color a node can have. *Problem amplification* transforms one conflict coloring problem instance into another, as follows: given an input to a problem A, each node computes its input to another problem B (perhaps by exchanging information along the way), with the property that, 1. having a solution to B, a simple one round algorithm computes a solution to A, and 2. the list-size-to-conflict-degree (l/d) ratio of B is larger than that of A. Note that the first property essentially determines the conflicts in B, and usually a color in B is a set of colors in A. The importance of the second property stems from the concept of *0-round solvability*: an instance of a problem B with large enough l/d ratio can be solved in 0 rounds, i.e., with no communication.

From here, the plan is simple: take problem P_0, which is the list coloring problem, recast it as a conflict coloring problem, and amplify it into problems $P_1, \ldots, P_t$, so that P_t is 0-round solvable. Then we can cascade down to a solution of problem P_0, in t rounds. Crucially, in order to do the above, we need P_0 to have sufficiently large l/d ratio to begin with (which explains the particular list size requirements in our theorems). The input m-coloring is used for tie-breaking in the 0-round solution of P_t.

In [23], *local conflict coloring* is the main problem type, where the conflict between two colors depends on who the colors belong to, i.e., two colors can conflict along one edge of the graph and not conflict on another one. Their framework allows solving any local conflict

coloring problem, and by re-modeling a problem with an arbitrary colorspace via mapping each list to an interval $[1, l]$ of natural numbers, one can redefine local conflicts and "forget" about the real size of the colorspace (hence colorspace independence). When computing the input of P_i (given P_{i-1}), in order to maintain manageable conflict degree, *nodes exchange messages* to filter out colors in P_i that cause too much conflict with any neighboring node. These messages are huge (recall that a color in P_i is a set of colors in P_{i-1}). Thus, the input to P_i is usually the topology, P_0-lists and conflicts in the i-hop neighborhood of a node. The goal towards 0-round solvability is then to find a problem P_t whose l/d ratio is larger than the number of all t-hop neighborhood patterns (i.e., inputs). The complicated nature of the input to P_t also makes the 0-round solvability proof rather conceptually involved. The number t of problems required is about $3\log^*(m + \Delta)$.

Our framework, on the other hand, is based on special *global conflict coloring* instances, where the conflict relation of two colors does not depend on the edge across which they are. This limits us to solving only ordinary list problems P_0. Our key insight (see Section 3.3), which sets Theorems 1.1 and 1.2 apart, is that in our setting *nodes do not need to communicate* for computing the input to problems $P_1, \ldots, P_t$. To achieve this, we show that when forming the lists for P_i from the input to P_{i-1}, it suffices to drop "universally bad" colors (sets of colors in P_{i-1}), whose absence is enough to ensure moderate conflict degree towards any (!) other node. We achieve this by crucially exploiting the symmetry of the particular conflict coloring problems arising from ordinary list coloring.

Thus, the input of a node in P_t is just its input in P_0. This makes the 0-round solution (of P_t) particularly simple. The only communication happens when we cascade down from a solution of P_t to that of P_0. With $t = 2$, we get our main theorem. Since here we have only two problems, the message size is limited (the first round is needed to *learn the P_0-lists* of neighbors, while the second one consists of a *small auxiliary message*). Taking larger t would reduce the requirement on the initial list size but increase message size (see Section 5). Since $t = 2$ is sufficient for our applications, we limit our exposition to that case. Setting $t = 1$ does not give anything non-trivial for list coloring, since the l/d ratio is not large enough, but when all P_0-lists are equal, it gives *an alternative proof of Linial's color reduction* (Section 2.3). In fact, P_1 is essentially the problem of finding a *low intersecting set family*, which Linial's algorithm is based on, while P_2 is a "higher-dimensional" variant of it. Thus, at the core of our result there is an (offline) construction of certain set families over the given color space: given those, the algorithm is easy. This way, we believe our paper also provides a deeper insight into the framework of Thm. 1.1. Our result can also be seen as a bridge between the results of [23] and the recently popular concept of speedup (see Section 5).

1.3 Further Related Work

Most results on distributed graph coloring until roughly 2013 are covered in the excellent monograph by Barenboim and Elkin [12]. An overview of more recent work can be found in [37]. Due to the large volume of published work on distributed graph coloring, we limit this section to an informative treatment of a selected subset. While we have covered most literature on truly local vertex coloring algorithms, there are many known algorithms that trade the high Δ-dependence in the runtime with lower n-dependence. All **deterministic** algorithms in this category (for general input graphs) involve a $\Omega(\log n)$ factor. From the early 90s until very recently, the complexity of $(\deg+1)$-list coloring (and $(\Delta+1)$-coloring) in terms of n was $2^{O(\sqrt{\log n})}$ [1, 45], with algorithms based on *network decomposition* (into small diameter components). A recent breakthrough in network decomposition algorithms [47]

brought the runtime of $(deg+1)$-list coloring down to $\operatorname{poly}\log n$ in LOCAL (it also applies to many other symmetry breaking problems; see [47, 29, 26]). A little later, [7] found a $\operatorname{poly}\log n$ round CONGEST algorithm.

Historically, decompositions into subgraphs that are equipped with low outdegree orientations as used in our results, in [23], and in [8] are closely related to the notion of arboricity. To the best of our knowledge, [9] was the first paper to introduce low out-degree orientations as a tool for distributed graph coloring. First, they showed that one can compute $O(a)$-outdegree orientations in graphs with arboricity a in $O(\log n)$ rounds, and used it to devise several algorithms to color graphs with bounded arboricity. [9] is also the first paper to notice that the degree bound of Δ in Linial's color reduction can be replaced with a bound on the outdegree. Then, [10] devised methods to recursively partition into graphs with small arboricity yielding an $O(\log\Delta\log n)$-round algorithm for $O(\Delta^{1+\varepsilon})$-coloring and an $O(\Delta^{\varepsilon}\log n)$-round algorithm for $O(\Delta)$ coloring. Recently, this recursive technique was extended to $(\deg+1)$-list coloring, giving a $(2^{O(\sqrt{\log\Delta})}\log n)$-round algorithm [37]; the runtime of [37] has a hidden dependence on the color space. While [9, 10, 37] have an inherent $O(\log n)$-factor in their runtime, [8] showed that one can decompose a graph into small arboricity subgraphs (equipped with a small outdegree orientation) without inferring a $O(\log n)$ factor, yielding the first sublinear in Δ algorithm for $\Delta+1$ coloring. In the aftermath, [13] improved the runtime for computing the underlying decompositions (and also simplified the algorithm). Thus, the best forms of our results, [23] and [8] are obtained by using [13] to compute decompositions into subgraphs of small arboricity (equipped with small outdegree orientations).

Note that our results, [23] and [8] only require a bound on the outdegree of the subgraphs' orientations and are oblivious to their arboricity. While bounded outdegree in a graph with a given orientation implies bounded arboricity, computing a bounded outdegree orientation in a graph with bounded arboricity requires $\Omega(\log n)$ rounds, as shown in [9].

Recent **randomized** coloring algorithms rely on the *graph shattering* technique [15]. In the *shattering phase*, a randomized algorithm computes a partial coloring of the graph, after which every uncolored connected component of the graph has small size (say, $\operatorname{poly}\log n$). Then, in the *post-shattering* phase, deterministic $(\deg+1)$-list coloring is applied on all uncolored components in parallel. The runtime of the shattering phase has progressed from $O(\log\Delta)$ [15], over $O(\sqrt{\log\Delta})$ [33] to $O(\log^*\Delta)$ [20]. Combined with the $\operatorname{poly}\log n$-round list coloring algorithm of [47], this gives the current best runtime $\operatorname{poly}\log\log n$, for $(\Delta+1)$-coloring [20], and $O(\log\Delta)+\operatorname{poly}\log\log n$, for $(deg+1)$-list coloring [15].

While special graph classes are out of the scope of this paper, we mention the extensively studied case of distributed **edge coloring**. Here, $\operatorname{poly}\log n$-round algorithms were designed for progressively improving number of colors, from $(2+\varepsilon)\Delta$ [31, 28] to $(2\Delta-1)$ [22, 32], then to $(1+\varepsilon)\Delta$ [30, 32, 48]. The truly local complexity of $(2\Delta-1)$-edge coloring has improved from $O(\Delta)$ [44] to $2^{O(\sqrt{\log\Delta})}$ [37] then to quasi $\operatorname{poly}\log\Delta$ [6] (in addition to $O(\log^* n)$). $O(\Delta^{1+\varepsilon})$-edge colorings can be computed in $O(\log\Delta+\log^* n)$ rounds [11].

Little is known on coloring **lower bounds** (in contrast to other symmetry breaking problems, e.g., maximal matching, MIS or ruling sets [38, 3, 4]). Linial's $\Omega(\log^* n)$ lower bound is extended to randomized algorithms in [43]. The deterministic bound has recently been re-proven in a topological framework [24]. A $\Omega(\Delta^{1/3})$ lower bound for $O(\Delta)$-coloring holds in a weak variant of the LOCAL model [34]. Several works characterized coloring algorithms which can only spend a single communication round [49, 39, 34]. None of these results gives anything non-trivial for two rounds. Also, the *speedup* technique (e.g., [16, 17, 3, 18, 4, 2]), which proved very successful for MIS lower bounds, is poorly understood for graph coloring. We briefly discuss the technique and its relation to our result in Sec. 5. There are lower

bounds for more restricted variants of coloring. There is a $\Omega(\log n)$ ($\Omega(\log\log n)$) lower bound for deterministic [17] (randomized [19]) Δ-coloring, as well as for $(\Delta-2)$-defective 2-coloring [5]. Further, [25] provides a $\Omega(\log n/\log\log n)$ lower bound for greedy coloring. Similar bounds hold for coloring trees and bounded arboricity graphs with significantly fewer than Δ colors [41, 9].

1.4 Roadmap

Section 2.1 introduces our version of conflict coloring together with the 0-round solvability lemma. Section 2.2 defines the problems P_0 and P_1 and provides further notation. Section 2.3 contains the first result of our framework: an alternative proof of Linial's algorithm. Theorem 1.2 (Linial for Lists) is proved in Section 3. Theorem 1.3 ($(deg+1)$-list coloring) is proved in Section 4. We conclude with a discussion of the results and open problems in Section 5.

2 Basic Setup and Linial's Color Reduction

In this section, we first introduce the conflict coloring framework that is the basis of our algorithm, then we show how it quickly implies an alternative variant of Linial's color reduction algorithm. For a set S and an integer $k \geq 0$, let $\mathcal{P}(S)$ and $\binom{S}{k}$ denote the set of all subsets and all size-k subsets of S, respectively. For a map f we use $f^{(i)}$ to denote the i-fold application of f, e.g., $\mathcal{P}^{(2)}(S) = \mathcal{P}(\mathcal{P}(S))$.

2.1 Global Conflict Coloring

A *list family* $\mathcal{F} \subseteq \mathcal{P}(\mathcal{C})$ is a set of subsets of a color space $\mathcal{C}$. Given a *symmetric conflict relation* $\mathcal{R} \subseteq \{\{c,c'\} \mid c,c' \in \mathcal{C}\}$, the *conflict degree* of a family $\mathcal{F}$ in $\mathcal{R}$ is the maximum number of colors in a list L that conflict with a single color in a list L' (possibly same as L), i.e. , $d_{\mathcal{R}}(\mathcal{F}) = \max_{L,L'\in\mathcal{F},c\in L} |\{c' \in L' \mid \{c,c'\} \in \mathcal{R}\}|$. An instance $\mathfrak{P} = (\mathcal{C},\mathcal{R},\mathcal{F},\mathcal{L})$ of the *global conflict coloring problem* on the graph G is given[5] by a color space $\mathcal{C}$, a symmetric conflict relation $\mathcal{R}$ on $\mathcal{C}$, a list family $\mathcal{F}$, and an assignment $\mathcal{L} : V \to \mathcal{F}$ of lists $\mathcal{L}(v) \in \mathcal{F}$ of colors to each vertex v. The goal is to assign each vertex a color from its list such that no pair of neighboring vertices get conflicting colors $\{c,c'\} \in \mathcal{R}$. The *conflict degree* of $\mathfrak{P}$ is $d_{\mathcal{R}}(\mathcal{F})$. Note that the conflict degree *does not depend on* G *or* $\mathcal{L}$.

▶ **Lemma 2.1** (Zero Round Solution). *An instance $(\mathcal{C},\mathcal{R},\mathcal{F},\mathcal{L})$ of the conflict coloring problem on a graph G can be solved without communication if G is m-colored, $m,\mathcal{R},\mathcal{F}$ are globally known, and every list in $\mathcal{F}$ has size at least $l > m \cdot |\mathcal{F}| \cdot d_{\mathcal{R}}(\mathcal{F})$.*

Proof. Every vertex v has a *type* $(\psi_v, \mathcal{L}(v)) \in [m] \times \mathcal{F}$, which is uniquely determined by its input color ψ_v and list $\mathcal{L}(v)$. Note that adjacent vertices have distinct types, and there are $t = m \cdot |\mathcal{F}|$ (globally known) possible types. Below, we show how to greedily assign each type a color from its list s.t. different types get non-conflicting colors. The conflict coloring problem is then solved by running this algorithm locally and consistently by all vertices, where each vertex gets the color assigned to its type.

[5] Formally, G is also part of the problem, but we omit it since it is always clear from the context. These definitions crucially differ from local conflict coloring in [23], where a pair of colors can conflict along one edge and not conflict along another.

Let $\{T_i = (m_i, L_i)\}_{i=1}^{t}$ be a fixed ordering of $[m] \times \mathcal{F}$. Assign T_1 a color $\phi(T_1) \in L_1$ arbitrarily. For any $i \geq 1$, given the colors $\phi(T_1), \ldots, \phi(T_i)$ of preceding types, assign T_{i+1} a color from L_{i+1} that does not conflict with $\phi(T_1), \ldots, \phi(T_i)$. This can be done since each of the i fixed colors conflicts with at most $d_{\mathcal{R}}(\mathcal{F})$ colors in L_{i+1}, i.e., there are at most $i \cdot d_{\mathcal{R}}(\mathcal{F}) \leq m \cdot |\mathcal{F}| \cdot d_{\mathcal{R}}(\mathcal{F})$ colors that T_{i+1} cannot take, and this is less than the size of L_{i+1}, as assumed. ◀

2.2 Basic Problems: P_0 and P_1

Let $\mathcal{C}$ be a fixed and globally known color space (which may depend on the graph G). An *i-list* is a subset $L \subseteq \mathcal{P}^{(i)}(\mathcal{C})$; e.g., the initial color list $L_v \subseteq \mathcal{C}$ of a vertex v is a 0-list. Below, we introduce two problems. Problem P_0 is the standard list coloring problem, which we would like to solve via Lemma 2.1. However, the Lemma may not apply, if the lists L_v are not large enough. We then introduce problem P_1, with parameters $0 < \tau \leq k$, which is a *low intersecting sublist selection* problem. On the one hand, P_1 can be reformulated as a conflict coloring problem with larger lists and color space (hence could be solvable via Lemma 2.1), and on the other hand, a solution to P_1 can be used to solve P_0. The input of a node v in both problems contains its list L_v. Formally, we have, for parameters τ and k,

- **P_0 (list coloring):** Node v has to output a color $c(v) \in L_v$ such that adjacent nodes' colors do not conflict, i.e., they are not equal.
- **P_1 (low intersecting sublists):** Node v has to output a 0-list $C_v \subseteq L_v$ such that $|C_v| = k$ and adjacent nodes' 0-lists do not τ-conflict.

  Two 0-lists $C, C' \subseteq \mathcal{C}$ do *τ-conflict* if $|C \cap C'| \geq \tau$.

Note that problems P_0 and P_1 are not conflict coloring problems in the formal sense defined above (e.g., we do not define a list family $\mathcal{F}$ or a list assignment $\mathcal{L} : V \to \mathcal{F}$). The aim with such definitions is to have a higher level and more intuitive (but still formal) problem statement. As the name suggests, in P_1 each node needs to compute a subset of its list such that the outputs form a low intersecting set family. P_1 can be reduced to a formal conflict coloring problem $\mathfrak{P}_1$ whose solution immediately solves the P_1 instance (see Thm. 2.2).

2.3 Warmup: Linial's Color Reduction (without Lists)

As a demonstration, we use the introduced framework to re-prove Linial's color reduction theorem [41, Thm. 4.1] (which was extended to directed graphs in [9]). An *r-cover-free family* of size k over a set U is a collection of k subsets $C_1, \ldots, C_k \subseteq U$ such that no set C_i is a subset of the union of r others. The obtained algorithm is essentially a *greedy construction* (via Lemma 2.1) of an r-cover-free family (with appropriate parameters) whose existence was proved via the probabilistic method in [41]. This greedy construction was first obtained in [35] but, to our knowledge, remained unnoticed in the distributed computing community.

While our aim is to make the proof below reusable for the later sections (hence the general statement in list coloring terms), we note that similar ideas can be used to obtain a less technical proof of Linial's color reduction (see the full version [42]).

▶ **Theorem 2.2** ([41, 9]). *Let the graph G be m-colored and oriented, with max outdegree β. All nodes have an identical color list $L_v = \mathcal{C}$ from a color space $\mathcal{C}$. Further, $\mathcal{C}$, m, and β are globally known. If $|L_v| = l_0 \geq 2e \cdot \beta^2 \cdot \lceil \log m \rceil$ holds then in 1 round in LOCAL, each node can output a color from its list s.t. adjacent nodes output distinct colors.*

Proof. Our goal is to solve P_0 with the given lists. To this end, it suffices to solve P_1 with parameters $\tau = \lceil \log m \rceil > 1$ and $k = \beta \cdot \tau$, without communication. Indeed, having selected the sublists $(C_v)_{v \in V}$, we need only one round to solve P_0: Node v learns sublists

of its outneighbors and outputs any color $c(v) \in C_v \setminus \cup_{u \in N_{out}(v)} C_u$. This can be done since for any outneighbor u, C_v and C_u do not τ-conflict ($|C_v \cap C_u| \leq \tau - 1$) and hence $|C_v \setminus \cup_{u \in N_{out}(v)} C_u| \geq k - (\tau - 1)\beta > 0$.

We recast the given P_1 instance with (identical) input lists $(L_v)_{v \in V}$ as a conflict coloring problem $\mathfrak{P}_1 = (\mathcal{C}_1, \mathcal{R}_1, \mathcal{F}_1, \mathcal{L}_1)$ with color space $\mathcal{C}_1 = \mathcal{P}(\mathcal{C})$ and the τ-conflict relation as $\mathcal{R}_1$. The list of a node $\mathcal{L}_1(v) = \binom{L_v}{k}$ in $\mathfrak{P}_1$ consists of all k-sized subsets of its input list L_v. As each L_v is identical to $\mathcal{C}$, we have that $\mathcal{L}_1$ maps each node v to the same list $\binom{L_v}{k} = \binom{\mathcal{C}}{k}$ and the list family $\mathcal{F}_1 = \{\binom{\mathcal{C}}{k}\}$ consists of that singleton. A solution to $\mathfrak{P}_1$ immediately solves P_1.

▷ Claim 2.3. The conflict degree of $\mathfrak{P}_1$ is upper bounded by $d_1 = \binom{k}{\tau} \cdot \binom{l_0 - \tau}{k - \tau}$.

Proof. Consider two arbitrary lists $L, L' \in \mathcal{F}_1$ and some 0-list $C \in L$ (that is of size k). Each 0-list $C' \in L'$ that τ-conflicts with C can be constructed by first choosing τ elements of C, and then adding $k - \tau$ elements from the rest of $\mathcal{C}$ which is of size $l_0 - \tau$. This can be done in at most d_1 many ways. ◁

▷ Claim 2.4. Let $l_1 = \binom{l_0}{k}$. For any $k \geq \tau > 1$, if $l_0 \geq 2ek^2/\tau$ then $l_1/d_1 > 2^\tau$.

Proof. We have

$$\frac{l_1}{d_1} = \frac{\binom{l_0}{k}}{\binom{l_0 - \tau}{k - \tau}\binom{k}{\tau}} > \left(\frac{l_0}{k}\right)^\tau \cdot \left(\frac{\tau}{ek}\right)^\tau = \left(\frac{l_0 \tau}{ek^2}\right)^\tau \geq 2^\tau \,, \tag{1}$$

where in the first inequality, we used the well-known approximation $\binom{k}{\tau} \leq (ek/\tau)^\tau$, and the following inequality, applied to $\binom{l_0}{k}/\binom{l_0 - \tau}{k - \tau}$: for integers $L > K > x > 0$,

$$\frac{\binom{L}{K}}{\binom{L-x}{K-x}} = \frac{L!(K-x)!(L-K)!}{K!(L-K)!(L-x)!} = \frac{L(L-1)\dots(L-x+1)}{K(K-1)\dots(K-x+1)} > \left(\frac{L}{K}\right)^x \,, \tag{2}$$

which follows as $(L-i)/(K-i) > L/K$ holds for $0 < i \leq x$. ◁

Since $\tau \geq \lceil \log m \rceil$ and $|\mathcal{F}_1| = 1$ the last claim implies $|\mathcal{L}_1(v)| = l_1 > md_1 \geq m|\mathcal{F}_1| d_{\mathcal{R}_1}(\mathcal{F}_1)$, hence we can solve $\mathfrak{P}_1$ (and thus also P_1) without communication, using Lemma 2.1. ◀

What we did above is *greedily* forming a Δ-*cover-free family* $C_1, \dots, C_m \subseteq [O(\Delta^2 \log m)]$ of *size* m. The same was done in [41], using the probabilistic method. Having such a family globally known, every vertex of input color x picks, in 0 rounds, the set C_x as its candidate output colors (neighboring vertices get distinct sets). Then, every vertex of color x learns the sets C of its neighbors, and based on the Δ-cover-free property and the fact that there are at most Δ neighbors, can select a color $c \in C_x$ that is not a candidate for any neighbor.

3 Linial for Lists

The goal of this section is to prove the following theorem.

▶ **Theorem 1.2** (Linial for Lists). *In a directed graph with max. degree Δ, max. outdegree β, and an input m-coloring, list coloring with lists L_v from a color space $\mathcal{C}$ and of size $|L_v| \geq l_0 = 4e\beta^2(4 \log \beta + \log\log|\mathcal{C}| + \log\log m + 8)$ can be solved in 2 rounds in LOCAL. Each node sends $l_0 + 1$ colors in the first round, and a l_0/β^2-bit message in the second.*

Assumption. Throughout this section, we assume that *the list of each node is exactly of size* l_0; if a node's list is larger it can select an arbitrary subset of size l_0.

3.1 The Problem P_2 (Low Intersecting Sublist Systems)

Recall that when applying our framework to the case where all lists were equal (Thm. 2.2), we essentially constructed a Δ-cover-free family over the color space, and this was sufficient because we only needed a family of size m: one set for each possible input pair (x, L) of a color x and list L, with *the same L for all nodes*. In order to replicate the construction for list coloring, we would need to construct a cover-free family with a set $C_{x,L}$ for every combination of color x and list L, and such that $C_{x,L} \subseteq L$. It is not hard to see that such a family does not exist. Instead, we introduce problem P_2, whose goal is to assign every input (x, L) a *collection of candidate subsets* $K_{x,L} = \{C_{x,L,1}, C_{x,L,2}, \ldots\}$, where each $C_{x,L,i} \subseteq L$. Further, we need that for every pair of distinct collections, there are not many pairs of subsets from the two collections that intersect much (in a sense formally defined below). This ensures that having such $K_{x,L}$, the nodes can compute the desired Δ-cover free family with one communication round, and use it to choose a color in another round.

Problem P_2 depends on parameters $0 < \tau \leq k \leq l_0$ and $0 < \tau' \leq k'$, and each node has a list $L_v \subseteq \mathcal{C}$ in its input. Instead of a color (as in P_0) or a sublist (as in P_1), each vertex v now needs to output a collection $K_v = \{C_1, C_2, \ldots\}$ of sublists of L_v, each of size k.

> ***P_2* (low intersecting sublist systems):** Node v has to output a 1-list $K_v \subseteq \mathcal{P}(L_v)$ s.t. adjacent nodes' 1-lists do not (τ', τ)-conflict and $|K_v| = k'$ and $|C| = k$ for all $C \in K_v$.
> Two 1-lists $K, K' \subseteq \mathcal{P}(\mathcal{C})$ do (τ', τ)-*conflict* if there are two sequences $C_1, \ldots, C_{\tau'} \in K$ and $C'_1, \ldots, C'_{\tau'} \in K'$, where at least one of the sequences has τ' distinct elements and for every $1 \leq i \leq \tau'$, C_i and C'_i τ-conflict.

We prove in Lemma 3.1 that with a suitable choice of the parameters a solution of $P_2(\tau, k, \tau', k')$ yields a solution of $P_1(\tau, k)$ and P_0, where we implicitly impose (and throughout this section assume) that P_0, P_1 and P_2 receive the same input lists $(L_v)_{v \in V}$.

3.2 Algorithm

Under the assumptions of Thm. 1.2, fix the following (globally known) four parameters: $\tau = \lceil 8 \log \beta + 2 \log \log |\mathcal{C}| + 2 \log \log m \rceil + 14$, $\tau' = 2^{\tau - \lceil \log(2e\beta^2) \rceil}$, $k = \beta \cdot \tau$ and $k' = \beta \cdot \tau'$. Note that, τ', k, k' are determined by τ and β. [6] We have the bound $l_0 \geq 2ek^2/\tau$ on list size.

The algorithm consists of two phases. In the first phase, nodes **locally** and without any communication compute a solution $(K_v)_{v \in V}$ of P_2 consisting of 1-lists (see Lemma 3.2). The second phase has two rounds of communication. In the **first round**, each node v learns the solution K_u to P_2 of each outneighbor $u \in N_{out}(v)$, and selects a 0-list $C_v \in K_v$ that does not conflict with the 0-lists in K_u, for $u \in N_{out}(v)$, and thus is a solution to P_1 (Lemma 3.1). In the **second round**, node v learns the lists C_u of outneighbors, and selects a color $c(v) \in C_v$ that does not appear in C_u, for $u \in N_{out}(v)$ (Lemma 3.1). This solves P_0.

▶ **Lemma 3.1** ($P_2 \to P_1 \to P_0$)**.** *Given a solution $(K_v)_{v \in V}$ of P_2 (a solution $(C_v)_{v \in V}$ of P_1), a solution of P_1 (of P_0, resp.) can be computed in one round.*

Proof. $P_2 \to P_1$: As K_v and K_u do not (τ', τ)-conflict for any $u \in N_{out}(v)$, there are at most $\tau' - 1$ 0-lists $C \in K_v$ that τ-conflict with a 0-list in K_u. By removing all C from K_v that τ-conflict with any $C' \in K_u$ for any outneighbor $u \in N_{out}(v)$ at least $|K_v| - \beta \cdot (\tau' - 1) = k' - \beta \cdot (\tau' - 1) \geq 1$ outputs remain; let C_v be any such 0-list. As the conflict relation is symmetric, P_1 is solved.

[6] It may also be helpful to note the similarity between this parameter setting and that in Thm. 2.2.

$P_1 \to P_0$: Since C_v, C_u do not τ-conflict, removing from C_v all the colors from the 0-lists of the outneighbors leaves at least $k - \beta \cdot (\tau - 1) \geq 1$ colors that v can select as $c(v)$. ◀

3.3 Zero Round Solution to P_2

The results in this section hold for parameters τ, τ', k' fixed as in Section 3.2, and for any $\tau \leq k \leq \beta\tau$. While we set $k = \beta\tau$ for solving P_0, we will use another value of k for our defective coloring result (see [42]). Note that we still have the bound $l_0 \geq 2ek^2/\tau$ on list size, for any such k. The goal of this section is to prove the following lemma.

▶ **Lemma 3.2** (P_2 in zero rounds). *Under the assumptions of Thm. 1.2, the problem $P_2(\tau, k, \tau', k')$ can be solved in zero rounds.*

To prove Lemma 3.2, we reduce (without communication) an instance of P_2 to a conflict coloring instance $\mathfrak{P}_2$ that can be solved in zero rounds with Lemma 2.1.

Reducing P_2 to a conflict coloring instance $\mathfrak{P}_2$ (without communication): Given input lists $(L_v)_{v \in V}$ and parameters $0 < \tau \leq k \leq l_0$ and $0 < \tau' \leq k$, the conflict coloring instance $\mathfrak{P}_2$ is given by the colorspace $\mathcal{P}^{(2)}(\mathcal{C})$, the (τ', τ)-conflict relation $\mathcal{R}_2$ on 1-lists, the list family $\mathcal{F}_2 = \mathrm{Im}(L_2) = \{L_2(S) \mid S \in \binom{\mathcal{C}}{l_0}\}$ and list $\mathcal{L}(v) = L_2(L_v)$ for node v, where $L_2 : \binom{\mathcal{C}}{l_0} \to \mathcal{P}^{(3)}(\mathcal{C})$ maps l_0-sized subsets of $\mathcal{C}$ to 2-lists and is defined below. The map L_2, the colorspace, the conflict relation and the set family $\mathcal{F}_2$ are global knowledge and no communication is needed to compute the list $\mathcal{L}(v)$ of a node in $\mathfrak{P}_2$.

To define the map L_2 we need another definition. For an integer $t \geq 0$ and a 2-list T, a 1-list $K \in T$ is (T, t, τ', τ)-*good* if there are less than t 1-lists $K' \in T$ such that K and K' do (τ', τ)-conflict. We define maps L_1, $\bar{L}_2$ and L_2, as follows. For $S \in \binom{\mathcal{C}}{l_0}$,

$$L_1(S) = \binom{S}{k} \qquad \text{(elements } C \text{ are 0-lists)}$$

$$\bar{L}_2(S) = \binom{L_1(S)}{k'} \qquad \text{(elements } K \text{ are 1-lists)}$$

$$L_2(S) = \{K \in \bar{L}_2(S) |\ K \text{ is } (\bar{L}_2(S), d_2, \tau', \tau)\text{-good}\} \qquad \text{(elements } K \text{ are 1-lists)},$$

where d_2 is chosen as in Lemma 3.4.[7] Due to the definition of the (τ', τ)-conflict relation and the map L_2, solving $\mathfrak{P}_2$ immediately solves P_2.

The sizes of $L_1(S), \bar{L}_2(S)$ and $L_2(S)$ do not depend on S. Let $l_1 = |L_1(S)| = \binom{l_0}{k}$, and $l_2 = |\bar{L}_2(S)|/2 = \binom{l_1}{k'}/2$. We will later show that $|L_2(S)| \geq l_2$. Let $\mathcal{F}_1 = \{L_1(S) \mid S \in \binom{\mathcal{C}}{l_0}\}$.

Some intuition: In the conflict coloring instance $\mathfrak{P}_2$, every node v has a list $\{K_1, K_2, \ldots\}$ of 1-lists, each a collection of subsets of its input list L_v. To ensure small conflict degree, but still large list size, it is enough that v only takes Ks that are "good", as defined above. Since being "good" only depends on L_v, node v can also compute its $\mathfrak{P}_2$-list locally.

In Lemmas 3.3 to 3.5, we show that lists $L_2(L_v)$ are large and that $\mathfrak{P}_2$ has small conflict degree. Before that, let us see how these lemmas imply 0-round solvability of P_2 (Lemma 3.2).

Proof of Lemma 3.2. To solve an instance of P_2 on input lists $(L_v)_{v \in V}$, nodes locally set up the conflict coloring instance $\mathfrak{P}_2$. Lemmas 3.3 and 3.4 show that the conflict degree of $\mathfrak{P}_2$ is bounded by $d_{\mathcal{R}_2}(\mathcal{F}_2) \leq d_2$, and that every list in $\mathcal{F}_2$ has size at least l_2. Note that $\mathcal{F}_2$ is globally known and $|\mathcal{F}_2| = \binom{|\mathcal{C}|}{l_0} < |\mathcal{C}|^{l_0}$, since each element in $\mathcal{F}$ can be written as $L_2(S)$

[7] The precise value is not important to understand how L_2 is formed.

for some $S \in \binom{\mathcal{C}}{l_0}$. Using Lemma 3.5 we obtain $l_2/d_2 \geq \frac{1}{8}2^{2^{\tau-\log(4e\beta^2)}} \geq m \cdot |\mathcal{C}|^{l_0} > m \cdot |\mathcal{F}_2|$, where the second inequality follows by a routine calculation using the definition of τ and l_0 (see [42]). Thus, Lemma 2.1 holds, and $\mathfrak{P}_2$ and P_2 can be solved in zero rounds. ◀

We continue with proving Lemmas 3.3 to 3.5. First, we bound the conflict degree of $\mathfrak{P}_2$. Recall that it is a property of the list family $\mathcal{F}_2$ and the conflict relation $\mathcal{R}_2$, and is independent of the graph and list assignment. The proof involves establishing an isomorphism between $L_2(S)$ and $L_2(S')$, for any S, S', which preserves their common elements. For this, it is crucial to have $|S| = |S'|$. This is why we need all input lists to have same size $|L_v| = l_0$.

▶ **Lemma 3.3** (Conflict Degrees). *Let $X, Y \in \binom{\mathcal{C}}{l_0}$ be 0-lists. Let $d_1 = \binom{k}{\tau} \cdot \binom{l_0-\tau}{k-\tau}$.*

1. *For any 0-list $C \in L_1(X)$, there are at most d_1 0-lists in $L_1(Y)$ that τ-conflict with C.*
2. *For any 1-list $K \in L_2(X)$, there are at most d_2 1-lists in $L_2(Y)$ that (τ', τ)-conflict with K. In particular, $d_{\mathcal{R}_2}(\mathcal{F}_2) \leq d_2$, and this holds irrespective of the value of d_2.*

Proof. The proof of the first claim is along the same lines as the proof of Claim 2.3, so we only prove the second claim here. Let $X_1 = L_1(X), X_2 = L_2(X)$ and $\bar{X}_2 = \bar{L}_2(X)$, and define $Y_1, Y_2, \bar{Y}_2$ similarly. As $|X| = |Y|$, there is a bijection $\alpha : X \to Y$ that is the identity on $X \cap Y$: if $c \in X \cap Y$ then $\alpha(c) = c$. Further, since $X_1 = \binom{X}{k}$ and $Y_1 = \binom{Y}{k}$, we have the bijection $\beta : X_1 \to Y_1$ given by $\beta(\{c_1, \ldots, c_k\}) = \{\alpha(c_1), \ldots, \alpha(c_k)\}$, and since $\bar{X}_2 = \binom{X_1}{k'}$ and $\bar{Y}_2 = \binom{Y_1}{k'}$, we have the bijection $\gamma : \bar{X}_2 \to \bar{Y}_2$, where $\gamma(\{C_1, \ldots, C_{k'}\}) = \{\beta(C_1), \ldots, \beta(C_{k'})\}$.

We show that the claim holds for any $t \geq 0$ and for any $K \in \bar{X}_2$ that is $(\bar{X}_2, t, \tau', \tau)$-good (which demonstrates that the actual value of d_2 is irrelevant). As $Y_2 \subseteq \bar{Y}_2$, it suffices to show that K does (τ', τ)-conflict with at most t 1-lists in $\bar{Y}_2$. Towards a contradiction, let $K \in \bar{X}_2$ (τ', τ)-conflict with each of t distinct 1-lists $K'_1, K'_2, \ldots, K'_t \in \bar{Y}_2$ and define $K_i = \gamma^{-1}(K'_i) \in \bar{X}_2$. We show that K also (τ', τ)-conflicts with each of the distinct (γ is a bijection) $K_1, \ldots, K_t \in \bar{X}_2$, which is a contradiction to K being $(\bar{X}_2, t, \tau', \tau)$-good: To ease notation, let us focus on K and K_1. Assume there are τ' distinct (case 2: not necessarily distinct) 0-lists $C'_1, C'_2, \ldots, C'_{\tau'}$ in K'_1, and τ' not necessarily distinct (case 2: distinct) 0-lists $C_1, C_2, \ldots, C_{\tau'}$ in K, such that C_i and C'_i τ-conflict. Then $\beta^{-1}(C'_i)$ and C_i τ-conflict, since α is the identity on $C_i \cap C'_i$, $\beta^{-1}(C'_i)$ are all distinct (since β is a bijection) and belong to K_1, therefore K and K_1 (τ', τ)-conflict. ◀

Next, we show that at most half of the elements $K \in \bar{L}_2$ fail to be good; this lemma crucially depends on the value of d_2. Below, we use the conflict degree d_1 from Lemma 3.3.

▶ **Lemma 3.4** (L_2 is large). *Let $d_2 = 4\binom{k'd_1}{\tau'} \cdot \binom{l_1-\tau'}{k'-\tau'}$. For any $S \in \binom{\mathcal{C}}{l_0}$, we have $|L_2(S)| \geq l_2$.*

Proof. Fix $S \in \binom{\mathcal{C}}{l_0}$ and consider the digraph $H = (V_H, E_H)$ over the vertex set $V_H = \bar{L}_2(S)$, where $(K, K') \in E_H$ iff K contains at least τ' lists, each in τ-conflict with a list in K' (in particular, for every K, $(K, K) \in E_H$). Note that a 1-list K is $(\bar{L}_2(S), d_2, \tau', \tau)$-good iff its undirected degree in H is at most d_2.

▷ Claim. The maximum outdegree of a node $K \in V_H$ is at most $d_2/4$.

Proof. Consider a fixed $K \in V_H$. Let $X \subseteq K$ be the set of 0-lists in $L_1(S)$ that τ-conflict with a 0-list in K. By Lemma 3.3 part 1, every $C \in K$ τ-conflicts with at most d_1 of 0-lists, hence $|X| \leq |K| \cdot d_1 = kd_1$. Every 1-list K', such that there are at least τ' 0-lists in K that are in τ-conflict with a 0-list in K', can be obtained by first choosing τ' 0-lists from X, and adding an arbitrary subset of $k' - \tau'$ other 0-lists. Clearly, this can be done in at most $\binom{k'd_1}{\tau'} \cdot \binom{l_1-\tau'}{k'-\tau'} = d_2/4$ many ways. ◁

The Claim implies that $|E_H| \leq |V_H| \cdot d_2/4$, hence the undirected average degree of a node in H is at most $2|E_H|/|V_H| \leq d_2/2$, and by Markov's inequality, at most half of the nodes have degree greater than d_2. Since $L_2(S)$ is the set of nodes of degree at most d_2, we conclude that $|L_2(S)| \geq |V_H|/2 = |\bar{L}_2(S)|/2 = l_2$. ◀

Finally, we bound the ratio l_2/d_2 based on the values of the remaining parameters.

▶ **Lemma 3.5** (l/d Ratio). *If $k \geq \tau \geq \lceil \log(2e\beta^2) \rceil$, $l_0 \geq 2ek^2/\tau$, $\tau' = 2^{\tau - \lceil \log(2e\beta^2) \rceil}$, and $k' = \beta\tau'$, then $l_2/d_2 > 2^{2^{\tau - \log(4e\beta^2)}}/8$.*

Proof. First, we get $l_1/d_1 \geq 2^\tau$, as in Eq. (1). Then, with $\binom{k'd_1}{\tau'} \leq (\frac{ekd_1}{\tau'})^{\tau'}$, and (2) applied to $\binom{l_1}{k'}/\binom{l_1-\tau'}{k'-\tau'}$, we lower bound l_2/d_2 as

$$\frac{l_2}{d_2} = \frac{1}{8}\frac{\binom{l_1}{k'}}{\binom{l_1-\tau'}{k'-\tau'}\binom{k'd_1}{\tau'}} > \frac{1}{8}\left(\frac{l_1}{k'} \cdot \frac{\tau'}{e(k'd_1)}\right)^{\tau'} \geq \frac{1}{8}\left(\frac{2^\tau}{e\beta^2\tau'}\right)^{\tau'} \geq \frac{2^{\tau'}}{8} \geq \frac{2^{2^{\tau - \log(4e\beta^2)}}}{8},$$

where the third and fourth inequalities hold since $\tau' \leq \frac{2^\tau}{2e\beta^2} \leq 2\tau'$. ◀

3.4 Proof of the Main Theorem

Proof of Thm. 1.2. Nodes solve P_2 in zero rounds (Lemma 3.2), and then use two rounds of communication to solve the input list coloring problem P_0 (see algorithm description and Lemma 3.1). We bound the messages sent by a node v during the algorithm. In the first round, v needs to send K_v to its neighbors. Note that K_v is uniquely determined by the list L_v and the input color ψ_v (see the proof of Lemma 2.1) , so it suffices to send (ψ_v, L_v), which can be encoded in $l_0\lceil \log |\mathcal{C}| \rceil + \lceil \log m \rceil$ bits. In the second round, v needs to send C_v. Since $C_v \in K_v$, and the neighbors know K_v, it suffices to send the index of C_v in K_v (in a fixed ordering). Recall that $|K_v| = k' < 2^\tau$, so v only needs to send $\tau \leq l/4e\beta^2$ bits. ◀

▶ Remark 3.6. Note that in both communication rounds of Theorem 1.2 each node only needs to send messages to its in-neighbors. In contrast, the results in Section 4 and the defective coloring results require bi-directional communication.

4 Application: $(\Delta + 1)$-Coloring and $(deg + 1)$-List Coloring

▶ **Theorem 1.3** ($(deg + 1)$-List Coloring). *In a graph with max. degree Δ, $(deg + 1)$-list coloring with lists $L_v \subseteq \mathcal{C}$ from a color space of size $|\mathcal{C}| = 2^{\text{poly}(\Delta)}$ [8] can be solved in $O\big(\sqrt{\Delta \log \Delta}\big) + \frac{1}{2} \cdot \log^* n$ rounds in LOCAL. Furthermore, each node only needs to broadcast to its neighbors a single non-CONGEST message consisting of a subset of its list.*

The proof combines Thm. 1.2 with the graph partitioning provided by [13], following the high level description in Sec. 1. A variant of this framework was also used in [37, 6]. We nevertheless present a proof for completeness, and also due to subtle but important differences from [23] (we have an additional finishing phase that is not present there).

The graph partitioning given by [13] aims at *arbdefective colorings*, as introduced in [10], but the main technical object provided by [13] (and which is all we need here) is a *low outdegree partition* of a graph. For a graph $H = (V, E)$, the collection $H_1, \ldots, H_k$ of *directed* graphs $H_i = (V_i, E_i)$ is a *β-outdegree partition* of H if their vertices span V, i.e.,

[8] We use the notation $\text{poly}(X) = O(X^c)$, for an absolute constant c, and $\tilde{O}(X) = X \cdot \text{poly}(\log X)$.

$V = V_1 \cup \cdots \cup V_k$, the underlying undirected graph of H_i is the induced subgraph $H[V_i]$ (so it is indeed a partition), and the max. outdegree of a node in H_i is at most β, for all i.

▶ **Lemma 4.1** (Lemmas 6.1-6.3, [13]). *There are constants $c, c' > 0$, s.t. for every $\beta \geq c$, given a graph H with an m-coloring, there is a deterministic algorithm that computes a β-outdegree partition $H_1, \ldots, H_k$ with $k = c'\Delta/\beta$ in $O(k + \log^* m)$ rounds in CONGEST.*

Proof of Thm. 1.3. We begin with computing an $m = O(\Delta^2)$-coloring in $\frac{1}{2}\log^* n + O(1)$ rounds [41, 49]. The main algorithm consists of $t = \log_2(\Delta/\Delta^{1/4})$ phases. After phase j, we have colored a subset of vertices, s.t. the maximum degree Δ_j of the graph $G[U_j]$ induced by uncolored vertices is upper bounded as $\Delta_j \leq \Delta/2^j$. Before describing a phase j, let us show how we finish the coloring after the phase t, in a **final phase**. Consider the graph $G[U_t]$ at the end of phase t. Note that it has maximum degree $\Delta_t = O(\Delta^{1/4})$. We compute an m'-coloring of $G[U_t]$ with $m' = O(\Delta_t^2) = O(\sqrt{\Delta})$ from the initial m-coloring in $O(1)$ rounds [41]. In each of the final m' rounds $i = 1, \ldots, m'$, vertices with color i pick a color from their list not picked by a neighbor (can be done since $|L_v| > \Delta$ and no two neighbors pick simultaneously). The runtime of the final phase is $O(\sqrt{\Delta})$.

The following happens in **phase** $j = 1, \ldots, t$. At the beginning of the phase, we have the set U_{j-1} of uncolored vertices, where $U_0 = V(G)$. Let $X = 4e \cdot (4\log\Delta + \log\log|\mathcal{C}| + \log\log m + 8) = O(\log\Delta)$ and for $j = 0, \ldots, t-1$ let $\beta_j = \sqrt{\Delta_j/(2X)}$ and $k_j = c' \cdot \Delta_j/\beta_j$, where c' is the constant in Lemma 4.1.[9] We partition $G[U_{j-1}]$ into β_j-outdegree subgraphs $H_1, H_2, \ldots, H_{k_j}$, using Lemma 4.1. The phase consists of k_j stages $i = 1, \ldots, k_j$, each consisting of 3 rounds. In **stage** i, we partially color H_i, as follows. For every uncolored vertex $v \in H_i$, let $L_{v,j,i}$ be the set of colors in L_v that have not been taken by a neighbor of v. Let $W_i = \{v \in H_i : |L_{v,j,i}| \geq \beta_j^2 X\}$. Color the graph $H_i[W_i]$ using Linial for Lists (Thm. 1.2) with color space $\mathcal{C}$, the β_j-outdegree orientation and the m-coloring. This is a valid application of the theorem, by the definition of X, β_j and W_i. In the third round of the stage, all nodes in W_i send their color to their neighbors. This completes the algorithm description. Clearly, phase j takes $3k_j$ rounds.

It remains to show that $\Delta_j \leq \Delta/2^j$. We do this by induction, with base $j = 0$, $\Delta_0 = \Delta$. Assume $\Delta_j \leq \Delta/2^j$ holds for some $j \geq 0$. Let $v \in U_j$ be a node that is uncolored at the end of phase j. We know that $|L_{v,j,i}| < \beta_j^2 X = \Delta_j/2$, in a stage i. Recall that $L_{v,j,i}$ is the set of colors in L_v not taken by a neighbor of v. Since $|L_v|$ is larger than the number of neighbors of v, $|L_{v,j,i}|$ is larger than the number of *uncolored* neighbors of v. Therefore v has at most $|L_{v,j,i}| < \Delta_j/2 \leq \Delta/2^{j+1}$ neighbors in U_j, which proves the induction: $\Delta_{j+1} \leq \Delta/2^{j+1}$.

Recall that $X = O(\log\Delta)$ and bound the runtime as follows:

$$\frac{1}{2}\log^* n + O(1) + \sum_{j=1}^{t} 3k_j + m' = \frac{1}{2}\log^* n + \sum_{j=1}^{t} 3c'\sqrt{\frac{X\Delta}{2^{j-1}}} + O(\sqrt{\Delta}) = \frac{1}{2}\log^* n + O(\sqrt{\Delta\log\Delta}) \,.$$

The second claim easily follows, recalling the message complexity of Linial for Lists. ◀

Note that the final phase in the algorithm above is necessary as otherwise, if the recursion continued until the maximum degree of uncolored nodes was, say, $O(\log^{(3)}\Delta)$, their reduced list size would be similarly small, and we could no longer apply Theorem 1.2, which requires lists of size $\Omega(\log\log|\mathcal{C}|) = \Omega(\log\log\Delta)$, as the color space does not change in the recursion.

The (simple) proof of the following corollary is deferred to the full version [42].

---

[9] In order to apply the lemma, we need $\beta_j \geq c$. Since $\beta_j \geq \beta_t = \Omega(\sqrt{\sqrt{\Delta}/\log\Delta})$, $\beta_j \geq c$ holds if Δ is large enough. For $\Delta = O(1)$, Thm. 1.3 holds via a $O(\Delta) + 1/2\log^* n$ round algorithm (see e.g. [13]).

▶ **Corollary 4.2.** *In a graph with max. degree $\Delta = \tilde{O}(\log n)$, $(deg+1)$-list coloring with lists $L_v \subseteq \mathcal{C}$ from a color space of size $|\mathcal{C}| = \text{poly}(\Delta)$ can be solved in $\tilde{O}(\sqrt{\Delta}) + \frac{1}{2} \cdot \log^* n$ rounds in CONGEST.*

5 Discussion

We conclude with several observations on our results, as well as open problems.

1. It is possible to define problems $P_3, \ldots, P_t$ for any t, as we defined P_1 and P_2. Lemma 3.3 extends naturally to these problems, so the input of a node v in P_i is again only its initial list L_v. We need t rounds, instead of 2, to derive a solution of P_0 from a solution of P_t (which also implies larger messages). On the other hand, we have somewhat smaller list size requirement: $c\beta^2(\log\beta + \log^{(t)}|\mathcal{C}| + \log^{(t)} m)$, for a constant $c > 0$. In particular, one can list color in $O(\log^* \max\{|\mathcal{C}|, m\})$ rounds if lists are at least $c\beta^2 \log\beta$ for a sufficiently large constant $c > 0$.
2. Unlike in [23], our bound on the list size does not depend on Δ. This result implies that, e.g., given a graph with a β-outdegree orientation and an input coloring with $2^{\text{poly}(\beta)}$ colors and list sizes of at least $c\beta^2 \cdot \log\beta$ from a color space of size $2^{\text{poly}(\beta)}$, for a constant $c > 0$, it is possible to list-color the graph in 2 rounds. By the remark above, one can have even larger color space, by increasing the runtime accordingly.
3. A lower bound in [49] suggests that the coloring in Theorem 1.2 cannot be done in a single round. In particular, if one is willing to keep the doubly-logarithmic dependence on m in the list size, then one has to pay a factor exponential in β. On the other hand, we do not know how to eliminate the $\log\beta$ term, even if we use more communication.
4. The recently popular *speedup* technique has mostly been used to prove lower bounds, e.g., [16, 17, 3, 18, 4, 2]. Here, a problem P_0 is mechanically (and without communication!) transformed into a problem P_1 whose complexity is exactly one round less. Then, if P_1 cannot be solved locally one deduces that P_0 cannot be solved in 1 round. By iterating this process, one can derive larger lower bounds. However, the description complexity of derived problems grows exponentially, and it is very important to be able to simplify the problem description, in order to iterate the process. If P_0 is the $(\Delta+1)$-vertex coloring problem, this process has only been understood in the special case of $\Delta = 2$, which corresponds to Linial's $\Omega(\log^* n)$ lower bound [41, 40]. While [23] also performs a similar transformation, it is different from the speedup technique, since the transformation is not mechanical, requiring nodes to communicate for building the new problems. It may rather be seen as a transformation of *problem instances* (that depend on the graph) than problems. In contrast, our transformations are mechanical, and the input and output labels live in the same universe as it is the case for mechanical speedup.
5. While our present treatment of the proof of Thm. 1.2 in terms of conflict coloring problems and problems P_0, P_1 and P_2 has the aim of connecting to the framework of [23] as well as to the speedup framework, we note that the proof can be stated entirely in terms of set systems, just like the proof of Linial's color reduction.

Open Problem: Remove the $\log\beta$ term in Thm. 1.2 while keeping the runtime $o(\sqrt{\log\beta})$. This question is particularly of interest because $\log\beta$ is the source of the $\sqrt{\log\Delta}$ factor in in Thm. 1.3 (note that the terms depending on $m, |\mathcal{C}|$ can be reduced, by the remarks above). The non-list $O(\Delta^2)$-coloring by Linial uses, in addition to his main color reduction, a $O(\Delta^3)$-to-$O(\Delta^2)$ color reduction, using polynomials over finite fields [41]. With a more sophisticated use of polynomials [8] constructs a cover-free family for list coloring but it requires a much smaller outdegree. It is not clear if polynomials help with our question.

- **Open Problem:** More generally, prove or rule out a truly local $(\Delta+1)$-coloring algorithm with Δ-dependence $f(\Delta) = o(\sqrt{\Delta})$.

References

1. Baruch Awerbuch, Andrew V. Goldberg, Michael Luby, and Serge A. Plotkin. Network decomposition and locality in distributed computation. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, pages 364–369, 1989.
2. Alkida Balliu, Sebastian Brandt, Yuval Efron, Juho Hirvonen, Yannic Maus, Dennis Olivetti, and Jukka Suomela. Classification of distributed binary labeling problems. In *Proceedings of the International Symposium on Distributed Computing (DISC)*, 2020.
3. Alkida Balliu, Sebastian Brandt, Juho Hirvonen, Dennis Olivetti, Mikaël Rabie, and Jukka Suomela. Lower bounds for maximal matchings and maximal independent sets. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, pages 481–497, 2019.
4. Alkida Balliu, Sebastian Brandt, and Dennis Olivetti. Distributed lower bounds for ruling sets. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, 2020.
5. Alkida Balliu, Juho Hirvonen, Christoph Lenzen, Dennis Olivetti, and Jukka Suomela. Locality of not-so-weak coloring. In *Proceedings of the International Colloquium on Structural Information and Communication Complexity (SIROCCO)*, pages 37–51, 2019.
6. Alkida Balliu, Fabian Kuhn, and Dennis Olivetti. Distributed edge coloring in time quasi-polylogarithmic in delta. In *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, 2020.
7. Philipp Bamberger, Fabian Kuhn, and Yannic Maus. Efficient deterministic distributed coloring with small bandwidth. In *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, 2020.
8. Leonid Barenboim. Deterministic $(\Delta + 1)$-coloring in sublinear (in Δ) time in static, dynamic, and faulty networks. *Journal of the ACM*, 63(5):47:1–47:22, 2016.
9. Leonid Barenboim and Michael Elkin. Sublogarithmic distributed MIS algorithm for sparse graphs using Nash-Williams decomposition. *Distributed Comput.*, 22(5-6):363–379, 2010.
10. Leonid Barenboim and Michael Elkin. Deterministic distributed vertex coloring in polylogarithmic time. *Journal of the ACM*, 58(5):23:1–23:25, 2011.
11. Leonid Barenboim and Michael Elkin. Distributed deterministic edge coloring using bounded neighborhood independence. In *Proceedings of the 30th Annual ACM Symposium on Principles of Distributed Computing, PODC 2011, San Jose, CA, USA, June 6-8, 2011*, pages 129–138, 2011. doi:10.1145/1993806.1993825.
12. Leonid Barenboim and Michael Elkin. *Distributed Graph Coloring: Fundamentals and Recent Developments*. Morgan & Claypool Publishers, 2013.
13. Leonid Barenboim, Michael Elkin, and Uri Goldenberg. Locally-Iterative Distributed $(\Delta+1)$-Coloring below Szegedy-Vishwanathan Barrier, and Applications to Self-Stabilization and to Restricted-Bandwidth Models. In *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, pages 437–446, 2018.
14. Leonid Barenboim, Michael Elkin, and Fabian Kuhn. Distributed (Delta+1)-Coloring in Linear (in Delta) Time. *SIAM J. Comput.*, 43(1):72–95, 2014.
15. Leonid Barenboim, Michael Elkin, Seth Pettie, and Johannes Schneider. The locality of distributed symmetry breaking. *Journal of the ACM*, 63(3):20:1–20:45, 2016.
16. Sebastian Brandt. An automatic speedup theorem for distributed problems. In *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, pages 379–388, 2019.
17. Sebastian Brandt, Orr Fischer, Juho Hirvonen, Barbara Keller, Tuomo Lempiäinen, Joel Rybicki, Jukka Suomela, and Jara Uitto. A lower bound for the distributed lovász local lemma. In *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, pages 479–488, 2016.

18 Sebastian Brandt and Dennis Olivetti. Truly tight-in-δ bounds for bipartite maximal matching and variants. In *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, 2020.

19 Yi-Jun Chang, Tsvi Kopelowitz, and Seth Pettie. An exponential separation between randomized and deterministic complexity in the LOCAL model. *SIAM J. Comput.*, 48(1):122–143, 2019.

20 Yi-Jun Chang, Wenzheng Li, and Seth Pettie. An optimal distributed (Δ+1)-coloring algorithm? In *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, pages 445–456, 2018.

21 Paul Erdös, Peter Frankl, and Zóltan Füredi. Families of finite sets in which no set is covered by the union of r others. *Israel Journal of Mathematics*, 51:79–89, 1985.

22 Manuela Fischer, Mohsen Ghaffari, and Fabian Kuhn. Deterministic distributed edge-coloring via hypergraph maximal matching. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, pages 180–191, 2017.

23 Pierre Fraigniaud, Marc Heinrich, and Adrian Kosowski. Local conflict coloring. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, pages 625–634, 2016.

24 Pierre Fraigniaud and Ami Paz. The topology of local computing in networks. In *Proceedings of the International Colloquium on Automata, Languages, and Programming (ICALP)*, pages 128:1–128:18, 2020. doi:10.4230/LIPIcs.ICALP.2020.128.

25 Cyril Gavoille, Ralf Klasing, Adrian Kosowski, Lukasz Kuszner, and Alfredo Navarra. On the complexity of distributed graph coloring with local minimality constraints. *Networks*, 54(1):12–19, 2009.

26 Mohsen Ghaffari, David G. Harris, and Fabian Kuhn. On derandomizing local distributed algorithms. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, pages 662–673, 2018.

27 Mohsen Ghaffari, Juho Hirvonen, Fabian Kuhn, and Yannic Maus. Improved distributed delta-coloring. In *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, pages 427–436, 2018.

28 Mohsen Ghaffari, Juho Hirvonen, Fabian Kuhn, Yannic Maus, Jukka Suomela, and Jara Uitto. Improved distributed degree splitting and edge coloring. In *Proceedings of the International Symposium on Distributed Computing (DISC)*, pages 19:1–19:15, 2017.

29 Mohsen Ghaffari, Fabian Kuhn, and Yannic Maus. On the complexity of local distributed graph problems. In *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, pages 784–797, 2017.

30 Mohsen Ghaffari, Fabian Kuhn, Yannic Maus, and Jara Uitto. Deterministic distributed edge-coloring with fewer colors. In *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, pages 418–430, 2018.

31 Mohsen Ghaffari and Hsin-Hao Su. Distributed degree splitting, edge coloring, and orientations. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2505–2523, 2017.

32 David G. Harris. Distributed local approximation algorithms for maximum matching in graphs and hypergraphs. In *Proceedings of the Symposium on Foundations of Computer Science (FOCS)*, pages 700–724, 2019.

33 David G. Harris, Johannes Schneider, and Hsin-Hao Su. Distributed (Δ +1)-coloring in sublogarithmic rounds. *Journal of the ACM*, 65(4):19:1–19:21, 2018.

34 Dan Hefetz, Fabian Kuhn, Yannic Maus, and Angelika Steger. Polynomial lower bound for distributed graph coloring in a weak LOCAL model. In *Proceedings of the International Symposium on Distributed Computing (DISC)*, pages 99–113, 2016.

35 Frank K. Hwang and Vera T. Sós. Non-adaptive hypergeometric group testing. *Studia scient. Math. Hungaria*, 22:257–263, 1987.

36 Fabian Kuhn. Weak graph colorings: distributed algorithms and applications. In *Proceedings of the ACM Symposium on Parallelism in Algorithms and Architecture (SPAA)*, pages 138–144, 2009.

37 Fabian Kuhn. Faster deterministic distributed coloring through recursive list coloring. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1244–1259, 2020.

38 Fabian Kuhn, Thomas Moscibroda, and Roger Wattenhofer. Local computation: Lower and upper bounds. *Journal of the ACM*, 63(2), 2016.

39 Fabian Kuhn and Roger Wattenhofer. On the complexity of distributed graph coloring. In *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, pages 7–15, 2006.

40 Juhana Laurinharju and Jukka Suomela. Brief announcement: Linial's lower bound made easy. In *Proceedings of the ACM Symposium on Principles of Distributed Computing (PODC)*, pages 377–378, 2014.

41 Nathan Linial. Locality in distributed graph algorithms. *SIAM J. Comput.*, 21(1):193–201, 1992.

42 Yannic Maus and Tigran Tonoyan. Local conflict coloring revisited: Linial for lists, 2020. `arXiv:2007.15251`.

43 Moni Naor. A lower bound on probabilistic algorithms for distributive ring coloring. *SIAM J. Discret. Math.*, 4(3):409–412, 1991.

44 Alessandro Panconesi and Romeo Rizzi. Some simple distributed algorithms for sparse networks. *Distributed Comput.*, 14(2):97–100, 2001.

45 Alessandro Panconesi and Aravind Srinivasan. Improved distributed algorithms for coloring and network decomposition problems. In *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, pages 581–592, 1992.

46 David Peleg. *Distributed Computing: A Locality-Sensitive Approach*. SIAM, 2000.

47 Václav Rozhon and Mohsen Ghaffari. Polylogarithmic-time deterministic network decomposition and distributed derandomization. In *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, pages 350–363, 2020.

48 Hsin-Hao Su and Hoa T. Vu. Towards the locality of vizing's theorem. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 355–364, 2019.

49 Mario Szegedy and Sundar Vishwanathan. Locality based graph coloring. In *Proceedings of the ACM Symposium on Theory of Computing (STOC)*, pages 201–207, 1993.