

Towards Optimal Distributed Edge Coloring with Fewer Colors

Manuel Jakob  

TU Graz, Austria

Yannic Maus  

TU Graz, Austria

Florian Schager  

TU Graz, Austria

Abstract

There is a huge difference in techniques and runtimes of distributed algorithms for problems that can be solved by a sequential greedy algorithm and those that cannot. A prime example of this contrast appears in the edge coloring problem: while $(2\Delta - 1)$ -edge coloring – where Δ is the maximum degree – can be solved in $\mathcal{O}(\log^*(n))$ rounds on constant-degree graphs, the seemingly minor reduction to $(2\Delta - 2)$ colors leads to an $\Omega(\log n)$ lower bound [Chang, He, Li, Pettie & Uitto, SODA’18]. Understanding this sharp divide between very local problems and inherently more global ones remains a central open question in distributed computing and it is a core focus of this paper.

As our main contribution we design a deterministic distributed $\mathcal{O}(\log n)$ -round reduction from the $(2\Delta - 2)$ -edge coloring problem to the much easier $(2\Delta - 1)$ -edge coloring problem. This reduction is optimal, as the $(2\Delta - 2)$ -edge coloring problem admits an $\Omega(\log n)$ lower bound that even holds on the class of constant-degree graphs, whereas the $2\Delta - 1$ -edge coloring problem can be solved in $\mathcal{O}(\log^* n)$ rounds. By plugging in the $(2\Delta - 1)$ -edge coloring algorithms from [Balliu, Brandt, Kuhn & Olivetti, PODC’22] running in $\mathcal{O}(\log^{12} \Delta + \log^* n)$ rounds, we obtain an optimal runtime of $\mathcal{O}(\log n)$ rounds as long as $\Delta = 2^{\mathcal{O}(\log^{1/12} n)}$. Previously, such an optimal algorithm was only known for the class of constant-degree graphs [Brandt, Maus, Narayanan, Schager & Uitto, SODA’25]. Furthermore, on general graphs our reduction improves the runtime from $\tilde{\mathcal{O}}(\log^3 n)$ to $\tilde{\mathcal{O}}(\log^{5/3} n)$.

In addition, we also obtain an optimal $\mathcal{O}(\log \log n)$ -round randomized reduction of $(2\Delta - 2)$ -edge coloring to $(2\Delta - 1)$ -edge coloring. This leads to a $\tilde{\mathcal{O}}(\log^{5/3} \log n)$ -round $(2\Delta - 2)$ -edge coloring algorithm, which beats the (very recent) previous state-of-the-art taking $\tilde{\mathcal{O}}(\log^{8/3} \log n)$ rounds from [Bourreau, Brandt & Nolin, STOC’25].

Lastly, we obtain an $\mathcal{O}(\log_\Delta n)$ -round reduction from the $(2\Delta - 1)$ -edge coloring, albeit to the somewhat harder maximal independent set (MIS) problem.

2012 ACM Subject Classification Theory of computation → Distributed algorithms

Keywords and phrases distributed graph algorithms, edge coloring, LOCAL model

Digital Object Identifier 10.4230/LIPIcs.DISC.2025.37

Related Version *Full Version:* <https://arxiv.org/pdf/2504.13003> [51]

Funding This research was funded in whole or in part by the Austrian Science Fund (FWF) <https://doi.org/10.55776/P36280>, <https://doi.org/10.55776/I6915>.

Acknowledgements For open access purposes, the author has applied a CC BY public copyright license to any author-accepted manuscript version arising from this submission.

1 Introduction

There is a huge difference in the techniques and runtimes of distributed algorithms for locally checkable problems that can be solved by a sequential greedy algorithm and those that cannot. For example, on bounded degree graphs, the former problems have distributed complexity



© Manuel Jakob, Yannic Maus, and Florian Schager;
licensed under Creative Commons License CC-BY 4.0
39th International Symposium on Distributed Computing (DISC 2025).

Editor: Dariusz R. Kowalski; Article No. 37; pp. 37:1–37:26



Leibniz International Proceedings in Informatics

LIPICs Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

$\Theta(\log^* n)$ [24, 54, 60] while the latter require at least $\Omega(\log n)$ time [17, 19]. Understanding this sharp and potentially wide divide between very locally solvable problems and inherently more global ones remains a central open question in distributed computing.

A prime example of this contrast – explained in detail next and forming the main topic of this paper – appears in the edge coloring problem. A *k-edge coloring* of a graph $G = (V, E)$ is a function $\varphi : E \rightarrow \{1, \dots, k\}$ such that no two adjacent edges receive the same color. This problem [60, 25, 28, 31, 4, 16], but also other *greedy problems* like $(\Delta + 1)$ -vertex coloring, maximal independent set, and maximal matching [56, 1, 11, 6, 20, 2, 65, 43, 29, 37], has been extensively studied in the field of distributed computing, particularly in the LOCAL model.

In this model the communication network is abstracted as an undirected graph $G = (V, E)$. Each vertex hosts a processor that may communicate with its neighbors in synchronous rounds. In one round, each processor may send unbounded-size messages to all its neighbors in G and perform unbounded local computations. The goal is to minimize the number of synchronous rounds until every node outputs (a local part of) the solution. The related CONGEST model additionally imposes a maximum message size of $\mathcal{O}(\log n)$ bits per message.

In order to make progress in many parts of the graph in parallel, many distributed algorithms crucially rely on the property that color choices made by other vertices do not affect the solvability on the remaining graph. While this property is satisfied for $k = 2\Delta - 1$ (hence we call this the *greedy threshold*), for any $k < 2\Delta - 1$ there are partial k -edge colorings that cannot be extended to the entire graph [5]. On the other hand of the spectrum, it is known that every graph admits a $(\Delta + 1)$ -edge coloring by a celebrated result of Vizing [69].

The intuition that coloring with fewer colors is significantly more challenging – particularly in the distributed setting – is evidenced by strong lower bounds: Any deterministic edge coloring algorithm using less than $2\Delta - 1$ colors requires at least $\Omega(\log_\Delta n)$ rounds, and even randomized algorithms face an $\Omega(\log_\Delta \log n)$ -round lower bound [17]. On trees, where these lower bounds are established, there is even a deterministic algorithm that solves the greedy version of the problem in strongly sublogarithmic time, specifically in $\mathcal{O}(\log^{12/13} n)$ rounds [16]. On bounded degree graphs, the gap is even larger as the greedy version has deterministic and randomized complexity $\Theta(\log^* n)$, yet the lower bounds for algorithms that use fewer colors – $\Omega(\log n)$ deterministic and $\Omega(\log \log n)$ randomized – still apply.

Despite these challenges, there has been extensive work on designing increasingly faster deterministic and randomized algorithms for edge coloring with fewer than $2\Delta - 1$ colors – both in the LOCAL model [12, 13, 15, 17, 18, 44, 48], the more restricted CONGEST model [44, 47], and in other models such as streaming [5, 23, 66]. The fastest known deterministic distributed algorithm that operates below the greedy threshold runs in $\tilde{\mathcal{O}}(\log^3 n)$ rounds¹ [15], while the fastest randomized algorithm, derived from a recent more general result on Δ -vertex coloring, achieves a runtime of $\tilde{\mathcal{O}}(\log^{8/3} \log n)$ rounds [14]. See the related work section (Section 1.3) for further results on edge coloring with fewer colors.

This work aims to better understand the size and nature of the complexity gap between greedy-solvable problems and those below the greedy threshold. Understanding this latter class of problems is also central to the poorly understood regime in the complexity landscape of local graph problems, see, e.g., [30, 22, 58].

¹ We use the notation $\tilde{\mathcal{O}}(f(n))$ to hide factors polylogarithmic in $f(n)$.

1.1 Our contributions

As our main contribution, we show that the considerably harder $(2\Delta - 2)$ -edge coloring problem can be efficiently reduced to its greedy cousin $(2\Delta - 1)$ -edge coloring² in $\mathcal{O}(\log n)$ rounds of the LOCAL model.

► **Theorem 1.** *There is a deterministic LOCAL algorithm computing a $(2\Delta - 2)$ -edge coloring in any n -vertex graph with maximum degree Δ in $\mathcal{O}(\log n) + T_{2\Delta-1}(n, \Delta - 1)$ rounds.*

Secondly, allowing the use of maximal independent set (MIS) as a subroutine improves the runtime of our reduction at the cost of reducing to a harder problem.

► **Theorem 2.** *There is a deterministic LOCAL algorithm computing a $(2\Delta - 2)$ -edge coloring in any n -vertex graph with maximum degree Δ in $\mathcal{O}(\log_\Delta n) + T_{\text{MIS}}(\Delta^2 \cdot n, \text{poly}(\Delta))$ rounds.*

While the latter result might sound like a strict improvement over the former – especially given the fact that the runtimes of the current state-of-the-art algorithms for $(2\Delta - 1)$ -edge coloring and MIS on general graphs coincide – we argue that the former version is, in fact, the stronger result. A classic reduction from Luby [56] shows that $(2\Delta - 1)$ -edge coloring is no harder than MIS. Consequently, the $\tilde{\mathcal{O}}(\log^{5/3} n)$ deterministic algorithm for MIS [37] also solves the $(2\Delta - 1)$ -edge coloring problem. On the other hand, there is a $\mathcal{O}(\log^{12} \Delta + \log^* n)$ -round $(2\Delta - 1)$ -edge coloring algorithm from [4], which provably cannot be achieved by any MIS algorithm due to the $\Omega(\min(\Delta, \log n / \log \log n))$ lower bound established in [3]. We obtain the following corollary.

► **Corollary 3.** *There are deterministic LOCAL algorithms computing a $(2\Delta - 2)$ -edge coloring in any n -vertex graph with maximum degree Δ within either $\mathcal{O}(\log^{12} \Delta + \log n)$ or $\tilde{\mathcal{O}}(\log^{5/3} n)$ rounds.*

For the class of graphs with maximum degree Δ as large as $2^{\mathcal{O}(\log^{1/12} n)}$, the first point of this corollary yields an optimal runtime of $\mathcal{O}(\log n)$ rounds. Previously, an optimal runtime was only known for the class of bounded-degree graphs [15]. The second point significantly improves the upper bound on the complexity of $(2\Delta - 2)$ -edge coloring on general graphs, which previously stood at $\tilde{\mathcal{O}}(\log^3 n)$ rounds [15]. Additionally, we present an exponentially faster randomized reduction to the $(2\Delta - 1)$ -edge coloring problem.

► **Theorem 4.** *There is a randomized LOCAL algorithm computing a $(2\Delta - 2)$ -edge coloring in any n -vertex graph with maximum degree Δ in $\mathcal{O}(\log \log n) + T_{2\Delta-1}^{\text{R}}(n, \Delta - 1)$ rounds.*

Theorem 4 yields the following corollary by plugging in the algorithms of [4] and [37].

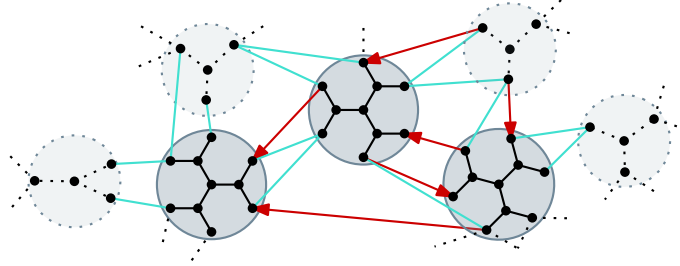
► **Corollary 5.** *There are randomized distributed algorithms computing a $(2\Delta - 2)$ -edge coloring in any n -vertex graph with maximum degree Δ within either $\mathcal{O}(\log^{12} \Delta + \log \log n)$ or $\tilde{\mathcal{O}}(\log^{5/3} \log n)$ rounds of the LOCAL model.*

For the class of graphs with maximum degree Δ as large as $2^{\mathcal{O}(\log^{1/12} \log n)}$, the first point of this corollary yields an optimal runtime of $\mathcal{O}(\log \log n)$ rounds. The second point beats the (very recent) previous state-of-the-art for randomized $(2\Delta - 2)$ -edge coloring on general graphs, which previously stood at $\tilde{\mathcal{O}}(\log^{8/3} \log n)$ rounds [14].

² We use $T_{2\Delta-1}(n, \Delta)$ ($T_{2\Delta-1}^{\text{R}}(n, \Delta)$) to denote the runtime of a deterministic (randomized) $(2\Delta - 1)$ -edge coloring algorithm and $T_{\text{MIS}}(n, \Delta)$ to denote the runtime of a deterministic MIS algorithm for n -vertex graphs with maximum degree Δ .

1.2 Our technique in a nutshell

To illustrate the difficulty of the $(2\Delta - 2)$ -edge coloring problem, let us first try to design an $\mathcal{O}(1)$ -round reduction to the MIS and the $(2\Delta - 1)$ -edge coloring problem; recall that such a reduction provably cannot exist. One approach is to use the MIS algorithm to compute a clustering of the graph, e.g., by computing an MIS of the power graph G^2 and letting each vertex join the cluster of the closest node in the MIS; see Figure 1 for an example. As a result, we can partition the edge set of G into edges E_{intra} (black) within the clusters and inter-cluster edges E_{inter} (turquoise) that go between the different clusters. We could simply color the edges in E_{intra} independently and in parallel for each cluster, as the edges of different clusters are non-adjacent. Due to the constant diameter of each cluster, this can be done in $\mathcal{O}(1)$ rounds and even using the existentially optimal number of colors, i.e., $\Delta + 1$ colors. However, now the problem of coloring E_{intra} is no easier than the original $(2\Delta - 2)$ -edge coloring problem as it essentially corresponds to coloring a general graph but with some colors – the ones used by adjacent edges in E_{intra} – being forbidden for certain edges. In fact, it might not even be possible to complete the coloring at all.



■ **Figure 1** A matching of the intercluster edges that assigns two edges to each clusters. The matching edges are marked red. Arrows point towards the cluster the edge belongs to.

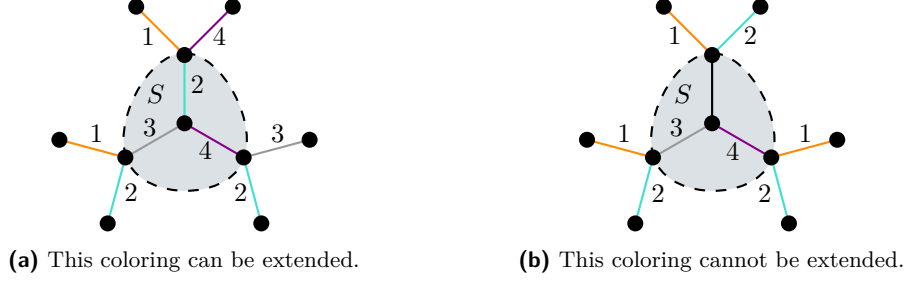
Instead, in our algorithm, we first color the edges between the clusters and then try to complete it within the clusters. First, observe that the graph induced by E_{inter} has maximum degree $\Delta' = \Delta - 1$, as each node has at least one adjacent edge within a cluster. Thus, we can color E_{inter} with one instance of greedy-type edge coloring, using only $2\Delta' - 1 = 2\Delta - 3$ colors. The main challenge that we address in this paper is how to ensure that the coloring can be extended to the edges within the clusters.

For the rest of this technical overview we assume that the graph is Δ -regular with a sufficiently high girth such that each cluster looks like a tree; our full proof does not require this assumption. In addition, let us assume for now that each cluster is a perfect Δ -ary tree, that is, all leaves are at the same level. Then, a simplified version of our main technical contribution gives a useful condition under which we can complete the coloring.

► **Lemma 6** (Simplified version of Lemma 14). *Consider a $(2\Delta - 2)$ -edge colored graph except for a perfect Δ -ary subtree $T \subseteq G$. If the number of distinct colors appearing on the incident edges of any leaf in T combined is at least Δ , then the coloring can be extended to T .*

For the remainder of this paper we will refer to the above condition as the *colorful condition*. To build some intuition for it, consider the simplest possible case, where the set of uncolored edges forms a star graph $S \subseteq G$. The only case, in which φ cannot be extended to S is if each edge in S has the exact same set of $\Delta - 1$ colors appearing among its adjacent edges, which we illustrate in Figure 2. By setting up a bipartite graph with available colors

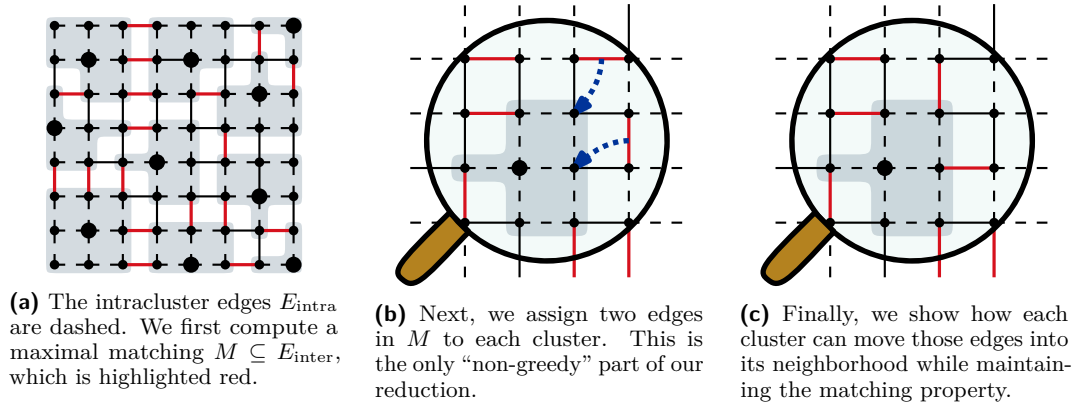
on one side and edges on the opposite side, an application of Hall's marriage theorem shows that this is indeed the only bad case. In the proof of Lemma 14, we generalize this condition from uncolored stars to uncolored trees.



■ **Figure 2** Extendability of $(2\Delta - 2)$ -colorings for $\Delta = 3$.

Next, we turn our attention to how the partial edge coloring of E_{inter} can be modified in order to satisfy the *colorful condition* for every cluster of uncolored edges. The core observation is that each leaf in the cluster already has $\Delta - 1$ incident edges in E_{inter} , whose colors must all be distinct. Therefore, if we can manage to keep the set of colors fixed for one leaf and modify at most one color at another leaf, the colorful condition is already satisfied. Notably, our partial coloring of E_{inter} only uses $2\Delta - 3$ of the $2\Delta - 2$ available colors. As a result, every edge already has at least one free color it can switch to if needed.

The main difficulty now is to construct a set of edges such that each cluster can perform these changes in parallel and without causing any conflicts. Hence, our desired set of edges needs to satisfy two key properties: (1) Each cluster needs exclusive access to two edges in its 1-hop neighborhood. (2) The set of all such edges needs to be a matching. Computing a greedy maximal matching satisfies (2), but not necessarily (1). Therefore, we modify a maximal matching $M \subseteq E_{\text{inter}}$ by letting each cluster send proposals to matching edges in its 2-hop neighborhood.



■ **Figure 3** In order to complete the coloring on the edges within the clusters, our algorithm ensures that each cluster can independently control the colors of two edges adjacent to the cluster. For illustration purposes this example has many cycles that are dealt with separately in our algorithm.

To solve this non-greedy problem of assigning matching edges to clusters, we set up a hypergraph H . Each cluster in G corresponds to a vertex in H and each edge $e \in M$ defines a hyperedge containing all clusters that sent at least one proposal to e . Note that a

hypergraph is in fact necessary here, since each edge in M may receive proposals from up to 2Δ different clusters. Now our assignment problem can then be viewed as a hypergraph orientation problem, where each hyperedge is said to be outgoing for exactly one of its vertices and ingoing for all others. In this setting, the problem has already been studied in the literature under the name *hypergraph sinkless orientation*. As long as each cluster sends out sufficiently more proposals than any edge in M receives, it can be solved in $\mathcal{O}(\log_{\Delta} n)$ rounds deterministically and $\mathcal{O}(\log_{\Delta} \log n)$ rounds randomized [15]. This is the only “non-greedy” part of our reduction and it dominates the runtime. Every other step runs in $\mathcal{O}(\log^* n)$ rounds for constant-degree graphs, which shows that this is an essential part of our reduction in light of the $\Omega(\log_{\Delta} n)$ lower bound for edge-coloring with less than $2\Delta - 1$ colors.

Now as a final technical remark, while each cluster may own two assigned edges, these may not be directly adjacent to the cluster and controlling their color would still be useless in terms of obtaining solvability of the cluster. To solve this issue we will show that cluster C can move its assigned edges into its 1-hop neighborhood along a shortest path to C , satisfying (1), while maintaining the matching property (2). Finally, the colorful condition is satisfied for every cluster and we can complete the $(2\Delta - 2)$ -edge coloring for each cluster in parallel.

1.3 Related work

Distributed graph coloring is one of the most extensively studied problems in the field of distributed computing. Many foundational results are covered in the excellent book by Barenboim and Elkin [8]. In this section, we survey both, results for greedy and non-greedy edge coloring, highlighting a range of deterministic and randomized algorithms.

Greedy distributed edge coloring

Deterministic algorithms. In a seminal paper from 1987, Linial showed that any deterministic 3-coloring algorithm on the n -cycle takes at least $1/2 \cdot \log^* n$ rounds in the LOCAL model [54]. Naor later extended this lower bound to apply to randomized algorithms as well [59]. On the upper bound side, Linial also presented an $\mathcal{O}(\log^* n)$ -round $\mathcal{O}(\Delta^2)$ -coloring algorithm [54]. Combined with the classical one-color-per-round reduction this directly yields a $(2\Delta - 1)$ -edge coloring algorithm in $\mathcal{O}(\Delta^2 + \log^* n)$ rounds. With some clever tricks, subsequent works [6, 9, 10, 60, 53, 33, 57] have improved on this simple color reduction scheme, culminating in the current state-of-the-art for $(2\Delta - 1)$ -edge coloring using $\mathcal{O}(\log^{12} \Delta + \log^* n)$ rounds [4]. Another line of research focuses on the complexity of graph coloring as a function of n , independent of Δ . Early works by Panconesi & Srinivasan [61, 63] show that using network decompositions, one can find a $(2\Delta - 1)$ -edge coloring in $\exp(\mathcal{O}(\sqrt{\log n}))$ rounds. A long standing open question in the field asked for $(2\Delta - 1)$ -edge coloring algorithm in polylogarithmic time. Barenboim and Elkin made progress by showing that the $\Delta^{1+\epsilon}$ -coloring problem can be solved in $\mathcal{O}(\log^2 n)$ rounds [7]. Ghaffari & Su later gave a $\text{poly}(\log n)$ algorithm for $(2\Delta - 1)(1 + o(1))$ -edge coloring [45], which in turn was improved by [42]. Fischer, Ghaffari, and Kuhn [31] fully resolved the question with a breakthrough $\mathcal{O}(\log^9 n)$ -round algorithm based on a novel hypergraph maximal matching procedure. Since then, the field has picked up the pace with a lot of new algorithms being developed in the last decade, e.g. [4, 29, 36, 37, 39, 43, 48, 49]. The fastest algorithm for general graphs currently comes from the MIS algorithm in [37] and runs in $\tilde{\mathcal{O}}(\log^{5/3} n)$ rounds.

Randomized algorithms. The first randomized algorithm for greedy edge coloring follows from the seminal $\mathcal{O}(\log n)$ -round MIS algorithm independently developed by Luby [56] and Alon, Babai and Itai [1]. This upper bound stood for almost three decades until it was

surpassed by the $\mathcal{O}(\log \Delta) + 2^{\mathcal{O}(\sqrt{\log \log n})}$ -round algorithm introduced in [11], pioneering the now ubiquitous shattering framework for randomized algorithms. This algorithm was later improved to $\mathcal{O}(\log^* \Delta) + T_{\deg+1}^{\det}(\text{poly } \log n)$ by Chang, Li, and Pettie [20]. Here, $T_{\deg+1}^{\det}(n')$ is the deterministic complexity of the $(\deg + 1)$ -list coloring problem on instances of size n' [21]. A major breakthrough came with the first polylogarithmic network decomposition algorithm [65], reducing the complexity of the second term in both results and enabling $(2\Delta - 1)$ -edge coloring in $\text{poly}(\log \log n)$ rounds. Subsequent works [29, 36, 37, 38, 64] continued to refine the runtime with the current state-of-the-art standing at $\tilde{\mathcal{O}}(\log^{5/3} \log n)$ rounds [37].

Non-greedy distributed edge coloring

Despite the more challenging nature of the problem, there has also been a lot of progress in the non-greedy regime of coloring with $k < 2\Delta - 1$ colors. On the lower bound side, the authors of [17] established that coloring the edges of a graph with at most $2\Delta - 2$ colors takes at least $\Omega(\log_{\Delta} n)$ rounds deterministically and $\Omega(\log_{\Delta} \log n)$ rounds randomized. While most of the algorithms breaking the greedy color barrier were only developed in the last decade, there are two remarkable results before the turn of the millennium.

Δ -vertex coloring algorithms. In 1992, Panconesi & Srinivasan [61, 62] presented the first distributed Δ -coloring algorithm running in $\exp(\mathcal{O}(\sqrt{\log n}))$ rounds. This deterministic result also implies the first distributed $(2\Delta - 2)$ -edge coloring algorithm. After two decades without improvements, the authors of [40] finally decreased the upper bound for the randomized complexity of Δ -coloring to $\mathcal{O}(\log \Delta) + 2^{\mathcal{O}(\sqrt{\log \log n})}$. Combining these seminal results with the new polylogarithmic algorithms for network decomposition [65, 38, 36, 29, 64, 37] yields runtimes of $\text{poly}(\log n)$ and $\mathcal{O}(\log \Delta) + \text{poly}(\log \log n)$, respectively. Recently, Bourreau, Brandt and Nolin [14] showed that Δ -coloring can be solved in $\mathcal{O}(\log^4 \Delta + \log^2 \Delta \cdot \log n \log^* n)$ deterministic time and $\tilde{\mathcal{O}}(\log^{8/3} \log n)$ randomized time. The randomized version of this result is also the current state-of-the-art for $(2\Delta - 2)$ -edge coloring. Moreover, for $\Delta = \omega(\log^{21} n)$ the randomized complexity of Δ -coloring collapses to $\mathcal{O}(\log^* n)$ due to [32]. Finally, a very recent result [50] gives an optimal Δ -coloring algorithm for a restricted graph class. However, since line graphs are not included in this class, it does not yield any new edge coloring results.

Randomized algorithms via the distributed Lovász Local Lemma. In 1998, Dubhashi, Grable & Panconesi [27] presented an $(1 + \varepsilon)\Delta$ -edge coloring algorithm using the *Rödl nibble method*, which runs in $\mathcal{O}(\log n)$ rounds, assuming $\Delta > (\log n)^{1+\gamma}$. After the turn of the millennium, there has been no progress on the problem, until Elkin, Pettie & Su [28] improved upon [27] in 2015. Their randomized $(1 + \varepsilon)\Delta$ -edge coloring algorithm reduces the problem to the distributed Lovász Local Lemma and runs in $\mathcal{O}(\log^* \Delta + \frac{\log n}{\Delta^{1-o(1)}})$ rounds. In both cases, ε need not be constant, but it is not clear how small it can be made as a function of Δ . Focusing on palette size over speed, the authors of [17] give a randomized $(\Delta + \tilde{\Theta}(\sqrt{\Delta}))$ -edge coloring algorithm that runs in $\text{poly}(\log n)$ rounds – nearly matching the natural threshold $\Delta + \Theta(\sqrt{\Delta})$, the smallest palette size for which there still is a constant probability of being able to color any edge given a random feasible coloring of its neighborhood. Recently, new advances in variants of the distributed Lovász Local Lemma [47, 26] have reduced the complexity of $(1 + \varepsilon)\Delta$ -edge coloring further to $\text{poly}(\log \log n)$.

Deterministic algorithms. In a 2018 paper [44], the authors present a $\mathcal{O}(\Delta^9 \cdot \text{poly}(\log n))$ -time algorithm for $(3\Delta/2)$ -edge coloring based on maximum matchings in bipartite graphs. Their result has since seen a series of improvements. First, Harris [49] presented improved

algorithms for hypergraph maximal matchings, reducing the runtime to $\tilde{O}(\Delta^4 \cdot \log^6 n)$ rounds. Next, a novel algorithm for hypergraph sinkless orientation [15] further reduced the complexity to $\mathcal{O}(\Delta^2 \cdot \log n)$ rounds. Using a splitting technique from [42, 41] this algorithm can also be adapted to a $(3/2 + \varepsilon)\Delta$ -edge coloring algorithm with runtime $\tilde{O}(\varepsilon^{-2} \cdot \log^2 \Delta \cdot \log n)$. Notably, this implies a $\mathcal{O}(\log^3 n)$ -round algorithm for $(2\Delta - 2)$ -edge coloring in general graphs.

Vizing chain algorithms. A Vizing chain is an alternating path used to swap colors and resolve conflicts when trying to assign a new color to an edge in a graph that cannot be colored greedily. Vizing chains form a central role in Vizing’s celebrated result stating that every graph is $(\Delta + 1)$ -edge colorable [69]. Recent years have brought significant progress on distributed versions of Vizing’s theorem. In [68], the authors give the first $\text{poly}(\Delta, \log n)$ algorithms for randomized $(\Delta + 2)$ -edge coloring and deterministic $\Delta + \mathcal{O}(\log n / \log \log n)$ edge coloring. Further, Bernshteyn and Dhawan [12, 13] recently gave a deterministic $(\Delta + 1)$ -edge coloring algorithm in $\tilde{O}(\Delta^{84} \cdot \log^5 n)$ rounds, together with a faster randomized version running in $\mathcal{O}(\text{poly}(\Delta) \cdot \log^2 n)$ rounds. While the exponent in Δ in the runtime is certainly not yet optimized, there is a fundamental barrier to this approach. The authors of [17] have shown that it requires recoloring subgraphs of diameter $\Omega(\Delta \cdot \log n)$ in the worst case.

1.4 Organization of the paper

In Section 3 we present our main technical contribution: a sufficient condition under which partial $(2\Delta - 2)$ -edge colorings can be extended. Building on this, we introduce our new algorithms in Section 4. We start by presenting our reduction from $(2\Delta - 2)$ -edge coloring to MIS. Later on, in Section A, we replace the MIS and maximal matching subroutines to achieve a reduction to just $(2\Delta - 1)$ -edge coloring. Then, in Section B we adapt the reduction to obtain a faster randomized algorithm. For convenience, we collect all subroutines from the literature that we use in our algorithms in the appendix, to serve as a point of reference. The full version of our paper [51] also contains a table giving an overview of existing algorithms, the number of colors they use, and their runtime.

2 Preliminaries

We will use the notation $[k] := \{1, \dots, k\}$. Let $G = (V, E)$ be an undirected graph. The *neighborhood* of a vertex $v \in V$ in G is $N_G(v) = \{w \in V : vw \in E\}$ and the *edge-neighborhood* of v is $N_E(v) = \{e \in E : |e \cap \{v\}| = 1\}$. Further, for a subset of vertices $W \subseteq V$ we write $N_E(W) = \{e \in E : |e \cap W| = 1\}$. The *distance* between two vertices $u, v \in V$ is defined as the number of edges on a shortest path connecting u and v in G and is denoted by $\text{dist}(u, v)$. We define the distance between an edge $e = uv \in E$ and a vertex $w \in V$ as

$$\text{dist}(e, w) := \min(\text{dist}(u, w), \text{dist}(v, w)).$$

For any positive integer $k > 0$ we define the *power graph* $G^k = (V, E^k)$ by adding an edge for every pair of vertices $u, v \in V$ with $\text{dist}(u, v) \leq k$ to E^k . For any positive integer $k > 0$ and any vertex $v \in V$ we define the *k-hop neighborhood* of v to be $N^k(v) = N_{G^k}(v)$. For a subset of edges $F \subseteq E$ we write $G - F$ to denote the graph obtained by removing all edges in F from G . Further, for a subset $V' \subseteq V$, we denote by $G[V'] = (V', E')$ the subgraph of G induced by V' . The edge set E' is given by the subset of edges with both endpoints in V' . Similarly, for a subset of edges $E' \subseteq E$ we define the *edge-induced subgraph* $G[E'] = (V', E')$, where V' is given by the subset of vertices that are incident to at least one edge in E' . We

define the *degree* $\deg(e)$ of an edge $e \in E$ as the number of edges adjacent to e . A *hypergraph* is a generalization of a graph in which each edge can contain more than two vertices. The *rank* of a hyperedge e is the number of vertices contained in e . A *proper c -edge coloring* is a function $\varphi : E \rightarrow [c]$ that assigns different colors to adjacent edges. In this paper we will work a lot with *partial edge colorings* that assign colors only to a subset of edges.

► **Definition 7.** Let $G = (V, E)$ be a graph, $F \subseteq E$ and $\varphi : E \setminus F \rightarrow [2\Delta - 2]$ a partial edge coloring of G . Then we call φ^* an extension of φ to F if φ^* is a proper $(2\Delta - 2)$ -edge coloring of G and $\varphi^*(e) = \varphi(e)$ for all $e \in E \setminus F$. Further, we say a color c is available for an edge $e \in F$, if no adjacent edge already has color c .

► **Definition 8** ((α, β) -ruling set). For two integers $\alpha, \beta \geq 1$, an (α, β) -ruling set for a subset $W \subseteq V$ is a set of nodes $S \subseteq V$ such that the distance between any two nodes in S is at least α and any node $w \in W$ has a distance of at most β to the closest node in S . If $V = W$, we call S an (α, β) -ruling set.

Ruling sets generalize maximal independent sets, which are $(2, 1)$ -ruling sets. In our algorithm we will use ruling sets as a tool to partition the vertex sets into clusters with some specific properties. Most importantly, we will need that for each cluster C there is a central vertex $r \in C$ such that the whole k -hop neighborhood around r is contained in C , for some parameter $k > 0$. This ensures that each cluster has sufficiently many vertices, which will become important when we try to assign two exclusive matching edges to each cluster.

► **Definition 9** ((α, β) -clustering). Let $G = (V, E)$ be a graph and $\mathcal{R} \subseteq V$ a subset of vertices. Then, we call a partition $\bigcup_{r \in \mathcal{R}} C(r) = V$ of the vertex set into pairwise disjoint subsets a (α, β) -clustering with respect to $\mathcal{R}$ if $N^\alpha(r) \subseteq C(r)$ and $\text{diam}(C(r)) \leq \beta$ for all $r \in \mathcal{R}$.

► **Remark 10.** Computing a $(2, \beta)$ -ruling set on the power graph G^k and letting each vertex join the cluster of one of its closest ruling set nodes leads to a $(\lfloor \frac{k}{2} \rfloor, k \cdot \beta)$ -clustering.

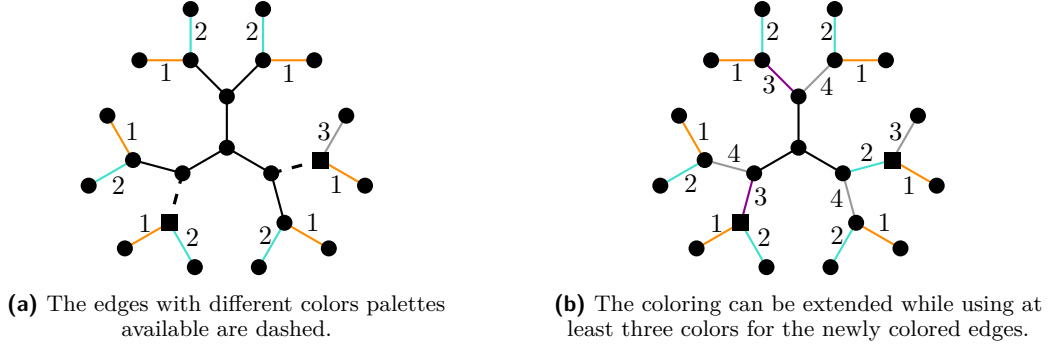
In the third step of our algorithm, our goal is to assign two exclusive edges to each cluster, such that the set of all such edges forms a matching in G . Starting with a maximal matching, we construct an auxiliary hypergraph H that lets us modify the matching in order to satisfy this property. This can be seen as an instance of the *hypergraph sinkless orientation* problem, which is a natural generalization of the fundamental *sinkless orientation* problem.

► **Definition 11** (Hypergraph sinkless orientation (HSO)). Let $H = (V, E)$ be a hypergraph. The objective of the hypergraph sinkless orientation problem is to orient the hyperedges of H such that every vertex $v \in V$ has at least one outgoing hyperedge. We define an oriented hyperedge to be outgoing for exactly one of its incident vertices and incoming for all others.

If the minimum degree $\delta = \min_{v \in V} \deg(v)$ and the maximum rank $r = \max_{e \in E} \text{rank}(e)$ of H satisfy $\delta > r$, then this problem can be solved efficiently using the algorithms in Theorems 34 and 35. We follow the standard assumption that each vertex in the LOCAL model knows the number of nodes $n = |V|$ of the graph, as well as the maximum degree Δ .

3 Under which condition can we extend partial $(2\Delta - 2)$ -colorings?

The goal of this section is to establish conditions under which we can extend φ to a proper $(2\Delta - 2)$ -edge coloring of the entire graph G . As a first step, we examine the case where the subgraph induced by the uncolored edges is a simple star. In this setting, the only forbidden configuration occurs when all the leaves of the star graph share the same set of $\Delta - 1$ incident colors, which can be seen in Figure 2b.



■ **Figure 4** Extending the coloring to a tree.

► **Lemma 12** (Reaching for the stars). *Let $G = (V, E)$ be a graph and let φ be a partial $(2\Delta - 2)$ -edge coloring of G that only leaves the edges of a star graph $T = (V_T, E_T) \subseteq G$ uncolored. If $|\varphi(N_E(V_T))| \geq \Delta$, then there is an extension φ^* of φ to E_T .*

To prove this lemma we invoke the following classical combinatorial result.

► **Theorem 13** (Hall's theorem [46]). *Let $G = (V \cup U, E)$ be a bipartite graph. For any subset $S \subseteq U$, let $N(S)$ denote the neighborhood of S in G . Then G has a U -saturating matching, if and only if $|N(S)| \geq |S|$ for all $S \subseteq U$.*

Proof of Lemma 12. We construct the auxiliary bipartite graph $B = (E_T \cup [2\Delta - 2], F)$ as follows: For every edge $e \in E_T$ and every color $c \in [2\Delta - 2]$ we add the edge (e, c) to F if and only if c is available for e under φ . Observe that any E_T -saturating matching in B corresponds to an extension of φ to E_T . By Theorem 13 such a matching exists if and only if for any subset $S \subseteq E_T$ it holds that $|N_B(S)| \geq |S|$. Since any edge in T has at least $\Delta - 1$ colors available and $|E_T| \leq \Delta$, the only set left to check is $S = E_T$. For contradiction sake, assume that $|N_B(E_T)| = \Delta - 1$ and let $r \in V_T$ be the center of the star. This implies that for every $e = rv \in E_T$ all colors in $C := [2\Delta - 2] \setminus N_B(E_T)$ are already occupied. Since $|C| = \Delta - 1 \geq |N_E(v) \cap N_E(V_T)|$, this is only possible if $\varphi(N_E(v) \cap N_E(V_T)) = C$ for all $e \in E_T$ and therefore $\varphi(N_E(V_T)) = C$. But $|\varphi(N_E(V_T))| \geq \Delta$, a contradiction! ◀

As the next step we show that this condition generalizes to trees as well. More precisely, we will show that if φ assigns at least Δ distinct colors to the edges adjacent to at least one vertex at distance k from the root, then there is always an extension of φ that assigns at least Δ distinct colors to the edges adjacent to any vertex at distance $k - 1$ from the root.

► **Lemma 14** (Colorful leaves make any tree happy). *Let $G = (V, E)$ be a Δ -regular graph, $T = (V_T, E_T) \subseteq G$ a tree, $r \in V_T$ and $\varphi : E \setminus E_T \rightarrow [2\Delta - 2]$ a partial edge coloring of G . Let*

$$V_k := \{v \in V_T : \text{dist}_T(v, r) = k\} \quad \text{and} \quad E_k = \{e \in E_T : \text{dist}_T(e, r) = k\}.$$

If there is a $k \in \mathbb{N}$ such that $|\varphi(N_E(V_k))| \geq \Delta$, then there exists an extension φ^ of φ to E_T .*

Proof. Let $\ell \in \mathbb{N}$ such that $|\varphi(N_E(V_\ell))| \geq \Delta$. First we observe that for every $k > 0$, each edge $e \in E_k$ has exactly one neighboring edge in E_{k-1} . Therefore, we can always find at least one available color for every such edge, as long as the set E_{k-1} remains uncolored. Using this observation we can assign arbitrary available colors to all edges in $\bigcup_{k \geq \ell} E_k$ by processing these edge sets in sequence, starting with the set with highest index and going

through each of these sets in an arbitrary order. However, after we finish coloring E_ℓ we need to start being more deliberate with our color choices. For any $k < \ell$ we need to ensure that $|\varphi(N_E(V_k))| \geq \Delta$ holds after we colored all edges of E_k . This is due to the fact that we need $|\varphi(N_E(V_1))| \geq \Delta$ in order to apply Lemma 12, which guarantees that we can complete the coloring on the final layer of edges E_0 connected to the root. Now we show that given $|\varphi(N_E(V_{k+1}))| \geq \Delta$ we can color the edges in E_k in such a way that $|\varphi(N_E(V_k))| \geq \Delta$ for any $1 \leq k < \ell$. For each $e \in E_k$ let $N^+(e) := N_E(e) \cap N_E(V_{k+1})$. Since $|N^+(e)| = \Delta - 1$ we can always find two edges $e, e' \in E_k$ such that $\varphi(N^+(e)) \neq \varphi(N^+(e'))$. We first assign colors to all edges in E_k that are not adjacent to e or e' . Since we assume Δ to be sufficiently large, there always exists a vertex $v \in V_k$ that is not incident to either e or e' . Thus, after coloring all $\Delta - 1$ edges in $N_E(v) \setminus E_{k-1}$ incident at v we already have $|\varphi(N_E(V_k))| \geq \Delta - 1$. If $|\varphi(N_E(V_k))| \geq \Delta$, we are already done. Otherwise, if $|\varphi(N_E(V_k))| = \Delta - 1$, then the set $\varphi(N_E(V_k) \cup (N^+(e) \cap N^+(e')))$ contains at most $2\Delta - 3$ colors. Therefore, either e or e' must have a color available in $[2\Delta - 2] \setminus \varphi(N_E(V_k))$. We pick that color for e or e' , which ensures that $|\varphi(N_E(V_k))| \geq \Delta$. The remaining edges in E_k can then be assigned arbitrary available colors again. For an illustration of this procedure we refer to Figure 4. Finally, to color the last layer E_0 containing the edges incident to the root, we can apply Lemma 12, since the number of distinct adjacent colors is at least Δ . $\blacktriangleleft$

The proofs of the next two lemmata argue in a similar way to previous work on Δ -vertex coloring and are therefore deferred to the appendix. First, we show that as soon as a cluster C contains a cycle of even length or a vertex of degree less than Δ , then we can extend any partial $(2\Delta - 2)$ -edge coloring without modifications to $E(C)$.

► **Lemma 15.** *Let $G = (V, E)$ and $\varphi : E \rightarrow [2\Delta - 2] \cup \{\perp\}$ be a partial $(2\Delta - 2)$ -edge coloring of G . Assume that the subgraph $H \subseteq G$ induced by the uncolored edges is connected and contains either a cycle of even length or a vertex of degree less than Δ . Then there exists an extension φ^* of φ , which is a proper $(2\Delta - 2)$ -edge coloring of the entire graph G .*

On the other hand, if a cluster does not contain a cycle of even length, we need to ensure that it can satisfy the colorful condition from Lemma 14. To this end, it is crucial that the cluster contains a sufficient number of leaves. We apply a similar argument to the one used in the analysis of the Δ -vertex coloring algorithm in [40] to show that a connected component that does not contain an even cycle must be expanding exponentially.

► **Lemma 16.** *Let $G = (V, E)$ be a Δ -regular graph, $k > 0, v \in V$ and $C \subseteq G$ be a connected subgraph of G such that $N^k(v) \subseteq C$. Further assume that C does not contain a cycle of even length. Then, any BFS-tree T of C rooted at v contains at least $(\Delta - 2)^k$ leaves.*

4 Deterministic $(2\Delta - 2)$ -edge coloring algorithm

We first present the simpler version of our algorithm, which reduces the $(2\Delta - 2)$ -edge coloring problem deterministically to MIS in $\mathcal{O}(\log_\Delta n)$ rounds of the LOCAL model.

4.1 High level overview of our reduction to MIS

Recall that HSO (Theorem 34) aims to orient the hyperedges of a hypergraph such that each vertex has at least one outgoing hyperedge. In Phase 2 of Algorithm 1, it is used to rearrange the existing matching, ensuring that each cluster is assigned two exclusive edges.

■ **Algorithm 1** Reduction from $(2\Delta - 2)$ -edge coloring to MIS (high level overview).

Input: a graph $G = (V, E)$	
Output: $(2\Delta - 2)$ -edge coloring φ of G .	
Phase 1: Partition vertices into clusters (Section 4.2)	
1: Compute an MIS $\mathcal{I} \subseteq V$ on the power graph G^8	▷ Theorem 36
2: $\mathcal{C}, E_T \leftarrow \text{CLUSTER}(G, \mathcal{I})$	▷ Algorithm 2
3: Compute a $(2\Delta - 3)$ -edge coloring φ of $G - E_T$	▷ Lemma 18
Phase 2: Assign two exclusive edges to each cluster (Section 4.3)	
4: Compute a maximal matching M of $G - E_T$	▷ Lemma 19
5: Set up auxiliary hypergraph $H = (\mathcal{C}, M)$	▷ Definition 21
6: Assign two matching edges to each cluster via HSO on H	▷ Theorem 34
7: Move the assigned edges into the neighborhood of the respective clusters	▷ Lemma 24
Phase 3: Switch colors in order to complete the coloring (Section 4.4)	
8: for each cluster in parallel do	
9: Change the colors of the assigned edges to fulfil the colorful condition	▷ Lemma 25
10: end for	
11: Extend φ to a $(2\Delta - 2)$ -edge coloring φ^* of G	▷ Lemma 14

Phase 1: Partition vertices into clusters (Section 4.2). We start by computing a maximal independent set $\mathcal{I}$ on the power graph G^8 . Next we partition the vertex set into disjoint clusters. Each vertex joins the cluster of one of its closest MIS-nodes, breaking ties arbitrarily, but consistently, which leads to a $(4, 8)$ -clustering $\mathcal{C}$. In addition, we store the BFS-tree (V, E_T) which arises naturally from this clustering. Computing the MIS on G^8 ensures that the individual clusters all contain sufficiently many vertices, which will become important later. Next, we remove the union of all BFS-trees from G to obtain a subgraph $G - E_T \subseteq G$ of maximum degree at most $\Delta - 1$. Hence, it suffices to invoke a greedy $(2\Delta - 1)$ -edge coloring on $G - E_T$ in order to get a $(2\Delta - 3)$ -edge coloring φ of $G - E_T$.

Phase 2: Assign two exclusive edges to each cluster (Section 4.3). Finally we need to modify φ such that the colorful condition from Lemma 14 is satisfied for every cluster. We will show that this can be achieved by giving each cluster control over the colors of two edges adjacent to the cluster. However, in order for the clusters to switch the colors of their respective assigned edges, the union of all those edge sets needs to be independent. Therefore we initially compute a maximal matching M of $G - E_T$ to use as a starting point. Note that the maximality of the matching does not guarantee that any fixed cluster receives any adjacent matching edges, but only that there are sufficiently many matching edges in the 2-hop neighborhood around that cluster. Hence, we then let each cluster propose to matching edges in its 2-hop neighborhood. This process naturally defines a hypergraph, where each matching edge collects the clusters which sent a proposal to it as vertices in a hyperedge. Then, the problem of assigning matching edges to clusters can be modeled as an instance of hypergraph sinkless orientation. In order to be able to solve HSO efficiently we need to ensure that the number of votes each matching edge receives is dominated by the number of votes each cluster casts in total. Here we require that each cluster contains enough vertices.

Phase 3: Switch colors in order to complete the coloring (Section 4.4). Finally, we show that each cluster can flip the colors of their assigned matching edges in parallel in order to fulfil the colorful condition from Lemma 14. Thus, this new coloring can now be extended to the remaining uncolored edges by iteratively coloring the layers of each BFS-tree in parallel. Since each cluster has diameter of at most 8 this last step can be executed in constant time.

► **Theorem 2.** *There is a deterministic LOCAL algorithm computing a $(2\Delta - 2)$ -edge coloring in any n -vertex graph with maximum degree Δ in $\mathcal{O}(\log_\Delta n) + T_{\text{MIS}}(\Delta^2 \cdot n, \text{poly}(\Delta))$ rounds.*

4.2 Phase 1: Partition vertices into clusters

■ **Algorithm 2** $\text{CLUSTER}(G, \mathcal{I})$.

Input: a graph $G = (V, E)$ and an independent set $\mathcal{I} \subseteq V$
Output: a $(4, 8)$ -clustering $\mathcal{C} = \{C^{-1}(r) : r \in \mathcal{I}\}$ and a forest $T = (V, E_T)$

```

1:  $E_T \leftarrow \emptyset$ 
2: for each  $v \in V$  in parallel do
3:    $C(v) \leftarrow \perp$ 
4:   if  $v \in \mathcal{I}$  then
5:      $C(v) \leftarrow v$ 
6:   end if
7:   while  $C(v) = \perp$  do
8:     receive messages from neighbors
9:     if  $\exists u \in N(v) : C(u) \neq \perp$  then
10:       $C(v) \leftarrow C(u)$ 
11:       $E_T \leftarrow E_T \cup \{uv\}$  ▷ break ties arbitrarily
12:      send message  $C(v)$  to neighbors
13:    end if
14:  end while
15: end for

```

► **Lemma 17** (Clustering algorithm). *The set $\mathcal{C} = \{C^{-1}(r) : r \in \mathcal{I}\}$ computed by Algorithm 2 is a $(4, 8)$ -clustering with respect to $\mathcal{I}$.*

Proof. First we show that $N^4(r) \subseteq C^{-1}(r)$ for all $r \in \mathcal{I}$. Let $r \in \mathcal{I}, v \in N^4(r)$ and assume for contradiction sake that $C(v) = r' \neq r$. Then, $\text{dist}(v, r') \leq \text{dist}(v, r) \leq 4$ and therefore $\text{dist}(r, r') \leq 8$, contradicting the fact that r and r' are independent in G^8 . For the second property assume that there exist $r \in \mathcal{I}$ and $v \in C^{-1}(r)$ such that $\text{dist}(v, r) > 8$. This implies that $\text{dist}(v, r') \geq \text{dist}(v, r) > 8$ for all $r' \in \mathcal{I}$, because otherwise v would have joined the cluster of r' instead of r . But then v would be independent of all $r' \in \mathcal{I}$ on G^8 , which contradicts the maximality of $\mathcal{I}$. ◀

For each $r \in \mathcal{I}$, let $T(r) = T[C^{-1}(r)]$ denote the BFS-tree of the cluster $C^{-1}(r)$ rooted at r .

► **Lemma 18.** *The subgraph $G - E_T$ has maximum degree of at most $\Delta - 1$ and can be $(2\Delta - 3)$ -edge colored in $T_{\text{MIS}}(\Delta^2 \cdot n, 4\Delta)$ rounds of the LOCAL model.*

Proof. We observe that the edge set E_T returned by Algorithm 2 satisfies $E_T = \bigcup_{r \in \mathcal{I}} E(T(r))$. For every vertex $v \in V$ there is an $r \in \mathcal{I}$ such that $v \in C^{-1}(r)$. Since $T(r)$ is a spanning tree of $C^{-1}(r)$, we get that v is incident to at least one edge in $E(T(r))$. Thus, the maximum degree of $G - E_T$ is at most $\Delta - 1$. Using the standard reduction from vertex coloring to MIS [56] we can compute a $(2\Delta - 3)$ -edge coloring of $G - E_T$ in $T_{\text{MIS}}(\Delta^2 \cdot n, 4\Delta)$ rounds. ◀

► **Lemma 19.** *We can compute a maximal matching M of the edges in $G - E_T \subseteq G$ deterministically in $T_{\text{MIS}}(\Delta \cdot n, 2\Delta - 2)$ rounds of the LOCAL model.*

Proof. A maximal matching of $G - E_T$ is simply an MIS on the line graph $L(G - E_T)$. ◀

4.3 Phase 2: Assign two exclusive edges to each cluster

In this phase, we rearrange the edges in the maximal matching M in order to assign two exclusive edges to each cluster, while preserving the matching property. These exclusive edges are essential for adjusting the coloring of the intercluster edges, as laid out in Lemma 25. This step relies on the HSO subroutine from Theorem 34, and we now detail the specific instance used in our algorithm. The high level idea is that each cluster sends proposals to nearby edges in the matching M . Each edge accepts exactly one proposal and we swap the edge towards the cluster that *won* it. This process can be modeled as an HSO instance where each matching edge in M represents an hyperedge that contains as vertices all clusters that sent a proposal to that edge. Since a directed hyperedge e is outgoing for exactly one of its vertices $v \in e$, we will say that v *won* the proposal for e .

However, the hypergraph sinkless orientation problem only guarantees at least one outgoing hyperedge for each vertex. Therefore, in order to achieve two exclusive hyperedges for each cluster, we simply split each cluster into two virtual nodes and let both participate in the HSO independently. In the following we formalize this idea and prove that this provides the desired properties as well as a deterministic runtime of $\mathcal{O}(\log n)$ rounds. First, we let each cluster $C^{-1}(r) \in \mathcal{C}$ that is not already edge-degree choosable select a subset of vertices $S \subseteq V(C^{-1}(r))$ to send proposals to a matching edges in their neighborhood. Let $\mathcal{C}'$ denote the set of clusters, which are not already edge-degree choosable. Crucially, in order to satisfy the colorful condition from Lemma 14 we require that the vertices in S are all leaves in the spanning tree $T(r) \subseteq C^{-1}(r)$ at the same distance from the root r .

► **Lemma 20.** *For each cluster $C^{-1}(r) \in \mathcal{C}'$, there exists an integer $k > 0$ and a subset $S \subseteq V(C^{-1}(r))$ of size at least $4 \cdot \Delta^3$ such that every vertex $v \in S$ is a leaf in the tree $T(r)$ that satisfies $\text{dist}(r, v) = k$.*

Proof. Since $\mathcal{C}$ is a $(4, 8)$ -clustering we have that $C^{-1}(r)$ necessarily contains all vertices in $N^4(r)$. Thus, according to Lemma 16 the corresponding BFS-tree contains at least $(\Delta - 2)^4$ leaves. Therefore, by the pigeonhole principle there exists a layer V_k for $k = 1, \dots, 8$ such that V_k contains at least $(\Delta - 2)^4 / 8 > 4 \cdot \Delta^3$ leaves for Δ sufficiently large. ◀

For each cluster we select an arbitrary set S of at least $4 \cdot \Delta^3$ leaves which are all at the same distance from the center. The subset S is the subset of leaves that send requests. Each vertex in $v \in S$ sends exactly one proposal. If v is incident to a matching edge $e \in M$, then it proposes to e . Otherwise, v sends a proposal to an arbitrary matching edge that is one hop away from e . Such an edge must exist, due to the maximality of M .

► **Definition 21 (Auxiliary hypergraph).** *We define the auxiliary hypergraph $H = (V_H, E_H)$:*

- *For each cluster $C \in \mathcal{C}'$ we add a vertex v_C to V_H .*
- *For each matching edge $e \in M$ we add a hyperedge e_H to E_H that contains all vertices v_C for which there exists at least one vertex $v \in V(C)$ that proposed to e .*

► **Lemma 22.** *The minimum degree δ_H and the maximum rank r_H of H satisfy $\delta_H > \Delta \cdot r_H$.*

Proof. First we observe that a single matching edge $e = uv$ can get proposals only from u, v and vertices that are exactly one hop away from either u or v . Hence, there may be two proposals coming from u and v plus $2 \cdot (\Delta - 1)$ proposals coming from neighbors of u or v . Thus, the maximum rank of H is at most $r_H = 2\Delta$. Next, each cluster contains at least $4 \cdot \Delta^3$ vertices sending proposals. Since each vertex sends exactly one proposal and each edge receives at most 2Δ proposals the vertices of each cluster must propose to at least $2\Delta^2$ edges in total. Hence, the minimum degree of H can be lower bounded by $\delta = 2\Delta^2$. ◀

► **Lemma 23.** *We can compute an orientation of E_H such that each vertex in V_H has at least two outgoing hyperedges in $\mathcal{O}(\log_\Delta n)$ rounds deterministically and $\mathcal{O}(\log_\Delta \log n)$ rounds randomized.*

Proof. To compute an orientation with two outgoing edges per cluster we simply split each vertex into two virtual nodes with half the degree each. Thus, the minimum degree of the resulting hypergraph H' is still at least Δ^2 . Therefore, we can solve HSO in $\mathcal{O}(\log_{\delta/r} n) = \mathcal{O}(\log_\Delta n)$ rounds deterministically and in $\mathcal{O}(\log_{\delta/r} \delta + \log_{\delta/r} \log n) = \mathcal{O}(\log_\Delta \log n)$ rounds randomized using the algorithms from Theorem 34 and Theorem 35. ◀

The orientation of H yields an assignment of matching edges to clusters such that each cluster receives exclusive access to two matching edges. However, the matching edges are not necessarily adjacent to their respective clusters yet. Therefore we modify the matching M by moving each edge closer to the vertex which won the proposal for it.

► **Lemma 24.** *There is a matching $M' = \chi(M)$ such that for all $e \in M$ it holds that $\chi(e)$ is adjacent to the vertex that won the proposal for e .*

Proof. For each $e \in M$, let v be the vertex (in the original graph G), which sent the winning proposal to e . If v is incident to e , then we simply set $\chi(e) = e$. Otherwise, let u be the endpoint of e which is adjacent to v and set $\chi(e) = uv$. With this procedure, we ensure that each cluster is adjacent to at least two edges in M' since each of the two partitions of the cluster receives at least one edge based on the HSO property. If a cluster is assigned more than two edges in M' , it discards any edges beyond the required two. In order to show that the edge set M' is still a proper matching, we recall that every vertex $v \in S \subseteq V$ sends out exactly one proposal. Hence, there is no vertex that wins more than one edge in M . Assume that v won the proposal for the edge $e = uv$. If v is incident to e there is no change from M to M' . Otherwise, let u be adjacent to v . In this case, the edge $uv \in M$ is flipped to $vw \in M'$. Hence, v is the only vertex that gains an incident edge in M' compared to M . Since e is one of the closest edges to v in M there cannot be an edge $e' \in M$ that is already incident to v . Therefore every vertex that wins its only proposal is incident to exactly one edge in M' . On the other hand, every vertex that does not win its proposal cannot gain any additional adjacent edges in M' . Therefore M' is still a matching. ◀

4.4 Phase 3: Switch colors in order to complete the coloring

Now that each cluster has been assigned two exclusive edges located at the same distance from its central vertex, we show how to modify their colors to ensure that the colorful condition from Lemma 14 is satisfied for every cluster. To prove that this procedure can be applied to all clusters in parallel, we argue that the colorful condition can still be fulfilled even if the colors of all edges not adjacent to the two exclusive edges are modified adversarially.

► **Lemma 25 (Color switching).** *Let $G = (V, E)$ be a Δ -regular graph, $T = (V_T, E_T) \subseteq G$ a tree, $r \in V_T$ and $\varphi : E \setminus E_T \rightarrow [2\Delta - 3]$ a partial edge coloring of G . Then, for any two independent edges $uv, u'v' \in E_k$, where u and u' are leaves in T , there exists a coloring $\varphi' : E \setminus E_T \rightarrow [2\Delta - 2]$ such that $|\varphi'(N_E(V_k))| \geq \Delta$ and $\varphi(e) = \varphi'(e)$ for all $e \notin \{uv, u'v'\}$.*

Proof. Recall that $V_k := \{v \in V_T : \text{dist}_T(v, r) = k\}$. Since u and u' are leaves in T , they are incident to exactly $\Delta - 1$ edges in $E \setminus E_T$. Hence, $\varphi(N_E(V_k))$ already contains at least $\Delta - 1$ distinct colors. If $|\varphi(N_E(V_k))| \geq \Delta$ we simply set $\varphi' = \varphi$. Otherwise, if $|\varphi(N_E(V_k))| = \Delta - 1$, we set $\varphi'(uv) = 2\Delta - 2$. Since $\varphi(e) < 2\Delta - 2$ for all $e \in E_k$ we only need to argue that



(a) Before switching colors: $\varphi(N_E(V_1)) = \{1, 2\}$. (b) After switching colors: $\varphi(N_E(V_1)) = \{1, 2, 4\}$.

■ **Figure 5** A simplified example for the color switching procedure, where $\Delta = 3$. The edges in the BFS-tree T_r are represented by solid lines, while edges outside of T_r are dashed. The assigned matching edges e_1, e_2 are marked orange.

the color $\varphi(uv)$ still remains in $\varphi'(E_k)$. Consider the sets $N := \{uw : uw \in E \setminus E_T\}$ and $N' := \{u'w : u'w \in E \setminus E_T\}$. Since u, u' are both leaves in T , both N and N' have cardinality $\Delta - 1$. Moreover, since $|\varphi(E_k)| = \Delta - 1$ and any two edges in N (or N') are adjacent, we have that $\varphi(N) = \varphi(N')$. Hence, there exists an edge $u'v''$ such that $\varphi(uv) = \varphi(u'v'') = \varphi'(u'v'') \in \varphi'(E_k)$. Thus, we have $\varphi'(E_k) = \Delta$. Note that this argument holds even if the colors of the edges not adjacent to u or v are modified adversarially. ◀

4.5 Proof of Theorem 2

► **Theorem 2.** *There is a deterministic LOCAL algorithm computing a $(2\Delta - 2)$ -edge coloring in any n -vertex graph with maximum degree Δ in $\mathcal{O}(\log_\Delta n) + T_{\text{MIS}}(\Delta^2 \cdot n, \text{poly}(\Delta))$ rounds.*

Proof. For $\Delta = \mathcal{O}(1)$ we apply the $(3\Delta/2)$ -edge coloring algorithm from Theorem 39 to get a $(2\Delta - 2)$ -edge coloring in $\mathcal{O}(\log n)$ rounds. Hence, for the remainder of the proof we assume $\Delta \geq \Delta_0$ for a sufficiently large constant Δ_0 . Since the power graph G^8 has n vertices and maximum degree polynomial in Δ , an MIS can be computed on it in time $T_{\text{MIS}}(n, \text{poly}(\Delta))$. Next, the clustering is possible in a constant number of rounds, since the radius of each cluster is at most 8. Also, both computing the $(2\Delta - 3)$ -edge coloring φ of $G - E_T$ and finding a maximal matching in E_T can be reduced to MIS, and thus can be solved in time $T_{\text{MIS}}(\Delta^2 \cdot n, \text{poly}(\Delta))$. Then, according to Section 4.3 we can modify the matching in a way that each cluster receives two exclusive edges. The rearrangement can be modeled as an instance of HSO with minimum degree $\frac{\Delta^3}{4}$ and maximum rank Δ^2 . Since each cluster has constant diameter, simulating this hypergraph incurs just a constant overhead factor. Thus, the deterministic HSO algorithm from Theorem 34 runs in $\mathcal{O}(\log_\Delta n)$ rounds. Next, we apply Lemma 25 to ensure that for each cluster there is a $k \leq 8$ such that $|\varphi(N_E(V_k))| \geq \Delta$. By Lemma 14 we can extend φ to a proper $(2\Delta - 2)$ -edge coloring of G . ◀

References

- 1 Noga Alon, László Babai, and Alon Itai. A fast and simple randomized parallel algorithm for the maximal independent set problem. *J. Algorithms*, 7(4):567–583, 1986. doi:10.1016/0196-6774(86)90019-2.
- 2 Alkida Balliu, Sebastian Brandt, Juho Hirvonen, Dennis Olivetti, Mikaël Rabie, and Jukka Suomela. Lower bounds for maximal matchings and maximal independent sets. In David Zuckerman, editor, *60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019, Baltimore, Maryland, USA, November 9–12, 2019*, pages 481–497. IEEE Computer Society, 2019. doi:10.1109/FOCS.2019.00037.

- 3 Alkida Balliu, Sebastian Brandt, Juho Hirvonen, Dennis Olivetti, Mikaël Rabie, and Jukka Suomela. Lower bounds for maximal matchings and maximal independent sets. *J. ACM*, 68(5):39:1–39:30, 2021. doi:10.1145/3461458.
- 4 Alkida Balliu, Sebastian Brandt, Fabian Kuhn, and Dennis Olivetti. Distributed edge coloring in time polylogarithmic in Δ . In Alessia Milani and Philipp Woelfel, editors, *PODC '22: ACM Symposium on Principles of Distributed Computing, Salerno, Italy, July 25 - 29, 2022*, pages 15–25. ACM, 2022. doi:10.1145/3519270.3538440.
- 5 Amotz Bar-Noy, Rajeev Motwani, and Joseph Naor. The greedy algorithm is optimal for on-line edge coloring. *Inf. Process. Lett.*, 44(5):251–253, 1992. doi:10.1016/0020-0190(92)90209-E.
- 6 Leonid Barenboim. Deterministic $(\Delta + 1)$ -coloring in sublinear (in Δ) time in static, dynamic, and faulty networks. *J. ACM*, 63(5):47:1–47:22, 2016. doi:10.1145/2979675.
- 7 Leonid Barenboim and Michael Elkin. Distributed deterministic edge coloring using bounded neighborhood independence. *Distributed Comput.*, 26(5-6):273–287, 2013. doi:10.1007/S00446-012-0167-7.
- 8 Leonid Barenboim and Michael Elkin. *Distributed Graph Coloring: Fundamentals and Recent Developments*. Morgan & Claypool Publishers, 2013. doi:10.2200/S00520ED1V01Y201307DCT011.
- 9 Leonid Barenboim, Michael Elkin, and Uri Goldenberg. Locally-iterative distributed $(\Delta + 1)$ -coloring and applications. *J. ACM*, 69(1):5:1–5:26, 2022. doi:10.1145/3486625.
- 10 Leonid Barenboim, Michael Elkin, and Fabian Kuhn. Distributed $(\Delta+1)$ -coloring in linear (in Δ) time. *SIAM J. Comput.*, 43(1):72–95, 2014. doi:10.1137/12088848X.
- 11 Leonid Barenboim, Michael Elkin, Seth Pettie, and Johannes Schneider. The locality of distributed symmetry breaking. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 321–330. IEEE Computer Society, 2012. doi:10.1109/FOCS.2012.60.
- 12 Anton Bernshteyn. A fast distributed algorithm for $(\Delta + 1)$ -edge-coloring. *J. Comb. Theory B*, 152:319–352, 2022. doi:10.1016/J.JCTB.2021.10.004.
- 13 Anton Bernshteyn and Abhishek Dhawan. Fast algorithms for vizing’s theorem on bounded degree graphs. *CoRR*, abs/2303.05408, 2023. doi:10.48550/arXiv.2303.05408.
- 14 Yann Bourreau, Sebastian Brandt, and Alexandre Nolin. Faster distributed Δ -coloring via ruling subgraphs. *arXiv preprint arXiv:2503.04320*, 2025. doi:10.48550/arXiv.2503.04320.
- 15 Sebastian Brandt, Yannic Maus, Ananth Narayanan, Florian Schager, and Jara Uitto. On the locality of hall’s theorem. In Yossi Azar and Debmalya Panigrahi, editors, *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2025, New Orleans, LA, USA, January 12-15, 2025*, pages 4198–4226. SIAM, 2025. doi:10.1137/1.9781611978322.143.
- 16 Sebastian Brandt and Ananth Narayanan. Towards optimal deterministic local algorithms on trees, 2025. doi:10.48550/arXiv.2505.01410.
- 17 Yi-Jun Chang, Qizheng He, Wenzheng Li, Seth Pettie, and Jara Uitto. The complexity of distributed edge coloring with small palettes. In Artur Czumaj, editor, *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 2633–2652. SIAM, 2018. doi:10.1137/1.9781611975031.168.
- 18 Yi-Jun Chang, Qizheng He, Wenzheng Li, Seth Pettie, and Jara Uitto. Distributed edge coloring and a special case of the constructive lovász local lemma. *ACM Trans. Algorithms*, 16(1), November 2019. doi:10.1145/3365004.
- 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. doi:10.1137/17M1117537.
- 20 Yi-Jun Chang, Wenzheng Li, and Seth Pettie. An optimal distributed $(\Delta+1)$ -coloring algorithm? In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *Proceedings of*

- the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018, pages 445–456. ACM, 2018. doi:10.1145/3188745.3188964.
- 21 Yi-Jun Chang, Wenzheng Li, and Seth Pettie. Distributed $(\Delta + 1)$ -coloring via ultrafast graph shattering. *SIAM Journal of Computing*, 49(3):497–539, 2020. doi:10.1137/19M1249527.
 - 22 Yi-Jun Chang and Seth Pettie. A time hierarchy theorem for the LOCAL model. *SIAM J. Comput.*, 48(1):33–69, 2019. doi:10.1137/17M1157957.
 - 23 Ilan Reuven Cohen, Binghui Peng, and David Wajc. Tight bounds for online edge coloring. In David Zuckerman, editor, *60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019, Baltimore, Maryland, USA, November 9-12, 2019*, pages 1–25. IEEE Computer Society, 2019. doi:10.1109/FOCS.2019.00010.
 - 24 Richard Cole and Uzi Vishkin. Deterministic coin tossing with applications to optimal parallel list ranking. *Inf. Control.*, 70(1):32–53, 1986. doi:10.1016/S0019-9958(86)80023-7.
 - 25 Andrzej Czygrinow, Michal Hanckowiak, and Michal Karonski. Distributed $o(\Delta \log(n))$ -edge-coloring algorithm. In Friedhelm Meyer auf der Heide, editor, *Algorithms - ESA 2001, 9th Annual European Symposium, Aarhus, Denmark, August 28-31, 2001, Proceedings*, volume 2161 of *Lecture Notes in Computer Science*, pages 345–355. Springer, 2001. doi:10.1007/3-540-44676-1_29.
 - 26 Peter Davies. Improved distributed algorithms for the lovász local lemma and edge coloring. In Nikhil Bansal and Viswanath Nagarajan, editors, *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023*, pages 4273–4295. SIAM, 2023. doi:10.1137/1.9781611977554.CH163.
 - 27 Devdatt P. Dubhashi, David A. Grable, and Alessandro Panconesi. Near-optimal, distributed edge colouring via the nibble method. *Theor. Comput. Sci.*, 203(2):225–251, 1998. doi:10.1016/S0304-3975(98)00022-X.
 - 28 Michael Elkin, Seth Pettie, and Hsin-Hao Su. $(2\Delta - 1)$ -edge-coloring is much easier than maximal matching in the distributed setting. In Piotr Indyk, editor, *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2015, San Diego, CA, USA, January 4-6, 2015*, pages 355–370. SIAM, 2015. doi:10.1137/1.9781611973730.26.
 - 29 Salwa Faour, Mohsen Ghaffari, Christoph Grunau, Fabian Kuhn, and Václav Rozhon. Local distributed rounding: Generalized to mis, matching, set cover, and beyond. In Nikhil Bansal and Viswanath Nagarajan, editors, *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023*, pages 4409–4447. SIAM, 2023. doi:10.1137/1.9781611977554.CH168.
 - 30 Manuela Fischer and Mohsen Ghaffari. Sublogarithmic distributed algorithms for lovász local lemma, and the complexity hierarchy. In Andréa W. Richa, editor, *31st International Symposium on Distributed Computing, DISC 2017, October 16-20, 2017, Vienna, Austria*, volume 91 of *LIPICs*, pages 18:1–18:16. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPICS.DISC.2017.18.
 - 31 Manuela Fischer, Mohsen Ghaffari, and Fabian Kuhn. Deterministic distributed edge-coloring via hypergraph maximal matching. In Chris Umans, editor, *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pages 180–191. IEEE Computer Society, 2017. doi:10.1109/FOCS.2017.25.
 - 32 Manuela Fischer, Magnús M. Halldórsson, and Yannic Maus. Fast distributed brooks’ theorem. In Nikhil Bansal and Viswanath Nagarajan, editors, *Proceedings of the 2023 ACM-SIAM Symposium on Discrete Algorithms, SODA 2023, Florence, Italy, January 22-25, 2023*, pages 2567–2588. SIAM, 2023. doi:10.1137/1.9781611977554.CH98.
 - 33 Pierre Fraigniaud, Marc Heinrich, and Adrian Kosowski. Local conflict coloring. In Irit Dinur, editor, *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 625–634. IEEE Computer Society, 2016. doi:10.1109/FOCS.2016.73.
 - 34 Beat Gfeller and Elias Vicari. A randomized distributed algorithm for the maximal independent set problem in growth-bounded graphs. In Indranil Gupta and Roger Wattenhofer,

- editors, *Proceedings of the Twenty-Sixth Annual ACM Symposium on Principles of Distributed Computing, PODC 2007, Portland, Oregon, USA, August 12-15, 2007*, pages 53–60. ACM, 2007. doi:10.1145/1281100.1281111.
- 35 Mohsen Ghaffari. An improved distributed algorithm for maximal independent set. In Robert Krauthgamer, editor, *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 270–277. SIAM, 2016. doi:10.1137/1.9781611974331.CH20.
 - 36 Mohsen Ghaffari and Christoph Grunau. Faster deterministic distributed MIS and approximate matching. In Barna Saha and Rocco A. Servedio, editors, *Proceedings of the 55th Annual ACM Symposium on Theory of Computing, STOC 2023, Orlando, FL, USA, June 20-23, 2023*, pages 1777–1790. ACM, 2023. doi:10.1145/3564246.3585243.
 - 37 Mohsen Ghaffari and Christoph Grunau. Near-optimal deterministic network decomposition and ruling set, and improved MIS. In *65th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2024, Chicago, IL, USA, October 27-30, 2024*, pages 2148–2179. IEEE, 2024. doi:10.1109/FOCS61266.2024.00007.
 - 38 Mohsen Ghaffari, Christoph Grunau, and Václav Rozhon. Improved deterministic network decomposition. In Dániel Marx, editor, *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 2904–2923. SIAM, 2021. doi:10.1137/1.9781611976465.173.
 - 39 Mohsen Ghaffari, David G. Harris, and Fabian Kuhn. On derandomizing local distributed algorithms. In Mikkel Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 662–673. IEEE Computer Society, 2018. doi:10.1109/FOCS.2018.00069.
 - 40 Mohsen Ghaffari, Juho Hirvonen, Fabian Kuhn, and Yannic Maus. Improved distributed delta-coloring. In Calvin Newport and Idit Keidar, editors, *Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing, PODC 2018, Egham, United Kingdom, July 23-27, 2018*, pages 427–436. ACM, 2018. URL: <https://dl.acm.org/citation.cfm?id=3212764>.
 - 41 Mohsen Ghaffari, Juho Hirvonen, Fabian Kuhn, Yannic Maus, Jukka Suomela, and Jara Uitto. Improved distributed degree splitting and edge coloring. In Andréa W. Richa, editor, *31st International Symposium on Distributed Computing, DISC 2017, October 16-20, 2017, Vienna, Austria*, volume 91 of *LIPICs*, pages 19:1–19:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPICs.DISC.2017.19.
 - 42 Mohsen Ghaffari, Juho Hirvonen, Fabian Kuhn, Yannic Maus, Jukka Suomela, and Jara Uitto. Improved distributed degree splitting and edge coloring. *Distributed Comput.*, 33(3-4):293–310, 2020. doi:10.1007/S00446-018-00346-8.
 - 43 Mohsen Ghaffari and Fabian Kuhn. Deterministic distributed vertex coloring: Simpler, faster, and without network decomposition. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 1009–1020. IEEE, 2021. doi:10.1109/FOCS52979.2021.00101.
 - 44 Mohsen Ghaffari, Fabian Kuhn, Yannic Maus, and Jara Uitto. Deterministic distributed edge-coloring with fewer colors. In Ilias Diakonikolas, David Kempe, and Monika Henzinger, editors, *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2018, Los Angeles, CA, USA, June 25-29, 2018*, pages 418–430. ACM, 2018. doi:10.1145/3188745.3188906.
 - 45 Mohsen Ghaffari and Hsin-Hao Su. Distributed degree splitting, edge coloring, and orientations. In Philip N. Klein, editor, *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2017, Barcelona, Spain, Hotel Porta Fira, January 16-19*, pages 2505–2523. SIAM, 2017. doi:10.1137/1.9781611974782.166.
 - 46 Philip Hall. On representatives of subsets. *Classic Papers in Combinatorics*, pages 58–62, 1987.
 - 47 Magnús M. Halldórsson, Yannic Maus, and Alexandre Nolin. Fast distributed vertex splitting with applications. In Christian Scheideler, editor, *36th International Symposium on Distributed*

- Computing, DISC 2022, October 25-27, 2022, Augusta, Georgia, USA*, volume 246 of *LIPICs*, pages 26:1–26:24. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022. doi:10.4230/LIPICs.DISC.2022.26.
- 48 David G. Harris. Distributed local approximation algorithms for maximum matching in graphs and hypergraphs. In David Zuckerman, editor, *60th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2019, Baltimore, Maryland, USA, November 9-12, 2019*, pages 700–724. IEEE Computer Society, 2019. doi:10.1109/FOCS.2019.00048.
 - 49 David G. Harris. Distributed local approximation algorithms for maximum matching in graphs and hypergraphs. *SIAM J. Comput.*, 49(4):711–746, 2020. doi:10.1137/19M1279241.
 - 50 Manuel Jakob and Yannic Maus. Towards optimal distributed delta coloring. *CoRR*, abs/2504.03080, 2025. doi:10.48550/arXiv.2504.03080.
 - 51 Manuel Jakob, Yannic Maus, and Florian Schager. Towards optimal distributed edge coloring with fewer colors, 2025. arXiv:2504.13003.
 - 52 Fabian Kuhn, Yannic Maus, and Simon Weidner. Deterministic distributed ruling sets of line graphs. In Zvi Lotker and Boaz Patt-Shamir, editors, *Structural Information and Communication Complexity - 25th International Colloquium, SIROCCO 2018, Ma'ale HaHamisha, Israel, June 18-21, 2018, Revised Selected Papers*, volume 11085 of *Lecture Notes in Computer Science*, pages 193–208. Springer, 2018. doi:10.1007/978-3-030-01325-7_19.
 - 53 Fabian Kuhn and Roger Wattenhofer. On the complexity of distributed graph coloring. In Eric Ruppert and Dahlia Malkhi, editors, *Proceedings of the Twenty-Fifth Annual ACM Symposium on Principles of Distributed Computing, PODC 2006, Denver, CO, USA, July 23-26, 2006*, pages 7–15. ACM, 2006. doi:10.1145/1146381.1146387.
 - 54 Nathan Linial. Distributive graph algorithms-global solutions from local data. In *28th Annual Symposium on Foundations of Computer Science, Los Angeles, California, USA, 27-29 October 1987*, pages 331–335. IEEE Computer Society, 1987. doi:10.1109/SFCS.1987.20.
 - 55 Nathan Linial. Locality in distributed graph algorithms. *SIAM J. Comput.*, 21(1):193–201, 1992. doi:10.1137/0221015.
 - 56 Michael Luby. A simple parallel algorithm for the maximal independent set problem. In Robert Sedgewick, editor, *Proceedings of the 17th Annual ACM Symposium on Theory of Computing, May 6-8, 1985, Providence, Rhode Island, USA*, pages 1–10. ACM, 1985. doi:10.1145/22145.22146.
 - 57 Yannic Maus and Tigran Tonoyan. Local conflict coloring revisited: Linial for lists. In Hagit Attiya, editor, *34th International Symposium on Distributed Computing, DISC 2020, October 12-16, 2020, Virtual Conference*, volume 179 of *LIPICs*, pages 16:1–16:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2020. doi:10.4230/LIPICs.DISC.2020.16.
 - 58 Yannic Maus and Jara Uitto. Efficient CONGEST algorithms for the lovász local lemma. In Seth Gilbert, editor, *35th International Symposium on Distributed Computing, DISC 2021, October 4-8, 2021, Freiburg, Germany (Virtual Conference)*, volume 209 of *LIPICs*, pages 31:1–31:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICs.DISC.2021.31.
 - 59 Moni Naor. A lower bound on probabilistic algorithms for distributive ring coloring. *SIAM J. Discret. Math.*, 4(3):409–412, 1991. doi:10.1137/0404036.
 - 60 Alessandro Panconesi and Romeo Rizzi. Some simple distributed algorithms for sparse networks. *Distributed Comput.*, 14(2):97–100, 2001. doi:10.1007/PL00008932.
 - 61 Alessandro Panconesi and Aravind Srinivasan. Improved distributed algorithms for coloring and network decomposition problems. In S. Rao Kosaraju, Mike Fellows, Avi Wigderson, and John A. Ellis, editors, *Proceedings of the 24th Annual ACM Symposium on Theory of Computing, May 4-6, 1992, Victoria, British Columbia, Canada*, pages 581–592. ACM, 1992. doi:10.1145/129712.129769.
 - 62 Alessandro Panconesi and Aravind Srinivasan. The local nature of delta-coloring and its algorithmic applications. *Comb.*, 15(2):255–280, 1995. doi:10.1007/BF01200759.

- 63 Alessandro Panconesi and Aravind Srinivasan. On the complexity of distributed network decomposition. *J. Algorithms*, 20(2):356–374, 1996. doi:10.1006/JAGM.1996.0017.
- 64 Václav Rozhon, Michael Elkin, Christoph Grunau, and Bernhard Haeupler. Deterministic low-diameter decompositions for weighted graphs and distributed and parallel applications. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*, pages 1114–1121. IEEE, 2022. doi:10.1109/FOCS54457.2022.00107.
- 65 Václav Rozhon and Mohsen Ghaffari. Polylogarithmic-time deterministic network decomposition and distributed derandomization. In Konstantin Makarychev, Yuri Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy, editors, *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, pages 350–363. ACM, 2020. doi:10.1145/3357713.3384298.
- 66 Amin Saberi and David Wajc. The greedy algorithm is not optimal for on-line edge coloring. In Nikhil Bansal, Emanuela Merelli, and James Worrell, editors, *48th International Colloquium on Automata, Languages, and Programming, ICALP 2021, July 12-16, 2021, Glasgow, Scotland (Virtual Conference)*, volume 198 of *LIPIcs*, pages 109:1–109:18. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPICS.ICALP.2021.109.
- 67 Johannes Schneider, Michael Elkin, and Roger Wattenhofer. Symmetry breaking depending on the chromatic number or the neighborhood growth. *Theor. Comput. Sci.*, 509:40–50, 2013. doi:10.1016/J.TCS.2012.09.004.
- 68 Hsin-Hao Su and Hoa T. Vu. Towards the locality of vizing’s theorem. In Moses Charikar and Edith Cohen, editors, *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019*, pages 355–364. ACM, 2019. doi:10.1145/3313276.3316393.
- 69 Vadim G Vizing. On an estimate of the chromatic class of a p-graph. *Diskret analiz*, 3:25–30, 1964.

A

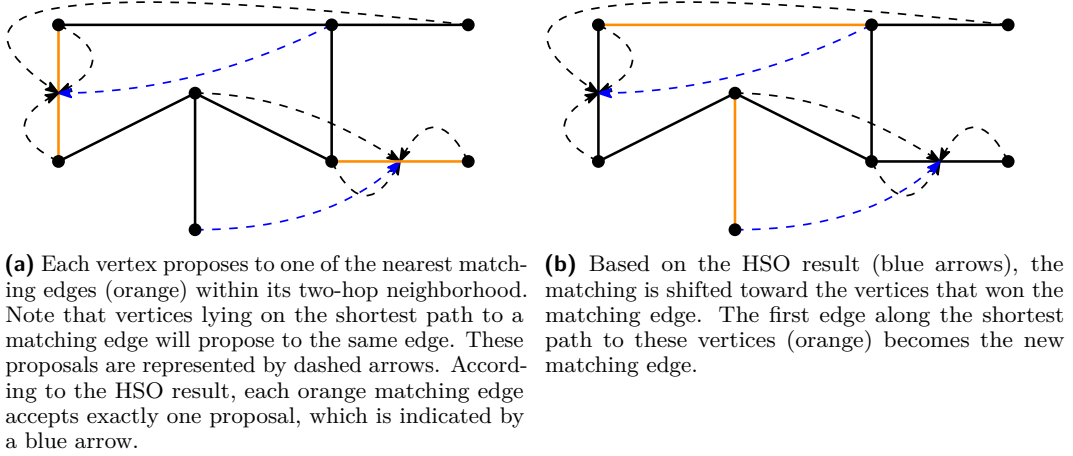
 Replacing MIS and maximal matching (Proof of Theorem 1)

In this subsection we show that computing an MIS is not strictly necessary for our reduction. Indeed, using a $(2, k)$ -ruling set changes only the diameter of our clusters. Since increasing the cluster diameter increases the overhead for simulating the auxiliary hypergraph, this induces a tradeoff between k and the runtime of computing a $(2, k)$ -ruling set. Setting $k = \mathcal{O}(\log \Delta)$ turns out to be the best choice for our purposes. Secondly, we show how to substitute the maximal matching by a 2-edge ruling set. This change is technically more involved, as it requires us to generalize the definition of our auxiliary hypergraph. The necessary changes to Algorithm 1 are also highlighted red in Algorithm 3.

Algorithm 3 Reduction from $(2\Delta - 2)$ -edge coloring to $(2\Delta - 1)$ -edge coloring (high level overview).

-
- | | |
|---|----------------|
| 1: Compute a $(2, \mathcal{O}(\log \Delta))$ -ruling set $\mathcal{R}$ on G^8 | ▷ Corollary 31 |
| 2: $\mathcal{C}, E_T \leftarrow \text{CLUSTER}(G, \mathcal{R})$ | ▷ Algorithm 2 |
| 3: Compute a $(2\Delta - 3)$ -edge coloring φ of $G - E_T$ | ▷ Lemma 18 |
| 4: Compute a 2-edge ruling set $E_{\mathcal{R}}$ of T | ▷ Theorem 33 |
| 5: Compute 2-HSO $\psi : \mathcal{C} \rightarrow E_{\mathcal{R}}$ in $H = (\mathcal{C}, E_{\mathcal{R}})$ | ▷ Theorem 34 |
| 6: Rearrange the matching edges via $\chi : E_{\mathcal{R}} \rightarrow E'_{\mathcal{R}}$ | ▷ Lemma 24 |
| 7: for each cluster $\mathfrak{C} \in \mathcal{C}$ in parallel do | |
| 8: Change the colors of the edges in $\chi(\psi(\mathfrak{C}))$ if necessary | ▷ Lemma 25 |
| 9: end for | |
| 10: Extend φ to a $(2\Delta - 2)$ -edge coloring φ^* of G | ▷ Lemma 14 |
-

Secondly, we show that we can replace the maximal matching M by a cheaper 2-edge ruling set $E_{\mathcal{R}}$ of $G - E_T$. In this case, every leaf-vertex of a cluster sees at least one ruling set edge in its 3-hop neighborhood. Here we need to be a little bit more careful in how we set up our hypergraph. Firstly, each vertex may only send one proposal to one of its closest ruling set edges. Secondly, each vertex may only propose to a ruling set edge $e \in E_{\mathcal{R}}$, if all the vertices on a shortest path towards this have proposed to e as well. For an illustration we refer to Figure 6.



■ **Figure 6** Illustration of a simplified example showing how the HSO is modeled as a proposal instance for assigning clusters two exclusive edges.

For each cluster $C \in \mathcal{C}$ we select an arbitrary set $S(C) \subseteq V(C)$ of at least $4 \cdot \Delta^3$ leaves which are all at the same distance from the center.

► **Definition 26** (Proposal scheme). *Each cluster $C \in \mathcal{C}$ sends proposals to matching edges according to the following rules:*

- *Each vertex $v \in S(C)$ proposes to exactly one matching edge $e \in E_{\mathcal{R}}$.*
- *If v proposes to e , there may not be an edge $e' \in E_{\mathcal{R}}$ that is closer to v than e .*
- *If v proposes to e , every vertex on the shortest path from v to e must also propose to e .*

► **Definition 27.** *Let $e \in E_{\mathcal{R}}$ and $v \in V$ be the vertex which sent the winning proposal to e .*

- *If v is incident to e , then we simply add e to M' .*
- *If v is one hop away from e , let u be the endpoint of e which is adjacent to v and add the edge uv to M' .*
- *If v is two hops away from e , let u be the next vertex on a shortest path from v to e and add the edge uv to M' .*

Now we prove an analogous result to Lemma 24.

► **Lemma 28.** *The edge set M' is a proper matching in $G - E_T$ and every edge $e \in M'$ is adjacent to the vertex that won the proposal for e .*

Proof. Recall that every vertex $v \in V$ sends out at most one proposal. Hence, there is no vertex that wins more than one edge in $E_{\mathcal{R}}$. Assume that v won the proposal for the edge $e = uv \in E_{\mathcal{R}}$. If v is incident to e there is no change between $E_{\mathcal{R}}$ and M' . If v is one hop away from e , then v is the only vertex that gains an incident edge in M' compared to $E_{\mathcal{R}}$. Since e is one of the closest edges to v in $E_{\mathcal{R}}$ there cannot be an edge $e' \in E_{\mathcal{R}}$ that is already

incident to v . In the final case that v is two hops away from e , both v and the next vertex u on a shortest path from v to e gain an incident edge in $E_{\mathcal{R}}$. Note that every vertex is at most two hops away from its closest edge in $E_{\mathcal{R}}$, since $E_{\mathcal{R}}$ is a 2-edge ruling set. By our proposal scheme, u must also have proposed to v . Therefore u cannot have won its proposal and u cannot be incident to an edge in $E_{\mathcal{R}}$. Hence, both v and u are incident to only one edge in M' . Now assume that there is another vertex $v' = v$ that adds the edge $v'u$ to M' . This means that v' must have proposed to the same edge as u . But v already won the proposal for this edge, a contradiction! Therefore every vertex is incident to at most one edge in M' . ◀

► **Theorem 1.** *There is a deterministic LOCAL algorithm computing a $(2\Delta - 2)$ -edge coloring in any n -vertex graph with maximum degree Δ in $\mathcal{O}(\log n) + T_{2\Delta-1}(n, \Delta - 1)$ rounds.*

Proof. The $(2, \mathcal{O}(\log \Delta))$ -ruling set can be computed in $\mathcal{O}(\log \Delta + \log^* n)$ rounds according to Corollary 31. As an immediate consequence, the diameter of our clusters now increases to $\mathcal{O}(\log \Delta)$. This implies that simulating the hypergraph now incurs a $\mathcal{O}(\log \Delta)$ factor overhead. Therefore the complexity of computing a HSO increases to $\mathcal{O}(\log_{\Delta} n \cdot \log \Delta) = \mathcal{O}(\log n)$ rounds. The $(2\Delta - 3)$ -edge coloring of $G - E_T$ remains unchanged. Further, we can also relax the requirements for our matching M to 2-edge ruling set, which can be computed in $\mathcal{O}(\log^* n)$ rounds. Therefore the total runtime of our reduction works out to be

$$\mathcal{O}(\log \Delta + \log^* n) + \mathcal{O}(\log \Delta) + T_{2\Delta-1}(n) + \mathcal{O}(\log^* n) + \mathcal{O}(\log \Delta \cdot \log n) = \mathcal{O}(\log n) + T_{2\Delta-1}(n). \blacktriangleleft$$

► **Corollary 3.** *There are deterministic LOCAL algorithms computing a $(2\Delta - 2)$ -edge coloring in any n -vertex graph with maximum degree Δ within either $\mathcal{O}(\log^{12} \Delta + \log n)$ or $\tilde{\mathcal{O}}(\log^{5/3} n)$ rounds.*

Proof. For the first claim we plug in the $\mathcal{O}(\log^{12} \Delta + \log^* n)$ -round algorithm from Theorem 38 into Theorem 1. For the latter we plug in the $\tilde{\mathcal{O}}(\log^{5/3} n)$ -round MIS algorithm from Theorem 36 into Theorem 2 instead. ◀

B Randomized $(2\Delta - 2)$ -edge coloring algorithm

In order to achieve a $\mathcal{O}(\log \log n)$ -round randomized reduction, we only need to replace the ruling set algorithm by a faster randomized variant in Corollary 32 and the deterministic HSO algorithm by its randomized counterpart in Theorem 35. The changes to the optimized deterministic version are also highlighted red in Algorithm 4.

■ **Algorithm 4** Randomized $(2\Delta - 2)$ -edge coloring (high level overview).

1: Compute a $(2, \mathcal{O}(\log \Delta) \cap \mathcal{O}(\log \log n))$ -ruling set $\mathcal{R}$ on G ⁸	▷ Corollary 32
2: $\mathcal{C}, E_T \leftarrow \text{CLUSTER}(G, \mathcal{R})$	▷ Algorithm 2
3: Compute a $(2\Delta - 3)$ -edge coloring φ of $G - E_T$	▷ Lemma 18
4: Compute a 2-edge ruling set $E_{\mathcal{R}}$ of T	▷ Theorem 33
5: Compute 2-HSO $\psi : \mathcal{C} \rightarrow E_{\mathcal{R}}$ in $H = (\mathcal{C}, E_{\mathcal{R}})$	▷ Theorem 35
6: Rearrange the matching edges via $\chi : E_{\mathcal{R}} \rightarrow E'_{\mathcal{R}}$	▷ Lemma 24
7: for each cluster $\mathfrak{C} \in \mathcal{C}$ in parallel do	
8: Change the colors of the edges in $\chi(\psi(\mathfrak{C}))$ if necessary	▷ Lemma 25
9: end for	
10: Extend φ to a $(2\Delta - 2)$ -edge coloring of G	▷ Lemma 14

► **Theorem 4.** *There is a randomized LOCAL algorithm computing a $(2\Delta - 2)$ -edge coloring in any n -vertex graph with maximum degree Δ in $\mathcal{O}(\log \log n) + T_{2\Delta-1}^{\{\frac{1}{2}\}}(n, \Delta - 1)$ rounds.*

Proof. The $(2, \mathcal{O}(\log \Delta) \cap \mathcal{O}(\log \log n))$ -ruling set can be computed in $\mathcal{O}(\log \log n)$ rounds according to Corollary 32. Hence, each cluster has diameter at most $\mathcal{O}(\log \log n)$ and can be computed in $\mathcal{O}(\log \log n)$ rounds as well. For the 2-edge ruling set the deterministic algorithm from Theorem 33 running in $\mathcal{O}(\log^* n)$ rounds is still fast enough for our purposes here. According to Theorem 35 and Section 4.3 the hypergraph sinkless orientation can be computed in $\mathcal{O}(\log_{\Delta} \log n)$ rounds on the cluster graph, where we incur a $\mathcal{O}(\log \Delta)$ -round simulation overhead. Finally, the extension to the edges inside each cluster can be computed in $\mathcal{O}(\log \log n)$ rounds. Hence, the total runtime works out to be

$$\mathcal{O}(\log \log n) + T_{2\Delta-1}^{\{\frac{1}{2}\}}(n) + \mathcal{O}(\log^* n) + \mathcal{O}(\log \Delta \cdot \log_{\Delta} \log n) = \mathcal{O}(\log \log n) + T_{2\Delta-1}^{\{\frac{1}{2}\}}(n). \blacktriangleleft$$

► **Corollary 5.** *There are randomized distributed algorithms computing a $(2\Delta - 2)$ -edge coloring in any n -vertex graph with maximum degree Δ within either $\mathcal{O}(\log^{12} \Delta + \log \log n)$ or $\tilde{\mathcal{O}}(\log^{5/3} \log n)$ rounds of the LOCAL model.*

Proof. To prove the first claim we plug in the deterministic $\mathcal{O}(\log^{12} \Delta + \log^* n)$ -round algorithm from Theorem 38 into Theorem 4, while the second claim uses the randomized $\tilde{\mathcal{O}}(\log^{5/3} \log n)$ -round $(2\Delta - 1)$ -edge coloring algorithm from Corollary 37. $\blacktriangleleft$

Similar to the deterministic case we could also get a $\mathcal{O}(\log_{\Delta} \log n)$ -round reduction to MIS via replacing the $(2, \mathcal{O}(\log \Delta) \cap \mathcal{O}(\log \log n))$ -ruling set by an MIS. However, applying the shattering framework of [35] to the deterministic MIS algorithm from [37] still takes $\mathcal{O}(\log \Delta) + \tilde{\mathcal{O}}(\log^{8/3} \log n)$ rounds. Hence, the additional cost of computing an MIS currently outweighs the time saved from the faster reduction.

C Deferred proofs

Proof of Lemma 15. Let C be a cycle of even length and assume that all edges in $N(C)$ have been colored already. Our task is now to find a proper edge-coloring of C , where every edge has a list of available colors of size two. If all edges have the same two colors available, then we can color the cycle alternately with these two colors. Otherwise, there exist two adjacent edges e_1, e_2 with different sets of available colors. Color e_1 with a color that is not available for e_2 . Then we traverse the cycle starting at e_1 and going in the opposite direction of e_2 and let each edge pick one of its available colors. Since e_1 picked a color that is not available at e_2 , there is also a color remaining for e_2 at the end. Since H is connected, every edge lies on a path towards an even cycle. Fix such an even cycle C and define

$$E_k = \{e \in E(H) : \text{dist}(e, C) = k\}.$$

Since every edge in E_k has at least one neighbor in E_{k-1} for $k > 1$, we can greedily assign colors to these edges, starting from the highest layer. Finally, every edge in C has a list of two available colors and can therefore be properly colored as well. $\blacktriangleleft$

Proof of Lemma 16. For two nodes $u, u' \in V(C)$ let $P(u, u')$ denote the unique path in T between u and u' . We claim that each node $u \in V(C)$ is incident to at most one edge that is not part of $E(T)$ and thus the degree of all internal nodes of T is at least $\Delta - 1$. First we observe that any edge in $E(C) \setminus E(T)$ can only connect vertices which are at most one level apart in T . Any additional edge in $E(C) \setminus E(T)$ from a vertex v_{ℓ} in layer ℓ to a

vertex $v_{\ell+1}$ in layer $\ell + 1$ in T immediately induces an even cycle. To wit, let u be the least common ancestor of v_ℓ and $v_{\ell+1}$ and observe that the cycle $P(u, v_\ell) \cup \{v_\ell, v_{\ell+1}\} \cup P(u, v_{\ell+1})$ has length $2 \cdot |P(u, v_{\ell+1})|$. Next, let $u \in V(C)$ be a vertex in layer ℓ with two neighbors w, w' in the same layer ℓ . Then, $P(v, w) \cup \{w, u\} \cup \{u, w'\} \cup P(v, w')$ is a cycle of length $2(\ell + 1)$. Hence each vertex $u \in V(C)$ is adjacent to at most one edge from $E(C) \setminus E(T)$. Since C contains the complete k -hop neighborhood of v this implies that there are at least $(\Delta - 2)^k$ vertices in layer k of T . Every such vertex either has a leaf-descendant in T or is a leaf itself. $\blacktriangleleft$

D Complementary subroutines

All subroutines from the existing literature used throughout our algorithms are stated here explicitly.

D.1 Ruling sets

In this subsection we collect and combine various ruling set algorithms from the distributed literature that we will use as subroutines for our edge coloring algorithm.

► **Theorem 29** (ruling set algorithm [34, Theorem 8]). *There is a randomized distributed algorithm that computes a $(1, \mathcal{O}(\log \log \Delta))$ -ruling set with induced degree at most $\mathcal{O}(\log^5 n)$ in any n -vertex graph with maximum degree Δ in $\mathcal{O}(\log \log \Delta)$ rounds*

► **Theorem 30** (ruling set algorithm [67, Theorem 3]). *Let $G = (V, E)$ and $W \subseteq V$. Given a d -coloring of G , there is a deterministic distributed algorithm that computes a $(2, c)$ -ruling set for W in time $\mathcal{O}(c \cdot d^{1/c})$.*

► **Corollary 31.** *There is a deterministic distributed algorithm that computes a $(2, \mathcal{O}(\log \Delta))$ -ruling set in any graph G in $\mathcal{O}(\log \Delta + \log^* n)$ rounds of the LOCAL model.*

Proof. We start by computing a $\mathcal{O}(\Delta^2)$ -vertex coloring φ of G in $\mathcal{O}(\log^* n)$ rounds using Linial's algorithm [54, 55]. Then, we apply Theorem 30 to φ for $c = \mathcal{O}(\log \Delta)$ to get a $(2, \mathcal{O}(\log \Delta))$ -ruling set in time $\mathcal{O}(c \cdot d^{1/c}) = \mathcal{O}(\log \Delta \cdot \Delta^{2/\log \Delta}) = \mathcal{O}(\log \Delta)$. $\blacktriangleleft$

► **Corollary 32.** *There is a randomized distributed algorithm that computes a $(2, \mathcal{O}(\log \Delta) \cap \mathcal{O}(\log \log n))$ -ruling set in any graph G in $\mathcal{O}(\log \log n)$ rounds of the LOCAL model.*

Proof. If $\Delta = \mathcal{O}(\log n)$, we can simply use the deterministic algorithm from Corollary 31 to get an $(2, \mathcal{O}(\log \Delta))$ -ruling set (which is also a $(2, \mathcal{O}(\log \log n))$ -ruling set) in $\mathcal{O}(\log \Delta + \log^* n) = \mathcal{O}(\log \log n)$ rounds. Otherwise, for $\Delta = \Omega(\log n)$ we first apply Theorem 29 to get a $(1, \mathcal{O}(\log \log \Delta))$ -ruling set S_0 with induced degree at most $\mathcal{O}(\log^5 n)$. Next we compute a $\mathcal{O}(\log^{10} n)$ -coloring χ of $G[S_0]$ using Linial's algorithm [54, 55]. Finally using χ as the input coloring and setting $c = \mathcal{O}(\log \log n)$, we get a $(2, \mathcal{O}(\log \log n))$ ruling set S_1 for S_0 in

$$\mathcal{O}(c \cdot d^{1/c}) = \mathcal{O}(\log \log n \cdot \log^{10/\log \log n} n) = \mathcal{O}(\log \log n).$$

randomized time. Since every vertex $v \in V$ is at most $\mathcal{O}(\log \log \Delta) = \mathcal{O}(\log \log n)$ hops away from a vertex in S_0 and every vertex in S_0 is at most $\mathcal{O}(\log \log n) = \mathcal{O}(\log \Delta)$ hops away from a vertex in S_1 , we get that S_1 is both a $(2, \mathcal{O}(\log \Delta))$ and a $(2, \mathcal{O}(\log \log n))$ -ruling set for V . $\blacktriangleleft$

Another subroutine that we will use in the optimized version of our algorithm computes a relaxation of a maximal matching.

► **Theorem 33** (2-ruling edge set algorithm [52, Theorem 1.3]). *There is a deterministic distributed algorithm that computes a 2-ruling edge set of G in $\Theta(\log^* n)$ rounds in the CONGEST model.*

D.2 Hypergraph sinkless orientation

► **Theorem 34** (deterministic HSO [15, Theorem 1.4]). *There is a deterministic distributed algorithm for computing an HSO in any n -vertex multihypergraph $H = (V, E)$ with maximum rank r and minimum degree $\delta > r$ in $\mathcal{O}(\log_{\delta/r} n)$ rounds.*

► **Theorem 35** (randomized HSO [15, Theorem 1.5]). *There is a randomized distributed algorithm that w.h.p. computes an HSO on any hypergraph with maximum rank r and minimum degree $\delta \geq 320r \log r$ with runtime $\mathcal{O}(\log_{\delta/r} \delta + \log_{\delta/r} \log n)$.*

D.3 Maximal Independent Set

► **Theorem 36** (deterministic MIS [37, Theorem 1.3]). *There is a deterministic distributed algorithm that in any n -vertex graph $G = (V, E)$ computes an MIS in $\tilde{\mathcal{O}}(\log^{5/3} n)$ rounds of the LOCAL model. This complexity also applies for maximal matching, $(\deg + 1)$ -list vertex coloring, and $(2 \cdot \deg - 1)$ -list edge coloring.*

► **Corollary 37** (randomized MIS [37, Theorem 1.3] & [20]). *There is a randomized distributed algorithm, in the LOCAL model, that computes a $(\Delta + 1)$ -vertex coloring in $\tilde{\mathcal{O}}(\log^{5/3} \log n)$ rounds. The same holds also for the $(2\Delta - 1)$ -edge coloring problem.*

D.4 Edge coloring

► **Theorem 38** (deterministic $(2\Delta - 1)$ -edge coloring [4, Theorem D.4]). *The $(\deg(e) + 1)$ -list edge coloring problem can be solved in $\mathcal{O}(\log^7 C \cdot \log^5 n + \log^* n)$ deterministic rounds in the LOCAL model, where C is the size of the color space.*

► **Theorem 39** (deterministic $(3\Delta/2)$ -edge coloring [15, Theorem 1.2]). *There is a deterministic $\mathcal{O}(\Delta^2 \cdot \log n)$ -round LOCAL algorithm that computes a $(3\Delta/2)$ -edge coloring on any n -vertex graph with maximum degree Δ .*