$(\Delta + 1)$ Coloring in the Congested Clique Model

Merav Parter
Weizmann IS, Rehovot, Israel
merav.parter@weizmann.ac.il

Abstract

In this paper, we present improved algorithms for the $(\Delta + 1)$ (vertex) coloring problem in the *Congested Clique* model of distributed computing. In this model, the input is a graph on n nodes, initially each node knows only its incident edges, and per round each two nodes can exchange $O(\log n)$ bits of information.

Our key result is a randomized $(\Delta + 1)$ vertex coloring algorithm that works in $O(\log\log\Delta \cdot \log^* \Delta)$-rounds. This is achieved by combining the recent breakthrough result of [Chang-Li-Pettie, STOC'18] in the LOCAL model and a degree reduction technique. We also get the following results with high probability: (1) $(\Delta + 1)$-coloring for $\Delta = O((n/\log n)^{1-\epsilon})$ for any $\epsilon \in (0, 1)$, within $O(\log(1/\epsilon)\log^* \Delta)$ rounds, and (2) $(\Delta + \Delta^{1/2+o(1)})$-coloring within $O(\log^* \Delta)$ rounds.
Turning to *deterministic* algorithms, we show a $(\Delta+1)$-coloring algorithm that works in $O(\log \Delta)$ rounds. Our new bounds provide *exponential* improvements over the state of the art.

2012 ACM Subject Classification Theory of computation → Distributed algorithms

Keywords and phrases Distributed Graph Algorithms, Coloring, Congested Clique

Digital Object Identifier 10.4230/LIPIcs.ICALP.2018.160

Related Version A full version of the paper is available at [22], `https://arxiv.org/abs/1805.02457`.

Acknowledgements I am very grateful to Hsin-Hao Su, Eylon Yogev, Seth Pettie, Yi-Jun Chang and the anonymous reviewers for helpful comments.

1 Introduction & Related Work

Graph coloring is one of the most central symmetry breaking problems, with a wide range of applications to distributed systems and wireless networks. The most studied coloring problem is the $(\Delta + 1)$ *vertex* coloring in which all nodes are given the *same* palette of $\Delta + 1$ colors, where Δ is the maximum degree in the graph. Vertex coloring among other LCL problems[1] (e.g., MIS, matching) are traditionally studied in the LOCAL model in which any two neighboring vertices in the input graph can exchange arbitrarily long messages.

In recent years there has been a tremendous progress in the understanding of the randomized and the deterministic complexities of many LCL problems in the LOCAL model [6, 20, 8, 9, 1]. Putting our focus on the $(\Delta + 1)$ coloring problem, in a seminal work, Schneider and Wattenhofer [23] showed that increasing the number of colors from $\Delta + 1$ to $(1 + \epsilon)\Delta$ has a dramatic effect on the round complexity and coloring can be computed in just $O(\log^* n)$ rounds when $\epsilon = \Omega(1)$ and $\Delta > \mathsf{poly}\log n$. This has led to two recent breakthroughs. Harris, Schneider and Su [13] showed an $O(\sqrt{\log \Delta})$-round algorithm for $(\Delta + 1)$ coloring, providing a separation for the first time between MIS and coloring (due to the MIS lower bound of [17]). In a recent follow-up breakthrough, Chang, Li and Pettie [7] extended the

[1] LCL stands for Locally Checkable Labelling problems, see [21].

45th International Colloquium on Automata, Languages, and Programming (ICALP 2018).
Editors: Ioannis Chatzigiannakis, Christos Kaklamanis, Dániel Marx, and Donald Sannella;
Article No. 160; pp. 160:1–160:14
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

technique of [13] to obtain the remarkable and quite extraordinary round complexity of $O(\log^* n + \mathsf{Det}_{\deg}(\mathsf{polylog} n))$ for the $(\Delta+1)$-list coloring problem where $\mathsf{Det}_{\deg}(n')$ is the deterministic round complexity of $(\deg+1)$ list coloring algorithm[2] in n'-vertex graph. Both of these recent breakthroughs use messages of large size, potentially of $\Omega(n)$ bits.

In view of these recent advances, the understanding of LCL problems in bandwidth-restricted models is much more lacking. Among these models, the congested clique model [19], which allows all-to-all communication has attracted a lot of attention in the last decade and more recently, in the context of LCL problems [4, 15, 14, 5, 12, 24]. In the congested clique model, each node can send $O(\log n)$ bits of information to any node in the network (i.e., even if they are not connected in the input graph). The ubiquitous of overlay networks and large scale distributed networks make the congested clique model far more relevant (compared to the LOCAL and the CONGEST models) in certain settings.

Randomized LCL in the Congested Clique Model. Starting with Barenboim et al. [2], currently, all efficient randomized algorithms for classical LCL problems have the following structure: an initial randomized phase and a *post-shattering* deterministic phase. The shattering effect of the randomized phase which dates back to Beck [3], breaks the graph into subproblems of $\mathsf{poly}\log n$ size to be solved deterministically. In the congested-clique model, the shattering effect has an even more dramatic effect. Usually, a node survives (i.e., remained undecided) the randomized phase with probability of $1/\mathsf{poly}(\Delta)$. Hence, in expectation the size of the remaining unsolved graph is[3] $O(n)$. At that point, the entire unsolved subgraph can be solved in $O(1)$ rounds, using standard congested clique tools (e.g., the routing algorithm by Lenzen [18]). Thus, as long as the main randomized part uses short messages, the congested clique model "immediately" enjoys an improved round complexity compared to that of the LOCAL model.

In a recent work [12], Ghaffari took it few steps farther and showed an $\widetilde{O}(\sqrt{\log \Delta})$-round randomized algorithm for MIS in the congested clique model, improving upon the state-of-the-art complexity of $O(\log \Delta + 2^{O(\sqrt{\log\log n})})$ rounds in the LOCAL model, also by Ghaffari [11]. When considering the $(\Delta+1)$ coloring problem, the picture is somewhat puzzling. On the one hand, in the LOCAL model, $(\Delta+1)$ coloring is provably simpler then MIS. However, since all existing $o(\log \Delta)$-round algorithms for $(\Delta+1)$ coloring in the LOCAL model, use large messages, it is not even clear if the power of all-to-all communication in the congested clique model can compensate for its bandwidth limitation and outperform the LOCAL round complexity, not to say, even just match it. We note that on hind-sight, the situation for MIS in the congested clique was somewhat more hopeful (compared to coloring), for the following reason. The randomized phase of Ghaffari's MIS algorithm although being in the LOCAL model [11], used *small* messages and hence could be implemented in the CONGEST model with the same round complexity. To sum up, currently, there is no $o(\log \Delta)$-round algorithm for $(\Delta+1)$ coloring in any bandwidth restricted model, not even in the congested-clique.

Derandomization of LCL in the Congested-Clique Model. There exists a curious gap between the known complexities of randomized and deterministic solutions for local problems in the LOCAL model ([6, 20]). Censor et al. [5] initiated the study of *deterministic* LCL algorithms in the congested clique model by means of derandomization. The main take home message of [5] is as follows: for most of the classical LCL problems there are $\mathsf{poly}\log n$ round

[2] In the $(\deg+1)$ list coloring problem, each vertex v is given a palette with $\deg(v, G) + 1$ colors.
[3] Using the bounded dependencies between decisions, this holds also with high probability.

randomized algorithms (even in the CONGEST model). For these algorithms, it is usually sufficient that the random choices made by vertices are *almost* independent. This implies that each round of the randomized algorithm can be simulated by giving all nodes a shared random seed of $\mathsf{poly}\log n$ bits. To dernadomize a single round of the randomized algorithm, nodes should compute (deterministically) a seed which is at least as “good"[4] as a random seed would be. To compute this seed, they need to estimate their “local progress" when simulating the random choices using that seed. Combining the techniques of conditional expectation, pessimistic estimators and bounded independence leads to a simple “voting"-like algorithm in which the bits of the seed are computed *bit-by-bit*. Once all bits of the seed are computed, it is used to simulate the random choices of that round. For a recent work on other complexity aspects in the congested clique, see [16].

1.1 Main Results and Our Approach

In this paper, we show that the power of all-to-all communication compensates for the bandwidth restriction of the model:

▶ **Theorem 1.** *There is a randomized algorithm that computes a $(\Delta + 1)$ coloring in $O(\log\log\Delta \cdot \log^* n)$ rounds of the congested clique model, with high probability*[5].

This significantly improves over the state-of-the-art of $O(\log\Delta)$-round algorithm for $(\Delta + 1)$ in the congested clique model. It should also be compared with the round complexity of $(2^{O(\sqrt{\log\log n})})$ in the LOCAL model, due to [7]. As noted by the authors, reducing the LOCAL complexity to below $O((\log\log n)^2)$ requires a radically new approach.

Our $O(\log\log\Delta \cdot \log^* n)$ round algorithm is based on a recursive degree reduction technique which can be used to color any almost-clique graph with $\Delta = \widetilde{O}(n^{1-o(1)})$ in essentially $O(\log^* n)$ rounds.

▶ **Theorem 2.** *(i) For every $\epsilon \in (0, 1)$, there is a randomized algorithm that computes a $(\Delta + 1)$ coloring in $O(\log(1/\epsilon) \cdot \log^* n)$ rounds for graphs with $\Delta = O((n/\log n)^{1-\epsilon})$, (ii) This also yields a $(\Delta + \Delta^{1/2+o(1)})$ coloring in $O(\log^* n)$ rounds, with high probability.*

Claim (ii) improves over the $O(\Delta)$-coloring algorithm of [14] that takes $O(\log\log\log n)$ rounds in *expectation*. We also provide fast *deterministic* algorithms for $(\Delta + 1)$ list coloring. The stat-of-the-art in the LOCAL model is $\widetilde{O}(\sqrt{\Delta}) + \log^* n$ rounds due to Fraigniaud, Heinrich, Marc and Kosowski [10].

▶ **Theorem 3.** *There is a deterministic algorithm that computes a $(\Delta + 1)$ coloring in $O(\log\Delta)$ rounds of the congested clique model and an $O(\Delta^2)$ coloring in $O(1)$ rounds.*

In [5], a deterministic algorithm for $(\Delta + 1)$ coloring in $O(\log\Delta)$ round was shown only for the case where $\Delta = O(n^{1/3})$. Here it is extended for $\Delta = \Omega(n^{1/3})$. This is done by derandomizing an $(\Delta + 1)$-list coloring algorithm which runs in $O(\log n)$ rounds. Similarly to [5], we first show that this algorithm can be simulated when the random choices made by the nodes are pairwise independent. Then, we enjoy the small search space and employ the method of conditional expectations. Instead of computing the seed bit by bit, we compute it in chunks of $\lfloor\log n\rfloor$ bits at a time, by fully exploiting the all-to-all power of the model.

[4] The random seed is usually shown provide a large progress in expectation. The deterministically computed seed should provide a progress at least as large as the expected progress of a random seed.
[5] As usual, by high probability we mean $1 - 1/n^c$ for some constant $c \geq 1$.

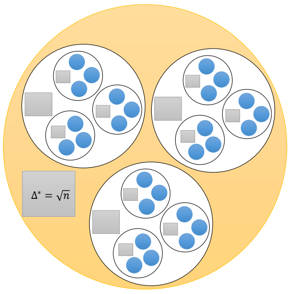


Figure 1 Illustration of the recursive sparsification. Gray boxes correspond to subgraphs with maximum degree $O(\sqrt{\Delta})$.

The Challenges and the Degree Reduction Technique. Our starting observation is that the CLP algorithm [7] can be implemented in $O(\log^* n)$ rounds in congested clique model for $\Delta = O(\sqrt{n})$. When $\Delta = O(\sqrt{n})$, using Lenzen's routing algorithm [18], each node can learn in $O(1)$ rounds, the palettes of all its neighbors along with the neighbors of its neighbors. Such knowledge is mostly sufficient for the CLP algorithm to go through.

To handle large degree graphs, we design a graph sparsification technique that essentially reduces the problem of $(\Delta + 1)$ coloring for an arbitrarily large $\Delta = \widetilde{O}(n^{1-\epsilon})$ into $\ell = O(\log(1/\epsilon))$ (non-independent) subproblems. In each subproblem, one has to compute a $(\Delta' + 1)$ coloring for a subgraph with $\Delta' = O(\sqrt{n})$, which can be done in $O(\log^* n)$ rounds, using a modification of the CLP algorithm, that we describe later on. Since there many dependencies between these ℓ sub-problems, it is required by our algorithm to solve them one-by-one, leading to a round complexity of $O(\log(1/\epsilon)\log^* n)$. See Figure 1 for an illustration of the recursion levels. To get an intuition into our approach and the challenges involved, consider an input graph G with maximum degree $\Delta = (n/\log n)^{1-\epsilon}$ and a palette $\mathsf{Pal}(G) = \{1, \ldots, \Delta + 1\}$ given to each node in G. A natural approach (also taken in [14]) for handling a large degree graph is to decompose it (say, randomly) into k vertex disjoint graphs $G_1, G_2, \ldots, G_k$, allocate a *distinct* set of colors for each of the subgraphs taken from $\mathsf{Pal}(G)$ and solve the problem recursively on each of them, enjoying (hopefully) smaller degrees in each G_i. Intuitively, assigning a *disjoint* set of colors to each G_i has the effect of *removing* all edges connecting nodes in different subgraphs. Thus, the input graph G is sparsified into a graph $G' = \bigcup G_i$ such that a legal coloring of G' (with the corresponding palettes given to the nodes) is a legal coloring for G. The main obstacle in implementing this approach is that assigning a distinct set of $\Delta(G_i) + 1$ colors to each of the G_i subgraph might be beyond the budget of $\Delta + 1$ colors. Indeed in [14] this approach led to $O(\Delta)$ coloring rather than $(\Delta + 1)$. To reduce the number of colors allocated to each subgraph G_i, it is desirable that the maximum degree $\Delta(G_i)$ would be as small as possible, for each G_i. This is exactly the problem of (k, p) *defective coloring* where one needs to color the graph with k colors such that the number of neighbors with the same color is at most p. To this point, the best defective coloring algorithm for large degrees is the randomized one: let each node pick a subgraph G_i (i.e., a color in the defective coloring language) uniformly at random. By a simple application of Chernoff bound, it is easy to see that the partitioning is "almost" perfect: w.h.p., for every i, $\Delta(G_i) \leq \Delta/k + \sqrt{\log n \cdot \Delta/k}$. Hence, allocating $\Delta(G_i) + 1$ colors to each subgraphs consumes $\Delta + \widetilde{O}(\sqrt{\Delta k})$ colors. To add insult to injury, this additive penalty of $\widetilde{O}(\sqrt{\Delta k})$ is only for one recursion call!

It is interesting to note that the parameter k – number of subgraphs (colors) – plays a key role here. Having a large k has the benefit of sharply decreasing the degree (i.e., from Δ

to Δ/k). However, it has the drawback of increasing the standard deviation and hence the total number of colors used. Despite these opposing effects, it seems that for whatever value of k chosen, increasing the number of colors to $\Delta + \Delta^{\epsilon}$ is unavoidable.

Our approach bypasses this obstacle by partitioning only a large fraction of the vertices into small-degree subgraphs *but not all of them*. Keeping in mind that we can handle efficiently graphs with maximum degree $\sqrt{n}$, in every level of the recursion, roughly $1-1/\sqrt{\Delta}$ of the vertices are partitioned into subgraphs $G_1, \ldots, G_k$. Let $\Delta(G_i)$ be the maximum degree of G_i. The remaining vertices join a left-over subgraph G^*. The number of subgraphs, k, is chosen carefully so that allocating $\Delta(G_i)+1$ colors to each of the k subgraphs, consumes at most Δ colors, on the one hand; and that the degree reduction in each recursion level is large enough on the other hand. These subgraphs are then colored recursively, until all remaining subgraphs have degree of $O(\sqrt{n})$. Once all vertices in these subgraphs are colored, the algorithm turns to color the left-over subgraph G^*. Since the maximum degree in G^* is $O(\sqrt{n})$, it is tempting to use the CLP algorithm to complete the coloring, as this can be done in $O(\log^* n)$ rounds for such bound on the maximum degree. This is not so immediate for the following reasons. Although the degree of v in G^* is $O(\sqrt{n})$, the graph G^* cannot be colored independently (as at that point, we ran out of colors to be solely allocate to G^*). Instead, the coloring of G^* should agree with the coloring of the rest of the graph and each v might have $\Omega(\Delta) = n^{1-\epsilon}$ neighbors in G. At first glance, it seems that this obstacle is easily solved by letting each $v \in G^*$ pick a subset of $\deg(v, G^*) = O(\sqrt{n})$ colors from its palette (i.e., removing the colors taken by its neighbors in $G \setminus G^*$). Now, one can consider only the graph G^* with maximum degree $\sqrt{n}$, where each vertex has a palette of $\deg(v, G^*)+1$ colors. Unfortunately, this seemingly plausible approach has a subtle flaw: for the CLP algorithm it is essential that each vertex receives a palette with *exactly* $\Delta(G^*)+1$ colors. This is indeed crucial and as noted by the authors adopting their algorithm to a $(\deg + 1)$ coloring algorithm is highly non-trivial and probably calls for a different approach.

In our setting, allocating each vertex $v \in G$ the exact same number of colors seems to be impossible as the number of available colors of each v depends on the number of its neighbors in $G \setminus G^*$, and this number has some fluctuations due to the random partitioning of the vertices. To get out of this impasse, we show that after coloring all vertices in $G \setminus G^*$, every vertex $v \in G^*$ has $r_v \in [\Delta(G^*) \pm (\Delta(G^*))^{3/5}]$ available colors in its palette where $r_v \geq \deg(v, G^*)$. In other words, all vertices can be allocated “almost" the same number of colors, but not exactly the same. We then carefully revise the basic definitions of the CLP algorithm and show that the analysis still goes through (upon minor changes) for this narrow range of variation in the size of the palettes.

Paper Organization. In Section 2, we explain how the CLP algorithm of [7] can be simulated in $O(\log^* n)$ congested-clique rounds when $\Delta = O(\sqrt{n})$. In Section 3.1, we illustrate the degree-reduction technique on the case where $\Delta = O((n/\log n)^{3/4})$. Section 3.2 extends this approach for $\Delta = O((n/\log n)^{1-\epsilon})$ for any $\epsilon \in (0,1)$, and Section 3.3 handles the general case and provides the complete algorithm. Finally, Section 4 discusses deterministic coloring algorithms. Missing proofs are deferred to the full version.

2 The Chang-Li-Pettie (CLP) Alg. in the Congested Clique

High-level Description of the CLP Alg. in the LOCAL Model. In the description below, we focus on the main randomized part of the CLP algorithm [13].

Harris-Schneider-Su algorithm is based on partitioning the graph into an ϵ-sparse subgraph and a collection of vertex-disjoint ϵ-dense components, for a given input parameter ϵ. Since the CLP algorithm extends this partitioning, we next formally provide the basic definitions from [13]. For an $\epsilon \in (0,1)$, an edge $e=(u,v)$ is an ϵ-friend if $|N(u)\cap N(v)| \geq (1-\epsilon)\cdot\Delta$. The endpoints of an ϵ-friend edge are ϵ-friends. A vertex v is ϵ-dense if v has at least $(1-\epsilon)\Delta$ ϵ-friends, otherwise it is ϵ-sparse. A key structure that arises from the definition of ϵ dense vertices is that of ϵ-*almost clique* which is a connected component of the subgraph induced by the ϵ-dense vertices and ϵ-friend edges. The dense components, ϵ-*almost* cliques, have some nice properties: each component C has at most $(1+\epsilon)\Delta$ many vertices, each vertex $v \in C$ has $O(\epsilon\Delta)$ neighbors outside C (called *external* neighbors) and $O(\epsilon\Delta)$ vertices in C which are not its neighbors. In addition, C has weak diameter at most 2. Coloring the dense vertices consists of $O(\log_{1/\epsilon}\Delta)$ phases. The efficient coloring of dense regions is made possible by generating a random proper coloring inside each clique so that each vertex has a small probability of receiving the same color as one of its external neighbors. To do that, in each cluster a random permutation is computed and each vertex selects a tentative color from its palette excluding the colors selected by lower rank vertices. Since each component has weak diameter at most 2, this process is implemented in 2 rounds of the LOCAL model. The remaining *sparse* subgraph is colored using a Schneider-Wattenhofer style algorithm [23] within $O(\log(1/\epsilon))$ rounds.

In Chang-Li-Pettie algorithm the vertices are partitioned into $\ell = \lceil \log\log\Delta \rceil$ layers in decreasing level of density. This hierarchical partitioning is based on a sequence of ℓ sparsity thresholds $\epsilon_1,\ldots,\epsilon_\ell$ where $\epsilon_i = \sqrt{\epsilon_{i-1}}$. Roughly speaking, level i consists of the vertices which are ϵ_i-dense but ϵ_{i-1}-sparse. Instead of coloring the vertices layer by layer, the algorithm partitions the vertices in level i into large and small components and partitions the layers into $O(\log^*\Delta)$ strata. It then colors the vertices in $O(\log^*\Delta)$ phases, giving priority to vertices in small components. The algorithms used to color these vertices are of the same flavor of the dense-coloring procedure of Harris-Schneider-Su. The key benefit in having the hierarchical structure is that the dense-coloring procedure is applied for $O(1)$ many phases on each stratum, rather than applying it for $O(\log_{1/\epsilon}\Delta)$ phases as in [13].

An $O(\log^*\Delta)$-Round Alg. for $\Delta = O(\sqrt{n})$ in the Congested Clique. We next observe that the randomized part of the CLP algorithm [7] can be implemented in the congested clique model when $\Delta = O(\sqrt{n})$ within $O(\log^*\Delta)$ rounds. We note that we obtain a round complexity of $O(\log\Delta^*)$ rather than $O(\log^* n)$ as in [7], due to the fact that the only part of the CLP algorithm that requires $O(\log^* n)$ rounds was for coloring a subgraph with maximum constant degree. In the congested-clique model such a step can be implemented in $O(1)$ rounds using Lenzen's routing algorithm. We show:

▶ **Theorem 4.** *For every graph with maximum degree $\Delta = O(\sqrt{n})$, there is an $O(\log^*\Delta)$-round randomized algorithm that computes $(\Delta+1)$-list coloring in the congested clique model.*

The main advantage of having small degrees is that it is possible for each node to collect its 2-neighborhood in $O(1)$ rounds (i.e., using Lenzen's routing [18]). As we will see, this is sufficient in order to simulate the CLP algorithm in $O(\log^*\Delta)$ rounds. The hierarchical decomposition of the vertices depends on the computation of ϵ-dense vertices. By collecting the neighbors of its neighbors, every vertex can learn its ϵ-dense friends and based on that deduce if it is an ϵ-dense vertex for every ϵ. In particular, for every edge (u,v), v can learn the minimum i such that u and v are ϵ_i-friends. To allow each vertex v compute the ϵ-almost cliques to which it belongs, we do as follows. Each vertex v sends to each of its neighbors $N(v)$, the minimum ϵ_i such that u,v are ϵ_i-friends, for every $u \in N(v)$. Since the weak diameter of each almost-clique is at most 2, each vertex has collected all the required information from its 2^{nd} neighborhood to locally compute its ϵ_i-almost cliques for every ϵ_i. Overall, each vertex sends $O(\Delta)$ messages and receives $O(\Delta^2) = O(n)$ messages, collection this information can be done in $O(1)$ rounds for all nodes, using Lenzen's routing algorithm. The next obstacle is the simulation of the algorithm that colors the ϵ-dense vertices. Since each ϵ-almost clique C has $(1+\epsilon)\Delta = O(\sqrt{n})$ vertices, we can make the leader of each such C learn the palettes of all the vertices in its clique as well as their neighbors in $O(1)$ rounds. The leader can then locally simulate the dense-coloring procedure and notify the output color to each of its almost-clique vertices. Finally, coloring the sparse regions in a Schneider-Wattenhofer style uses messages of size $O(\Delta)$ and hence each vertex is the target of $O(\Delta^2) = O(n)$ messages which again can be implemented in $O(1)$ many rounds. A more detailed description appears in the full version [22]. By the above description, we also have:

▶ **Corollary 5.** *Given q vertex-disjoint subgraphs $G_1, \ldots, G_q$ each with maximum degree $\Delta = O(\sqrt{n})$, a $(\Delta+1)$ coloring can be computed in $O(\log^* \Delta)$ rounds, for all subgraphs simultaneously.*

Handling Non-Equal Palette Size for $\Delta = O(\sqrt{n})$. The CLP algorithm assumes that each vertex is given a list of *exactly* $(\Delta+1)$ colors. Our coloring algorithms requires a more relaxed setting where each vertex v is allowed to be given a list of $r_v \in [\Delta - \Delta^{3/5}, \Delta+1]$ colors where $r_v \geq \deg(v, G) + 1$. In this subsection we show:

▶ **Lemma 6.** *Given a graph G with $\Delta = O(\sqrt{n})$, if every vertex v has a palette with $r_v \geq \deg(v, G) + 1$ colors and $r_v \in [\Delta - \Delta^{3/5}, \Delta+1]$ then a list coloring can be computed in $O(\log^* \Delta)$ rounds in the congested clique model.*

The key modification for handling non-equal palette sizes is in definition of ϵ-friend (which affects the entire decomposition of the graph). Throughout, let $q = \Delta^{3/5}$ and say[6] that u, v are (ϵ, q)-friends if $|N(u) \cap N(v)| \geq (1-\epsilon) \cdot (\Delta - q)$. Clearly, if u, v are ϵ-friends, they are also (ϵ, q)-friends. A vertex v is an (ϵ, q)-*dense* if it has at least $(1-\epsilon) \cdot (\Delta - q)$ neighbors which are (ϵ, q)-friends. An (ϵ, q)-almost clique is a connected component of the subgraph induced by (ϵ, q)-dense vertices and their (ϵ, q) friends edges. We next observe that for the ϵ-values used in the CLP algorithm, the converse is also true up to some constant. The full proof of Theorem 6 is in the full version.

▶ **Observation 7.** *For any $\epsilon \in [\Delta^{-10}, K^{-1}]$, where K is a large constant, and for $q = \Delta^{3/5}$, it holds that if u, v are (ϵ, q) friends, they are (2ϵ)-friends. Also, if v is an (ϵ, q)-dense, then it is 2ϵ-dense.*

3 $(\Delta + 1)$-Coloring for $\Delta = \Omega(\sqrt{n})$

In this section, we describe a new recursive degree-reduction technique. As a warm-up, we start with $\Delta = O((n/\log n)^{3/4})$. We make use of the following fact.

▶ **Theorem 8** (Simple Corollary of Chernoff Bound)**.** *Suppose $X_1, X_2, \ldots, X_\ell \in [0,1]$ are independent random variables, and let $X = \sum_{i=1}^{\ell} X_i$ and $\mu = \mathbb{E}[X]$. If $\mu \geq 5 \log n$, then w.h.p. $X \in \mu \pm \sqrt{5\mu \log n}$, and if $\mu < 5 \log n$, then w.h.p. $X \leq \mu + 5 \log n$.*

[6] The value of q is chosen to be a bit above the standard deviation of $\sqrt{\Delta \log n}$ that will occur in our algorithm.

3.1 An $O(\log^* \Delta)$-round algorithm for $\Delta = O((n/\log n)^{3/4})$

The algorithm partitions G into $O(\Delta^{1/3})$ subgraphs as follows. Let $\ell = \lceil \Delta^{1/3} \rceil$. We define $\ell + 1$ subsets of vertices $V_1, \ldots, V_\ell, V^*$. A vertex joins each V_i with probability

$$p_i = 1/\ell - 2\sqrt{5 \log n}/(\Delta^{1/3} \cdot \ell),$$

for every $i \in \{1, \ldots, \ell\}$, and it joins V^* with the remaining probability of $p^* = 2\sqrt{5 \log n}/\Delta^{1/3}$.

Let $G_i = G[V_i]$ be the induced subgraph for every $i \in \{1, \ldots, \ell, *\}$. Using Chernoff bound of Theorem 8, the maximum degree Δ' in each subgraph G_i, $i \in \{1, \ldots, \ell\}$ is w.h.p.:

$$\Delta' \leq \Delta/\ell - 2\Delta^{2/3}\sqrt{5 \log n}/\ell + \sqrt{5\Delta \log n/\ell} \leq \Delta/\ell - 1$$

In the first phase, all subgraphs $G_1, \ldots, G_\ell$ are colored independently and simultaneously. This is done by allocating a distinct set of $(\Delta' + 1)$ colors for each of these subgraphs. Overall, we allocate $\ell \cdot (\Delta' + 1) \leq \Delta$ colors. Since $\Delta' = O(\Delta^{2/3}) = O(\sqrt{n})$, we can apply the $(\Delta' + 1)$-coloring algorithm of Theorem 5 on all the graphs $G_1, \ldots, G_\ell$ simultaneously. Hence, all the subgraphs $G_1, \ldots, G_\ell$ are colored in $O(\log^* \Delta)$ rounds.

Coloring the remaining left-over subgraph G^*. The second phase of the algorithm completes the coloring for the graph G^*. This coloring should agree with the colors computed for $G \setminus G^*$ computed in the previous phase. Hence, we need to color G^* using a *list* coloring algorithm. We first show that w.h.p. the maximum degree Δ^* in G^* is $O(\sqrt{n})$. The probability of vertex to be in G^* is $p^* = 2\sqrt{5 \log n}/\Delta^{1/3}$. By Chernoff bound of 8, w.h.p., $\Delta^* \leq p^* \cdot \Delta + \sqrt{5p^* \cdot \Delta \cdot \log n}$. Since $\Delta \leq (n/\log n)^{3/4}$, $\Delta^* = O(\sqrt{n})$. To be able to apply the modified CLP of Theorem 6, we show:

▶ **Lemma 9.** *Every $v \in G^*$ has at least $\Delta^* - (\Delta^*)^{3/5}$ available colors in its palette after coloring all its neighbors in $G \setminus G^*$.*

Proof. First, consider the case where $\deg(v, G) \leq \Delta - (\Delta^* - \sqrt{5\Delta^* \cdot \log n})$. In such case, even after coloring all neighbors of v, it still has an access of $\Delta^* - \sqrt{5\Delta^* \cdot \log n} \geq \Delta^* - (\Delta^*)^{3/5}$ colors in its palette after coloring $G \setminus G^*$ in the first phase. Now, consider a vertex v with $\deg(v, G) \geq \Delta - (\Delta^* - \sqrt{5\Delta^* \cdot \log n})$. Using Chernoff bound, w.h.p., $\deg(v, G^*) > (\Delta - (\Delta^* - \sqrt{\Delta^* \cdot 5 \log n})) \cdot p^* - \sqrt{5 \log n \Delta p^*} \geq \Delta^* - (\Delta^*)^{3/5}$. ◀

Also note that a vertex $v \in G^*$ has at least $\deg(v, G^*) + 1$ available colors, since all its neighbors in G^* are uncolored at the beginning of the second phase and initially it was given $(\Delta + 1)$ colors. Eventhough, $v \in G^*$ might have $\Omega(\Delta)$ neighbors not in G^*, to complete the coloring of G^*, by Theorem 9, after the first phase, each v can find in its palette $r \in [\Delta^* - (\Delta^*)^{3/5}, \Delta^* + 1]$ available colors and this sub-palette is sufficient for its coloring in G^*. Since $\Delta^* = O(\sqrt{n})$, to color G^* (using these small palettes), one can apply the $O(\log^* \Delta)$ round list-coloring algorithm of Theorem 6.

3.2 An $O(\log(1/\epsilon) \cdot \log^* \Delta)$-round algorithm for $\Delta = O((n/\log n)^{1-\epsilon})$

Let $N = n/(5 \log n)$. First assume that $\Delta \leq N/2$ and partitions the range of relevant degrees $[\sqrt{n}, N/2]$ into $\ell = \Theta(\log \log \Delta)$ classes. The y^{th} range contains all degrees in $[N^{1-1/2^y}, N^{1-1/(2^{y+1})}]$ for every $y \in \{1, \ldots, \ell\}$. Given a graph G with maximum degree $\Delta = O(N^{1-1/(2^{y+1})})$, Algorithm RecursiveColoring colors G in $y \cdot O(\log^* \Delta)$ rounds, w.h.p.

Step (I): Partitioning (Defective-Coloring). For $i \in \{0, \ldots, y-1\}$, in every level i of the recursion, we are given a graph G', with maximum degree $\Delta_i = O(N^{1-1/2^{y-i+1}})$, and a palette Pal_i of $(\Delta_i + 1)$ colors. For $i = 0$, $\Delta_0 = \Delta$ and the palette $\mathsf{Pal}_0 = \{1, \ldots, \Delta + 1\}$.

The algorithm partitions the vertices of G' into $q_i + 1$ subsets: $V'_1, \ldots, V'_{q_i}$ and a special left-over set V^*. The partitioning is based on the following parameters. Set $x = 2^{y-i}$ and

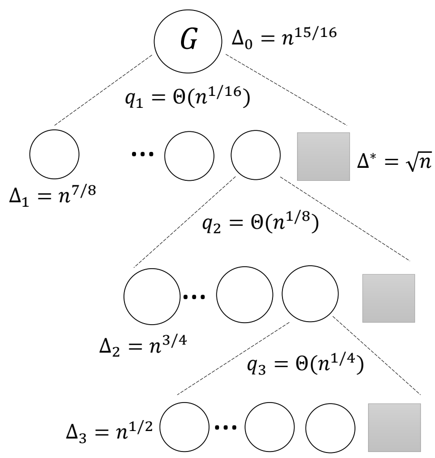
$$q_i = \lceil \Delta_i^{1/(2x-1)} \rceil \text{ and } \delta_i = 2\sqrt{5 \log n} \cdot q_i^{3/2} / \sqrt{\Delta_i}.$$

Each vertex $v \in V(G')$ joins V'_j with probability $p_j = 1/q_i - \delta_i/(q_i)^2$ for every $j \in \{1, \ldots, q_i\}$, and it joins V^* with probability $p^* = \delta_i/q_i$. Note that $p^* \in (0, 1)$ as $x \geq 2$. For every $j \in \{1, \ldots, q_i\}$, let $G'_j = G'[V'_j]$ and let $G^* = G'[V^*]$.

Step (II): Recursive coloring of $G'_1, \ldots, G'_{q_i}$. Denote by $\widetilde{\Delta}_j$ to be the maximum degree in G'_j for every $j \in \{1, \ldots, q_i\}$ and by Δ^*, the maximum degree in G^*. The algorithm allocates a distinct subset of $(\widetilde{\Delta}_j + 1)$ colors from Pal_i for every $j \in \{1, \ldots, q_i\}$. In the analysis, we show that w.h.p. Pal_i contains sufficiently many colors for that allocation. The subgraphs $G'_1, \ldots, G'_{q_i}$ are colored recursively and simultaneously, each using its own palette. It is easy to see that the maximum degree of each G'_j is $O(N^{1-1/2^{y-i}})$ (which is indeed the desire degree for the subgraphs colored in level $i+1$ of the recursion).

Step (III): Coloring the left-over graph G^*. Since the algorithm already allocated at most Δ_i colors for coloring the G'_j subgraphs, it might run out of colors to allocate for G^*. This last subgraph is colored using a list-coloring algorithm only after all vertices of $G'_1, \ldots, G'_{q_i}$ are colored. Recall that Δ^* is the maximum degree of G^*. In the analysis, we show that w.h.p. $\Delta^* = O(\sqrt{n})$. For every $v \in G^*$, let $\mathsf{Pal}(v) \subseteq \mathsf{Pal}_i$ be the remaining set of available colors after coloring all the vertices in $V'_1, \ldots, V'_{q_i}$. Each vertex $v \in G^*$ computes a new palette $\mathsf{Pal}^*(v) \subseteq \mathsf{Pal}(v)$ such that: (i) $|\mathsf{Pal}^*(v)| \geq \deg(v, G^*) + 1$, and (ii) $|\mathsf{Pal}^*(v)| \in [\Delta^* - \Delta^{3/5}, \Delta^* + 1]$. In the analysis section, we show that w.h.p. this is indeed possible for every $v \in G^*$. The algorithm then applies the modified CLP algorithm, and G^* gets colored within $O(\log^* \Delta)$ rounds.

Example:



Assume that input graph G has maximum degree $\Delta_0 = n^{15/16}$. The algorithm partitions G into $k_0 = n^{1/15}$ subgraphs in the following manner. For sake of clarity, we omit logarithmic factors in the explanation. With probability $1/(\Delta_0)^{1/2+o(1)}$, v joins a left-over subgraph G^*, and with the remaining probability it picks a subgraph $[1, k_0]$ uniformly at random. It is easy to see that the maximum degree in each of these subgraphs is at most $\Delta_1 = n^{7/8} + n^{7/16}$. A distinct set of Δ_1 colors from $[1, \Delta_0 + 1]$ is allocated to each of the k subgraphs. Each such subgraph is now partitioned into $k_1 = n^{1/8}$ subgraphs plus a left-over subgraph. This continues until all subgraphs have their degrees sharply concentrated around $\sqrt{n}$. At that point, the modified CLP algorithm can be applied on all the subgraphs in the last level ℓ. Once these subgraphs are colored, the left-over subgraphs in level $\ell - 1$ are colored, this continues until the final left-over subgraph of the first level is colored. We next provide a compact and high level description of the algorithm.

Algorithm 1 $\mathsf{RecursiveColoring}(G', \mathsf{Pal}_i)$

Input: Graph G' with maximum degree $\Delta_i = O(N^{1-1/2^{y-i+1}})$.
A palette Pal_i of $(\Delta_i + 1)$ colors (same for all nodes).

- Partitions G' into $q_i + 1$ vertex-disjoint subgraphs:
  - q_i vertex-subgraphs $G'_1, \ldots, G'_{q_i}$ with maximum degree $\Delta_{i+1} = O(N^{1-1/2^{y-i}})$.
  - Left-over subgraph G^* with maximum degree $\Delta^* = O(\sqrt{n})$.
- Allocate a distinct palette $\mathsf{Pal}_j \subset \mathsf{Pal}_i$ of $(\Delta_{i+1} + 1)$ colors for each $j \le q_i$.
- Apply $\mathsf{RecursiveColoring}(G_j, \mathsf{Pal}_j)$ for every $j \le q_i$ simultaneously.
- Apply a $(\Delta_i + 1)$-list coloring restricted to Pal_i, to complete the coloring of $G[V^*]$.

Analysis.

▶ **Lemma 10.** *(i) For every $j \in \{1, \ldots, q_i\}$, w.h.p., $\widetilde{\Delta}_j = O(N^{1-1/2^{y-i}})$. (ii) One can allocate $(\widetilde{\Delta}_j + 1)$ distinct colors from Pal_i for each G'_j, $j \in \{1, \ldots, q_i\}$.*

Proof. Using Chernoff bound of Theorem 8, w.h.p., for every $j \in \{1, \ldots, q_i - 1\}$, the maximum degree $\widetilde{\Delta}_j$ in G'_j is at most $\widetilde{\Delta}_j = O(\Delta_i/q_i)$. Since $\Delta_i = O(N^{1-2^{y-i+1}})$, claim (i) follows. We now bound the sum of all colors allocated to these subgraphs:

$$\widetilde{\Delta}_j \le \Delta_i/q_i - (\Delta_i \cdot \delta_i)/(q_i)^2 + \sqrt{5 \log n \cdot \Delta_i/q_i} \le \Delta_i/q_i - 1 \ .$$

where the last inequality follows by the value of δ_i. We get that $\sum_{j=1}^{q_i}(\widetilde{\Delta}_j + 1) \le \Delta_i$ and since Pal_i contains $\Delta_i + 1$ colors, claim (ii) follows. ◀

We next analyze the final step of the algorithm and begin by showing that, w.h.p., the maximum degree in the left-over graph G^* is $O(\sqrt{n})$. By Chernoff bound of Theorem 8, w.h.p., the maximum degree $\Delta^* \le \Delta_i \cdot \delta_i/q_i + \sqrt{\log n \cdot \Delta_i \cdot \delta_i/q_i}$. Since $\Delta_i = O(N^{1-1/2^{y-i+1}})$, we get that $\Delta^* = O(\sqrt{n})$. We now claim:

▶ **Lemma 11.** *After coloring for all the vertices in $G'_1, \ldots, G'_{q_i}$, each vertex $v \in G^*$ has a palette $\mathsf{Pal}^*(v)$ of free colors such that (i) $|\mathsf{Pal}^*(v)| \ge \deg(u, G^*) + 1$, and (ii) $|\mathsf{Pal}^*(v)| \in [\Delta^* - (\Delta^*)^{3/5}), \Delta^* + 1]$.*

Proof. Since each vertex $v \in G'$ has a palette of size $(\Delta_i + 1) \ge \deg(v, G')$, after coloring all its neighbors in $G'_1, \ldots, G'_{q_i}$, it has at least $\deg(v, G^*) + 1$ free colors in its palette. Claim (ii) follows the same argument as in Theorem 9. We show that the palette of v has at least $\Delta^* - O(\sqrt{\Delta^*} \cdot 5 \log n) \ge (\Delta^* - (\Delta^*)^{3/5})$ available colors after coloring all the vertices in

$G \setminus G^*$. First, when $\deg(v, G') \leq \Delta_i - (\Delta^* - \sqrt{\Delta^* \cdot 5 \log n})$, then even after coloring all neighbors of v in G', it still has an access of $\Delta^* - \sqrt{\Delta^* \cdot 5 \log n}$ colors in its palette. Consider a vertex v with $\deg(v, G') \geq \Delta_i - (\Delta^* - \sqrt{\Delta^* \cdot 5 \log n})$. By Chernoff, w.h.p. it holds that:

$$\begin{aligned}\deg(v, G^*) &\geq (\Delta_i - (\Delta^* - \sqrt{\Delta^* \cdot 5 \log n})) \cdot p^* - \sqrt{5 \log n \cdot \Delta_i \cdot p^*} \\ &\geq \Delta_i \cdot \delta_i / q_i - O(\sqrt{\Delta_i \cdot \delta_i \log n / q_i}) \geq \Delta^* - (\Delta^*)^{3/5} .\end{aligned}$$

Hence, by combining with claim (i), the lemma follows. ◀

This completes the proof of Theorem 2(i).

$(\Delta + \Delta^{1/2+\epsilon})$ Coloring in Log-Star Rounds

▶ **Lemma 12.** *For any fixed $\epsilon \in (0, 1)$, one can color, w.h.p., a graph with $(\Delta + \Delta^{1/2+\epsilon})$ colors in $O(\log(1/\epsilon) \cdot \log^* \Delta)$ rounds.*

Proof. Due to Theorem 4, it is sufficient to consider the case where $\Delta = \Omega(\sqrt{n})$. Partition the graph into $k = \lfloor \Delta^\epsilon \rfloor$ subgraphs $G_1, \ldots, G_k$, by letting each vertex independently pick a subgraph uniformly at random. By Chernoff bound of Theorem 8, the maximum degree Δ_i in each subgraph G_i is at most $\Delta_i \leq \Delta^{1-\epsilon} + \Delta^{1/2-\epsilon/2} \cdot \sqrt{5 \log n}$. Allocate a distinct set of $\Delta_i + 1$ colors Pal_i to each subgraph G_i. Since $\Delta^{1-\epsilon} = O((n/\log n)^{1-\epsilon/2})$, we can apply Alg. RecursiveColoring on each of these subgraphs which takes $O(\log(1/\epsilon) \cdot \log^* \Delta)$ rounds. It is easy to see, that since the subgraphs are vertex disjoint, Alg. RecursiveColoring can be applied on all k subgraphs simultaneously with the same round complexity. Overall, the algorithm uses $\Delta + \Delta^{1/2+\epsilon}$ colors. ◀

3.3 $(\Delta + 1)$ Coloring Algorithm for General Graphs

For graphs G with $\Delta \leq n/(10 \log n)$, we simply apply Alg. RecursiveColoring. Plugging $\epsilon = 1/\log n$ in Theorem 2, we get that this is done in $O(\log \log \Delta \cdot \log^* \Delta)$ rounds. It remains to handle graphs with with $\Delta \in [n/(10 \log n), n]$. We partition the graph into $\ell = \lceil 5 \log n \rceil$ subgraphs $G_1, G_2, \ldots, G_\ell$ and a left-over graph G^* in the following manner. Each $v \in V$ joins G_i with probability $p = 1/\ell - 2\sqrt{5 \log n/(\Delta \cdot \ell)}$ for every $i \in \{1, \ldots, \ell\}$, and it joins G^* with probability $p^* = 1 - \ell \cdot p = \Theta(\log n/\sqrt{\Delta})$. By Chernoff bound, the maximum degree in G_i for $i \in \{1, 2, \ldots, \ell\}$ is $\Delta_i \leq \Delta/\ell - 2\sqrt{(\Delta \cdot 5 \log n)/\ell} + \sqrt{\Delta \cdot 5 \log n/\ell} \leq \Delta/\ell - 1$. Hence, we have the budget to allocate a distinct set Pal_i of Δ_i colors for each G_i.

The first phase applies Algorithm RecursiveColoring on each (G_i, Pal_i) simultaneously for every i. Since $\Delta_i = O(n/\log n)$, and the subgraphs are vertex-disjoint, this can be done in $O(\log \log \Delta \cdot \log^* \Delta)$ rounds for all subgraphs simultaneously (see Theorem 2(i)).

After all the vertices of $G \setminus G^*$ get colored, the second phase colors the left-over subgraph G^*. The probability of a vertex v to be in G^* is $O(\log n/\sqrt{\Delta}) = O(\log^2 n/\sqrt{n})$. Hence, G^* contains $O(\log^2 n \cdot \sqrt{n})$ vertices with high probability. We color G^* in two steps. First, we use the deg +1 list coloring Algorithm OneShotColoring from [2] to reduce the uncolored-degree of each vertex to be $O(\sqrt{n}/\log^2 n)$ with high probability. This can be done in $O(\log \log n)$ rounds. In the second step, the entire uncolored subgraph $G'' \subset G^*$ has $O(n)$ edges and can be solved locally in $O(1)$ rounds. Note that for each $v \in G''$, it is sufficient to consider a palette with $\deg(v, G'') + 1$ colors, and hence sending all these palettes can be done in $O(1)$ rounds as well. The complete proof of Theorem 1 is in the full-version.

Algorithm 2 **FastColoring**(G)

- If $\Delta \leq n/(10\log n)$, call RecursiveColoring$(G, [1, \Delta+1])$.
- Else, partition G into vertex-subgraphs as follows:
  - $G'_1, \ldots, G'_q$ with the maximum degree $\Theta(\Delta/\log n)$, and
  - a left-over subgraph G^* with maximum degree $\Delta^* = O(\sqrt{n})$.
- Allocate a distinct palette $\mathsf{Pal}_j \subset [1, \Delta+1]$ of $(\Delta(G_j)+1)$ colors for each $j \leq q_i$.
- Apply RecursiveColoring(G_j, Pal_j) for all $G_1, \ldots, G_q$ simultaneously.
- Apply a $(\deg+1)$-list coloring algorithm on G^* for $O(\log\log n)$ rounds.
- Solve the remaining uncolored subgraph locally.

4 Deterministic Coloring Algorithms

In this section, we provide deterministic coloring algorithms using the tools of bounded inpdendent and conditional expectation introduced in [5].

▶ **Theorem 13.** *There is a deterministic* $(\Delta+1)$ *list coloring using* $O(\log\Delta)$ *rounds, in the congested clique model.*

In [5], a deterministic $(\Delta+1)$ coloring was presented only for graphs with maximum degree $\Delta = O(n^{1/3})$. Here, we handle the case of $\Delta = \Omega(n^{1/3})$. We derandomize the following simple $(\Delta+1)$-algorithm that runs in $O(\log n)$ rounds.

Algorithm 3 Round $\boldsymbol{i}$ of Algorithm **SimpleRandColor** (for node $\boldsymbol{v}$ with palette $\mathsf{\mathbf{Pal}}_{\boldsymbol{v}}$)

- Let F_v be the set of colors taken by the colored neighbors of v.
- Pick a color c_v uniformly at random from $\mathsf{Pal}_v \setminus F_v$.
- Send colors to neighbors and if $c_v \neq 0$ and legal, halt.

▶ **Observation 14.** *The correctness of Algorithm* SimpleRandColor *is preserved, even if the coin flips are* pairwise-*independent.*

The goal of *phase* i in our algorithm is to compute a seed that would be used to simulate the random color choices of *round* i of Alg. SimpleRandColor. This seed will be shown to be good enough so that at least 1/4 of the currently uncolored vertices, get colored when picking their color using that seed. Let V_i be the set of uncolored vertices at the beginning of phase i. We need the following construction of bounded independent hash functions:

▶ **Lemma 15.** *[25] For every* $\gamma, \beta, d \in \mathbb{N}$, *there is a family of* d*-wise independent functions* $\mathcal{H}_{\gamma,\beta} = \left\{h : \{0,1\}^\gamma \to \{0,1\}^\beta\right\}$ *such that choosing a random function from* $\mathcal{H}_{\gamma,\beta}$ *takes* $d \cdot \max\{\gamma, \beta\}$ *random bits, and evaluating a function from* $\mathcal{H}_{\gamma,\beta}$ *takes time poly*(γ, β, d).

For our purposes, we use Theorem 15 with $d = 2, \gamma = \log n, \beta = \log\Delta$ and hence the size of the random seed is $\alpha \cdot \log n$ bits for some constant α. Instead of revealing the seed bit by bit using the conditional expectation method, we reveal the assignment for a *chunk* of $z = \lfloor\log n\rfloor$ variables at a time. To do so, consider the i'th chunk of the seed $Y'_i = (y'_1, \ldots, y'_z)$. For each of the n possible assignments $(b'_1, \ldots, b'_z) \in \{0,1\}^z$ to the z variables in Y', we assign a leader u that represent that assignment and receives the conditional expectation values from all the uncolored nodes V_i, where the conditional expectation is computed based on assigning $y'_1 = b'_1, \ldots, y'_z = b'_z$. Unlike the MIS problem, here the vertex's success depends only on its neighbors (i.e., and does not depend on its second neighborhood). Using the partial seed and the IDs of its neighbors, every vertex v can compute the probability that it gets colored based on the partial seed. It then sends its probability of being colored using a particular assignment $y'_1 = b'_1, \ldots, y'_z = b'_z$ to the leader u responsible for that assignment. The leader node u of each assignment $y'_1 = b'_1, \ldots, y'_z = b'_z$ sums up all the values and obtains the expected number of colored nodes conditioned on the assignment. Finally, all nodes send to the leader their computed sum and the leader selects the assignment $(b^*_1, \ldots, b^*_z) \in \{0,1\}^z$ of largest value. After $O(1)$ many rounds, the entire assignment of the $O(\log n)$ bits of the seed are revealed. Every yet uncolored vertex $v \in V_i$ uses this seed to simulate the random choice of Alg. SimpleRandColor, that is selecting a color in $\{0, 1, 2, \ldots, \Delta + 1\} \setminus F_v$ and broadcasts its decision to its neighbors. If the color $c_v \neq 0$ is legal, v is finally colored and it notifies its neighbors. By the correctness of the conditional expectation approach, we have that least $1/4 \cdot |V_i|$ vertices got colored. Hence, after $O(\log n) = O(\log \Delta)$ rounds, all vertices are colored. In the full version, we also show a deterministic $O(\Delta^2)$ coloring in $O(1)$ rounds.

References

1 Alkida Balliu, Juho Hirvonen, Janne H. Korhonen, Tuomo Lempiäinen, Dennis Olivetti, and Jukka Suomela. New classes of distributed time complexity. *CoRR*, abs/1711.01871, 2017. arXiv:1711.01871.

2 Leonid Barenboim, Michael Elkin, Seth Pettie, and Johannes Schneider. The locality of distributed symmetry breaking. *Journal of the ACM (JACM)*, 63(3):20, 2016.

3 József Beck. An algorithmic approach to the lovász local lemma. i. *Random Structures & Algorithms*, 2(4):343–365, 1991.

4 Andrew Berns, James Hegeman, and Sriram V Pemmaraju. Super-fast distributed algorithms for metric facility location. In *International Colloquium on Automata, Languages, and Programming*, pages 428–439. Springer, 2012.

5 Keren Censor-Hillel, Merav Parter, and Gregory Schwartzman. Derandomizing local distributed algorithms under bandwidth restrictions. In *31st International Symposium on Distributed Computing, DISC 2017, October 16-20, 2017, Vienna, Austria*, pages 11:1–11:16, 2017.

6 Yi-Jun Chang, Tsvi Kopelowitz, and Seth Pettie. An exponential separation between randomized and deterministic complexity in the LOCAL model. In *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 615–624, 2016.

7 Yi-Jun Chang, Wenzheng Li, and Seth Pettie. An optimal distributed $(\Delta + 1)$ coloring algorithm? *arXiv preprint arXiv:1711.01361*, 2018.

8 Yi-Jun Chang and Seth Pettie. A time hierarchy theorem for the local model. *FOCS*, 2017.

9 Manuela Fischer and Mohsen Ghaffari. Sublogarithmic distributed algorithms for Lov\'asz local lemma, and the complexity hierarchy. *DISC*, 2017.

10 Pierre Fraigniaud, Marc Heinrich, and Adrian Kosowski. Local conflict coloring. In *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*, pages 625–634. IEEE, 2016.

11 Mohsen Ghaffari. An improved distributed algorithm for maximal independent set. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 270–277. Society for Industrial and Applied Mathematics, 2016.

12 Mohsen Ghaffari. Distributed MIS via all-to-all communication. In *Proceedings of the ACM Symposium on Principles of Distributed Computing, PODC 2017, Washington, DC, USA, July 25-27, 2017*, pages 141–149, 2017.

13 David G Harris, Johannes Schneider, and Hsin-Hao Su. Distributed $(\delta + 1)$-coloring in sublogarithmic rounds. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing*, pages 465–478. ACM, 2016.

14 James W Hegeman and Sriram V Pemmaraju. Lessons from the congested clique applied to mapreduce. *Theoretical Computer Science*, 608:268–281, 2015.

15 James W Hegeman, Sriram V Pemmaraju, and Vivek B Sardeshmukh. Near-constant-time distributed algorithms on a congested clique. In *International Symposium on Distributed Computing*, pages 514–530. Springer, 2014.

16 Janne H Korhonen and Jukka Suomela. Brief announcement: Towards a complexity theory for the congested clique. In *LIPIcs-Leibniz International Proceedings in Informatics*, volume 91. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

17 Fabian Kuhn, Thomas Moscibroda, and Roger Wattenhofer. Local computation: Lower and upper bounds. *Journal of the ACM (JACM)*, 63(2):17, 2016.

18 Christoph Lenzen. Optimal deterministic routing and sorting on the congested clique. In *PODC*, pages 42–50, 2013.

19 Zvi Lotker, Boaz Patt-Shamir, Elan Pavlov, and David Peleg. Minimum-weight spanning tree construction in o (log log n) communication rounds. *SIAM Journal on Computing*, 35(1):120–131, 2005.

20 Y Maus, F Kuhn, and M Ghaffari. On the complexity of local distributed graph problems. In *Proceedings of the Annual ACM Symposium on Theory of Computing*, pages 784–797, 2017.

21 Moni Naor and Larry Stockmeyer. What can be computed locally? *SIAM Journal on Computing*, 24(6):1259–1277, 1995.

22 Merav Parter. $(\delta + 1)$ coloring in the congested clique model. *arXiv*, 2018. arXiv:1805.02457.

23 Johannes Schneider and Roger Wattenhofer. A new technique for distributed symmetry breaking. In *Proceedings of the 29th ACM SIGACT-SIGOPS symposium on Principles of distributed computing*, pages 257–266. ACM, 2010.

24 Gregory Schwartzman. Adapting sequential algorithms to the distributed setting. *arXiv preprint arXiv:1711.10155*, 2017.

25 Salil P. Vadhan. Pseudorandomness. *Foundations and Trends in Theoretical Computer Science*, 7(1-3):1–336, 2012. doi:10.1561/0400000010.