Local Conflict Coloring

Pierre Fraigniaud[1,2], Marc Heinrich[1,2], and Adrian Kosowski[1,2]

[1]CNRS and Paris Diderot University
[2]GANG Project, Inria Paris
E-mails: pierre.fraigniaud@liafa.univ-paris-diderot.fr, marc.heinrich@ens.fr, adrian.kosowski@inria.fr

***Abstract*—Locally finding a solution to *symmetry-breaking* tasks such as vertex-coloring, edge-coloring, maximal matching, maximal independent set, etc., is a long-standing challenge in *distributed network* computing. More recently, it has also become a challenge in the framework of *centralized local* computation. We introduce *conflict coloring* as a general symmetry-breaking task that includes all the aforementioned tasks as specific instantiations — conflict coloring includes all *locally checkable labeling* tasks from [Naor & Stockmeyer, STOC 1993]. Conflict coloring is characterized by two parameters l and d, where the former measures the amount of freedom given to the nodes for selecting their colors, and the latter measures the number of constraints which colors of adjacent nodes are subject to. We show that, in the standard LOCAL model for distributed network computing, if $l/d > \Delta$, then conflict coloring can be solved in $\widetilde{O}(\sqrt{\Delta}) + \log^* n$ rounds in n-node graphs with maximum degree Δ, where $\widetilde{O}$ ignores the polylog factors in Δ. The dependency in n is optimal, as a consequence of the $\Omega(\log^* n)$ lower bound by [Linial, SIAM J. Comp. 1992] for $(\Delta+1)$-coloring. An important special case of our result is a significant improvement over the best known algorithm for distributed $(\Delta+1)$-coloring due to [Barenboim, PODC 2015], which required $\widetilde{O}(\Delta^{3/4}) + \log^* n$ rounds. Improvements for other variants of coloring, including $(\Delta+1)$-list-coloring, $(2\Delta-1)$-edge-coloring, coloring with forbidden color distances, etc., also follow from our general result on conflict coloring. Likewise, in the framework of centralized local computation algorithms (LCAs), our general result yields an LCA which requires a smaller number of probes than the previously best known algorithm for vertex-coloring, and works for a wide range of coloring problems.**

***Index Terms*—Distributed Network Computing; Symmetry Breaking; List-coloring; $(\Delta+1)$-coloring; Local Computation Algorithm.**

I. Introduction

A. Context and Objective

Distributed network computing considers the computing model in which every node of a graph is an autonomous computing entity, and nodes exchange information by sending messages along the edges of the graph. In this context, *symmetry breaking*, which is arguably the most important problem in distributed network computing has attracted a lot of attention, and several local forms of symmetry breaking tasks have been considered, including the construction of proper *graph colorings* [5], [7], [25], [30], [33], [39], of *maximal independent sets* (MIS) [1], [26], of *maximal matchings* [18], [20], etc., to mention just a few. The main question in this framework is whether these tasks can be solved *locally*, i.e., by exchanging data between nodes at short distance in the network. To tackle the locality issue, the complexity of a distributed algorithm is measured in term of number of *rounds* in the LOCAL model [35], where a round consists in synchronously exchanging data along all the links of the network, and performing individual computations at each node. That is, a t-round algorithm is an algorithm in which every node exchanges data with nodes at distance at most t (i.e., at most t hops away) from it.

It is worth taking the example of coloring for understanding the computational challenges induced by the question of locality in distributed network computing. The main concern of distributed coloring is solving the $(\Delta+1)$*-coloring task*, in which the nodes of a network G are free to choose any color from the set $\{1,\dots,\Delta+1\}$, where Δ is the maximum degree of G, as long as each node output a color that is different from all the colors output by its neighbors.[1] Several breakthroughs were almost simultaneously obtained towards the end of the 1980's. Awerbuch, Goldberg, Luby, and Plotkin [2] devised a deterministic distributed $(\Delta+1)$-coloring algorithm running in a subpolynomial-in-n number of rounds, which was subsequently improved by Panconesi and Srinivasan [33] to run in $2^{O(\sqrt{\log n})}$ rounds. Despite a quarter of a century of intensive research, this is still the best known distributed deterministic algorithm for $(\Delta+1)$-coloring in general graphs. Around the same time, Goldberg, Plotkin and Shannon [17] and Linial [25] designed distributed $(\Delta+1)$-coloring algorithms, performing in $O(\Delta^2+\log^* n)$ rounds, where $\log^* n$ denotes the least number of times the log-function should be applied on n to get a value smaller than one[2]. These algorithms are significantly faster than the one in [33] for graphs with reasonably small maximum degree (e.g., $\Delta = O(\log^c n)$ for arbitrarily large constant $c > 0$). Interestingly, the achieved dependence in n is optimal for constant degree graphs, as [25] also proves that 3-coloring the n-node ring requires $\Omega(\log^* n)$ rounds, and this lower bound also holds for randomized algorithms [30]. As a consequence, since Linial's contributions to $(\Delta+1)$-coloring, lots of effort has been devoted to decreasing the time dependence in Δ of coloring algorithms.

Partially supported by ENS Paris, the ANR-14-CE25-0006 project of the French National Research Agency and the CNRS PICS-07315 project. Additional support from ANR project DISPLEXITY and NCN grant 2015/17/B/ST6/01897. A full version of this paper is available at http://arxiv.org/abs/1511.01287.

[1] Solving k-coloring for $k < \Delta+1$ cannot be local, even if G is Δ-colorable, because the decision of a node can impact nodes far away from it, as witnessed by 2-coloring even cycles [25].

[2] Formally, define $\log^{(0)} x = x$, and $\log^{(k+1)} x = \log\log^{(k)} x$ for $k \geq 0$; Then $\log^* x$ denotes the least integer k such that $\log^{(k)} x < 1$.

Szegedy and Vishwanathan [39] show that a wide class of *locally iterative* algorithms for $(\Delta + 1)$-coloring must perform in $\Omega(\Delta \log \Delta)$ rounds, where an algorithm belongs to the locally iterative class if it has the property that, at each round, every node considers only its own current color together with the current colors of its neighbors, and updates its color value accordingly. This result was made more explicit by Kuhn and Wattenhofer [24], who considered an almost identically defined model and proposed a locally iterative algorithm performing in $O(\Delta \log \Delta + \log^* n)$ rounds. Three years later, Barenboim and Elkin [5], and Kuhn [22] independently proposed distributed $(\Delta + 1)$-coloring algorithms performing in $O(\Delta + \log^* n)$ rounds (see also [8]). These latter algorithms are not iterative. Finally, Barenboim [3] recently presented a distributed $(\Delta + 1)$-coloring algorithm performing in $O(\Delta^{3/4} \log \Delta + \log^* n)$ rounds.

Other forms of coloring problems have also been tackled in the distributed network computing setting, including *relaxations* of the classical vertex coloring problem, such as: edge-coloring, weak-coloring, defective coloring, vertex coloring with more than $(\Delta + 1)$ colors, etc. (see, e.g., [7] for a survey). In a number of practical scenarios, nodes aiming at breaking symmetry are also subject to more specific individual constraints. This is typically the case in frequency assignments in radio networks [16], [41], in scheduling [29], and in digital signal processing [42], to mention just a few scenarios. In all these latter settings, each node u is not initially free to choose any value from a color set $\mathcal{C}$, but is a priori restricted to choose only from some subset $L(u) \subseteq \mathcal{C}$ of colors. This framework is not captured by classical coloring, but rather by *list-coloring*. As in the case of vertex coloring, distributed list-coloring can be approached from a locality perspective only if the lists satisfy $|L(u)| \geq \deg_G(u) + 1$ for every node u of a graph G having degree $\deg_G(u)$.

Vertex $(\Delta + 1)$-coloring, as well as all of its previously mentioned relaxed variants, can be solved in $o(\Delta) + O(\log^* n)$ rounds [7]. However, the more complex task of $(\Delta + 1)$-list-coloring was (prior to this work) only known to be solvable in $\widetilde{O}(|\mathcal{C}|^{3/4}) + O(\log^* n)$ [3] rounds, which is sublinear-in-Δ only for $|\mathcal{C}| = o(\Delta^{4/3})$. Moreover, no sublinear (in Δ) algorithms are known for MIS or maximal matching, for which the currently best algorithms run is $O(\Delta) + \log^* n$ rounds [5], [7], [22]. (Again, the additional factor $\log^* n$ is unavoidable, and can be seen as an inherent cost of distributed symmetry breaking [38]). In fact, there is evidence suggesting that no sublinear algorithms exist for these problems. For instance, for maximal matching, a time lower bound of $\Omega(\Delta + \log^* s)$ is known to hold for an anonymous variant of the LOCAL model in which edges are equipped with locally unique identifiers from the range $\{1, \ldots, s\}$ [20]. In the standard LOCAL model, a lower bound of $\Omega(\Delta)$ is known to hold for the *fractional* variant of the maximal matching problem [18], while an $\Omega(\Delta/\log \Delta + \log^* n)$ lower bound holds for an extension of MIS called *greedy coloring* [15].

In order to better understand which tasks can be solved in a number of rounds sublinear in Δ, we focus on the general class of *locally checkable labelings* (LCL) introduced by Naor and Stockmeyer [31], which includes all tasks mentioned so far in this paper. Recall that a LCL is defined as a set of *bad* labeled balls in graphs, where the ball of radius $r \geq 0$ centered at node u in a graph G is the subgraph of G induced by all nodes at distance at most r from u in G (excluding edges between nodes at distance exactly r from u), and where a label is assigned to each node. For instance, the bad balls for coloring are the balls of radius 1 in which the center node has the same label as one of its neighbors. Similarly, the bad balls for MIS are the balls of radius 1 for which either the center of the ball as well as one of its neighbors are both in the MIS, or none of the nodes in the ball are in the MIS. Every ball which is not bad is *good*. To each LCL is associated a distributed task in which all nodes of an unlabeled graph G must collectively compute a label at each node, such that all balls are good. Thus, our general objective is to tackle the following question:

What LCL tasks can be deterministically solved in $o(\Delta) + O(\log^* n)$ rounds?

Given the state-of-the-art, we know since recently that answer to the above question is affirmative for $(\Delta + 1)$-coloring [3], and there is also some very partial evidence hinting that this may not be true for MIS-type problems [15], [18]. This also leads us to ask what makes $(\Delta + 1)$-coloring and MIS so different? In the study of the randomized LOCAL model, a separation in time complexity between $(\Delta + 1)$-coloring and MIS has very recently been obtained by contrasting the randomized $(\Delta+1)$-coloring algorithms of Harris, Schneider, and Su [19] with lower bounds for MIS due to Kuhn, Moscibroda, and Wattenhoffer [23]. However, this separation does not carry over directly to the deterministic setting. Here, in an attempt to advance understanding of the question for the deterministic scenario, we put forward the framework of *conflict coloring*, and show that efficient solutions to problems in the LOCAL model can be obtained by taking advantage of their amenability to the conflict coloring framework.

B. Our Results

a) The setting: We define the general *conflict coloring* task, which can be instantiated so as to correspond to any given LCL task. Roughly, conflict coloring is defined by a list of candidate colors given to each node (in the same spirit as list-coloring), and a list of conflicts between colors associated to each edge (following a convention used, e.g., when formulating unique games, CSP-s with binary conflict relations, etc.). For edge $\{u, v\}$, a conflict is a pair of the form (c_u, c_v), indicating that a coloring where u has color c_u and v has color c_v is illegal. Intuitively, given a LCL, the corresponding instance of conflict coloring is obtained by giving the list of all good balls centered at u to every node u, and two balls given to adjacent nodes are in conflict whenever they are not consistent. Every LCL task is therefore a possible instantiation of conflict coloring (a given LCL task may have more than one conflict coloring representation). Note however that the power of conflict coloring extends beyond such a formulation of LCL tasks: depending on the instance, two colors in conflict along an edge e do not, in general, need to be in conflict along another edge $e' \neq e$.

We will speak of a conflict coloring with lists of length l and conflict degree d, or more compactly of (l, d)-conflict-coloring, when all color lists given to the nodes are of length at least l, and for every edge e and color c, the number of colors conflicting with color c on edge e does not exceed d. Intuitively, the larger the value of l, the easier the problem is, as every node has a choice among a large number of outputs. Conversely, the larger d is, the harder the problem becomes as some nodes have to deal with many conflicts with at least one of their neighbors.

b) Distributed algorithm: Our main result is the design of a generic distributed algorithm which solves the conflict coloring task whenever $l/d > \Delta$ in graphs with maximum degree Δ. In the classical LOCAL model for distributed network computing, our algorithm performs in $\widetilde{O}(\sqrt{\Delta}) + \log^* n$ rounds in n-node graphs, where the $\widetilde{O}$ notation disregards polylogarithmic factors in Δ.

The implications of our result are the following. There exists a trivial representation of $(\Delta+1)$-coloring as a conflict coloring task with $l/d \geq \Delta + 1$. Therefore, our algorithm can be used to solve $(\Delta+1)$-coloring in $\widetilde{O}(\sqrt{\Delta}) + \log^* n$ rounds, which outperforms the currently fastest known $(\Delta + 1)$-coloring algorithm by Barenboim [3] performing in $\widetilde{O}(\Delta^{3/4}) + \log^* n$ rounds. In fact, for most classical variants of coloring, including $(2\Delta-1)$-edge-coloring, $(\Delta+1)$-list-coloring, coloring with forbidden color-distance sets [36] given a sufficiently large palette, etc., our algorithm solves all these tasks in $\widetilde{O}(\sqrt{\Delta}) + \log^* n$ rounds, also improving the best results known for each of them. For small values of Δ, our (deterministic) algorithm for conflict coloring is even faster than the best known *randomized* algorithms for $(\Delta + 1)$-coloring [19].

Interestingly, the bound $l/d > \Delta$ is essentially the best bound for which there exists a generic algorithm solving conflict coloring locally. Indeed, for every l and d such that $l/d \leq \Delta$, there exists an instance of conflict coloring for which no solutions can be sequentially computed by a greedy algorithm selecting the nodes in arbitrary order. That is, the output of a node can impact the possible legal outputs of far away nodes in the network (like for Δ-coloring [25]). In particular, we are not aware of any instantiations of conflict coloring for MIS or maximal matching satisfying $l/d > \Delta$, which prevents us from solving these problems with a generic algorithm for conflict coloring. It might well be the case that there are no instantiation of conflict coloring for these problems satisfying $l/d > \Delta$, which might be another hint that there are no algorithms running in $o(\Delta) + O(\log^* n)$ rounds for these tasks.

c) The techniques: From a technical point of view, the design of our algorithm required the development of a new technique, called a *simplification mechanism*. This mechanism aims at iteratively reducing the difficulty of a given problem until it becomes simple enough to be trivially solved. More specifically, let P_0 be the problem we are aiming at solving. Our mechanism constructs a sequence $P_1, \ldots, P_t$ of problems with the following three properties: (1) P_{k+1} is "easier" to solve than P_k, and can be constructed from P_k in $O(1)$ rounds, (2) P_t is simple enough to be solved individually at each node, without any communication, and (3) given a solution to P_{k+1}, there is a $O(1)$-round algorithm computing a solution to P_k. Conflict coloring is perfectly suited to an application of the aforementioned simplification mechanism. Indeed, the set of colors in P_0 are those in the lists given to the nodes in G. Constructing P_{k+1} from P_k increases the size of the lists (which is good), but the number of conflicts between colors also increases (which is bad). However, the increase rate of the number of conflicts will be shown to be lower than the increase rate of the size of the lists, which will eventually ensure that P_t is easily solvable thanks to large lists, but a relatively small number of conflicts.

In conflict coloring, the main difficulty lies in obtaining a $O(\log^* n)$-round algorithm for solving an instance with ratio $l/d \geq 10\vec{\Delta}^2 \ln \Delta$, given a graph with maximum degree Δ and edge orientation with maximum outdegree $\vec{\Delta}$. Subsequently, the conflict coloring problem then turns out to be directly amenable to an application of the *arbdefective* coloring approach (cf. [6], [7]), without having to resort to constructions of polynomials of the type used in [3] during the recombination phase. This is because the class of conflict coloring problems solved by our algorithm includes *precoloring extension* (i.e., completing a partially given coloring of a graph), which can be handled directly through a modification of color lists available to vertices. By a careful (adaptive) choice of parameters of the arbdefective coloring, the complexity of our algorithm is reduced to $\widetilde{O}(\sqrt{\Delta}) + \log^* n$ rounds.

Disregarding polylogarithmic-in-Δ factors, the $\sqrt{\Delta}$-running time of our algorithm appears to be the limit of the precoloring extension technique, unless radically new algorithms are found to construct colorings in $O(\log^* n)$ rounds using significantly fewer colors than $\widetilde{O}(\Delta^2)$. This latter problem has resisted all attempts for more than 20 years, since the publication of [25].

d) Additional results: Our result has also impact on *centralized local* computation [12], [27], [28], [32], [37]. In this model, the local computation algorithm (LCA) is executed by a single computing unit which has access to the whole input graph, and needs to answer queries about a solution to the considered problem (e.g., "is node u in the MIS?"). For answering queries, the LCA probes the input graph, learning in each probe about some node u and its neighborhood. The answers to the queries provided by the LCA must be consistent, that is, there must exist an implicit global solution that fits with the answers of the LCA. The complexity of such an algorithm is the number of probes that the LCA performs per query. Using our algorithm for conflict coloring, we show that there is a deterministic oblivious LCA for solving $(\Delta+1)$-list-coloring (and thus also $(\Delta + 1)$-coloring) using only $\Delta^{O(\sqrt{\Delta}\log^{5/2}\Delta)} \log^* n$ probes, improving the bound in [12].

C. Other Related Work

In addition to the aforementioned deterministic algorithms for $(\Delta+1)$-coloring, it is worth mentioning the *randomized* algorithms for MIS in [1], [26], which both perform in $O(\log n)$ rounds, with high probability. Both algorithms can be transformed into randomized $(\Delta+1)$-coloring algorithms with the same round-complexity (e.g., using the reduction in [25]). A "direct" randomized algorithm for $(\Delta+1)$-coloring with the same performances as these latter algorithms can be found in [7]. As a function of Δ and n, the best known randomized algorithms for $(\Delta+1)$-coloring, as well as for $(\Delta+1)$-list-coloring, perform in $O(\sqrt{\log\Delta})+2^{O(\sqrt{\log\log n})}$ rounds with high probability [19]. This result, combined with a previous lower bound on MIS of $\Omega(\log\Delta/\log\log\Delta)$ rounds [23], which also holds for randomized algorithms, implies a separation between the $(\Delta+1)$-coloring and MIS problems in the randomized case. On the positive side, MIS can be solved in $O(\log^2\Delta)+2^{O(\sqrt{\log\log n})}$ rounds with high probability [9]. We remark that the randomized and deterministic flavors of the LOCAL model are significantly different, and in fact admit an exponential time separation, which has been recently shown for specific case of the problem of coloring a tree with Δ colors [10]. Whether a similar separation between randomized and deterministic complexity holds for MIS and the general $(\Delta+1)$-coloring problem is one of the main open questions of the field.

The list-coloring problem was introduced independently by Vizing [40], and Erdös, et al. [11]. It is defined as follows. Let G be a graph, let $\mathcal{C}$ be a set of colors, and let $L: V\to 2^{\mathcal{C}}$. If there exists a function $f: V\to\mathcal{C}$ such that $f(v)\in L(v)$ for every $v\in V(G)$, and $f(u)\neq f(v)$ for every $\{u,v\}\in E(G)$, then G is said to be L-list-colorable. A graph is *k-choosable*, or *k-list-colorable*, if it has a list-coloring no matter how one assigns a list of k colors to each node. The *choosability number* $\mathsf{ch}(G)$ of a graph G is the least number k such that G is k-choosable. Clearly, $\mathsf{ch}(G)\geq\chi(G)$, where $\chi(G)$ denotes the chromatic number of G. Computing the choosability number is actually believed to be harder than computing the chromatic number, because deciding the former is Π_2^{P}-complete, while deciding the latter is NP-complete. In a distributed setting, $(\Delta+1)$-list-coloring is solvable in $O(\Delta+\log^* n)$ rounds by reduction to vertex-coloring. It was also recently proved to be solvable in time dependent on the size of the set of allowed colors, in $\widetilde{O}(|\mathcal{C}|^{3/4})+O(\log^* n)$ rounds [3].

It is also worth specifically mentioning the *weak-coloring* problem, which asks for a coloring of the nodes such that every non isolated node has at least one neighbor colored differently from it. It was proved in [31] that, in bounded-degree graphs with odd degrees, 2-weak-coloring can be solved in a constant number of rounds. This is one of the rare non-trivial distributed symmetry-breaking tasks that are known to be solvable in a constant number of rounds (in general, it is undecidable whether a solution to a locally checkable task can be constructed in constant time [31]). In graphs with constant maximum degree, for all locally checkable tasks, as well as their probabilistic extension [14], any randomized construction algorithm running in a constant number of rounds can be derandomized into a deterministic algorithm running in the same number of rounds [13], [31]. However, this derandomization result does not necessarily hold for randomized algorithms running in a *non-constant* numbers of rounds. For example, it is not known whether there exists a deterministic $(\Delta+1)$-coloring algorithm running in a polylogarithmic number of rounds, or in other words, it is not known whether randomization helps for distributed $(\Delta+1)$-coloring.

Many other types of coloring have been considered in the literature, including using a larger number of colors, coloring the edges instead of the nodes, defective coloring, etc., and some of these tasks have been tackled in specific classes of graphs (planar, bounded arboricity, etc.). We refer to [7] for an excellent survey, also describing the various techniques of reductions between coloring, MIS, maximal matching, etc.

Regarding the centralized local model, essentially the same problems as for the distributed LOCAL model have been studied, such as, e.g., maximal independent set [37], and Maximum Matching [27], for which algorithms were devised. A recent paper [12] studies the relationship between the LOCAL model and the centralized local computation model, including ways to adapt algorithms from the LOCAL model to the centralized local setting. The resulting LCAs are deterministic and oblivious (they do not require to store information between queries), and, above all, they require a smaller number of probes than previously known algorithms. In particular, the method from [12] yields a centralized Δ^2-coloring LCA running in $O(\mathrm{poly}(\Delta)\cdot\log^* n)$ probes per query, and a centralized $(\Delta+1)$-coloring LCA running in $\Delta^{O(\Delta^2)}\cdot\log^* n$ probes per query.

II. Model, Problem Setting, and Preliminaries

A. The LOCAL Model

We consider the usual framework for the analysis of locality in network computing, namely the LOCAL model [35]. In this model, a network is modeled as a *connected* and *simple* n-node graph (i.e., no loops, and no multiple edges). Each node v of a network is given an *identity*, denoted by $\mathsf{id}(v)$. This identity is a positive integer that is assumed to be encoded on $O(\log n)$ bits, and the identities of the nodes in the same network are pairwise distinct. In addition, every node v may also be given an *input* $\mathsf{inp}(v)\in\{0,1\}^*$.

For the sake of defining conflict coloring, we assume that the edges incident to a degree-δ node are identified by pairwise distinct labels in $\{1,\dots,\delta\}$, called *port numbers*. No consistency between the port numbers at different nodes is assumed (in particular, an edge may have two different port numbers at its two extremities). Again, these port numbers are solely used for describing the input to every node in the context of conflict coloring, and provide no additional computing power to the LOCAL model (since nodes have identities).

In any execution of an algorithm $\mathcal{A}$ in the LOCAL model, all nodes start at the same time. Initially, every node is only aware of its identity, and its input. As is usual in the framework of network computing, and for simplifying the description of the algorithm, we also assume that each node initially knows a polynomial upper bound on the total number n of nodes. (See [21] for techniques enabling to get rid of this assumption). Then all nodes perform a sequence of *synchronous rounds*. At each round, every node sends a message to its neighbors, receives the messages of its neighbors, and performs some individual computation. Which messages to send, and what computation to perform depend on the algorithm $\mathcal{A}$. The complexity of algorithm $\mathcal{A}$ in n-node graphs is the maximum, taken over all n-node graphs G, of the number of rounds performed by $\mathcal{A}$ in G until all nodes terminate.

Note that, whenever t is known a priori, an algorithm $\mathcal{A}$ performing in t rounds can be simulated by an algorithm $\mathcal{B}$ performing in two phases: First, in a network G, every node v collects all data from nodes at hop distance at most t from v (i.e., their identities, their inputs, as well as the structure of the connections between these nodes); Second, every node simulates the execution of $\mathcal{A}$ in $B_G(v,t)$, where $B_G(v,t)$ is the *ball* of radius t around node v in graph G, that is, $B_G(v,t)$ is the subgraph of G induced by all nodes at distance at most t from v, excluding the edges between the nodes at distance exactly t from v. Hence, the LOCAL model enables to measure the *locality* of a problem.

An algorithm satisfying the property that the output of every node is the same for all possible identity assignments to the nodes of the network is called *identity-oblivious*, or *ID-oblivious* for short.

a) Notation: We denote by $\deg_G(v)$ the *degree* of a node v in a graph G, that is the number of neighbors of v in G, or, alternatively, the number of edges incident to v in G (recall that G is a simple graph). We denote by $\Delta_G = \max_{v \in V(G)} \deg_G(v)$ the maximum degree of the nodes in G. The set of neighbors of node v in graph G is denoted by $N_G(v)$. Given an orientation of the edges of G, the set of out-neighbors of v (nodes connected to v by edges having their tail at v) is denoted by $\vec{N}_G(v)$, and the maximum node outdegree is denoted by $\vec{\Delta}_G$. When the graph G is clear from the context, the index G will be omitted from notation.

B. Conflict Coloring

Conflict coloring is defined as follows. Let $\mathcal{C}$ be a finite set, whose elements are called colors. In graph G, each node $u \in V(G)$ is given as input

- a list $L(u)$ of colors in $\mathcal{C}$, and
- for every port number $i \in \{1, \dots, \deg_G(u)\}$, a list $C_i(u) = \big((c_1, c'_1), \dots, (c_k, c'_k)\big)$ of conflicts, where $c_j \in L(u)$ and $c'_j \in \mathcal{C}$ for every $j = 1, \dots, k$.

To be well defined, the instance must satisfy the constraint that if $(c, c') \in C_i(u)$ and u' is the neighbor of u reachable from u via port i, then $(c', c) \in C_j(u')$, where j is the port number of edge $\{u, u'\}$ at u'. Each node u in G must output a color $\mathsf{out}(u) \in L(u)$ such that, for every edge $\{u, v\}$ with port number i at u, we have $(\mathsf{out}(u), \mathsf{out}(v)) \notin C_i(u)$. That is, two adjacent nodes cannot be colored with a pair of colors that is indicated as a conflict for that edge. A given conflict coloring instance has conflict degree d if, for all colors c, there are at most d pairs of the form $(c, \cdot)$ in any of the lists $C_i(u)$. The conflict degree d represents the maximum number of possible conflicts of one colors with other colors of one given neighbor.

For instance, $(\Delta + 1)$-coloring is the instance of conflict coloring with $L(u) = \{1, \dots, \Delta + 1\}$, and all conflict lists are of the form (c, c) for all $c \in \{1, \dots, \Delta + 1\}$. Expressing MIS as an instance of conflict coloring is not as straightforward. One way of doing this is the following. Assign lists of the form $L(u) = \{0, 1\} \times \{1, \dots, \Delta\}$ to every node u. A color is thus a pair of integer values, where a color in the form of a pair $(1, i)$, for any $i \in \{1, \dots, \Delta\}$, is interpreted as "$u \in$ MIS", and a color $(0, i)$ is likewise interpreted as "$u \notin$ MIS, but the neighbor of u reachable via port i belongs to the MIS". We set a conflict along the edge from vertex v, following the i-th port to a neighboring vertex u, for all color pairs of the form $(1, j)$ at v and $(1, k)$ at u, for all $j, k \in \{1, \dots, \Delta\}$, as well as for all color pairs of the form $(0, i)$ at v and $(0, j)$ at u, for all $j \in \{1, \dots, \Delta\}$.

In fact, any LCL task can be expressed as an instance of conflict coloring. To see why, let us revisit MIS, and let us define MIS as an instance of conflict coloring in a brute force manner. One assigns $L(u) = \{S_1, \dots, S_{2^\delta}\}$ to every node u of degree δ, where S_1 is the $(\delta + 1)$-node star with center labeled 1 and all leaves labeled 0, and, for $j > 1$, S_j is a $(\delta + 1)$-node star with center labeled 0, $\binom{\delta}{x}$ leaves labeled 1 for some $x \in \{1, \dots, \delta\}$, and all other leaves labeled 0. Conflicts in $C_i(u)$ are between incompatible stars S_j at u and S'_k at u' where the latter is the neighboring node of u reachable from u via port i. More generally, any LCL task can be expressed as an instance of conflict coloring by assigning to every node u a list of colors consisting of all good balls centered at u, and conflicts are between inconsistent balls between neighboring nodes.

For the sake of describing our algorithm, we define the *conflict graph* F associated to an instance of conflict coloring on $G = (V, E)$. The conflict graph F is the simple undirected graph with vertex set

$$V(F) = \{(v, c) : v \in V(G), c \in L(v)\},$$

and edge set

$$E(F) = \{\{(u, c), (u', c')\} : (c, c') \in C_i(u)\}.$$

where i is the port number of edge $\{u, u'\} \in E(G)$ at node u. In other words, to every edge $e = \{u, u'\} \in E(G)$ corresponds a bipartite graph with partitions $L(u)$ and $L(u')$, and there is an edge between a color $c \in L(u)$ and a color $c' \in L(u')$ if and only if these two colors are in conflict for edge e. For a conflict coloring in a graph of maximum degree Δ, and conflict degree d, the conflict graph has degree at most $d\Delta$.

Let us note that, in conflict coloring, there is an interplay between the size, l, of the lists of available colors at each node (the larger the better as far as solving the task is concerned), and the conflict degree, d, of the colors along each edge (the smaller the better). We define (l, d)-*conflict coloring* as conflict coloring with all lists of size l, and the conflict degree at most d. In the rest of the paper, we shall show that if the ratio between these two quantities is large enough, namely $l/d > \Delta$, then (l,d)-conflict coloring in solvable in a sublinear (in Δ) number of rounds. For instance, $(\Delta+1)$-list-coloring corresponds to $l = \Delta + 1$, and $d = 1$, hence the ratio l/d is sufficient to be covered by our approach. By contrast, for the previously described representation of MIS as conflict coloring, we have $l = 2\Delta$ and $d = \Delta$, hence $l/d = 2$.[3]

C. Organization and Proof Outline

In Sections III and IV we provide the techniques and algorithms for solving (l,d)-conflict coloring for $l/d > \Delta$. Section III lays out the main ingredient, namely, a routine for conflict coloring in $O(\log^* n)$ rounds when $l/d \geq 10\Delta^2 \ln\Delta$ in a graph of maximum degree Δ, or more generally when $l/d \geq 10\vec{\Delta}^2 \ln\Delta$ and an orientation of the edges of the graph with outdegree $\vec{\Delta}$ is given. This is achieved by an application of our instance simplification technique, since the existence of color lists in the problem description precludes the application of simpler color reduction mechanisms (e.g., of the sort used by Linial [25] for Δ^2-vertex coloring). In Section IV we then solve any conflict coloring problem with $l/d > \Delta$ by applying the routines from Section III on specific vertex-disjoint oriented subgraphs of G. These subgraphs are carefully constructed using the technique of arbdefective coloring [3], in such a way as to have sufficiently small outdegree β for the condition $l/d \geq 10\beta^2 \ln\Delta$ to hold within them. Finally, in Section V we discuss implications of our conflict coloring routines for centralized LCAs, both in the case of $l/d \geq 10\Delta^2 \ln\Delta$ and $l/d > \Delta$.

III. Instance Simplification

Our simplification mechanism, which allows us to generate progressively easier conflict coloring problems on a graph G, is now captured by the following key lemma. We will apply it to "simplify" a (l,d)-conflict-coloring problem $P = P_0$, such that $l/d \geq 10\vec{\Delta}^2 \ln\Delta$, into one with a larger ratio l/d.

Lemma III.1. *Let G be a graph with maximum degree Δ and a given edge orientation of outdegree at most $\vec{\Delta}$. Let P_i be an instance of a (l_i, d_i)-conflict-coloring problem on graph G. Then, for some integers l_{i+1}, d_{i+1}, there exists an instance P_{i+1} of (l_{i+1}, d_{i+1})-conflict-coloring on graph G, such that:*

1) *There exists an ID-oblivious single-round local distributed algorithm which, given the* input *of each node in P_i, outputs for each node its input in P_{i+1}.*
2) *There exists an ID-oblivious single-round local distributed algorithm which, given any valid* output *of each node in P_{i+1}, outputs for each node a valid output for P_i.*
3) *The following condition is fulfilled for any $\varepsilon > 0$, when $\vec{\Delta}$ is larger than a sufficiently large constant:*

$$\frac{l_{i+1}}{d_{i+1}} > \frac{1}{\Delta} \exp\left(\frac{1}{(e^2+\varepsilon)\vec{\Delta}^2} \frac{l_i}{d_i}\right).$$

We construct instance $P_{i+1} = (L_{i+1}, F_{i+1})$ over color set $\mathcal{C}_{i+1}$ from instance $P_i = (L_i, F_i)$ over color set $\mathcal{C}_i$ as follows. We define the color set $\mathcal{C}_{i+1}$ as the collection of all the subsets of size $k_i = \lfloor \frac{l_i}{e^2 d_i \vec{\Delta}} \rfloor$ of $\mathcal{C}_i$. For each node v, we will now appropriately define its color list $L_{i+1}(v) \subseteq \binom{L_i(v)}{k_i}$ by selecting into $L_{i+1}(v)$ a constant proportion of all k_i-element-subsets of $L_i(v)$. The adopted value of parameter k_i is the result of a certain tradeoff: increasing k_i further would indeed increase the list length l_{i+1}, but would also result in an explosion of the number of conflicts d_{i+1} (the ratio l_{i+1}/d_{i+1} needs to be controlled in view of Clause 3). The details of the construction of lists L_{i+1} are deferred until later in the proof.

Next, let $\tau_i = \lfloor \frac{k_i}{\vec{\Delta}} \rfloor - 1$ be a threshold parameter, which we will use to define the edge set of the conflict graph F_{i+1}. For a pair of neighboring nodes $\{u, v\} \in E$, we denote by $S_i^u(v, c_v)$ the set of all colors at vertex u in conflict with color c_v at vertex v in problem P_i. We now define the following symmetric conflict relation $(\sim)$ on $V \times \mathcal{C}_{i+1}$ for the problem P_{i+1}:

$$(u, C_u) \sim (v, C_v) \Leftrightarrow \left\{ \begin{array}{cl} & \left|C_u \cap \bigcup_{c_v \in C_v} S_i^u(v, c_v)\right| > \tau_i \\ \text{or} & \left|C_v \cap \bigcup_{c_u \in C_u} S_i^v(u, c_u)\right| > \tau_i \end{array} \right.$$

When looking a the left-hand-side of the above relation, it is convenient to think of C_u and C_v as candidates for color values, which are being considered for inclusion in the lists $L_{i+1}(u)$ and $L_{i+1}(v)$ of nodes u and v, respectively, in problem P_{i+1}. When looking at the right-hand side, we treat C_u and C_v as sets of colors with respect to problem P_i. Subsequently, when defining the color lists in problem P_{i+1}, we will eliminate those configurations of candidates which generate too many conflicts in node neighborhoods in problem P_i.

The above relation, when restricted to permissible vertex colors, defines conflict edges for F_{i+1}: given colors $C_u \in L_{i+1}(u)$ and $C_v \in L_{i+1}(v)$ (where we recall that $C_u \subseteq L_i(u)$ and $C_v \subseteq L_i(v)$), we put:

$$\{(u, C_u), (v, C_v)\} \in E(F_{i+1}) \iff (u, C_u) \sim (v, C_v). \quad (1)$$

For this definition of the edge set of F_{i+1}, we immediately show how to convert any valid solution to P_{i+1} into a solution for P_i in a single communication round. Indeed, observe that if a node v knows its output $\mathsf{out}_{i+1}(v)$ for P_{i+1} and the outputs of all its out-neighbors in the considered orientation, then it can obtain a valid color in P_i by returning an arbitrary element of the set $\mathsf{out}_{i+1}(v)$ which does not conflict with any of the colors belonging to the corresponding sets of its out-neighbors:

$$\mathsf{out}_i(v) \in \mathsf{out}_{i+1}(v) \setminus \bigcup_{u \in \vec{N}_G(v)} \bigcup_{c_u \in \mathsf{out}_{i+1}(u)} S_i^v(u, c_u). \quad (2)$$

[3] A simple argument illustrating that $l/d = 2$ is essentially the best ratio which can be achieved when using natural conflict-coloring-based representations of MIS is given in the full version of the paper.

Since, by assumption, the considered solution to P_{i+1} was correct, we have $(u, \mathsf{out}_{i+1}(u)) \not\sim (v, \mathsf{out}_{i+1}(v))$. It follows from the definition of relation $(\sim)$ that in the right-hand-side of expression (2), each element of the union over $u \in \vec{N}_G(v)$ eliminates at most τ_i elements from the set $\mathsf{out}_{i+1}(v)$. Moreover, since we have $|\mathsf{out}_{i+1}(v)| = k_i \geq \vec{\Delta}\tau_i + 1$, the set from which we are choosing $\mathsf{out}_i(v)$ is always non-empty. Finally, the construction of (2) is such that color $\mathsf{out}_i(v)$ cannot conflict with any other color assigned to any of its neighbors in the obtained solution to P_i. Thus the obtained solution to P_i is conflict-free with respect to every edge of G, which completes the proof of Clause 2 of the Lemma.

In the rest of the construction, we focus on a careful construction of color lists $L_{i+1}(v) \subseteq \binom{L_i(v)}{k_i}$, so as to ensure the local constructibility of the input instance to P_{i+1} in a single round (Clause 1) and a sufficiently large ratio l_{i+1}/d_{i+1} (Clause 3). The value of d_{i+1} will be fixed as:

$$d_{i+1} := 8\Delta \binom{k_i d_i}{\tau_i} \binom{l_i}{k_i - \tau_i}.$$

We will proceed with the construction of lists L_{i+1} by including all k_i-element subsets of $L_i(v)$ in $L_{i+1}(v)$, and then we eliminate some colors from $L_{i+1}(v)$ which would generate too many conflicts in P_{i+1} with any of the possible colors for neighbors $u \in N_G(v)$. Formally, for all $v \in V$, we set:

$$D_{i,v}(u) := \left\{ C_v : |\{C_u : (u, C_u) \sim (v, C_v)\}| > \frac{d_{i+1}}{2} \right\} \quad (3)$$

$$L_{i+1}(v) := \binom{L_i(v)}{k_i} \setminus \bigcup_{u \in N_G(v)} D_{i,v}(u) \quad (4)$$

The above setting guarantees that the conflict degree bound of d_{i+1} is indeed satisfied by problem P_{i+1}. We now show that the condition $|L_{i+1}(v)| \geq \frac{1}{2}\binom{l_i}{k_i}$ is met for all vertices. To lower bound the size of $L_{i+1}(v)$, we will prove that for each neighbor u of a node v, at most $\frac{1}{2\Delta}\binom{l_i}{k_i}$ subsets are removed from $L_{i+1}(v)$ when considering conflicts between u and v.

Claim III.1. *For any $v \in V$ and $u \in N_G(v)$, we have:*

$$|D_{i,v}(u)| \leq \frac{1}{2\Delta}\binom{l_i}{k_i}$$

Proof. Consider the bipartite graph with vertex partition $A_v \cup A_u$, where $A_v = \{(v, C_v) : C_v \in \binom{L_i(v)}{k_i}\}$ and $A_u = \{(u, C_u) : C_u \in \binom{L_i(u)}{k_i}\}$, and a set of edges $E_\sim$ defined by the conflict relation $(u, C_u) \sim (v, C_v)$ on its nodes. Our goal is to bound the number of vertices in partition A_v having degree at least $\frac{d_{i+1}}{2}$ with respect to $E_\sim$. We will first bound the number of edges in $E_\sim$ as follows. For a fixed set $C_u \in \binom{L_i(u)}{k_i}$, we bound the number x_1 of sets $C_v \in \binom{L_i(v)}{k_i}$ satisfying the first of the conditions which appear in the definition of relation $(\sim)$:

$$\left| C_u \cap \bigcup_{c_v \in C_v} S_i^u(v, c_v) \right| > \tau_i. \quad (5)$$

Taking into account that P_i is an instance of conflict coloring with conflict degree at most d_i, for any color c_v at v we have $S_i^u(v, c_v)$, and so $|\bigcup_{c_v \in C_v} S_i^u(v, c_v)| \leq \sum_{c_v \in C_v} d_i = k_i d_i$. It follows that x_1 can be bounded by the following expression:

$$x_1 \leq \binom{k_i d_i}{\tau_i}\binom{l_i}{k_i - \tau_i} = \frac{1}{8\Delta} d_{i+1}.$$

Thus, overall, the number of edges of $E_\sim$ satisfying Eq. (5) is at most $x_1 |A_u| \leq \frac{1}{8\Delta} d_{i+1}\binom{l_i}{k_i}$. By a symmetric argument, the number of edges contributed by the other condition in the definition of relation $(\sim)$ (i.e., $\left|C_v \cap \bigcup_{c_u \in C_u} S_i^v(u, c_u)\right| > \tau_i$), is also $\frac{1}{8\Delta} d_{i+1}\binom{l_i}{k_i}$. Overall, we have:

$$|E_\sim| \leq \frac{1}{4\Delta} d_{i+1}\binom{l_i}{k_i}.$$

The average degree $\delta_\sim$ of a node in A_v with respect to $E_\sim$ is thus bounded by $\delta_\sim \leq \frac{1}{4\Delta} d_{i+1}$. Since only at most $\frac{|A_v|}{2\Delta} = \frac{1}{2\Delta}\binom{l_i}{k_i}$ nodes in A_v can have a degree higher than $2\Delta\delta_\sim \leq \frac{d_{i+1}}{2}$, the claim follows. □

As a direct corollary of the above claim and of the definition of $L_{i+1}(v)$ in (3), we have obtained the sought bound $|L_{i+1}(v)| \geq \frac{1}{2}\binom{l_i}{k_i}$. Formally, to guarantee that P_{i+1} is an instance of a (l_{i+1}, d_{i+1})-conflict-coloring problem with lists of size precisely equal to:

$$l_{i+1} := \frac{1}{2}\binom{l_i}{k_i},$$

in the case when the size of some $L_{i+1}(v)$ still exceeds l_{i+1}, node v removes arbitrarily some elements of $L_{i+1}(v)$ so that its size becomes exactly l_{i+1}. Bearing in mind the description of color lists L_{i+1} according to Eq. (3) and the edges of the conflict graph F_{i+1} according to Eq. (1), a single-round distributed algorithm for computing an instance of P_{i+1} based on an instance of P_i follows directly from the construction. This completes the proof of Clause 1 of the Lemma.

Finally, we complete the proof by showing that Clause 3 is also satisfied by taking into account the known properties of l_{i+1}, d_{i+1}, k_i, and τ_i. For all omitted proofs, we refer the reader to the full version of the paper. □

The following lemma provides a criterion which allows us to determine the necessary number of iterations of the proposed simplification mechanism. It states that we can solve a (l, d)-conflict-coloring problem directly, without any further communication, given that the ratio l/d is sufficiently large, subject to some additional assumptions constraining the structure of the input instance. This is achieved through a greedy assignment of colors for the sufficiently simplified problem instance.

Lemma III.2. *Consider an instance of the (l, d)-conflict coloring problem on a graph G with maximum degree Δ, such that the list of colors available to all nodes is $\{1, \ldots, l\}$. Suppose the following information available to all nodes:*

- *Each node $v \in V$ receives its input $\mathsf{inp}(v)$ for the corresponding (l, d)-conflict coloring instance P for v, accompanied by an integer label $\lambda(v) \in \{1, \ldots, s\}$,* *such that $\lambda(V)$ is a s-vertex-coloring of the graph (i.e., $\lambda(u) \neq \lambda(v)$ for all $\{u,v\} \in E(G)$),*

- *A promise is given to all nodes $v \in V$ that $\mathsf{inp}(v) \in I$, where I is a set known to all nodes.*

If $\frac{l}{d} > \Delta s|I|$, then a solution to P can be found in a local manner without communication (in 0 rounds).

We can now combine the claims of Lemma III.1 and Lemma III.2 to show that any conflict coloring problem P_0, given a sufficiently large initial ratio l_0/d_0, will after a small number of rounds be simplified by iterated application of Lemma III.1 into a conflict coloring problem P_t, which is solvable without communication in view of Lemma III.2. This leads us to the main technical lemma of this Section.

Lemma III.3. *For a graph G with maximum degree Δ, a given orientation of edges with maximum outdegree $\vec{\Delta}$, and a s-coloring of the vertex set, any instance of the (l,d)-conflict coloring problem with $\frac{l}{d} \geq 10\vec{\Delta}^2 \ln \Delta$ can be solved with a local distributed algorithm in $O(\log^* s + \log^* \Delta + \log^* d)$ rounds.*

Lemma III.3 can be applied to solve (l,d)-conflict coloring on any graph G, using a $O(\Delta^2)$ initial coloring. This coloring is computed in $\log^* n + O(1)$ rounds using Linal's algorithm [25]. We thus obtain the following theorem.

Theorem III.1. *There is a local distributed algorithm which solves the (l,d)-conflict-coloring problem in $O(\log^* d + \log^* \Delta) + \log^* n$ rounds when $\frac{l}{d} \geq 10\Delta^2 \ln \Delta$.* □

For example, for the special case of list coloring, this gives the following corollary.

Corollary III.1. *There is a local distributed algorithm which finds a $(10\Delta^2 \ln \Delta)$-list-coloring in $\log^* n + O(\log^* \Delta)$ rounds.*

In the next section, we will use Theorem III.1 as a building block for solving conflict coloring instances with a smaller value of ratio l/d.

IV. Conflict Coloring with a Small Number of Colors

In this section we show how to apply the techniques from Section III to obtain a distributed solution to (l,d)-conflict coloring problems with $l \geq d \cdot \Delta + 1$, such as $(\Delta+1)$-list-coloring.

Whereas we choose to speak of conflict colorings throughout the rest of the paper, we will no longer make use of the general structure of conflict colorings in our technical arguments. The reader focusing on results directly relevant to the $(\Delta+1)$-coloring problem may from now on assume that the problem being solved is $(\Delta+1)$-list-coloring (and specifically, that the conflict degree is $d=1$), and in this context, may rely on Corollary III.1 instead of Theorem III.1 as the relevant ingredient used in the subsequent construction.

In the designed algorithm we will also make use of the following recent result on arbdefective coloring, shown by Barenboim [3]. For $\beta \geq 0$, a (possibly improper) vertex coloring of a graph G is said to be *β-arbdefective* if there is an orientation of the edges of G such that, for every node v, at most β out-neighbors of v have the same color as v.

Lemma IV.1 ([3]). *There is a distributed algorithm, parameterized by $k \geq 1$, which, given any graph G with a Δ^2-coloring of its vertex set, produces for $\beta = O(\frac{\Delta}{k} \log \Delta)$ a β-arbdefective k-coloring $V = V_1 \cup \ldots \cup V_k$ of G, together with a corresponding orientation of each $G[V_i]$ having outdegree at most β. The running time of the algorithm is $O(k \log \Delta)$ rounds.*

Our conflict coloring procedure will assume our graph G is already equipped with a Δ^2-coloring. This can be initially computed using Linial's algorithm [25], in $\log^* n$ rounds.

Lemma IV.2. *Given a Δ^2-vertex coloring of graph G of maximum degree at most Δ, there is an algorithm which solves any conflict-coloring instance on G having conflict degree at most d and color lists L such that $|L(v)| \geq d \cdot \deg_G(v) + 1$ for all $v \in V$, in at most $O(\sqrt{\Delta} \log^{1.5} \Delta (\log \Delta + \log^* d))$ rounds.*

Proof. We restrict considerations to the case where Δ is larger than some fixed constant $\Delta' > 0$; otherwise, an appropriate conflict coloring can be obtained in $O(\Delta'^2) = O(1)$ rounds by greedily assigning in each round colors to all vertices of successive independent sets, corresponding to color classes of the given Δ^2-coloring of G.

We will design a conflict-coloring procedure A, which satisfies the assumptions of the lemma. For a graph G, the procedure starts by constructing the β-arbdefective k-coloring $V = V_1 \cup \ldots \cup V_k$ from Lemma IV.1, for a certain parameter k that will be explicitly stated later. Each of the subgraphs $G[V_i]$ now has an edge orientation with maximum outdegree at most $\beta = \frac{\Delta}{k} \log \Delta$, and its vertices are also equipped with locally unique identifiers in the range $\{1, \ldots, \Delta^2\}$ by virtue of the given Δ^2-vertex coloring.

Now, we are ready to solve the conflict-coloring problem on G for a given assignment of lists L such that $|L(v)| \geq d \cdot \deg_G(v) + 1$ for all $v \in V$. Our algorithm proceeds in k stages, obtaining in the i-th stage a valid (final) conflict-coloring of $G[U_i]$ for a specifically defined subset $U_i \subseteq V_1 \ldots \cup V_i$ (we let $U_0 = \emptyset$), i.e., $\mathsf{out}(v) \in L(v)$ and the color pair $(\mathsf{out}(v), \mathsf{out}(u))$ is not forbidden for the edge (v,u), for all $v \in U_i$, $u \in N_{G[U_i]}(v)$. Let $S^v(u, c_u) \subseteq L(v)$ denote the set of colors available to a node v which are in conflict with a color c_u at neighboring node u; we recall that $|S^v(u,c_u)| \leq d$. For $i \geq 1$, given a valid conflict-coloring of $G[U_{i-1}]$ at the beginning of the stage, we create for each $v \in V_i$ a list of colors $L'(v) = L(v) \setminus \bigcup_{u \in U_{i-1} \cap N_G(v)} S^v(u, \mathsf{out}(u))$, which may be used at v to extend the conflict coloring of U_{i-1}. Now, we use Lemma III.3 to perform a conflict coloring, restricted to color lists L', for the oriented subgraph of $G[V_i]$ induced by those vertices $v \in V_i$, for which the assumptions of the Lemma are satisfied (i.e., $|L'(v)| \geq 10d\beta^2 \ln \Delta$). This coloring routine takes $O(\log^* \Delta + \log^* d)$ rounds.

We observe that if a vertex $v \in V_i$ is colored during the phase, then it receives a color $\mathsf{out}(v) \in L'(v) \subseteq L(v)$, which does not conflict with the colors of any of its neighbors in U_{i-1} or simultaneously colored vertices from V_i; we thus construct U_i by adding to U_{i-1} all vertices colored in the current phase.

If, on the other hand, if vertex $v \in V_i$ does not receive a color, then we must have $|L'(v)| < 10d\beta^2 \ln \Delta$. By definition, $L'(v)$ consists of the colors in $L(v)$ which are not in conflict with colors chosen in a previous step. For a previously colored neighbor $u \in U_i$, the color $\mathsf{out}(u)$ is in conflict with at most d colors in $L(v)$. Hence, the number of already colored neighbors is $|N_{G[U_{i-1}]}(v)| \geq (|L(v)| - |L'(v)|)/d > \deg_G(v) - 10\beta^2 \ln \Delta$. In other words, there are at most $10\beta^2 \ln \Delta$ neighbors of v who did not receive a color yet.

Finally, at the end of the k-th stage of the coloring process, we are left with a set $V^* = V \setminus U_k$ of uncolored vertices.

We observe that our conflict-coloring of G can now be completed correctly by conflict-coloring the graph $G^* = G[V^*]$. We define $\Delta^* = 10\beta^2 \ln \Delta$, having $\Delta^* \geq \Delta_{G^*}$. Moreover, we can complete the conflict-coloring of G by merging the so-far obtained coloring out on U_k with the conflict-coloring of G^*, with inherited conflicting color pairs and color lists L^* defined for $v \in V^*$ as:

$$L^*(v) = L(v) \setminus \bigcup_{u \in U_k \cap N_G(v)} S^v(u, \mathsf{out}(u)).$$

Since $|L(v)| \geq d \deg_G(v) + 1$ and $U_k \cap N_G(v) = \deg_G(v) - \deg_{G^*}(v)$, it follows that $|L^*(v)| \geq d \deg_{G^*}(v) + 1$, for all $v \in V^*$. Thus, we may now complete procedure A by recursively applying A to find a list-coloring on G^* with lists L^*, and merge the obtained colorings for U_k and V^*. By assumption, procedure A on G^* must be given a $(\Delta^*)^2$-vertex coloring of G^*, which we can compute using Linial's color reduction mechanism, based on the given Δ^2-coloring of G, in $\log^* \Delta$ rounds. Overall, denoting by $T_A(\Delta)$ an upper bound on the running time of algorithm A on a graph of maximum degree at most Δ, we obtain the following bound:

$$\begin{aligned} T_A(\Delta) \leq O(k \log \Delta) &+ O(k(\log^* \Delta + \log^* d)) \\ &+ O(\log^* \Delta) + T_A(O(\beta^2 \log \Delta)), \end{aligned}$$

where the first component of the sum comes from the routine of Lemma IV.1, the second one is the time of the k stages of coloring graphs $G[V_i]$, the third stage is the time of $(\Delta^*)^2$-coloring graph G^*, and the final stage comes from the recursive application of procedure A. Taking into account that $\beta = O(\frac{\Delta}{k} \log \Delta)$, we obtain:

$$T_A(\Delta) \leq O(k(\log \Delta + \log^* d)) + T_A(O(\tfrac{\Delta^2}{k^2} \log^3 \Delta)).$$

The above expression is minimized for an appropriately chosen (sufficiently large) value $k = O(\sqrt{\Delta \log^3 \Delta})$, for which we eventually obtain $T_A(\Delta) = O(\sqrt{\Delta} \log^{1.5} \Delta(\log \Delta + \log^* d))$. □

We thus obtain the main result of our paper.

Theorem IV.1. *There is a distributed algorithm which solves any conflict-coloring instance on G with conflict degree at most d and color lists L such that $|L(v)| \geq d \deg_G(v) + 1$ for all $v \in V$, in at most $O(\sqrt{\Delta} \log^{1.5} \Delta(\log \Delta + \log^* d)) + \log^* n$ rounds.* □

We remark that, for any conflict coloring problem in which the conflict degree d is constant or bounded by any reasonable function of Δ (i.e., $\log^* d = O(\log \Delta)$), the obtained round complexity simplifies to $O(\sqrt{\Delta} \log^{2.5} \Delta) + \log^* n$. In particular, for the case of $(\Delta + 1)$-list-coloring, we have $d = 1$, giving the following corollary.

Corollary IV.1. *There is a distributed algorithm for the distributed $(\Delta + 1)$-list-coloring problem, performing in $O(\sqrt{\Delta} \log^{2.5} \Delta) + \log^* n$ rounds.*

V. A Centralized Local Algorithm for Conflict-Coloring

In this section, we provide algorithms for solving the conflict coloring problem in the model of centralized local computation. These LCAs are obtained by adapting our distributed algorithms for the LOCAL model to the centralized local model, using the guidelines in [12]. As a special case, we obtain an LCA for $(\Delta + 1)$-coloring algorithm with a smaller probe complexity (in terms of n and Δ) than the best previously known approach. Throughout this section we assume a reasonably small conflict degree for the problem (i.e., $\log^* d = O(\log \Delta)$).

Theorem V.1. *There is a deterministic oblivious LCA for solving an instance of (l, d)-conflict coloring, satisfying the following:*

- *if $l/d \geq 10\Delta^2 \ln \Delta$, then the algorithm performs $\Delta^{O(\log^* \Delta)} \log^* n$ probes per query.*
- *if $l/d > \Delta$, then the algorithm performs $\Delta^{O(\sqrt{\Delta} \log^{2.5} \Delta)} \log^* n$ probes per query.*

Considering list-coloring as a special case of conflict-coloring, we get the following corollary.

Corollary V.1. *There is a deterministic oblivious LCA for list-coloring, which runs in $\Delta^{O(\log^* \Delta)} \log^* n$ probes per query when all color lists are of length at least $10\Delta^2 \ln \Delta$, and in $\Delta^{O(\sqrt{\Delta} \log^{2.5} \Delta)} \log^* n$ probes per query when all color lists are of length at least $\Delta + 1$.*

VI. Conclusion

This paper presents the problem of (l, d)-conflict-coloring in a twofold light. First of all, we show that it is a generalization of numerous symmetry-breaking tasks, which can be solved efficiently in a distributed setting. Secondly, we rely on conflict coloring as a tool to describe intermediate instances of tasks when applying the simplification technique used in our algorithms (cf. Lemma III.1). In view of our results, the deterministic round complexities of $(\Delta+1)$-coloring, $(\Delta+1)$-list-coloring, and (l, d)-conflict-coloring with $l/d > \Delta$, all collapse to $\widetilde{O}(\sqrt{\Delta}) + \log^* n$ rounds. The sufficiently large value of the ratio l/d in the conflict coloring formulation appears to be what sets these problems apart from not easier tasks, such as MIS, for which no approaches with deterministic $o(\Delta) + \log^* n$ runtime are currently known.

We close the paper by remarking briefly on the amount of local computations which individual nodes need to perform to run the proposed algorithms. The most computationally-intensive steps are related to Lemma III.2, which relies on an enumeration of a potentially large set of inputs I to perform a color assignment to each element of the set. The size of this set I, and consequently the complexity of local computations of our algorithms, can be bounded as $2^{\Delta^{O(\log^* \Delta)}}$. This value is polynomially bounded with respect to n for values of $\Delta = (\log n)^{o(1/\log^* n)}$. Since the enumeration of set I is the only bottleneck in our algorithms, there exist several ways of speeding up local computations. For example, one can introduce into the algorithms an element of non-uniformity with respect to maximum degree Δ, and for a given upper bound on Δ, construct the solution in Lemma III.2 through a pre-computed hash function on set I, known to the algorithm, rather than a greedy color selection algorithm. This reduces the local computation time of our algorithms to $\Delta^{O(\log^* \Delta)}$.

References

[1] N. Alon, L. Babai, and A. Itai. A fast and simple randomized parallel algorithm for the maximal independent set problem. *J. Algorithms* **7**(4):567-583, 1986.

[2] B. Awerbuch, A.V. Goldberg, M. Luby, S.A. Plotkin. Network Decomposition and Locality in Distributed Computation. In Proc. *30th Annual Symposium on Foundations of Computer Science* (FOCS), pp. 364-369, 1989.

[3] L. Barenboim. Deterministic $(\Delta + 1)$-coloring in sublinear (in Δ) Time in Static, Dynamic and Faulty Networks. In Proc. *34th ACM Symp. on Principles of Distributed Computing* (PODC), pp. 345-354, 2015.

[4] L. Barenboim and M. Elkin. Sublogarithmic distributed MIS algorithm for sparse graphs using Nash-Williams decomposition. In Proc. *27th ACM Symp. on Principles of Distributed Computing* (PODC), pp. 25-34, 2008.

[5] L. Barenboim and M. Elkin. Distributed $(\Delta + 1)$-coloring in linear (in Δ) time. In Proc. *41th ACM Symp. on Theory of Computing* (STOC), pp. 111-120, 2009.

[6] L. Barenboim and M. Elkin. Deterministic distributed vertex coloring in polylogarithmic time. In Proc. *29th ACM Symp. on Principles of Distributed Computing* (PODC), pp. 410-419, 2010.

[7] L. Barenboim and M. Elkin. *Distributed Graph Coloring: Fundamentals and Recent Developments*. Synthesis Lectures on Distributed Computing Theory, Morgan & Claypool Publishers, 2013.

[8] L. Barenboim, M. Elkin, and F. Kuhn. Distributed $(\Delta + 1)$-coloring in linear (in Δ) time. *SIAM J. Comput.* **43**(1):72-95 (2014)

[9] L. Barenboim, M. Elkin, S. Pettie, and J. Schneider. The locality of distributed symmetry breaking. In Proc. *53rd IEEE Symp. on Foundations of Computer Science* (FOCS), pp. 321-330, 2012.

[10] Y-J. Chang, T. Kopelowitz, S. Pettie, An Exponential Separation Between Randomized and Deterministic Complexity in the LOCAL Model, http://arxiv.org/abs/1602.08166.

[11] P. Erdös, A.L. Rubin, and H. Taylor. Choosability in graphs. In Proc. *West Coast Conf. on Combinatorics, Graph Theory and Computing*, Congres. Num. 26, pp. 125-157, 1979.

[12] G. Even, M. Medina, and D. Ron. Deterministic Stateless Centralized Local Algorithms for Bounded Degree Graphs. Proc. *22nd Annual European Symposium on Algorithms* (ESA) pp. 394-405, 2014.

[13] L. Feuilloley and P. Fraigniaud. Randomized Local Network Computing. In Proc. *27th ACM Symp. on Parallelism in Algorithms and Architectures* (SPAA), pp. 340-349, 2015.

[14] P. Fraigniaud, A. Korman, and D. Peleg. Towards a complexity theory for local distributed computing. *J. ACM* **60**(5):35 (2013)

[15] C. Gavoille, R. Klasing, A. Kosowski, L. Kuszner, and A. Navarra. On the complexity of distributed graph coloring with local minimality constraints. *Networks* **54**(1):12-19 (2009)

[16] N. Garg, M. Papatriantafilou, and P. Tsigas. Distributed list-coloring: how to dynamically allocate frequencies to mobile base stations. In Proc. *8th IEEE Symp. on Parallel and Distributed Processing* (SPDP), pp. 18-25, 1996.

[17] A.V. Goldberg, S.A. Plotkin, G.E. Shannon. Parallel Symmetry-Breaking in Sparse Graphs. *SIAM J. Discrete Math.* **1**(4): 434-446 (1988).

[18] M. Göös, J. Hirvonen, and J. Suomela. Linear-in-Delta lower bounds in the LOCAL model. In Proc. *33rd ACM Symp. on Principles of Distributed Computing* (PODC), pp. 86-95, 2014.

[19] D.G. Harris, J. Schneider, H-H. Su: Distributed $(\Delta + 1)$-Coloring in Sublogarithmic Rounds, In Proc. *48th Annual Symposium on the Theory of Computing* (STOC), 2016, to appear.

[20] J. Hirvonen and J. Suomela. Distributed maximal matching: greedy is optimal. In Proc. *31st ACM Symp. on Principles of Distributed Computing* (PODC), pp. 165-174, 2012.

[21] A. Korman, J.-S. Sereni, and L. Viennot. Toward more localized local algorithms: removing assumptions concerning global knowledge. *Distributed Computing* **26**(5-6): 289-308 (2013)

[22] F. Kuhn. Weak graph colorings: distributed algorithms and applications. In Proc. *21st ACM Symp. on Parallel Algorithms and Architectures* (SPAA), pp. 138-144, 2009.

[23] F. Kuhn, T. Moscibroda, and R. Wattenhofer. What cannot be computed locally! In Proc. *23rd ACM Symp. on Principles of Distributed Computing* (PODC), pp. 300-309, 2004.

[24] F. Kuhn and R. Wattenhofer. On the complexity of distributed graph coloring. In Proc. *25th ACM Symp. on Principles of Distributed Computing* (PODC), pp. 7-15, 2006.

[25] N. Linial. Locality in Distributed Graph Algorithms. *SIAM J. Comput.* **21**(1): 193-201 (1992)

[26] M. Luby. A simple parallel algorithm for the maximal independent set problem. *SIAM J. on Computing* **15**:1036-1053 (1986)

[27] Y. Mansour, A. Rubinstein, S. Vardi, and N. Xie. Converting online algorithms to local computation algorithms. In Proc. *39th Int. Colloq. on Automata, Languages, and Programming* (ICALP), pp. 653-664, 2012.

[28] Y. Mansour and S. Vardi. A local computation approximation scheme to maximum matching. In proc. *16th APPROX- 17th RANDOM*, Springer LNCS 8096, pp. 260-273, 2013.

[29] D. Marx. Graph coloring problems and their applications in scheduling. *Periodica Polytechnica Ser. El. Eng.* **48**(1):11-16 (2004)

[30] M. Naor. A lower bound on probabilistic algorithms for distributive ring coloring. *SIAM J. on Discrete Mathematics* **4**(3):409-412, 1991.

[31] M. Naor and L.J. Stockmeyer. What Can be Computed Locally? *SIAM J. Comput.* **24**(6):1259-1277 (1995)

[32] H. Nguyen and K. Onak. Constant-time approximation algorithms via local improvements. In proc. *49th IEEE Symp. on Foundations of Computer Science* (FOCS), pp. 327-336, 2008.

[33] A. Panconesi and A. Srinivasan. Improved Distributed Algorithms for Coloring and Network Decomposition Problems. In Proc. *24th ACM Symp. on Theory of Computing* (STOC), pp. 581-592, 1992.

[34] M. Parnas and D. Ron. Approximating the minimum vertex cover in sublinear time and a connection to distributed algorithms. *Theor. Comp. Sci.* **381**(1-3):183-196 (2007)

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

[36] F. R. Roberts. T-colorings of graphs: recent results and open problems. *Discrete Mathematics* **93**:229-245 (1991)

[37] R. Rubinfeld, G. Tamir, S. Vardi, and N. Xie. Fast local computation algorithms. In Proc. *Innovations in Computer Science* (ICS), pp. 223-238, 2011.

[38] J. Suomela. Survey of local algorithms. *ACM Comput. Surv.* **45**(2):24 (2013)

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

[40] V. G. Vizing. Coloring the vertices of a graph in prescribed colors. *Diskret. Analiz.* **29**:3-10 (1976)

[41] W. Wang and X. Liu. List-coloring based channel allocation for open-spectrum wireless networks. In Proc. *62nd IEEE Vehicular Technology Conf.* (VTC), pp. 690-694, 2005.

[42] T. Zeitlhofer and B. Wess. List-coloring of interval graphs with application to register assignment for heterogeneous register-set architectures. *Signal Processing* **83**(7): 1411-1425 (2003)