

Perfectly Sampling $k \geq (8/3 + o(1))\Delta$ -Colorings in Graphs

Vishesh Jain
Stanford University
vishesh.vj@gmail.com

Ashwin Sah
Massachusetts Institute of Technology
asah@mit.edu

Mehtaab Sawhney
Massachusetts Institute of Technology
msawhney@mit.edu

Abstract

We present a randomized algorithm which takes as input an undirected graph G on n vertices with maximum degree Δ , and a number of colors $k \geq (8/3 + o_\Delta(1))\Delta$, and returns – in expected time $\tilde{O}(n\Delta^2 \log k)$ – a proper k -coloring of G distributed *perfectly* uniformly on the set of all proper k -colorings of G . Notably, our sampler breaks the barrier at $k = 3\Delta$ encountered in recent work of Bhandari and Chakraborty [STOC 2020]. We also sketch how to modify our methods to relax the restriction on k to $k \geq (8/3 - \epsilon_0)\Delta$ for an absolute constant $\epsilon_0 > 0$.

As in the work of Bhandari and Chakraborty, and the pioneering work of Huber [STOC 1998], our sampler is based on Coupling from the Past [Propp&Wilson, Random Struct. Algorithms, 1995] and the bounding chain method [Huber, STOC 1998; Häggström& Nelander, Scand. J. Statist., 1999]. Our innovations include a novel bounding chain routine inspired by Jerrum’s analysis of the Glauber dynamics [Random Struct. Algorithms, 1995], as well as a preconditioning routine for bounding chains which uses the algorithmic Lovász Local Lemma [Moser&Tardos, J.ACM, 2010].

1 Introduction

Let $G = (V(G), E(G))$ be an undirected graph with vertex set $V(G)$ and edge set $E(G)$. For an integer $k \geq 1$, a (*proper*) k -coloring of G is a map $\varphi : V(G) \rightarrow [k] := \{1, \dots, k\}$ such that for all $\{u, v\} \in E(G)$, $\varphi(u) \neq \varphi(v)$. In this paper, we study the problem of efficiently *perfectly* sampling a k -coloring of a graph with maximum degree Δ , uniformly at random from among all such colorings.

1.1 Sampling k -colorings: approximately and perfectly

The algorithmic problem of sampling a uniformly random k -coloring of a graph with maximum degree Δ has been intensely studied (see, e.g., the references in [5, 4]). Perhaps the major open problem in this area is to devise – for all $k \geq \Delta + 2$ – a (randomized) algorithm, with running time polynomial in $n = |V(G)|$, k , and $\ln(1/\epsilon)$, which outputs a distribution within total variation distance ϵ of the uniform distribution on the space of k -colorings; the lower bound corresponds to the minimum number of colors needed to ensure that the space of k -colorings is connected in a certain sense, and is within 1 color of the classical theorem of Brooks which asserts that $\Delta + 1$ colors are sufficient to color any graph of maximum degree Δ (and necessary for cliques and cycles of odd length).

Recall that the Glauber dynamics on the space of k -colorings is the Markov chain which, at a coloring χ , chooses a vertex uniformly at random from $V(G)$, and updates its color to be a color chosen uniformly at random from among those not already occupied by its neighbors; it is readily seen that this Markov chain is ergodic for $k \geq \Delta + 2$, and has the uniform distribution on the space of k -colorings as its stationary distribution. In a seminal work, Jerrum [9] showed that the Glauber dynamics mixes rapidly for $k > 2\Delta$, thereby providing an efficient algorithm for approximately sampling k -colorings for all $k > 2\Delta$. The lower bound on k was relaxed by Vigoda [15] to $11\Delta/6$ by using a different Markov chain based on ‘flip dynamics’, although by using comparison techniques, his proof also implies rapid mixing of the Glauber dynamics for $k > 11\Delta/6$. Recently, Chen, Delcourt, Moitra, Perarnau, and Postle [4] sharpened Vigoda’s analysis to further relax the lower bound to $k > (11/6 - \epsilon_0)\Delta$, where ϵ_0 is a small absolute constant ($\sim 10^{-4}$). Under additional assumptions on the degree and girth of G , even less restrictive lower bounds on k are known (see, e.g., the references in [5, 4, 2]).

The problem of efficiently (i.e. polynomial in n and k) perfectly sampling k -colorings, which is the focus of this paper, was first studied by Huber [8], who used Coupling from the Past (CFTP) [14] along with the bounding chain method [7, 8] to devise an efficient algorithm for perfectly sampling k -colorings, provided that $k > \Delta(\Delta + 2)$. One of the motivations of Huber’s work was that using perfect sampling algorithms in the general sampling-to-counting framework of Jerrum, Valiant, and Vazirani [10] can potentially be used to obtain faster algorithms for the problem of *approximately counting* the number of k -colorings of a graph, than can be obtained from approximate sampling algorithms [8, Theorem 7]. Another motivation for his work was that – in contrast to the approximate sampling algorithms discussed above, which always need to be run for the theoretical worst-case time in order to output a distribution guaranteed to be close to the uniform distribution – perfect sampling algorithms based on CFTP have the attractive property of coming with a well-defined termination criterion, which may be reached in practice well before the time suggested by worst-case analysis (in fact, for implementing such an algorithm, the practitioner need not have *any* knowledge of the worst-case running time).

In recent years, by exploiting the connection, due to Jerrum, Valiant, and Vazirani [10, Theorem 3.3], between *deterministic* approximate counting and perfect sampling, several improvements

of Huber’s result, which are efficient for graphs of *constant* maximum degree, have been obtained. Using the correlation decay technique, Gamarnik and Katz [6], respectively Lu and Yin [12], obtained perfect samplers with $k > 2.78\Delta$ for triangle free graphs, respectively $k > 2.58\Delta$ for general bounded degree graphs, both running in time $O(n^{O(\log k)})$; using Barvinok’s polynomial interpolation method, Liu, Sinclair, and Srivastava [11] provided a perfect sampler for $k \geq 2\Delta$ running in time $O(n^{\exp(\text{poly}(k))})$.

For general graphs, where Δ (or k) is allowed to grow with n , the only improvement of Huber’s result is the recent work of Bhandari and Chakraborty [2], who provided an efficient CFTP based perfect sampler for $k > 3\Delta$. The natural question left open by their work is whether one can devise efficient perfect samplers for $k < 3\Delta$ – indeed, the sampler in [2] may be viewed as implementing a two-stage process, with natural barriers at $k = 3\Delta$ encountered (for different reasons) at both the stages (see Section 3.3 for a quick overview and [2, Section 3] for a more detailed explanation).

1.2 Our result

As our main result, we obtain a CFTP based perfect sampler for $k > (8/3 + o(1))\Delta$. Pleasantly, our sampler has the same expected running time as in [2].

Theorem 1.1. *There is a (CFTP-based) randomized algorithm PERFECTSAMPLER (Algorithm 1) and an absolute constant $C_{1.1} > 0$ such that the following holds. Given an undirected graph G with maximum degree Δ , and a number of colors k with $k \geq 8\Delta/3 + C_{1.1}\sqrt{\Delta \log \Delta}$, PERFECTSAMPLER returns a uniformly random k -coloring of G , and runs in expected time $O(T_1 + T_2 + T_3)$, where $T_1, T_2, T_3 = O(n(\log n)^2 \Delta^2 (\log \Delta) (\log k))$.*

Remark. The proof shows that for Δ sufficiently large, taking $C_{1.1} = 2$ is sufficient. Moreover, in Section 6, we briefly indicate how our sampler may be modified to obtain a version of Theorem 1.1 for $k \geq (8/3 - \epsilon)\Delta$, where $\epsilon \approx 10^{-2}$ is an absolute constant. We decided not to pursue this improvement since (i) the details are a bit more technical and all the main ideas are already present in our current analysis, (ii) the improvement is relatively minor, and can anyway not reach $k > 5\Delta/2$, which we believe is a natural barrier for our methods (see Section 6 for a discussion of this).

1.3 Organization

The rest of this paper is organized as follows. In Section 2, we provide an introduction to coupling from the past, and the bounding chain method. In particular, the standard Lemma 2.1 reduces the proof of Theorem 1.1 to the construction of a certain procedure which we call SAMPLERUNIT. In Section 3, we provide an overview of this procedure – Section 3.1 contains some notation used throughout the paper, Section 3.2 contains a preliminary routine used by our algorithm, whose proof is presented in Appendix A, Section 3.3 provides a quick introduction to the sampler in [2], Section 3.4 provides a description of SAMPLERUNIT, modulo the details of some primitive routines, and finally, Section 3.5 provides a high-level discussion of the key ideas underpinning the construction and analysis of our sampler. Section 4 presents and analyses our main primitives – COMPRESS, SEEDING, and DISJOINT, Section 5 completes the analysis of SAMPLERUPDATE, and Section 6 concludes with some final remarks (including a brief sketch of how to relax the lower bound in Theorem 1.1 to $(8/3 - \epsilon_0)\Delta$) and directions for future research.

2 Coupling from the past and bounding chains

2.1 Coupling from the past

As in [8, 2], our perfect sampler is based on coupling from the past (CFTP), which is a general procedure due to Propp and Wilson [14] for sampling exactly from the stationary distribution of a Markov chain. The basic idea behind CFTP is that for an ergodic Markov chain started at time $-\infty$, its location at time 0 should be distributed according to the stationary distribution; hence, if we could determine the location at time 0 by only looking at the randomness generating the chain in the recent past, then we would have an efficient way of obtaining a sample from the stationary distribution of the chain.

Implementing this idea algorithmically for an ergodic Markov chain on a finite state space Ω typically amounts to the following: for $i = 1, 2, \dots, T$, we generate independent random maps $f_{-i} : \Omega \rightarrow \Omega$ with the property that if $\omega \in \Omega$ is distributed according to the stationary distribution, then $f_{-i}(\omega)$ is also distributed according to the stationary distribution. If it so happens that the composite function

$$F_{-1,-T} := f_{-1} \circ \dots \circ f_{-T}$$

is constant on Ω , then we are guaranteed that $F_{-1,-T}(\omega_0)$ (for any $\omega_0 \in \Omega$; note that the image does not depend on the choice of ω_0) is a sample from the stationary distribution. If $F_{-1,-T}$ is not constant, then we can consider $F_{-1,-T} \circ F_{-T-1,-2T}$ (by independently generating f_{-T-1}, \dots, f_{-2T}), and so on. More formally, [Theorem 1.1](#) follows from the following standard lemma, once we have constructed a suitable randomized algorithm `SAMPLERUNIT` and predicate Φ .

Lemma 2.1. *Let G be an undirected graph on n vertices with maximum degree Δ , let $k \geq 8\Delta/3 + C_{1.1}\sqrt{\Delta \log \Delta}$, and let Ω denote the set of k -colorings of G . Suppose there is a randomized algorithm `SAMPLERUNIT` for generating a distribution \mathcal{D} on functions $F : \Omega \rightarrow \Omega$, and a predicate $\Phi : \text{Supp}(\mathcal{D}) \rightarrow \{\text{TRUE}, \text{FALSE}\}$ with the following properties:*

- (P1) *If χ is uniformly distributed in Ω , and F is generated according to \mathcal{D} independently of χ , then $F(\chi)$ is also uniformly distributed in Ω .*
- (P2) *If $\Phi(F) = \text{TRUE}$, then F is constant on Ω .*
- (P3) $\mathbb{P}_{F \sim \mathcal{D}}[\Phi(F) = \text{TRUE}] \geq 1/2$.
- (P4) *`SAMPLERUNIT` runs in time T_1 , $\Phi(F)$ can be computed in time T_2 , and $F(\chi)$ can be computed in time T_3 .*

Then, the randomized algorithm `PERFECTSAMPLER` terminates in expected time $O(T_1 + T_2 + T_3)$ and returns a uniformly distributed k -coloring of G .

Proof. Let F_{-1}, F_{-2}, \dots be the i.i.d. samples from \mathcal{D} generated by `PERFECTSAMPLER`. Let χ be an independent and uniformly distributed k -coloring, let $\chi_i = F_{-1} \circ \dots \circ F_{-i}(\chi)$, and let χ^* be the output of the algorithm. By (P1), it follows that for all $i \geq 1$, χ_i is also a uniformly distributed k -coloring. Moreover, by (P2, P3), $\chi_i = \chi^*$ with probability at least $1 - 2^{-i}$ (since this happens whenever $\bigvee_{j=1}^i \Phi(F_{-j}) = \text{TRUE}$). In particular, by the coupling characterization of total variation distance, χ^* is within total variation distance 2^{-i} of the uniform distribution on the space of k -colorings. Finally, since $i \geq 1$ is arbitrary, it follows that χ^* is actually itself uniformly distributed. The claim about the running time follows easily by noting that the outer loop is executed at most 2 times in expectation. \square

Algorithm 1: PERFECTSAMPLER – Takes an input procedure SAMPLERUNIT and converts procedure into a perfect sampler.

```

1 Compute seeding set  $\mathcal{S}$  (as in Proposition 3.1)
2 for  $i = 1, 2, \dots$  do
3   Generate  $F_{-i}$  according to SAMPLERUNIT
4   if  $\Phi(F_{-1} \circ \dots \circ F_{-i}) = \text{TRUE}$  then
5     Output unique coloring in the image of  $F_{-1} \circ \dots \circ F_{-i}$  and TERMINATE

```

The main challenge in CFTP based algorithms is efficiently determining whether $\Phi(F) = \text{TRUE}$. *A priori*, this requires evaluating F for every $\omega \in \Omega$, which is infeasible if $|\Omega|$ is very large. However, in certain contexts where the domain Ω is equipped with a natural partial order compatible with the Markov chain, considerations of monotonicity or anti-monotonicity can reduce this task to evaluating F on only a small number of ‘extremal’ elements (see, e.g., [14, 7] for examples). Unfortunately, in our case, where Ω is the space of k -colorings, $|\Omega|$ is too large (potentially k^n) to permit direct evaluation of F , and moreover, there doesn’t seem to be any natural notion of (anti)monotonicity compatible with various Markov chains on the space of colorings.

2.2 Bounding chains

To overcome this issue, Huber [8] and independently Häggström and Nelander [7] introduced the method of bounding chains. The way this method is implemented in the case of k -colorings is the following: while evaluating $\Phi(F)$, where F is the composite function $F_{-1,-T}$ as in the previous subsection, instead of precisely keeping track of the intermediate images $f_{-j} \circ \dots \circ f_{-T}$, we instead maintain a set $L_{-j+1}(v)$ of colors for each vertex $v \in V(G)$ with the property that for all $j \in [T]$, the image of Ω under $f_{-j} \circ \dots \circ f_{-T}$ is contained in $L_{-j+1}(v_1) \times \dots \times L_{-j+1}(v_n)$. Then, if we can show that $|L_0(v)| = 1$ for all $v \in V(G)$, we will be done. The idea here is that the (product) space of sets of available colors at each vertex, while cruder, is more amenable to the design of CFTP algorithms (for instance, note that there is a natural partial order on this space induced by set-theoretic inclusion of the set of available colors at each vertex). The perfect samplers in [8, 2] are both based on CFTP and the bounding chain method. Our improvement stems from a novel implementation of this method (see Section 3.5 for a discussion of the key ideas); in particular, among other things, we find a way of lifting Jerrum’s analysis [9] of the rapid mixing of Glauber dynamics to bounding chains (Section 3.5.1).

3 Overview of SAMPLERUNIT

3.1 Notation

Throughout, G will be an undirected graph on n vertices with maximum degree Δ . A *bounding list* is a list $L = (L(v) : v \in V(G))$, where each $L(v)$ is a subset of colors in $[k]$. We will often refer to $L(v)$ as the *bounding set* of the vertex v . Given a vertex $v \in V(G)$, we let

$$S_L(v) = \bigcup_{w \in N(v)} L(w), \quad Q_L(v) = \bigcup_{\substack{w \in N(v) \\ |L(w)|=1}} L(w).$$

Here, as is standard, $N(v)$ denotes the neighborhood of a vertex v . A key quantity in our algorithm is the set

$$N_L^*(v) = \{w \in N(v) : |L(w)| = 2 \text{ and } L(w) \cap L(w') = \emptyset \text{ if } w' \in N(w), w' \neq w\},$$

and the set of *disjoint-pair colors* associated to v , defined by

$$D_L(v) = \bigcup_{w \in N_L^*(v)} L(w).$$

Finally, let

$$E_L(v) = S_L(v) \setminus (Q_L(v) \cup D_L(v)).$$

We will reserve the symbols χ, χ' for k -colorings, and say that χ is compatible with L , denoted by $\chi \sim L$, if $\chi(v) \in L(v)$ for all v . As in [2], we will associate update operations with tuples – specifically, we will use 6-tuples of the form

$$\alpha = (v, \tau, L, L', M, \gamma),$$

where $v \in V(G)$, $\tau \in [0, 1]$, L, L' are bounding lists, M is a sequence of at most $\Delta + 1$ distinct colors from $[k]$, and $\gamma \in [3]$ specifies the ‘type’ of the update. We will denote the update operation (i.e. the map from the space of proper colorings to itself) associated to the tuple α by f_α ; in particular, the sequence of random functions $f_{-1}, \dots, f_{-T}, \dots$ discussed in Section 2.1 will be specified by the sequence of random updates $\alpha_{-1}, \dots, \alpha_{-T}, \dots$.

As in [2], SAMPLERUNIT will consist of a sequence of T updates satisfying the following three key properties. Fix $t \in \{-T, \dots, -1\}$. First, the random vertex v_t is independent of $\alpha_{-T}, \dots, \alpha_{-t-1}$. Second, f_{α_t} implements the Glauber dynamics at v_t i.e. for any coloring χ , $f_{\alpha_t}(\chi)(w) = \chi(w)$ for all $w \neq v_t$ and $f_{\alpha_t}(\chi)(v_t)$ is uniformly distributed in $[k] \setminus \chi(N(v_t))$. Third, if $\chi \sim L_t$, then $f_{\alpha_t}(\chi) \sim L'_t$.

3.2 Finding a seeding set

The very first step of PERFECTSAMPLER consists of efficiently finding a set \mathcal{S} of *seeded vertices*, as defined in the following proposition. In fact, if we need to generate multiple samples, we can perform this step only once at the start, and use the same \mathcal{S} for all calls to PERFECTSAMPLER.

Proposition 3.1. *Fix $\eta \in (0, 1/3)$ and let $\Delta \geq C_\eta$. There is a set of vertices $\mathcal{S} \subseteq V(G)$ such that any $v \in V(G)$ satisfies*

$$|N(v) \cap \mathcal{S}^c| \leq (1 - \eta)\Delta, \quad |N(v) \cap \mathcal{S}| \leq \Delta/3. \quad (3.1)$$

Furthermore, there is a randomized algorithm that finds such a set with probability at least $1/2$ and runs in time $O(n\Delta + n \log n)$.

Remark. In fact, we can let C_η be an absolute constant for $\eta = 1/3 - 2\sqrt{(\log \Delta)/\Delta}$.

Proposition 3.1 is purely a statement about probabilistic combinatorics, and has nothing to do with graph colorings. Its proof is based on a standard application of the algorithmic Lovász Local Lemma due to Moser and Tardos [13], and is included in Appendix A for completeness.

3.3 Outline of the Bhandari-Chakraborty construction

Before presenting our construction of SAMPLERUNIT, it is instructive to briefly review the salient features of the corresponding construction in [2]; we refer the reader to [2, Section 1.2] for a more detailed overview. Recall that the goal of the block of T updates in SAMPLERUNIT is to ensure that, with probability at least $1/2$, the bounding list at the most recent time consists of sets of size 1. In [2], this is accomplished in two phases – the first phase (referred to as ‘collapsing’) serves to ensure that all bounding sets are of size at most 2, whereas the second phase (referred to as ‘coalescing’) makes all bounding sets of size 1 with probability at least $1/2$.

The two phases are themselves based on two types of updates, called COMPRESS and CONTRACT. The coalescing phase consists of a predetermined number of applications of CONTRACT at uniformly randomly chosen vertices. Whenever CONTRACT is applied at a vertex v , it results in the bounding set at v contracting to size at most 2, and with some probability, to size 1. However, to apply CONTRACT at v , one needs the promise that $|S_L(v)| < k - \Delta$; since bounding sets of size 2 (which is the only guarantee we have at the end of the collapsing phase) can in general lead to $|S_L(v)| = 2\Delta$, this is one source of the restriction $k > 3\Delta$. Also, while CONTRACT leads to bounding sets of size at most 2, it may very well happen that applying CONTRACT to a vertex which already has bounding set of size 1 leads to a larger bounding set of size 2. During the coalescing phase, in order for the repeated random applications of CONTRACT to lead to a ‘drift’ towards all bounding sets having size 1, one also needs the condition that $k > \Delta + \Delta \cdot \max_{v \in V(G)} |L(v)|$, which again leads to the restriction $k > 3\Delta$.

In contrast to the coalescing phase, where the vertices are chosen uniformly at random, the vertices chosen to update in the collapsing phase are predetermined (this is one of the chief innovations of [2]). Indeed, for an arbitrary ordering v_1, \dots, v_n of the vertices, the collapsing phase can be concisely represented as

$$\text{SPRUCEUP}(v_1), \text{CONTRACT}(v_1), \dots, \text{SPRUCEUP}(v_n), \text{CONTRACT}(v_n),$$

where the job of SPRUCEUP(v_i) is to ensure that the condition $|N_L(v_i)| < k - \Delta$, needed to apply CONTRACT(v_i), is satisfied. Finally, SPRUCEUP(v_i) is performed as follows: first, we pick an *arbitrary* set A of size Δ which non-trivially intersects the bounding sets of all neighbors of v_i preceding v_i (in the fixed ordering of vertices). Next, to each neighbor of v_i succeeding it in the ordering, we apply COMPRESS with input A – this has the effect of changing the bounding sets at these vertices to be the union of A and a color outside of A . Note that once SPRUCEUP(v_i) is completed, we indeed have $|N_L(v_i)| \leq 2\Delta$, since the vertices preceding v_i have already been contracted (and hence, can contribute at most one color outside of A) whereas the vertices succeeding v_i can also contribute at most one color outside of A (by definition of COMPRESS).

3.4 Our construction of SAMPLERUNIT

We are now ready to present our construction, which is based on three kinds of updates – COMPRESS (Algorithm 2), SEEDING (Algorithm 3), and DISJOINT (Algorithm 4). Throughout this subsection, let $\eta = 1/3 - 2\sqrt{(\log \Delta)/\Delta}$, let $k > (3 - \eta)\Delta$, and let \mathcal{S} be the set of vertices coming from Proposition 3.1 applied with η . For the sake of simplicity, let $s = |\mathcal{S}|$. Also, throughout the rest of this paper, we assume that $\Delta \geq C$, for some sufficiently large absolute constant C . This may be done without loss of generality since, in Theorem 1.1, the constant $C_{1.1}$ may be taken sufficiently large so that for $\Delta \leq C$, the lower bound on k is $k > 3\Delta$, at which point one may use either the sampler in [2], or indeed our sampler with a slightly more careful analysis.

We construct SAMPLERUNIT in the following four phases:

- Phase 1 (Seeding step): Arbitrarily order the vertices in \mathcal{S} as v_1, \dots, v_s . For $1 \leq i \leq s$, perform COMPRESS on all neighbors of v_i that are not in $\{v_1, \dots, v_{i-1}\}$ with associated set A being an arbitrary set of size Δ *completely containing* $L(w)$ for each $w \in N(v_i) \cap \{v_j : j < i\}$. Then, perform SEEDING on v_i and increment i by 1 (if $i < s$) or move to Phase 2 (if $i = s$). Note that at the end of this phase, all vertices in \mathcal{S} have bounding set of size at most 3.
- Phase 2 (Converting seeded vertices to size 2): For each $1 \leq i \leq s$, apply COMPRESS to all neighbors of v_i not in \mathcal{S} , with associated set A being an arbitrary set of size Δ *completely containing* $N(w)$ for all $w \in N(v_i) \cap \mathcal{S}$. Then, apply DISJOINT to v_i . Note that at the end of this phase, all vertices in \mathcal{S} have bounding set of size at most 2.
- Phase 3 (Converting remaining vertices to size 2): Mark all vertices in \mathcal{S} . Arbitrarily order the vertices in $V(G) \setminus \mathcal{S}$ as v_{s+1}, \dots, v_n . For $s+1 \leq i \leq n$ perform the following sequence of operations. Apply COMPRESS to all unmarked neighbors of v_i with associated set A of size Δ determined as follows: let L_m be the current bounding list, restricted to marked neighbors of v . We greedily take elements from $Q_{L_m}(v) \cup E_{L_m}(v)$ first, then (if the set constructed at this point has size less than Δ colors) elements from $D_{L_m}(v)$ (chosen in pairs $L_m(w)$ for $w \in N_{L_m}^*(v)$), and then (if we still do not have Δ colors) arbitrarily from the remaining colors. Then, apply DISJOINT to v_i , mark v_i , and increment i by 1 (if $i < n$) or move to Phase 4 (if $i = n$). Note that at the end of this phase, all vertices have bounding set of size at most 2.
- Phase 4 (Drifting to size 1): For $T_D = 2(k - \Delta)n \log n / (k - 5\Delta/2)$ steps, apply DISJOINT on a uniformly random vertex in the graph.

In [Section 5](#), we show how SAMPLERUNIT can be used to generate a distribution \mathcal{D} and a predicate Φ satisfying (P1)-(P4) in [Lemma 2.1](#) with T_1, T_2, T_3 as in [Theorem 1.1](#).

3.5 Key ideas

In this subsection, we provide an informal and high-level discussion of some of the key ideas underlying our construction and analysis of SAMPLERUNIT.

3.5.1 Lifting Jerrum's analysis of Glauber dynamics using $D_L(v)$

The main idea which enables us to bypass the obstacle at 3Δ encountered in the coalescing phase is a bounding list version of Jerrum's analysis in [\[9\]](#). Recall that in the standard (path) coupling argument proving rapid mixing of Glauber dynamics for $k > 3\Delta$, one couples two chains by generating a uniformly random pair $(v, c) \in V(G) \times [k]$, and updating the color at v to c whenever possible. Suppose we have two colorings χ and χ' differing only at a single vertex v_0 , with $\chi(v_0) = c_0$ and $\chi'(v_0) = c'_0$. Then, under this coupling, the distance between χ and χ' decreases by 1 iff for the random pair (v, c) , $v = v_0$ and c is one of the at most $k - \Delta$ colors not appearing in $|\chi(N(v_0))|$ (note that $\chi(N(v_0)) = \chi'(N(v_0))$). Also, the distance between χ and χ' increases by 1 iff for the random pair (v, c) , $v \in N(v_0)$ and $c \in \{c_0, c'_0\}$. Since there are at least $k - \Delta$ pairs which decrease the distance by 1, and at most 2Δ pairs which increase the distance by 1, we see (at least intuitively) that the distance drifts towards 0 if $k - \Delta > 2\Delta$ i.e. $k > 3\Delta$. Jerrum improved the lower bound to 2Δ by slightly modifying this coupling so that whenever c_0 (respectively c'_0) is selected by for χ , c'_0 (respectively c_0) is selected for χ' ; it is immediate that this halves the number of 'bad' pairs (v, c) , and leads to the weaker restriction $k - \Delta > \Delta$ i.e. $k > 2\Delta$.

In our algorithm, we perform a similar coupling of the colors in $D_L(v)$ (which are naturally paired up by definition) – this appears as one of the cases in the update DISJOINT. However, in order to obtain any improvement via such a coupling, we need $|D_L(v)|$ to constitute a non-trivial fraction of $|N_L(v)|$, which need not be the case (note that no such difficulty arises in Jerrum’s work). We overcome this issue using a win-win analysis based on a robust version (see (5.2)) of the extremal combinatorial fact that if $|S_L(v)| > 3\Delta/2$, then $|D_L(v)| \neq 0$.

While this idea takes care of the barrier at 3Δ *provided* the bounding list consists of sets of size at most 2, getting to this stage presents a different obstacle owing to the fact that the update CONTRACT in [2] requires $k > 3\Delta$ in the worst case to satisfy its promise. We circumvent this issue by using a combination of several ideas.

3.5.2 Preconditioning via SEEDING

In contrast to [2, 8], we make much greater use of the structure of the underlying graph G by first identifying a set \mathcal{S} of size $\approx n/3$ such that each vertex has no more than $\approx 2\Delta/3$ neighbors outside \mathcal{S} and no more than $\Delta/3$ neighbors inside \mathcal{S} . Phase 1 ensures that all vertices in \mathcal{S} have bounding sets of size at most 3 – in order to accomplish this, we introduce a new update called SEEDING which requires a weaker promise than CONTRACT, but comes at the cost of the bounding set being of size at most 3 (as opposed to 2). Specifically, SEEDING requires the guarantee that $k - \Delta \geq |S_L(v)|^2 / (\Delta + |S_L(v)|)$; when $|S_L(v)| \leq 2\Delta$ (as can be guaranteed by applying COMPRESS updates as in [2]), the right hand side is at most $4\Delta/3$, so that the restriction on k is only $k > 7\Delta/3$.

3.5.3 Substantially exploiting the flexibility in the choice of A

In [2], the set A used for COMPRESS updates to ‘spruce-up’ the neighborhood of v is always chosen to be simply a set of size Δ intersecting the bounding set of each neighbor of v preceding it in the order, and has no additional properties. In contrast, our construction of A is much more careful, and in fact, varies across phases to account for the different nature of the challenges encountered. In particular, when ‘sprucing-up’ a vertex $v \in \mathcal{S}$ in Phase 2, we take A to be a set of size Δ containing *all* the colors appearing in any bounding set of $N(v) \cap \mathcal{S}$ – by Phase 1 and the definition of \mathcal{S} , this is always possible. Then, note that applying COMPRESS to the at most $\approx 2\Delta/3$ neighbors of v not in \mathcal{S} can contribute at most one additional color each, so that after this sprucing-up procedure, $|S_L(v)| \leq \Delta + \approx 2\Delta/3 = \approx 5\Delta/3$. At this point, we could use the CONTRACT update from [2] to convert the bounding set of v to size at most 2, but for a streamlined treatment, we use our more refined DISJOINT update.

3.5.4 Refining CONTRACT by tracking $D_L(v)$

Phase 3 of our algorithm, whose analysis is the most involved, combines the previous idea of exploiting the flexibility in the choice of A with a variation of CONTRACT, called DISJOINT, which implements the idea in Section 3.5.1. Notably, as compared to CONTRACT, which requires the promise $k - \Delta > |S_L(v)|$, DISJOINT requires the more refined promise

$$|S_L(v)| - |Q_L(v)| < (k - \Delta) \left(\frac{k - |Q_L(v)|}{k - |Q_L(v)| - |D_L(v)|/2} \right);$$

note that when $|Q_L(v)| = 0 = |D_L(v)|$, this reduces to the promise required by CONTRACT. Once the desired properties of the DISJOINT update have been established (Lemma 4.3), the analysis of

Phase 3 boils down to checking that the promise required by DISJOINT is always satisfied; we show that by using a more intricate procedure for selecting the set A , this refined promise can be satisfied with around $8\Delta/3$ colors.

4 COMPRESS, SEEDING, and DISJOINT

In this section, we present and analyse our three main updates – COMPRESS, SEEDING, and DISJOINT. In each case, we explain how the update is generated, how it interacts with the bounding list, and how one can apply the resulting random function to colorings (i.e. ‘decode the update’) in order to simulate the Glauber dynamics at the appropriate vertex.

4.1 COMPRESS

The first update, COMPRESS, is exactly the same as in [2], which in turn builds on ideas in [8]; we sketch the analysis, as it serves as a warm-up for the analysis of our other updates. We define COMPRESS in Algorithm 2, and summarize its important properties in the following lemma.

Algorithm 2: COMPRESS – Takes an input update $\alpha_{\text{IN}} = (v_{\text{IN}}, \tau_{\text{IN}}, L_{\text{IN}}, L'_{\text{IN}}, M_{\text{IN}}, \gamma_{\text{IN}})$, a vertex v , and a set A of size Δ , and outputs a compatible “compressed update”.

1 **Function** COMPRESS.GEN:

Input : $\alpha_{\text{IN}} = (v_{\text{IN}}, \tau_{\text{IN}}, L_{\text{IN}}, L'_{\text{IN}}, M_{\text{IN}}, \gamma_{\text{IN}})$, $v \in V(G)$ and $A \subseteq [k]$ with $|A| = \Delta$

Output: $\alpha_{\text{F}} = (v_{\text{F}}, \tau_{\text{F}}, L_{\text{F}}, L'_{\text{F}}, M_{\text{F}}, \gamma_{\text{F}})$

2 $\gamma_{\text{F}} \leftarrow 1$; $v_{\text{F}} \leftarrow v$; $L_{\text{F}} \leftarrow L_{\text{IN}}$; $L'_{\text{F}} \leftarrow L'_{\text{IN}}$; $\tau_{\text{F}} \leftarrow \text{UNIF}[0, 1]$;

3 $c_1 \leftarrow \text{UNIF}([k] \setminus A)$; $L'_{\text{F}}(v) \leftarrow A \cup \{c_1\}$;

4 $M_{\text{F}} \leftarrow \text{UNIFPERMUTATION}(A)$; $M_{\text{F}} \leftarrow (M_{\text{F}}, c_1)$;

5 $\alpha_{\text{F}} \leftarrow (v_{\text{F}}, \tau_{\text{F}}, L_{\text{F}}, L'_{\text{F}}, M_{\text{F}}, \gamma_{\text{F}})$;

6 **Function** COMPRESS.DECODE:

Input : $\alpha = (v, \tau, L, L', M, \gamma)$ with $\gamma = 1$ and a coloring $\chi \sim L$

Output: $\chi' \sim L'$

7 $\chi' \leftarrow \chi$;

8 $p_{\chi}(v) \leftarrow \frac{k-\Delta}{k-|\chi(N(v))|}$;

9 $c_1 \leftarrow M[\Delta + 1]$

▷ Since $\gamma = 1$, M has length $\Delta + 1$

10 **if** $c_1 \notin \chi(N(v))$ and $\tau \leq p_{\chi}(v)$ **then**

11 $\chi'(v) \leftarrow c_1$;

12 **else**

13 $M' \leftarrow M[1, \Delta] \setminus \chi(N(v))$;

14 $\chi'(v) \leftarrow M'[1]$;

▷ Exists if $c_1 \in \chi(N(v))$

Lemma 4.1 ([2, Lemma 2.1]). *Let $k \geq \Delta + 1$. Let $\alpha_{\text{IN}} = (v_{\text{IN}}, \tau_{\text{IN}}, L_{\text{IN}}, L'_{\text{IN}}, M_{\text{IN}}, \gamma_{\text{IN}})$, and choose $v \in V(G)$ and $A \subseteq [k]$ with $|A| = \Delta$. Let $\alpha_{\text{F}} = (v_{\text{F}}, \tau_{\text{F}}, L_{\text{F}}, L'_{\text{F}}, M_{\text{F}}, \gamma_{\text{F}})$ be the output of COMPRESS.GEN $[\alpha_{\text{IN}}, v, A]$. Let χ be a coloring, and let $\chi' = \text{COMPRESS.DECODE}[\alpha_{\text{F}}, \chi]$. Then:*

(C1) $L_{\text{F}} = L'_{\text{IN}}$, $L'_{\text{F}}(u) = L_{\text{F}}(u)$ for $u \neq v$, and $L'_{\text{F}}(v) = A \cup \{c_1\}$ for some $c_1 \in [k] \setminus A$.

(C2) If $\chi \sim L_{\text{F}}$, then $\chi' \sim L'_{\text{F}}$.

(C3) For $\chi \sim L_F$, the random variable χ' is uniformly distributed over the set of colorings satisfying $\chi'(w) = \chi(w)$ for $w \neq v$ (i.e., this follows Glauber dynamics).

(C4) Other than copying L'_{IN} , the expected runtime of COMPRESS.GEN is $O(\Delta \log k + \log n)$. The runtime of COMPRESS.DECODE is $O(\Delta(\log \Delta \log k + \log n))$.

Proof Sketch. The first and second items follow trivially by construction, and the final item can also be justified easily (see [2, Lemma 2.1] for details); the technical heart of the above lemma is the third item, whose proof we now sketch.

Note that the randomness in χ' comes entirely from the COMPRESS.GEN routine. Consider some $\chi \sim L_F = L'_{IN}$. COMPRESS.GEN chooses $c_1 \in [k] \setminus A$ uniformly. COMPRESS.DECODE changes only the color of χ at v , in the following way: if $c_1 \notin \chi(N(v))$, then we let $\chi'(v) = c_1$ with probability $p_\chi(v)$. In all other cases, we let $\chi'(v)$ be a uniform color in $A \setminus \chi(N(v))$. Note that this set is empty only when $\chi(N(v)) = A$, which implies that $p_\chi(v) = (k - \Delta)/(k - |\chi(N(v))|) = 1$ and $c_1 \notin \chi(N(v))$, i.e., that the first case is always invoked. Hence the decoding algorithm is well-defined.

Finally, we check that the color $\chi'(v)$ is chosen with the correct probability. For this, note that we choose any fixed element $c \in [k] \setminus (A \cup \chi(N(v)))$ if $c_1 = c$ and $\tau \leq p_\chi(v)$, which happens with probability

$$\frac{1}{k - \Delta} \cdot \frac{k - \Delta}{k - |\chi(N(v))|} = \frac{1}{k - |\chi(N(v))|},$$

which is the correct probability according to the Glauber dynamics. By symmetry, the remaining probability is easily seen to be split equally among $A \setminus \chi(N(v))$, hence the probability distribution of $\chi'(v)$ is indeed uniform on $[k] \setminus \chi(N(v))$. \square

4.2 SEEDING

The second update, SEEDING, is a variant of CONTRACT in [2], and has the crucial property of operating under a weaker guarantee than $|S_L(v)| \leq k - \Delta$. The tradeoff in exchange for this weaker guarantee is that the bounding set is no longer guaranteed to be of size 2 but will instead be of size at most 3.

Lemma 4.2. *Let $\alpha_{IN} = (v_{IN}, \tau_{IN}, L_{IN}, L'_{IN}, M_{IN}, \gamma_{IN})$, and choose $v \in V(G)$ such that $\frac{|S_L(v)|^2}{\Delta + |S_L(v)|} \leq k - \Delta$. Let $\alpha_F = (v_F, \tau_F, L_F, L'_F, M'_F, \gamma'_F)$ be the output of SEEDING.GEN $[\alpha_{IN}, v]$. Let χ be a coloring, and let $\chi' = \text{SEEDING.DECODE}[\alpha_F, \chi]$. Then:*

(S1) $L_F = L'_{IN}$, $L'_F(u) = L_F(u)$ for $u \neq v$, and $|L'_F(v)| \leq 3$.

(S2) If $\chi \sim L_F$, then $\chi' \sim L'_F$.

(S3) For $\chi \sim L_F$, the random variable χ' is uniformly distributed over the set of colorings satisfying $\chi'(w) = \chi(w)$ for $w \neq v$ (i.e., this follows Glauber dynamics).

(S4) Other than copying L'_{IN} , the expected runtime of SEEDING.GEN is $O(\Delta(\log k + \log n))$. The runtime of SEEDING.DECODE is $O(\Delta(\log k + \log n))$.

Proof. The first two items are trivial, and the fourth item follows in the same way as in [2, Lemma 2.2(d)]. We now verify the key third item. First, note that $p_\chi(v) \in [0, 1]$ by the condition given. Indeed, since $|\chi(N(v))| \in (0, \Delta]$ we easily see that $p_\chi(v) \geq 0$, and moreover, that $p_\chi(v)$ is maximized by its value when $|\chi(N(v))| = 0$ or $|\chi(N(v))| = \Delta$ (since the denominator of $p_\chi(v)$ is a

Algorithm 3: SEEDING – Takes an input update $\alpha_{\text{IN}} = (v_{\text{IN}}, \tau_{\text{IN}}, L_{\text{IN}}, L'_{\text{IN}}, M_{\text{IN}}, \gamma_{\text{IN}})$ and a vertex v with $\frac{|S_L(v)|^2}{\Delta + |S_L(v)|} \leq k - \Delta$, and outputs a compatible “seeding update”.

1 **Function** SEEDING.GEN:

Input : $\alpha_{\text{IN}} = (v_{\text{IN}}, \tau_{\text{IN}}, L_{\text{IN}}, L'_{\text{IN}}, M_{\text{IN}}, \gamma_{\text{IN}})$, $v \in V(G)$ with $\frac{|S_L(v)|^2}{\Delta + |S_L(v)|} \leq k - \Delta$
Output: $\alpha_{\text{F}} = (v_{\text{F}}, \tau_{\text{F}}, L_{\text{F}}, L'_{\text{F}}, M_{\text{F}}, \gamma_{\text{F}})$
2 $\gamma_{\text{F}} \leftarrow 2$; $v_{\text{F}} \leftarrow v$; $L_{\text{F}} \leftarrow L'_{\text{IN}}$; $L'_{\text{F}} \leftarrow L_{\text{F}}$; $\tau_{\text{F}} \leftarrow \text{UNIF}[0, 1]$;
3 $c_1 \leftarrow \text{UNIF}([k] \setminus S_L(v))$; $c_2, c_3 \leftarrow \text{UNIF}(S_L(v))$; $L'_{\text{F}}(v) \leftarrow \{c_1, c_2, c_3\}$;
4 ▷ c_2, c_3 chosen with repetition
5 $M_{\text{F}} \leftarrow (c_1, c_2, c_3)$;
6 $\alpha_{\text{F}} \leftarrow (v_{\text{F}}, \tau_{\text{F}}, L_{\text{F}}, L'_{\text{F}}, M_{\text{F}}, \gamma_{\text{F}})$;

7 **Function** SEEDING.DECODE:

Input : $\alpha = (v, \tau, L, L', M, \gamma)$ with $\gamma = 2$, $\frac{|S_L(v)|^2}{\Delta + |S_L(v)|} \leq k - \Delta$, $M[1] \notin S_L(v)$,
 $M[2], M[3] \in S_L(v)$, and a coloring $\chi \sim L$
Output: $\chi' \sim L'$
8 $\chi' \leftarrow \chi$;
9 $p_{\chi}(v) \leftarrow \frac{|S_L(v)|^2}{(k - |\chi(N(v))|)(|\chi(N(v))| + |S_L(v)|)}$;
10 $(c_1, c_2, c_3) \leftarrow M[1, 3]$
11 **if** $\{c_2, c_3\} \subseteq \chi(N(v))$ or $\tau > p_{\chi}(v)$ **then**
12 $\chi'(v) \leftarrow c_1$;
13 **else if** $c_2 \notin \chi(N(v))$ **then**
14 $\chi'(v) \leftarrow c_2$;
15 **else**
16 $\chi'(v) \leftarrow c_3$;

concave function of $|\chi(N(v))|$ on $[0, \Delta]$. In the former case, we have $p_{\chi}(v) = |S_L(v)|/k \leq 1$. In the latter case, we have

$$p_{\chi}(v) = \frac{|S_L(v)|^2}{(k - \Delta)(\Delta + |S_L(v)|)} \leq 1$$

by the given condition.

Finally, consider any color $c \in S_L(v) \setminus \chi(N(v))$. It is chosen if and only if $\tau \leq p_{\chi}(v)$, and also either $c_2 = c$ or $c_3 = c$ (and the latter case clearly satisfies $c_2 \neq c$). This occurs with probability

$$\left(\frac{1}{|S_L(v)|} + \frac{|\chi(N(v))|}{|S_L(v)|} \cdot \frac{1}{|S_L(v)|} \right) \cdot \frac{|S_L(v)|^2}{(k - |\chi(N(v))|)(|\chi(N(v))| + |S_L(v)|)} = \frac{1}{k - |\chi(N(v))|},$$

as required. Furthermore, since colors in $[k] \setminus S_L(v)$ are symmetrically chosen, the result again immediately follows. \square

4.3 DISJOINT

We now define our most complicated update, DISJOINT, which can be seen as combining the CONTRACT update in [2] with a bounding list version of Jerrum’s analysis of the Glauber dynamics in [9], by pairing up colors in $D_L(v)$. This additional pairing, compared to the analysis in

[2, 8], is critical in obtaining a better drift estimate in the final coalescence phase and ensuring that the final stages of Phase 3 succeed for $k < 3\Delta$.

Algorithm 4: DISJOINT – Takes an input update $\alpha_{\text{IN}} = (v_{\text{IN}}, \tau_{\text{IN}}, L_{\text{IN}}, L'_{\text{IN}}, M_{\text{IN}}, \gamma_{\text{IN}})$ and a vertex v with $S - Q < (k - \Delta)(\frac{k-Q}{k-Q-D/2})$, and outputs a compatible “disjoint update”.

1 **Function** DISJOINT.GEN:

Input : $\alpha_{\text{IN}} = (v_{\text{IN}}, \tau_{\text{IN}}, L_{\text{IN}}, L'_{\text{IN}}, M_{\text{IN}}, \gamma_{\text{IN}})$, $v \in V(G)$ with $S - Q < (k - \Delta)(\frac{k-Q}{k-Q-D/2})$
 Output: $\alpha_{\text{F}} = (v_{\text{F}}, \tau_{\text{F}}, L_{\text{F}}, L'_{\text{F}}, M_{\text{F}}, \gamma_{\text{F}})$
2 $\gamma_{\text{F}} \leftarrow 3$; $v_{\text{F}} \leftarrow v$; $L_{\text{F}} \leftarrow L'_{\text{IN}}$; $L'_{\text{F}} \leftarrow L_{\text{F}}$; $\tau_{\text{F}} \leftarrow \text{UNIF}[0, 1]$;
3 **if** $\text{UNIF}[0, 1] > \frac{k-Q-D}{k-Q-D/2}$ **then**
4 $w \leftarrow \text{UNIF}(N_L^*(v))$; $L'_{\text{F}}(v) \leftarrow N(w)$; $M_{\text{F}} \leftarrow N(w)$; \triangleright Arbitrarily order $N(w)$
5 **else**
6 $c_1 \leftarrow \text{UNIF}([k] \setminus S_L(v))$; $c_2 \leftarrow \text{UNIF}(D_L(v))$;
7 $q(v) \leftarrow 1 - \frac{(k-Q-D/2)E}{(k-Q-D)(k-\Delta)}$; $p_{\Delta}(v) \leftarrow \frac{(\Delta-Q-D/2)D}{(k-\Delta)(k-Q-D)q(v)}$
8 **if** $\text{UNIF}[0, 1] \leq q(v)$ **then**
9 **if** $\text{UNIF}[0, 1] > p_{\Delta}(v)$ **then**
10 $L'_{\text{F}}(v) \leftarrow \{c_1\}$; $M_{\text{F}} \leftarrow (c_1)$;
11 **else**
12 $L'_{\text{F}}(v) \leftarrow \{c_1, c_2\}$; $M_{\text{F}} \leftarrow (c_1, c_2)$;
13 **else**
14 $c_2 \leftarrow \text{UNIF}(E_L(v))$; $L'_{\text{F}}(v) \leftarrow \{c_1, c_2\}$; $M_{\text{F}} \leftarrow (c_1, c_2)$;
15 $\alpha_{\text{F}} \leftarrow (v_{\text{F}}, \tau_{\text{F}}, L_{\text{F}}, L'_{\text{F}}, M_{\text{F}}, \gamma_{\text{F}})$;

16 **Function** DISJOINT.DECODE:

Input : $\alpha = (v, \tau, L, L', M, \gamma)$ with $\gamma = 3$, $S - Q < (k - \Delta)(\frac{k-Q}{k-Q-D/2})$, and a coloring
 $\chi \sim L$
 Output: $\chi' \sim L'$
17 $\chi' \leftarrow \chi$;
18 $q(v) \leftarrow 1 - \frac{(k-Q-D/2)E}{(k-Q-D)(k-\Delta)}$; $p_{\chi}(v) \leftarrow \frac{(|\chi(N(v))| - Q - D/2)D}{(k - |\chi(N(v))|)(k - Q - D)q(v)}$; $p'_{\chi}(v) \leftarrow \frac{k - \Delta}{k - |\chi(N(v))|}$;
19 **if** $|M| = 1$ or $M[1, 2] \subseteq D_L(v)$ **then**
20 $\chi'(v) \leftarrow M \setminus \chi(N(v))$; \triangleright This is size 1 due to the disjointness condition
21 **else**
22 **if** $M[2] \in D_L(v)$ **then**
23 $r_{\chi}(v) \leftarrow p_{\chi}(v)/p_{\Delta}(v)$;
24 **else**
25 $r_{\chi}(v) \leftarrow p'_{\chi}(v)$;
26 **if** $M[2] \in \chi(N(v))$ or $\tau > r_{\chi}(v)$ **then**
27 $\chi'(v) \leftarrow M[1]$;
28 **else**
29 $\chi'(v) \leftarrow M[2]$;

To begin, recall from [Section 3.1](#) that $S_L(v)$ is the set of colors appearing in the bounding lists $L(w)$ for neighbors w of v , $Q_L(v)$ is the set of colors that appear in some bounding list $L(w)$ for $w \in N(v)$ with $|L(w)| = 1$, i.e., the bounding list forces this color to appear in $\chi(N(v))$ if $\chi \sim L$,

$$N_L^*(v) = \{w \in N(v) : |L(w)| = 2 \text{ and } L(w) \cap L(w') = \emptyset \text{ if } w' \in N(w), w' \neq w\},$$

and the disjoint-pair colors associated to v are

$$D_L(v) = \bigcup_{w \in N_L^*(v)} L(w).$$

The key property of the disjoint-pair colors is that they appear in exactly one bounding set of a neighbor of v , and moreover, are naturally paired up with another disjoint-pair color via the bounding set of the *same* element of $N_L^*(v)$. Note that $Q_L(v) \cap D_L(v) = \emptyset$ and $D_L(v)$ is a disjoint union of pairs $L(w)$ for $w \in N_L^*(v)$; in particular, $\chi(N(v))$ always has at least $|Q_L(v)| + |D_L(v)|/2$ different colors. Recall also that $E_L(v) = S_L(v) \setminus (Q_L(v) \cup D_L(v))$.

Finally, for the sake of notational lightness, throughout this subsection and in the definition of [Algorithm 4](#), we let $S = |S_L(v)|$, $Q = |Q_L(v)|$, $D = |D_L(v)|$, and $E = |E_L(v)| = S - Q - D$.

Lemma 4.3. *Let $\alpha_{\text{IN}} = (v_{\text{IN}}, \tau_{\text{IN}}, L_{\text{IN}}, L'_{\text{IN}}, M_{\text{IN}}, \gamma_{\text{IN}})$, and choose $v \in V(G)$ such that $S - Q < (k - \Delta) \left(\frac{k - Q}{k - Q - D/2} \right)$. Let $\alpha_{\text{F}} = (v_{\text{F}}, \tau_{\text{F}}, L_{\text{F}}, L'_{\text{F}}, M'_{\text{F}}, \gamma'_{\text{F}})$ be the output of `DISJOINT.GEN` $[\alpha_{\text{IN}}, v]$. Let χ be a coloring and let $\chi' = \text{DISJOINT.DECODE}[\alpha_{\text{F}}, \chi]$. Then:*

- (D1) $L_{\text{F}} = L'_{\text{IN}}$, $L'_{\text{F}}(u) = L_{\text{F}}(u)$ for $u \neq v$, and $|L'_{\text{F}}(v)| \leq 2$. Moreover, $|L'_{\text{F}}(v)| = 1$ with probability $1 - \frac{S - Q}{k - \Delta} + \frac{D/2}{k - Q - D/2}$.
- (D2) If $\chi \sim L_{\text{F}}$, then $\chi' \sim L'_{\text{F}}$.
- (D3) For $\chi \sim L_{\text{F}}$, the random variable χ' is uniformly distributed over the set of colorings satisfying $\chi'(w) = \chi(w)$ for $w \neq v$ (i.e., this follows Glauber dynamics).
- (D4) Other than copying L'_{IN} , the expected runtime of `DISJOINT.GEN` is $O(\Delta(\log k + \log n))$. The runtime of `DISJOINT.DECODE` is $O(\Delta(\log k + \log n))$.

Proof. To begin, we check that the quantities $q(v), p_{\chi}(v), p_{\Delta}(v), p'_{\chi}(v)$ appearing in [Algorithm 4](#) lie in $[0, 1]$ and $p_{\chi}(v) \leq p_{\Delta}(v)$. Since $|\chi(N(v))| \leq \Delta < k$, it follows that $p'_{\chi}(v) \in [0, 1]$. Next, since

$$\frac{k - Q - D}{k - Q - D/2} \cdot (1 - q(v)) \cdot \frac{1}{E} \cdot p'_{\chi}(v) = \frac{1}{k - |\chi(N(v))|}, \quad (4.1)$$

it follows that $1 - q(v) > 0$. Also,

$$\begin{aligned} \frac{1}{k - Q - D/2} + \frac{k - Q - D}{k - Q - D/2} \cdot q \cdot \frac{1}{D} &= \frac{k - Q}{(k - Q - D/2)D} - \frac{E}{(k - \Delta)D} \\ &= \frac{k - Q}{(k - Q - D/2)D} - \frac{S - Q - D}{(k - \Delta)D} \\ &> \frac{S - Q}{(k - \Delta)D} - \frac{S - Q - D}{(k - \Delta)D} \\ &= \frac{1}{k - \Delta} \\ &\geq \frac{1}{k - |\chi(N(v))|} \end{aligned}$$

$$\geq \frac{1}{k - Q - D/2};$$

where the strictly inequality uses our assumption that $S - Q < (k - \Delta)(\frac{k-Q}{k-Q-D/2})$; this shows that $q(v) > 0$. Since

$$\frac{1}{k - Q - D/2} + \frac{k - Q - D}{k - Q - D/2} \cdot q(v) \cdot \frac{1}{D} \cdot p_\chi(v) = \frac{1}{k - |\chi(N(v))|}, \quad (4.2)$$

combining with the previous inequality shows that $p_\chi(v) \in [0, 1]$. A similar argument also shows that $p_\Delta \in [0, 1]$. Finally, since $|\chi(N(v))| \leq \Delta$, it follows that $p_\chi(v) \leq p_\Delta(v)$.

We now proceed to the proof of the items in the conclusion of the lemma. The second item is trivial, and the fourth item follows as in [2, Lemma 2.2(d)]. The only non-trivial part of the first item is the claim about the probability with which $|L'_r(v)| = 1$, which we will check at the end of the proof. We now verify the third item.

- The expression on the left hand side of (4.1) is the probability that a particular $c \in E_L(v) \setminus \chi(N(v))$ is chosen as $\chi'(v)$, since for this to happen, we must have chosen the second case of DISJOINT.GEN (which happens with probability $(k - Q - D)/(k - Q - D/2)$), then the second subcase of this (which happens independently with probability $1 - q(v)$), and then chosen $c \in E_L(v)$ from a uniform sample (which happens independently with probability $1/E$), all before choosing the last line of DISJOINT.DECODE (which happens independently with probability $p'_\chi(v)$).
- The expression on the left hand side of (4.2) is the probability that a particular $c \in D_L(v) \setminus \chi(N(v))$ is chosen as $\chi'(v)$.
 - The first term comes from the case where we generate $L'_r(v) = \{c, c'\} = N(w)$ for some $w \in N_L^*(v)$ (which happens with probability $(1 - (k - Q - D)/(k - Q - D/2)) \times 2/D = 1/(k - Q - D/2)$) – note that this always decodes to c since c is the unique element in $\{c, c'\}$ which is not in $\chi(N(v))$.
 - The second term is similar to the previous paragraph – specifically, we must choose the second case of DISJOINT.GEN (which happens with probability $(k - Q - D)/(k - Q - D/2)$), then the first subcase of that (which happens independently with probability $q(v)$), choose $c \in D_L(v)$ from a uniform sample (which independently happens with probability $1/D$), enter line 12 of DISJOINT.GEN (which happens independently with probability $p_\Delta(v)$), and finally, enter line 30 of DISJOINT.DECODE (which happens independently with probability $p_\chi(v)/p_\Delta(v)$).
- Finally, all remaining colors in $[k] \setminus \chi(N(v))$ are in $[k] \setminus S_L(v)$, and are treated uniformly, hence as before we have the desired uniformity.

Finally, we verify the remaining claim in the first item. Indeed, the bounding chain gives a set of size 1 with probability

$$\begin{aligned} \frac{k - Q - D}{k - Q - D/2} q(v) (1 - p_\Delta(v)) &= \frac{k - Q - D}{k - Q - D/2} \left(1 - \frac{(k - Q - D/2)E}{(k - Q - D)(k - \Delta)} \right. \\ &\quad \left. - \frac{(\Delta - Q - D/2)D}{(k - \Delta)(k - Q - D)} \right) \\ &= 1 - \frac{S - Q}{k - \Delta} + \frac{D/2}{k - Q - D/2}, \end{aligned}$$

as desired. \square

5 Analysis of SAMPLERUNIT

Let SAMPLERUNIT be defined as in Section 3.4. More formally, let T be the total number of updates used in the four phases, and starting from time $-T$, let $(\alpha_{-T})_F, \dots, (\alpha_{-1})_F$ be the T updates described in Section 3.4 generated as follows: for each $t \in [T]$, $(v_{-t})_{\text{IN}}$ and $(\gamma_{-t})_{\text{IN}}$ are chosen as described in Section 3.4. Moreover,

$$((\tau_{-t})_{\text{IN}}, (L_{-t})_{\text{IN}}, (L'_{-t})_{\text{IN}}, (M_{-t})_{\text{IN}}) = ((\tau_{-t-1})_F, (L_{-t-1})_F, (L'_{-t-1})_F, (M_{-t-1})_F),$$

with the initial conditions

$$((\tau_{-T})_{\text{IN}}, (L_{-T})_{\text{IN}}, (L'_{-T})_{\text{IN}}, (M_{-T})_{\text{IN}}) = (1, \prod_{v \in V(G)} [k], \prod_{v \in V(G)} [k], \emptyset).$$

We slightly overload notation (this does not create any confusion) by using F to refer both to the sequence of tuples $(\alpha_{-T})_F, \dots, (\alpha_{-1})_F$, as well as the composite function $(\alpha_{-1})_F.\text{DECODE} \circ \dots \circ (\alpha_{-T})_F.\text{DECODE}$, interpreted in the obvious way, generated by these tuples. Also, the predicate $\Phi(F)$ is defined as evaluating to TRUE iff $(L'_{-1})_F$ is a list of sets of size 1.

To complete the proof of Theorem 1.1, we need to check two things:

- (Q1) Our description of SAMPLERUNIT is well-defined i.e. the promise required to execute SEEDING and DISJOINT is satisfied at every step.
- (Q2) The resulting \mathcal{D}, Φ satisfy properties (P1)-(P4) of Lemma 2.1.

The next subsection contains an analysis of Phase 4 of SAMPLERUNIT, and completely addresses (Q2).

5.1 Drift analysis: Phase 4 succeeds with probability at least 1/2

The goal of this subsection is to show that after all bounding sets have been reduced to size at most 2, applying DISJOINT a sufficient number of times at a uniformly randomly chosen vertex gives coalescence with sufficiently high probability. This is the analogue of [2, Lemma 2.5]. As in [2, 8], we will make use of the following random walk lemma due to Huber [8] (stated below with minor indexing errors corrected).

Theorem 5.1 ([8, Theorem 4]). *Suppose that X_t is a random walk on $\{0, 1, \dots, n\}$ where 0 is a reflecting state and n is an absorbing state. Further, assume that $|X_{t+1} - X_t| \leq 1$, and $\mathbb{E}[X_{t+1} - X_t \mid X_t = i] \geq \kappa_i > 0$ for all $X_t < n$. Let e_i be the expected number of times the walk hits the state i . Then,*

$$\sum_{i=0}^{n-1} e_i \leq \sum_{i=0}^{n-1} \frac{1}{\kappa_i}.$$

Lemma 5.2. *Assume $k > 5\Delta/2$ and let $T_D = 2 \frac{k-\Delta}{k-5\Delta/2} n \log n$. Suppose that we have an update $\alpha_0 = \alpha$ with bounding list L' satisfying $|L'(v)| \leq 2$ for all $v \in V(G)$. Consider a random sequence of DISJOINT updates $\alpha_1, \dots, \alpha_{T_D}$ generated in sequence (note the forward time indexing) via*

$$\alpha_t = \text{DISJOINT.GEN}[\alpha_{t-1}, \text{UNIF}(V(G))]$$

for all $1 \leq t \leq T_D$. Let $\alpha_{T_D} = (\cdot, \cdot, \cdot, L'_0, \cdot, \cdot)$. Then $|L'_0(v)| = 1$ for all $v \in V(G)$ with probability at least 1/2.

Remark. Part of the assertion of this lemma is that the promise required to execute DISJOINT is satisfied throughout.

Proof. Let $W_t = \{v \in V(G) : |L'_t(v)| = 1\}$, let $X_t = |W_t|$, and let $\overline{W}_t = V(G) \setminus W_t$. Clearly, $X_t \in \{0, \dots, n\}$ and $|X_{t+1} - X_t| \leq 1$, since each update changes the bounding list for at most one vertex. We now show that

$$\mathbb{E}[X_{t+1} - X_t | X_t] \geq \frac{n - X_t}{n} \left(1 - \frac{3\Delta/2}{k - \Delta}\right). \quad (5.1)$$

After proving (5.1), the claim follows immediately by Theorem 5.1, since n is easily verified to be an absorbing state (because in such a situation, any vertex $v \in V(G)$ has $|S_L(v)| - |Q_L(v)| = |D_L(v)| = 0$).

To prove (5.1) we first show that if $|L(w)| \leq 2$ for all $w \in V(G)$, then

$$|S_L(v)| - |Q_L(v)| \leq \frac{3}{2} |\overline{W}_t \cap N(v)| + \frac{|N_L^*(v)|}{2} \leq \frac{3}{2} |\overline{W}_t \cap N(v)| + \frac{|D_L(v)|}{4} \quad (5.2)$$

To see this, consider assigning weights to each color: a color in $S_L(v) \setminus Q_L(v)$, which appears in m bounding lists, is assigned weight $1/m$, and any other color is assigned weight 0. In particular, for any $w \in N_L^*(v)$, both elements of $L(w)$ are weight 1 so that the sum of weights in $L(w)$ for $w \in N_L^*(v)$ is 2. Also, if $|L(w)| = 1$, then sum of weights is 0 by definition. Finally, in all other cases the sum of weights is at most $1 + 1/2 = 3/2$ (since $w \notin N_L^*(v)$ implies that at least one of the two colors must appear in at least two bounding lists). Now, (5.2) follows by noting that the leftmost quantity is the sum of all the weights in all $L(w)$ for $w \in N(v)$ (counting colors multiple times), whereas the middle quantity is a trivial upper bound for this sum given the information above, and the rightmost inequality follows by noting that $|N_L^*(v)| = |D_L(v)|/2$.

Equation (5.2) shows that the condition needed to apply DISJOINT.GEN is satisfied at every step it is used. Specifically, we find that

$$|S_L(v)| - |Q_L(v)| \leq \frac{3}{2}\Delta + \frac{|D_L(v)|}{4} \leq (k - \Delta) \frac{k - |Q_L(v)|}{k - |Q_L(v)| - |D_L(v)|/2},$$

where the second inequality holds since $D_L(v) \in [0, 2\Delta]$, $k \geq 2.5\Delta$, and $|Q_L(v)| \geq 0$.

Moreover, dividing (5.2) by $k - \Delta$, we immediately deduce that

$$\frac{|S_L(v)| - |Q_L(v)|}{k - \Delta} - \frac{|D_L(v)|/2}{k - |Q_L(v)| - |D_L(v)|/2} \leq \frac{3}{2} \frac{|\overline{W}_t \cap N(v)|}{k - \Delta}, \quad (5.3)$$

where for the second term on the left hand side, we have used that $k - |Q_L(v)| - |D_L(v)|/2 \leq k \leq 2(k - \Delta)$.

Therefore,

$$\begin{aligned} \mathbb{E}[X_{t+1} - X_t | L'_t] &= \frac{1}{n} \left[\sum_{v \notin W_t} \left(1 - \frac{|S_L(v)| - |Q_L(v)|}{k - \Delta} + \frac{|D_L(v)|/2}{k - |Q_L(v)| - |D_L(v)|/2}\right) \right. \\ &\quad \left. - \sum_{v \in W_t} \left(\frac{|S_L(v)| - |Q_L(v)|}{k - \Delta} - \frac{|D_L(v)|/2}{k - |Q_L(v)| - |D_L(v)|/2}\right) \right] \\ &= \frac{1}{n} \left[|\overline{W}_t| - \sum_{v \in V(G)} \left(\frac{|S_L(v)| - |Q_L(v)|}{k - \Delta} - \frac{|D_L(v)|/2}{k - |Q_L(v)| - |D_L(v)|/2}\right) \right] \end{aligned}$$

$$\begin{aligned}
&\geq \frac{1}{n} \left[|\overline{W}_t| - \frac{3}{2} \sum_{v \in V(G)} \frac{|\overline{W}_t \cap N(v)|}{k - \Delta} \right] \\
&\geq \frac{|\overline{W}_t|}{n} \left[1 - \frac{3\Delta/2}{k - \Delta} \right],
\end{aligned}$$

where the penultimate inequality uses (5.3) and the last inequality follows since the graph has maximum degree at most Δ . Finally, the law of total expectation and $|\overline{W}_t| = n - X_t$ gives the desired inequality. \square

In particular, Lemma 5.2 shows that once SAMPLERUNIT reaches Phase 4, it succeeds with probability at least $1/2$. Therefore, it only remains to check that the first three phases are always completed successfully i.e. the various guarantees required by COMPRESS, SEEDING, and DISJOINT are satisfied throughout the first three phases.

5.2 Phases 1, 2, and 3 always succeed

Lemma 5.3. *Phase 1 succeeds deterministically, i.e., every application of COMPRESS and SEEDING is guaranteed to satisfy its promise.*

Proof. First, note that for every $i \in [s]$, we can indeed find the necessary associated set $A - v_i$ has at most $\Delta/3$ neighbors in \mathcal{S} by construction, and each neighbor of v_i in $\{v_j : j < i\}$ has a bounding list of size at most 3 (since SEEDING has already been applied to such a vertex); hence A needs to contain a union of at most $\Delta/3$ sets of size at most 3.

Next, note that when we perform SEEDING on v_i , we trivially have $|S_L(v)| \leq 2\Delta$ since each vertex to which we apply COMPRESS (i.e. those neighbors of v_i which do not precede it in \mathcal{S}) contributes up to 1 additional color not present in A . The claim now follows upon noting that

$$\frac{|S_L(v)|^2}{\Delta + |S_L(v)|} \leq \frac{(2\Delta)^2}{\Delta + (2\Delta)} = \frac{4\Delta}{3} \leq k - \Delta. \quad \square$$

Lemma 5.4. *Phase 2 succeeds deterministically, i.e., every application of DISJOINT is guaranteed to satisfy its promise.*

Proof. The existence of the set A follows as in the previous proof by noting that each vertex has at most $\Delta/3$ neighbors in \mathcal{S} , each of which has a bounding list of size at most 3 at the end of Phase 1.

Further, since each v_i has at most $(1 - \eta)\Delta$ neighbors outside \mathcal{S} , it follows that after applying COMPRESS to all neighbors of v_i not in \mathcal{S} , we have $|S_L(v)| \leq \Delta + (1 - \eta)\Delta = (2 - \eta)\Delta$, where the first inequality follows since the $(1 - \eta)\Delta$ neighbors of v_i outside \mathcal{S} can each contribute at most 1 color outside of A to $S_L(v)$. The claim now follows upon noting that

$$|S_L(v)| - |Q_L(v)| \leq (2 - \eta)\Delta < k - \Delta \leq (k - \Delta) \left(\frac{k - |Q_L(v)|}{k - |Q_L(v)| - |D_L(v)|/2} \right). \quad \square$$

The analysis of Phase 3 is more nontrivial due to the intricate nature of choosing the set A of size Δ . Ultimately the proof is a routine casework check.

Lemma 5.5. *Phase 3 succeeds deterministically, i.e., every application of DISJOINT is guaranteed to satisfy its promise.*

Proof. Let $v = v_i$ for some $s + 1 \leq i \leq n$. As in the definition of Phase 3, let L_m be current the bounding list, restricted to marked neighbors of v . Also, let L be the bounding list *after* applying COMPRESS to all unmarked neighbors of v . Note that v has at most $(1 - \eta)\Delta$ unmarked neighbors; suppose that it has $\eta'\Delta$ unmarked neighbors (this is true even for the neighbors of v in \mathcal{S}^c). Recall that $\eta = 1/3 - 2\sqrt{(\log \Delta)/\Delta}$ for a sufficiently large constant C and that $S_{L_m}(v)$ is the disjoint union $Q_{L_m}(v) \cup E_{L_m}(v) \cup D_{L_m}(v)$. Recall also that for all marked neighbors w of v_i , $|L_m(w)| \leq 2$.

Case 1: $|S_{L_m}(v)| \leq \Delta$. In this case, we must have that $S_{L_m}(v) \subset A$, since colors in $S_{L_m}(v)$ are chosen to be in A before any other colors. Then, as in the proof of Lemma 5.4, we see that $|S_L(v)| \leq \Delta + (1 - \eta)\Delta$, so that as before,

$$|S_L(v)| - |Q_L(v)| \leq (2 - \eta)\Delta < k - \Delta \leq (k - \Delta) \frac{k - |Q_L(v)|}{k - |Q_L(v)| - |D_L(v)|/2}.$$

Case 2: $A \subseteq Q_{L_m}(v) \cup E_{L_m}(v)$. In particular, we must have $X = |Q_{L_m}(v) \cup E_{L_m}(v)| \geq \Delta$. Let $Y = |D_{L_m}(v)|$. Since the bounding list of each marked vertex has size at most 2, we may use a similar argument as in the proof of (5.2) to see that the total weight (as defined there) of the bounding list of each marked vertex w intersecting $Q_{L_m}(v) \cup E_{L_m}(v)$ is at most $3/2$. Since X is at least the sum of all the weights in all such lists (counting colors multiple times), it follows that there are at least $2X/3$ different (marked) $w \in N(v)$ with $L_m(w)$ intersecting $Q_{L_m}(v) \cup E_{L_m}(v)$. Therefore there are at most $\Delta - \eta'\Delta - 2X/3$ marked neighbors intersecting $D_{L_m}(v)$, so that

$$|S_{L_m}(v)| \leq X + 2(\Delta - \eta'\Delta - 2X/3) \leq 5\Delta/3 - 2\eta'\Delta.$$

Finally, after applying COMPRESS to the unmarked neighbors of v , we gain an additional at most $\eta'\Delta$ elements. Thus $|S_L(v)| \leq (5/3 - \eta')\Delta$, and as above, the result follows immediately since $5/3 \leq 2 - \eta$.

Case 3: $Q_{L_m}(v) \cup E_{L_m}(v) \subseteq A \subseteq S_{L_m}(v)$. Again let $X = |Q_{L_m}(v) \cup E_{L_m}(v)|$ and $Y = |D_{L_m}(v)|$. Thus $X \leq \Delta \leq X + Y$.

First, by repeating the computation in Case 2, but with the trivial lower bound $X \geq 0$, we see that $X + Y \leq 2(1 - \eta')\Delta$ and $|S_L(v)| \leq 2(1 - \eta')\Delta + \eta'\Delta = (2 - \eta')\Delta$. Thus if $\eta' \geq \eta$, we are done as before. Hence, we may assume that $\eta' \in [0, \eta]$.

We ultimately want to check the condition

$$|S_L(v)| - |Q_L(v)| < (k - \Delta) \frac{k - |Q_L(v)|}{k - |Q_L(v)| - |D_L(v)|/2}. \quad (5.4)$$

Since the left hand side is decreasing and the right hand side is increasing in $|Q_L(v)|$, it suffices to check $|S_L(v)| < (k - \Delta)k/(k - |D_L(v)|/2)$, i.e.

$$|S_L(v)| \left(k - \frac{|D_L(v)|}{2} \right) < k(k - \Delta). \quad (5.5)$$

Note that there are at least $\lfloor (X + Y - \Delta)/2 \rfloor$ pairs of colors $L_m(w)$, for $w \in N_{L_m}^*(v)$, inside $D_{L_m}(v) \setminus A$. Note also that every additional color coming from the $\eta'\Delta$ unmarked neighbors could be one of the following: (i) a color outside of $S_{L_m}(v)$ (ii) a color in $D_{L_m}(v)$ (observe that every such color prevents two colors from $D_{L_m}(v)$ from appearing in $D_L(v)$), and (iii) a color in $A \setminus D_{L_m}(v)$. Suppose we have $s\Delta$ colors of type (i) and $t\Delta$ colors of type (ii). Then,

$$|S_L(v)| = X + Y + s\Delta, \quad |D_L(v)| \geq 2\lfloor (X + Y - \Delta)/2 \rfloor - 2t\Delta, \quad s + t \leq \eta'.$$

Observe that if $|S_L(v)| = X + Y + s\Delta \leq k - \Delta$, then (5.5) is trivially satisfied. Therefore, we may assume that $X + Y \geq k - \Delta - s\Delta \geq (2 - \eta - s)\Delta$. Finally, let $X = x\Delta$, $Y = y\Delta$, and $k = \kappa\Delta$, so that (5.5) follows if

$$(x + y + s) \left(\kappa + t - \frac{x + y - 1 - \Delta^{-1}}{2} \right) < \kappa(\kappa - 1). \quad (5.6)$$

From the discussion above, we have the constraints $z = x + y \in [2 - \eta - s, 2 - 2\eta']$, $s, t \geq 0$, $s + t \leq \eta'$, $\eta' \in [0, \eta]$, and $\kappa \geq 3 - \eta$. Recall also that η is a fixed constant less than $1/3$. Also, increasing $S_L(v) \setminus D_L(v)$ can only make (5.5) harder to satisfy, we may assume that $s + t = \eta'$.

We see by taking derivatives that as long as $\kappa \geq 3/2$ (which is true in our case), the condition in (5.6) is most restrictive when κ is taken smaller. Therefore, we may let $\kappa = 3 - \eta$, to see that (5.6) is implied by

$$(z + s) \left(3 - \eta + t - \frac{z - 1 - \Delta^{-1}}{2} \right) < (2 - \eta)(3 - \eta). \quad (5.7)$$

Next, we see by taking derivatives that for $\eta \in [0, 1/3]$, the condition in (5.7) is strictly more restrictive when $\eta = 1/3$. Therefore, by taking $\eta = 1/3$ (note that for us, η is strictly smaller than $1/3$), we see that (5.7) is implied by

$$(z + s) \left(\frac{8}{3} + t - \frac{z - 1 - \Delta^{-1}}{2} \right) \leq \frac{40}{9}. \quad (5.8)$$

At this point, we assume $\Delta \geq 9$, and reduce to checking

$$(z + s) \left(\frac{29}{9} + t - \frac{z}{2} \right) \leq \frac{40}{9} \quad (5.9)$$

on the region carved out by $s, t \geq 0$, $s + t \leq 1/3$, $z \geq 5/3 - s$, and $z \leq 2 - 2s - 2t$.

Note that the left hand side is a downward quadratic in z with maximum at $z = 20/9 + t - s/2$, which is always bigger than $2 - 2s - 2t$. Hence, the left hand side is maximized at $2 - 2s - 2t$, which yields

$$(2 - s - 2t) \left(\frac{20}{9} + s + 2t \right) \stackrel{?}{\leq} \frac{40}{9};$$

this is clearly true since $s + 2t \geq 0$. □

5.3 Putting everything together

We now quickly check that (Q1) and (Q2) follow from our work so far. Indeed, Lemmas 5.2 to 5.5 shows that (Q1) is true. For (Q2), we note that (P2) follows from (C2), (S2) and (D2), and that (P1) follows from (C3), (S3), and (D3). Moreover, the same running time analysis as in [2] shows that (C4), (S4) and (D4) easily imply (P4). Finally, Lemma 5.2 implies (P3), which completes our analysis.

6 Conclusion and Open Problems

We first briefly elaborate on how the above analysis can be extended to push slightly beyond $k \geq (8/3 + o(1))\Delta$, i.e., to perfectly sample $(8/3 - \epsilon)\Delta$ colors for some absolute constant $\epsilon \approx 10^{-2}$. The current algorithm (for sufficiently large Δ) is only limited at $k = (8/3 + o(1))\Delta$ in Phase 2 (although this requires performing the analysis in Lemma 5.5 more carefully). In order to improve

Phase 2, a variant of [Algorithm 4](#) which allows for disjoint triples works. (There is an even more efficient routine using both disjoint pairs and disjoint triples.)

However, all of these techniques are currently limited at $k > 5\Delta/2$ due to [Lemma 5.2](#); the extremal configuration limiting the analysis here has shadows of the configurations which limit Jerrum’s [\[9\]](#) analysis for approximate sampling at $k > 2\Delta$ using the Glauber dynamics. However, incorporating the techniques of Vigoda [\[15\]](#) and more general path-coupling ideas [\[3\]](#) may allow one to break this barrier, but the interaction of these techniques with the bounding chain framework of [\[7, 8\]](#) is nontrivial and remains an interesting open question.

References

- [1] Noga Alon and Joel H. Spencer, *The probabilistic method*, fourth ed., Wiley Series in Discrete Mathematics and Optimization, John Wiley & Sons, Inc., Hoboken, NJ, 2016.
- [2] Siddharth Bhandari and Sayantan Chakraborty, *Improved bounds for perfect sampling of k -colorings in graphs*, Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, 2020, pp. 631–642.
- [3] Russ Bubley and Martin Dyer, *Path coupling: A technique for proving rapid mixing in markov chains*, Proceedings 38th Annual Symposium on Foundations of Computer Science, IEEE, 1997, pp. 223–231.
- [4] Sitan Chen, Michelle Delcourt, Ankur Moitra, Guillem Perarnau, and Luke Postle, *Improved bounds for randomly sampling colorings via linear programming*, Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SIAM, Philadelphia, PA, 2019, pp. 2216–2234.
- [5] Alan Frieze and Eric Vigoda, *A survey on the use of Markov chains to randomly sample colourings*, Combinatorics, complexity, and chance, Oxford Lecture Ser. Math. Appl., vol. 34, Oxford Univ. Press, Oxford, 2007, pp. 53–71.
- [6] David Gamarnik and Dmitriy Katz, *Correlation decay and deterministic FPTAS for counting colorings of a graph*, J. Discrete Algorithms **12** (2012), 29–47.
- [7] Olle Häggström and Karin Nelander, *Exact sampling from anti-monotone systems*, Statistica Neerlandica **52** (1998), 360–380.
- [8] Mark Huber, *Exact sampling and approximate counting techniques*, STOC ’98 (Dallas, TX), ACM, New York, 1999, pp. 31–40.
- [9] Mark Jerrum, *A very simple algorithm for estimating the number of k -colorings of a low-degree graph*, Random Structures Algorithms **7** (1995), 157–165.
- [10] Mark R. Jerrum, Leslie G. Valiant, and Vijay V. Vazirani, *Random generation of combinatorial structures from a uniform distribution*, Theoret. Comput. Sci. **43** (1986), 169–188.
- [11] Jingcheng Liu, Alistair Sinclair, and Piyush Srivastava, *A deterministic algorithm for counting colorings with 2 -delta colors*, 2019 IEEE 60th Annual Symposium on Foundations of Computer Science (FOCS), IEEE, 2019, pp. 1380–1404.

- [12] Pinyan Lu and Yitong Yin, *Improved FPTAS for multi-spin systems*, Approximation, randomization, and combinatorial optimization, Lecture Notes in Comput. Sci., vol. 8096, Springer, Heidelberg, 2013, pp. 639–654.
- [13] Robin A. Moser and Gábor Tardos, *A constructive proof of the general Lovász local lemma*, J. ACM **57** (2010), Art. 11, 15.
- [14] James Gary Propp and David Bruce Wilson, *Exact sampling with coupled Markov chains and applications to statistical mechanics*, Proceedings of the Seventh International Conference on Random Structures and Algorithms (Atlanta, GA, 1995), vol. 9, 1996, pp. 223–252.
- [15] Eric Vigoda, *Improved bounds for sampling colorings*, J. Math. Phys. **41** (2000), 1555–1569, Probabilistic techniques in equilibrium and nonequilibrium statistical physics.

A Proof of Proposition 3.1

Proof. The proof uses the symmetric Lovász Local Lemma (LLL; see [1]). We sample \mathcal{S} by selecting each vertex in $V(G)$ independently with probability $(\eta + 1/3)/2$. For each $v \in V(G)$, let \mathcal{B}_v denote the ‘bad’ event that v does not satisfy the condition in (3.1). Then, it is straightforward to verify that each bad event is mutually independent from all but at most $d = \Delta^2$ other bad events (corresponding to vertices with distance at most 2 from v). Moreover, a standard application of the Chernoff bound shows that each bad event occurs with probability at most $p = \exp(-\Omega_\eta(\Delta))$. Thus, for $\Delta \geq C_\eta$, we trivially have $ep(d+1) < 1$, which guarantees that a set S satisfying (3.1) for all $v \in V(G)$ exists.

In order to algorithmically generate such a set, we use the algorithmic version of LLL due to Moser and Tardos [13]. We set $x(v) = ep < 1/(d+1)$ for all $v \in V(G)$, which can easily be checked to satisfy the hypotheses of [13, Theorem 1.2]. Therefore, by [13, Theorem 1.2], the expected number of ‘resampling operations’ is at most

$$\sum_{v \in V(G)} \frac{x(v)}{1 - x(v)} < 2 \sum_{v \in V(G)} x(v) = O(n \exp(-\Omega_\eta(\Delta))).$$

For an analysis of the running time, note that it takes time $O(n\Delta)$ to initially sample and compute the number of neighbors in $\mathcal{S}, \mathcal{S}^c$ each vertex has, which we maintain as an array throughout. We also maintain a binary heap with the set of vertices violating (3.1). Since each resampling operation amounts to resampling the neighbors of a violating vertex, we see that, in particular, each resampling operation requires updating at most Δ^2 array elements (corresponding to vertices within distance 2 of the violating vertex at which resampling occurs). Finally, we remove vertices which no longer violate (3.1) from our binary heap, and add vertices which have turned into violators to the binary heap, which takes time $O(\Delta^2 \log n)$. Therefore the running time is $O(n\Delta + n(\log n)\Delta^2 \exp(-\Omega_\eta(\Delta)))$, which is $O(n\Delta + n \log n)$ as desired. Note that we terminate early if the number of resampling operations is twice the expectation in order to obtain the failure probability (by Markov’s inequality) and running time guarantee in the statement of the proposition. \square