MAX-PLANCK-INSTITUT FÜR INFORMATIK

Near–Optimal Distributed Edge Coloring

Devdatt Dubhashi, Alessandro Panconesi

MPI–I–94–136 July 1994

Im Stadtwald
66123 Saarbrücken
Germany

Near–Optimal Distributed Edge Coloring

Devdatt Dubhashi, Alessandro Panconesi

MPI–I–94–136 July 1994

Near-Optimal Distributed Edge Coloring*

Devdatt Dubhashi †
Max-Planck-Institute für Informatik
Im Stadtwald, 66123 Saarbrücken, Germany
dubhashi@mpi-sb.mpg.de

Alessandro Panconesi‡
Centrum voor Wiskunde en Informatica
413 Kruislaan, 1098 SJ, Amsterdam, Holland
ale@cwi.nl

Abstract

We give a distributed randomized algorithm to edge color a network. Given a graph G with n nodes and maximum degree Δ, the algorithm,

- For any fixed $\lambda > 0$, colours G with $(1+\lambda)\Delta$ colours in time $O(\log n)$.
- For any fixed positive integer s, colours G with $\Delta + \frac{\Delta}{(\log \Delta)^s} = (1+o(1))\Delta$ colours in time $O(\log n + \log^{2s} \Delta \log\log \Delta)$.

Both results hold with probability arbitrarily close to 1 as long as $\Delta(G) = \Omega(\log^{1+d} n)$, for some $d > 0$.

The algorithm is based on the Rödl Nibble, a probabilistic strategy introduced by Vojtech Rödl. The analysis involves a certain pseudo-random phenomenon involving sets at the vertices of the graph.

1 Introduction

The edge coloring problem is a basic problem in graph theory and combinatorial optimization. Its importance in distributed computing, and computer science generally, stems from the fact that several scheduling and resource allocation problems can be modeled as edge coloring problems [7, 9, 12, 14]. In this paper, we give a distributed randomized algorithm that computes a near-optimal edge coloring in time $O(\log n)$. By "near-optimal" we mean that the number of colors used is $(1+o(1))\Delta$ where Δ denotes the maximum degree of the network and the $o(1)$ term can be as small as $1/\log^s \Delta$, for any $s > 0$. Both performance guarentees – the running time and the number of colours used – hold with high probability as long as the maximum degree grows at least logarithmically with n. Our algorithm can be implemented directly in the PRAM model of computation.

Motivation and Related Work. The edge coloring problem can be used to model certain types

*Presented at the 15th International Symposium on Mathematical Programming, August 1994, Ann Arbour, Michigan, USA

†Supported by the ESPRIT Basic Research Actions Program of the EC under contract No. 7141 (project ALCOM II).

‡Supported by an Ercim postdoctoral fellowship.

of jobshop scheduling, packet routing, and resource allocation problems in a distributed setting. For example, the problem of scheduling I/O operations in some parallel architectures can be modeled as follows [7]. We are given a bipartite graph $G = (\mathcal{P}, \mathcal{R}, E)$ where, intuitively, $\mathcal{P}$ is a set of processes and $\mathcal{R}$ is a set of resources (say, disks). Each processor needs data from a subset of resources $R(p) \subseteq \mathcal{R}$. The edge set is defined to be $E = \{(p, r) : r \in R(p), p \in \mathcal{P}\}$. Due to hardware limitations each resource $r \in \mathcal{R}$ can service only one processor at a time. Under this constraints it is not hard to see that optimal edge colorings of the bipartite graph correspond to optimal schedules that is, schedules minimizing the overall completion time.

Clearly, if a graph G has maximum degree Δ then at least Δ colors are needed to edge color the graph. A classical theorem of Vizing shows that $\Delta + 1$ colors are always sufficient, and the proof is actually a polynomial time algorithm to compute such a coloring (see for example [4]). Interestingly, given a graph G, it is NP-complete to decide whether it is Δ or $\Delta + 1$ edge colorable [6], even for regular graphs [5]. Efforts at parallelizing Vizing's theorem have failed; the best PRAM algorithm known is a randomized algorithm by Karloff & Shmoys that computes an edge coloring using very nearly $\Delta + \sqrt{\Delta} = (1 + o(1))\Delta$ colors. The Karloff & Shmoys algorithm can be derandomized by using standard derandomization techniques [3, 13]. In the distributed setting the previously best known result was a randomized algorithm by Panconesi & Srinivasan that uses roughly $1.58\Delta + \log^2 n$ colors with high probability and runs in $O(\log n)$ time with high probability. For the interesting special case of bipartite graphs Lev, Pippinger & Valiant show that Δ-colorings can be computed in NC, whereas this is provably impossible in the distributed model of computation even if randomness is allowed (see [15]).

Our solution. To state our results precisely, we reproduce below our main theorem:

Theorem 1 *For any fixed $\lambda > 0$, given a graph with n vertices and maximum degee Δ, we can edge colour the graph with $(1+\lambda)\Delta$ colours in time $O(\log n)$ where n is the number of vertices in the graph. For any fixed positive integer s, we can edge colour it with $\Delta + \Delta/\log^s \Delta = (1+o(1))\Delta$ colours in time $O((\log \Delta)^{2s} \log\log \Delta + \log n)$. The results hold with failure probability decreasing to 0 faster than any polynomial (in n) provided that $\Delta = \Omega(\log^{1+d} n)$ for some $d > 0$.*

Our algorithm is based on the *Rödl Nibble*, a beautiful probabilistic strategy introduced by Vojtech Rödl to solve a certain covering problem in hypergraphs [2, 17]. The method has subsequently been used very successfully to solve other combinatorial problems such as asymptotically optimal coverings and colorings for hypergraphs [2, 8, 16, 18]. In this paper, we introduce it as a tool for the design and analysis of randomized algorithms.[1] Although the main component of our algorithm is the Rödl nibble and the intuition behind it rather compelling, the algorithm requires a non-trivial probabilistic analysis of a so called pseudo-random process. To explain what this is, it is perhaps best to give a brief outline of our algorithm. Starting with the input graph G_0 the algorithm generates a sequence $G_0, G_1, \ldots, G_t$ of graphs. One can view each edge e as possessing a palette of available colors, starting with the whole set of $[\Delta]$ colours initially. At an arbitrary stage, a small ϵ fraction of uncolored edges is selected, and each selected edge chooses a tentative color at random from its current palette. If the tentative color is not chosen by any neighboring edge it becomes final. Palettes of the remaining uncolored edges are updated in the obvious fashion– by deleting colors used by neighboring edges. The process is then repeated.

Like other proofs based on the same method our proof hinges on two key features of the Rödl nibble. The first key idea of the method is that if colors are chosen independently, the

[1] This research was originally prompted by a conversation that the second author had with Noga Alon and Joel Spencer, in which they suggested that the nibble approach should work. Noga Alon has recently informed us that he is already in possession of a solution with similar performance [1]. However, at the time of writing, a written manuscript was not available for comparison.

probability of color conflict is roughly ϵ^2, a negligible fraction of all edges attempting coloring at this stage. If the same "efficiency" is maintained throughout, the overall "wastage" will be very small. The second aspect of the Rödl nibble is a deeper mathematical phenomenon called *quasi-randomness* or *pseudo-randomness* (see [2]). In our context, pseudo-randomness means that the palettes of available colors at the edges at any stage are "essentially" truly independent random subsets of the original full palette. The crux of the analysis is to show that despite the potential of a complicated interaction regulated by the topology of the underlying graph, the "nibbling" feature of the coloring process ensures that the palettes are evolving almost independently of each other. In all applications of the nibble method, it is the pseudo-random aspect which is mathematically challenging and which usually requires a quite laborious probabilistic analysis.

2 Preliminaries

A *message-passing distributed network* is an undirected graph $G = (V, E)$ where vertices (or nodes) correspond to processors and edges to bi–directional communication links. Each processor has its unique ID. The network is *synchronous*, *i.e.*, computation takes place in a sequence of *rounds*; in each round, each processor reads messages sent to it by its neighbors in the graph, does any amount of local computation, and sends messages back to all of its neighbors. The time complexity of a distributed algorithm, or *protocol*, is given by the number of rounds needed to compute a given function. If one wants to translate an algorithm for this model into one for the PRAM then computation locally done by each processor must be charged for.

An *edge coloring* of a graph G is an assignment of colors to edges such that incident edges always have different colors. The edge coloring problem is to find an edge coloring with the aim of minimizing the number of colors used. Given that determining an optimal (minimal) coloring is an NP-hard problem this requirement is usually relaxed to consider approximate, hopefully even near-optimal, colorings. The edge coloring problem in a distributed setting is formulated as follows: a distributed network G wants to compute an edge coloring of its own topology. As remarked in the introduction such a coloring might be useful in the context of scheduling and resource allocation.

The set $\{1, 2, \ldots, n\}$ will be denoted by $[n]$. Given a graph G and a set of edges F, $G[F]$ denotes the subgraph of G whose edge set is F.

In the paper we will use the following approximations repeatedly: $(1 - 1/n)^n \approx e^{-1}$, and $e^\epsilon \approx 1 + \epsilon$ or $e^\epsilon \approx 1 + \epsilon + \epsilon^2/2$, for small values of ϵ. Whenever such an approximation is in effect, we will use the sign $\approx$ in place of the equality sign.

We will make use of a slight modification of a well-known vertex coloring algorithm by Luby [11]. Luby's algorithm computes a $(\Delta + 1)$–vertex coloring of a graph in expected time $O(\log n)$, where n is the number of vertices of a graph of maximum degree Δ. The running time of the algorithm is $O(\log n)$ with high probability [10, 11]. When applied to the line graph of G the algorithm computes a $(2\Delta - 1)$–edge coloring. In the original algorithm each vertex is initially given a palette of $\Delta + 1$ colors; it can be easily verified that the algorithm still works in the same fashion if each vertex u is given a palette of $\deg(u) + 1$ colors instead, where $\deg(u)$ is the degree of u. This modification is introduced for explanatory purposes.

3 The Algorithm

The algorithm is in two phases. The first phase is an application of the Rödl nibble and has the goal of coloring most of the edges using a palette of Δ colors. By the end of this phase we will be left with a graph whose maximum degree is at most $\kappa\Delta$ with high probability. In the

second phase the modified Luby's algorithm is used to color the remaining graph with at most $2\kappa\Delta$ fresh colors. As we shall see in section 4.1, the number of iterations needed to bring the degree down from Δ to $\kappa\Delta$ is $O(\log(1/\kappa)/\kappa^2)$. Hence, in order to get a $(1+\lambda)\Delta$, where $\lambda > 0$ is *any* fixed constant, the first phase takes constant time. To get a $(1+o(1))\Delta$ coloring takes $O((\log\Delta)^{2s}\log\log\Delta)$ time, where the $o(1)$ term is $1/(\log\Delta)^s$, for any $s > 0$. This holds with high probability. The exact probability of success will be determined in the section devoted to the analysis. We note here that an assumption on the maximum degree of the graph is needed, namely $\Delta(G) = \Omega(\log n)$ (n denotes the number of vertices of G). Phase 2 takes $O(\log n)$ time, with high probability.

The basic idea underlying the first phase of the algorithm is for each vertex to select a small "nibble" of edges incident upon it and assign tentative colors to them independently at random. Most of these edges are expected to avoid conflicts with other edges vying for coloring, and get successfully colored at this stage. This is because the nibble keeps the "efficiency" of the coloring close to 1 at each stage.

To describe the algorithm more precisely, we introduce some definitions that will also be used later in the analysis. At any stage $k \geq 1$, we have a graph $G_k(V, E_k)$. Initially, $G_0(V, E_0) := G(V, E)$, the input graph. By Δ_k we denote the maximum degree of the graph G_k (note $\Delta_0 = \Delta(G)$ initially). Each vertex has a palette of available colors, A_k^u with $A_0^u = [\Delta]$ initially.[2] The set of edges successfully colored at stage k is denoted by C_k. Then, $G_{k+1} := G_k[E - C_k]$ is the graph passed on to the next stage. In the algorithm, $t(\epsilon, \kappa)$ denotes the number of stages needed to bring the maximum degree of the graph from Δ to $\kappa\Delta$ with high probability, and has value

$$t(\epsilon, \kappa) = \left\lceil \frac{\ln(1/\kappa)}{\epsilon(1-\epsilon)e^{-4\epsilon}\kappa^2} \right\rceil.$$

The algorithm is more precisely described as follows

Phase 1. RODL NIBBLE

For $k = 1, 2, \ldots, t(\epsilon, \kappa)$ stages repeat the following:

- Each vertex u randomly selects an ϵ fraction of the edges incident on itself, and independently at random assigns them a tentative color from its palette A_k^u of currently available colors. If an edge $e = \{u, v\}$ is selected by both its endpoints, it is simply dropped and not considered for coloring at this stage.
- Let $e = \{u, v\}$ be a selected edge, and $c(e)$ its tentative color. Color $c(e)$ becomes the final color of e unless one of the following two conflict types arises: *i)* some edge incident on e is given the same tentative color, or *ii)* $c(e) \notin A_k^u \cap A_k^v$, *i.e.*, the tentative color given to e is not available at the other endpoint of e.
- The graph is updated by setting

$$A_{k+1}^u = A_k^u - \{c : \ e \text{ incident on } u,\ c(e) = c \text{ is the final color of } e\}$$

and $G_{k+1} = G_k[E_k - C_k]$, where C_k is the set of edges which got a final color at stage k.

Phase 2. Color $G_{t(\epsilon,\kappa)}$ with fresh new colors by using the modified Luby's algorithm.

[2]This assumption is not strictly necessary but it simplifies the analysis. It is sufficient to assume that $A_0^u = [\max_{w\in\delta(u)} \deg(w)]$.

4 Analysis

4.1 Intuitive Outline

Intuitively, the palettes A_k^u are more–or–less random subsets of the base set $[\Delta]$. Let us assume they are indeed truly random subsets of $[\Delta]$, so let us also assume that the palettes of each vertex is a uniformly and independently chosen random subset of $[\Delta]$ of the same size Δ_k at stage $k \geq 0$. Then, at stage k, the size of the common palette between any two vertices is Δ_k^2/Δ. So the probability that a colour chosen by a vertex as a tentative colour for an incident edge is also valid at the other end–point is Δ_k/Δ. Hence, the probability that an edge is successfully coloured at stage k is at least, roughly, $\epsilon\frac{\Delta_k}{\Delta}$ and we have the following recurrence for the vertex degree,

$$\Delta_{k+1} \leq (1 - \epsilon\frac{\Delta_k}{\Delta})\Delta_k \leq \exp(-\epsilon\frac{\Delta_k}{\Delta})\Delta_k$$

This recurrence implies that given a fixed $0 < \lambda < 1$, the vertex degree drops to $\lambda\Delta$ within a constant number of stages, or that for any positive integer $s > 0$, the degree drops to $\Delta/(\log \Delta)^s$ in a poly–logarithmic (in Δ) number of stages. This yields the required time complexity analysis for the algorithm.

Unfortunately, neither of the two assumptions above are in fact valid. First, because the graph G can have a very complex, irregular topology, it is not true that vertex degrees and palettes are uniform, at the outset, and they are even less likely to remain so at subsequent stages. In addition, the palettes are not truly independent random subsets either, as they can interact over the stages in a potentially complicated fashion governed by the topology of the graph. However, we show in § 4.3 below, that despite the possibility of a complex interaction in the graph, the "nibbling" feature of the colouring process leads to an essentially local interaction of the palettes. So, while the palettes are not quite truly random subsets, they behave essentially as such, specifically, with regard to the relative size and composition of the common palettes and the palettes themselves. *The crucial feature of the interaction is that the palettes of two neighbouring vertices are positively correlated*, that is, the probability of a colour being present in one palette conditional on its presence in the other palette is *no less* than the unconditional probability. Given this one simple, but crucial feature of the interaction of the palettes, we show in § 4.2 below, that the decay law is essentially as given above, however, we do not require to make any assumptions of uniformity in the graph.

4.2 Decay Rates with High Probability

Given vertices u, v of the graph at stage $k \geq 0$, G_k, and a colour $c \in [\Delta]$, denote

$$p_k^{u,c} := \Pr[c \in A_k^u]$$

and

$$q_k^{(u,v),c} := \Pr[c \in A_k^u \mid c \in A_k^v]$$

Now, we compute $\Pr[c \in A_{k+1}^u \mid c \in A_k^u]$ for a vertex $u \in G_{k+1}$. Assuming each vertex picks a ϵ–fraction "nibble" of neighbouring edges to which it assigns a tentative colour uniformly at random from its current palette, we have,

$$\Pr[c \notin A_{k+1}^u \mid c \in A_k^u] \quad = \quad \sum_{(u,w)\in E_k} \Pr[c_k(u,w) := c]$$

$$\geq \sum_{(u,w)\in E_k} \epsilon(1-\epsilon)e^{-4\epsilon}\left(\frac{q_k^{(w,u),c}}{\Delta_k^w} + \frac{q_k^{(w,u),c}}{\Delta_k^u}\right)$$

$$= \alpha \sum_{(u,w)\in E_k} \left(\frac{q_k^{(w,u),c}}{\Delta_k^w} + \frac{q_k^{(w,u),c}}{\Delta_k^u}\right) \quad (1)$$

where we introduce $\alpha := \epsilon(1-\epsilon)e^{-4\epsilon}$, and denote the event that an edge (u,w) is assigned a colour c at stage k by "$c_k(u,w) := c$". We used the fact that $\Pr[c \in A_k^w \mid c \in A_k^u] = q_k^{(w,u),c}$ in the second line. We also used the fact that the probability of a colour conflict at each endpoint of an edge is independently, very nearly $e^{-2\epsilon}$. We prove below in § 4.3 that *if the nibble is sufficiently small*,

- At every stage $k \geq 0$, for each $c \in [\Delta]$, for every two vertices u, v,

$$p_k^{u,c} \leq q_k^{(u,v),c} \quad (2)$$

That is, *the events "$c \in A_k^u$" and "$c \in A_k^w$" are positively correlated* for every two vertices u, v. Hence

$$q_k^{(w,u),c}/\Delta_k^w \geq p_k^{w,c}/\Delta_k^w \quad (3)$$

¿From symmetry, or the inductive proof in § 4.3, we also have that

- At any stage $k \geq 0$, for any vertex $u \in V_k$ and any two colours $c, c' \in A_0^u$,

$$p_k^{u,c} = \frac{\Delta_k^u}{\Delta_0^u} = p_k^{u,c'}. \quad (4)$$

Using equations 3 and 4 in equation 1, we get

$$\begin{aligned}
\Pr[c \notin A_{k+1}^u \mid c \in A_k^u] &\geq \alpha \sum_{(u,w)\in E_k} \frac{1}{\Delta_0^w} \\
&\geq \alpha \sum_{(u,w)\in E_k} \frac{1}{\Delta} \\
&= \alpha \frac{\Delta_k^u}{\Delta} \\
&= \alpha \frac{\Delta_0^u}{\Delta} p_k^u
\end{aligned}$$

(note $\Delta = \max_{u\in V_0} \Delta_0^u$ and also use $p_k^u = \frac{\Delta_k^u}{\Delta_0^u}$.) Hence we have the recurrence

$$p_{k+1}^{u,c} \leq (1 - \alpha\frac{\Delta_0^u}{\Delta} p_k^{u,c}) p_k^{u,c} \leq \exp(-\alpha\frac{\Delta_0^u}{\Delta} p_k^{u,c}) p_k^{u,c} \quad (5)$$

Given the above recurrence we can now determine the number of iterations needed to bring the degree down form Δ to $\kappa\Delta$. For a vertex $u \in V_0$ such that $\Delta_0^u/\Delta \geq \kappa$, we get the recurrence,

$$p_{k+1}^{u,c} \leq \exp(-\alpha\kappa p_k^{u,c}) p_k^{u,c}$$

Hence, for all such vertices $u \in V_k$ and for each colour $c \in [\Delta_0^u]$, we have

$$p_k^{u,c} \leq \eta_k$$

where η is defined via the recurrence

$$\begin{aligned} \eta_0 &:= 1, \\ \eta_{k+1} &:= e^{-\alpha\kappa\eta_k}\eta_k. \end{aligned}$$

Now, to get $p_k^{u,c} \leq \kappa$, it suffices to take $\eta_k < \kappa$. For the latter, in turn, it suffices to take $k > \frac{1}{\alpha\kappa^2}\log(\frac{1}{\kappa})$. For equation 2 to hold, we will see in the analysis below in § 4.3 that the nibble must be taken sufficiently small. Taking $\epsilon < 1/8$ at least, $\alpha = \alpha(\epsilon) \geq \frac{7}{16}\epsilon$, so we need $k > \frac{16}{7\epsilon\kappa^2}\ln(1/\kappa)$. These computations give us the expected number of stages by which the maximum degree is at most $\kappa\Delta$. We now argue that by stage k the maximum degree is around its expectation with high probability. In § 4.3, we will show that

- For any vertex $u \in V_k$, and any colours $c \in A_0^u$,

$$\Pr[c \in A_k^u \mid c' \in A_k^u, c'' \in A_k^u, \cdots] = \Pr[c \in A_k^u]$$

  for any other set of colours $c', c'', \cdots$.

that is, the events "$c \in A_k^u$", "$c' \in A_k^u$", $\cdots$ are *independent*. Hence, by an application of the Chernoff–Hoeffding bound, it follows that if we pick a k as above, then for any vertex $u \in G_k$ such that $\frac{\Delta_0^u}{\Delta} > \kappa$, and any $\delta > 0$, we have that,

$$\Pr[\Delta_k > (1+\delta)\kappa\Delta] < \exp(-\frac{\delta^2}{2}\kappa\Delta)$$

Of course, for the remaining vertices, namely those vertices u, for which $\frac{\Delta_0^u}{\Delta} \leq \kappa$, this continues to hold for each stage $k \geq 0$ with probability 1!

Hence, for any vertex, the degree drops to below $\kappa\Delta$ with at least the probability stated above. The probability that *all* vertex degrees drop to below $\kappa\Delta$ is then at least $1 - n \cdot \exp(-\frac{\delta^2}{2}\kappa\Delta)$ We have proved:

Theorem 2 *For any fixed $\lambda > 0$, given a graph with n vertices and maximum degee Δ, we can edge colour the graph with $(1+\lambda)\Delta$ colours in time $O(\log n)$ where n is the number of vertices in the graph. For any fixed positive integer s, we can edge colour it with $\Delta + \Delta/\log^s \Delta = (1+o(1))\Delta$ colours in time $O((\log\Delta)^{2s}\log\log\Delta + \log n)$. The results hold with failure probability decreasing to 0 faster than any polynomial (in n) provided that $\Delta = \Omega(\log^{1+d} n)$ for some $d > 0$.*

REMARK 1: One can improve the above probability estimate somewhat by using the Lovasz Local Lemma, [2]. However this does not allow us to remove the restriction that Δ grows with n as stated.

REMARK 2: It is unlikely that one can improve the above analysis to get a colouring better than the $\Delta + \Delta/(\log\Delta)^s$ bound above, while still retaining a poly–logarithmic running time (in n and Δ). One can show that for a wide class of non–decreasing functions, namely those functions g, that satisfy the condition that $g(n)/g(n+1)$ is nondecreasing, the least $k := k(n)$ such that $\eta_k \leq 1/g(n)$, is bounded below by the condition $k(n) = \Omega(g(n))$. So, if the shrinking of a vertex degree is governed by an equation of the form $\eta_{k+1} := e^{-\alpha\kappa\eta_k}\eta_k$ the number of iterations needed to bring the degree down to $\Delta/g(\Delta)$ is $k(\Delta) = \Omega(g(\Delta))$.

4.3 Pseudo–randomness in Nibbling

In this sub–section, we will prove the claims used in the analysis of the previous sub–section. For simplicity of the analysis, we will assume here that $\Delta_0^u = \Delta$ for all vertices u, initially [3]

We will prove that the palettes at the vertices, while not quite truly random subsets of $[\Delta]$ as assumed in § 4.1, are nevertheless *pseudo–random* in the precise sense specified by the following properties which we prove by induction on the stage $k \geq 0$:

1. **(Uniformity of Colours)**: For any vertex $u \in V_k$, and any colours $c, c' \in [\Delta]$,
$$p_k^{u,c} = \frac{\Delta_k^u}{\Delta} = p_k^{u,c'}.$$

2. **(Independence of Colours)**: For any $u \in V_k$ and any set of distinct colours $c, c', c'', \cdots \in [\Delta]$,
$$\Pr[c \in A_k^u \mid c' \in A_k^u, c'' \in A_k^u, \cdots] = \Pr[c \in A_k^u].$$

3. **(Independence of Colours)** For any two vertices $u, w \in V_k$, and any set of distinct colours $c, c', c'', \cdots \in [\Delta]$,
$$\Pr[c \in A_k^u \mid c' \in A_k^w, c'' \in A_k^w, \cdots] = \Pr[c \in A_k^u].$$

4. **(Positive Correlation of Palettes)**: For any two vertices u, v in the same connected component of G_k and any colour $c \in [\Delta]$,
$$q_k^{(u,v),c} \geq p_k^{u,c}.$$

The first two properties assert that a specific palette, by itself, is indeed a truly random subset of $[\Delta]$: the colours in it are *uniformly* and *independently* distributed. The third property asserts that this is also true for different colours in different palettes. All of these are intuitively plausible and in fact follow by reasons of symmetry, since the algorithm does not distinguish between colours, and also treats them independently of each other. Nonetheless, we will give a brief inductive verification of these statements at the end.

However, the key claim of this sub–section, and indeed of the whole analysis, is the fourth, namely that while the palettes of two different vertices are not necessarily independent, they are *positively correlated* for vertices in the same connected component, in that the probability of a colour being present in a given palette conditioned on its presence in a different palette is no less than the unconditional probability (when vertices are not in the same connected component we do not care). It is in this sense that the palettes are pseudo–random and not truly random as assumed in § 4.1. It is this claim that we set about to verify first.

The inductive assumption clearly holds for $k = 0$ where each $p_0^{u,c}$ and each $q_0^{(u,v),c}$ is 1. Then we assume simultaneously all the statements of the inductive hypothesis for some $k \geq 0$ and prove the corresponding statements for $k + 1$.

Let u, v be two vertices in the same connected component of G_{k+1}. So, there exists a shortest path in G_{k+1} (hence also in G_k),
$$u = x_0, x_1, \cdots, x_l = v$$
for some $l \geq 1$ and some distinct vertices $x_0, \cdots, x_l$. Note that since this is a shortest path, if in fact $(u, v) \in E_{k+1}$, then $l = 1$ and $x_1 = v$.

[3] This assumption is strictly not necessary for the claims in this section but it simplifies the exposition.

We start with some computations that give recurrences for the ps and qs. First, for any colour $c \in [\Delta]$,

$$\begin{aligned} p_{k+1}^{u,c} &:= \Pr[c \in A_{k+1}^u] \\ &= \Pr[c \in A_{k+1}^u \mid c \in A_k^u]\Pr[c \in A_k^u] \\ &= (1 - \Pr[c \notin A_{k+1}^u \mid c \in A_k^u])\Pr[c \in A_k^u] \\ &= (1 - \Pr[c \notin A_{k+1}^u \mid c \in A_k^u])p_k^{u,c} \end{aligned} \tag{6}$$

We will derive a similar recurrence for $q_k^{(u,v),c}$. First observe that if $\Pr[\bar{D} \mid A \wedge C], \Pr[\bar{D} \mid C], \Pr[\bar{B} \mid A \wedge C]$ are sufficiently small:[4]

$$\Pr[A \wedge B \mid C \wedge D] \approx \Pr[A \mid C](1 + (\Pr[\bar{D} \mid C] - \Pr[\bar{D} \mid A \wedge C]) - \Pr[\bar{B} \mid A \wedge C] + \Pr[\bar{B} \wedge \bar{D} \mid A \wedge C]).$$

To see this,

$$\begin{aligned} \Pr[AB|CD] &= \frac{\Pr[ABCD]}{\Pr[CD]} \\ &= \frac{\Pr[C]\Pr[A|C]\Pr[BD|AC]}{\Pr[C]\Pr[D|C]} \\ &= \Pr[A|C]\frac{1 - \Pr[\bar{B} \vee \bar{D}|AC]}{1 - \Pr[\bar{D}|C]} \\ &\approx \Pr[A|C](1 + \Pr[\bar{D}|C] - \Pr[[\bar{B} \vee \bar{D}|AC]) \\ &= \Pr[A|C](1 + \Pr[\bar{D}|C] - \Pr[[\bar{D}|AC] - \Pr[\bar{B}|AC] + \Pr[\bar{B}\bar{D}|AC]). \end{aligned}$$

Hence, with

$$A := c \in A_k^u, \bar{B} := c \notin A_{k+1}^u, C := c \in A_k^v, \bar{D} := c \notin A_{k+1}^v$$

we get for vertices u, v, colour c and stage $k \geq 0$,

$$q_{k+1}^{(u,v),c} \approx q_k^{(u,v),c}(1 + \delta) \tag{7}$$

where

$$\begin{aligned} \delta &= \Pr[c \notin A_{k+1}^v \mid c \in A_k^v] - \Pr[c \notin A_{k+1}^v \mid c \in A_k^u, c \in A_k^v] - \Pr[c \notin A_{k+1}^u \mid c \in A_k^u, c \in A_k^v] \\ &\quad + \Pr[c \notin A_{k+1}^u, c \notin A_{k+1}^v \mid c \in A_k^u, c \in A_k^v] \end{aligned} \tag{8}$$

¿From equations (6), (7) and (8), we see that using the induction hypothesis that $q_k^{(u,v),c} \geq p_k^{u,c}$, it suffices to prove that $(1 + \delta) \geq 1 - \Pr[c \notin A_{k+1}^u \mid c \in A_k^u]$. For this, it suffices to prove

$$\Pr[c \notin A_{k+1}^v \mid c \in A_k^v] \geq \Pr[c \notin A_{k+1}^v \mid c \in A_k^u, c \in A_k^v]$$

and, symmetrically,

[4] In the next sequence of equations leading upto equation (8), we use $\frac{1-x}{1-y} \approx 1 - (x - y)$, which holds for sufficiently small x and y. Whenever this approximation is in effect, we use $\approx$; this is consistent with our conventions in the preliminaries.

$$\Pr[c \notin A_{k+1}^u \mid c \in A_k^u] \geq \Pr[c \notin A_{k+1}^u \mid c \in A_k^u, c \in A_k^v]$$

(the term $\Pr[c \notin A_{k+1}^u, c \notin A_{k+1}^v \mid c \in A_k^u, c \in A_k^v]$ is ignored). Recalling that

$$\Pr[c \notin A_{k+1}^v \mid c \in A_k^v] = \sum_{w \in \delta(u)} \Pr[c_k(u,w) := c \mid c \in A_k^u],$$

it suffices to prove:

Lemma 1 *For any two vertices u, v connected by a shortest path*

$$u = x_0, x_1, \cdots, x_l = v$$

for a vertex $w \neq x_1$,

$$\Pr[c_k(u,w) := c \mid c \in A_k^u] \geq \Pr[c_k(u,w) := c \mid c \in A_k^u, c \in A_k^v],$$

and symmetrically, for a vertex $w \neq x_{l-1}$,

$$\Pr[c_k(v,w) := c \mid c \in A_k^v] \geq \Pr[c_k(v,w) := c \mid c \in A_k^u, c \in A_k^v].$$

PROOF. We sketch the proof of the statement for u, that for v being symmetrical. If $c \notin A_k^w$, then the statement holds as both sides are 0. So suppose $c \in A_k^w$. Denote the event that the edge (u,w) is assigned a tentative colour c at stage k by $t_k(u,w) := c$. Because each vertex chooses a colour uniformly and independently at random from its palette, we have

$$\Pr[t_k(u,w) := c \mid c \in A_k^u] = \epsilon(1-\epsilon)(\frac{1}{\Delta_k^u} + \frac{1}{\Delta_k^w}) = \Pr[t_k(u,w) := c \mid c \in A_k^u, A_k^v].$$

Hence the only possible change occurs in the probability of colour conflict at u and w. Now, suppose

$$u =: y_0, y_1, \cdots, y_l := w$$

is a shortest path in G_k. For each $y_i, y_{i+1}, i \geq 0$, we have, by the inductive hypothesis that $\Pr[c \in A_k^{y_{i+1}} \mid A_k^{y_i}] \geq \Pr[c \in A_k^{y_{i+1}}]$. Moreover this is the only manner in which the effect of conditioning at u can propagate to w. Hence $\Pr[c \in A_k^{y_{l-1}} \mid c \in A_k^u, A_k^v] \geq \Pr[c \in A_k^{y_{l-1}} \mid c \in A_k^v]$ which can only increases the conflict at w. Hence the conclusion of the lemma.

Finally, we give an inductive verification of the claim of the independence of different colours in palettes. In essence the reason is that each vertex treats different colours independently of each other, and that if two different vertices use two different colours, this interaction has no dependency either.

Lemma 2 *Let A, B, C, D be events such that*

1. *B and D are conditionally independent of $A \wedge C$, that is,*

$$\Pr[B \wedge D \mid A \wedge C] = \Pr[B \mid A \wedge C] \cdot \Pr[D \mid A \wedge C]$$

2. *B is conditionally independent of C and D of A, that is*

$$\Pr[B \mid A \wedge C] = \Pr[B \mid A], \textit{and } \Pr[D \mid A \wedge C] = \Pr[D \mid C].$$

Then

$$\Pr[A \wedge B \mid C \wedge D] = \Pr[A \mid C] \cdot \Pr[B \mid A].$$

PROOF. We have

$$\begin{aligned}
\Pr[A \wedge B \mid C \wedge D] &= \Pr[A \mid C] \cdot \frac{\Pr[B \wedge D \mid A \wedge C]}{\Pr[D \mid C]} \\
&= \Pr[A \mid C] \cdot \frac{\Pr[B \mid A \wedge C] \cdot \Pr[D \mid A \wedge C]}{\Pr[D \mid C]}, \text{ using (1)} \\
&= \Pr[A \mid C] \cdot \frac{\Pr[B \mid A] \cdot \Pr[D \mid C]}{\Pr[D \mid C]}, \text{ using (2)} \\
&= \Pr[A \mid C] \cdot \Pr[B \mid A].
\end{aligned}$$

For a vertex u, colours $c, c' \in [\Delta]$ and a stage $k \geq 0$, let

$$A := c \in A_k^u, B := c \in A_{k+1}^u, C := c' \in A_k^u, D := c' \in A_{k+1}^u.$$

Since a vertex chooses a tentative colour for a selected edge independently of the colours chosen at other edges, and using the inductive hypothesis of independence of different colours in different palettes, it can be verified by straightforward computations that the conditions of the lemma are satisfied. Applying the lemma and using the inductive hypothesis (in the second line) we have

$$\begin{aligned}
\Pr[c \in A_{k+1}^u \mid c' \in A_{k+1}^u] &= \Pr[c \in A_k^u \mid c' \in A_k^u] \cdot \Pr[c \in A_{k+1}^u \mid c \in A_k^u] \\
&= \Pr[c \in A_k^u] \cdot \Pr[c \in A_{k+1}^u \mid c \in A_k^u] \\
&= \Pr[c \in A_{k+1}^u].
\end{aligned}$$

Acknowledgments

We are indebted to Noga Alon and Joel Spencer for their suggestion that the Rödl nibble should work, and to Noga Alon for pointing out a mistake in a previous draft of this work and for several suggestions. We are grateful to Jirka Matousek who independently also suggested the use of the Rödl Nibble and who gave us a nice explanation of the swift working of the nibble. Kurt Melhorn provided, as usual, insightful remarks. We also thank Sem Borst and Marco Combe for useful discussions. The first author is grateful to the CWI and the Altec project for making possible an extended visit to CWI. The second author acknowledges the generous hospitality of MPI.

References

[1] N. Alon. Private Communication.

[2] N. Alon, J. Spencer, and P. Erdős. *The Probabilistic Method.* Wiley–Interscience Series, John Wiley & Sons, Inc., New York, 1992.

[3] B. Berger and J. Rompel. Simulating ($\log^c n$)-wise independence in NC. *J. Assoc. Comput. Mach.*, 38(4):1026–1046, 1991.

[4] B. Bollobás. *Graph Theory.* Springer Verlag, New York, 1979.

[5] Z. Galil and D. Leven. NP-completeness of finding the chromatic index of regular graphs. *J. of Algorithms*, 4:35–44, 1983.

[6] I. Holyer. The NP-completeness of edge coloring. *SIAM J. Comp.*, 10:718–720, 1981.

[7] R. Jain, K. Somalwar, J. Werth, and J. C. Browne. Scheduling parallel i/o operations in multiple bus systems. *Journal of Parallel and Distributed Computing*, 16(4):352–362, 1992.

[8] J. Kahn. Coloring nearly-disjoint hypergraphs with $n + o(n)$ colors. *J. Comb. Theory, Series A*, 59:31–39, 1992.

[9] H. J. Karloff and D. B. Shmoys. Efficient parallel algorithms for edge coloring problems. *Journal of Algorithms*, 8:39–52, 1987.

[10] R. M. Karp. Probabilistic recurrence relations. In *Proceedings of the ACM Symposium on Theory of Computing*, pages 190–197, 1991.

[11] M. Luby. Removing randomness in parallel computation without a processor penalty. In *Proceedings of the IEEE Symposium on Foundations of Computer Science*, pages 162–173, 1988. To appear in a special issue of *Journal of Computer and System Sciences*, devoted to FOCS 1988.

[12] N. A. Lynch. Upper bounds for static resource allocation in a distributed system. *Journal of Computer and System Sciences*, 23:254–278, 1981.

[13] R. Motwani, J. Naor, and M. Naor. The probabilistic method yields deterministic parallel algorithms. In *Proceedings of the IEEE Symposium on Foundations of Computer Science*, pages 8–13, 1989.

[14] A. Panconesi and A. Srinivasan. Fast randomized algorithms for distributed edge coloring. In *Proceedings of the ACM Symposium on Principles of Distributed Computing*, pages 251–262, 1992.

[15] A. Panconesi and A. Srinivasan. Improved distributed algorithms for coloring and network decomposition problems. In *Proceedings of the ACM Symposium on Theory of Computing*, pages 581–592, 1992.

[16] N. Pippinger and J. Spencer. Asymptotic behaviour of the chromatic index for hypergraphs. *J. Combinatorial Theory, Series A*, 51:24–42, 1989.

[17] V. Rödl. On a packing and covering problem. *European Journal of Combinatorics*, 5:69–78, 1985.

[18] J. Spencer. Asymptotically Good Coverings. *Pacific Journal of Mathematics*, 118(2):575–586, 1985.