New Linear-Time Algorithms for Edge-Coloring Planar Graphs

Richard Cole *   Łukasz Kowalik †

Abstract

We show efficient algorithms for edge-coloring planar graphs. Our main result is a linear-time algorithm for coloring planar graphs with maximum degree Δ with $\max\{\Delta, 9\}$ colors. Thus the coloring is optimal for graphs with maximum degree $\Delta \geq 9$. Moreover for $\Delta = 4, 5, 6$ we give linear-time algorithms that use $\Delta + 2$ colors. These results improve over the algorithms of Chrobak and Yung [1] and of Chrobak and Nishizeki [2] which color planar graphs using $\max\{\Delta, 19\}$ colors in linear time or using $\max\{\Delta, 9\}$ colors in $\mathcal{O}(n \log n)$ time.

Keywords. Edge-coloring, linear-time, algorithm, planar graph

1 Introduction

In the problem of edge-coloring the input is an undirected graph and the task is to assign colors to the edges so that edges with a common endpoint have different colors. This is one of the most natural graph coloring problems and arises in a variety of scheduling applications. Throughout the paper $\Delta(G)$ will denote the maximum degree in graph G; we write Δ for short when there is no ambiguity. Trivially at least Δ colors are needed to color the edges of any graph. Vizing [3] proved that $\Delta + 1$ colors always suffice. Unfortunately it is NP-complete even for cubic graphs to decide whether a

*Computer Science Department, New York University, 251 Mercer Street, New York, NY 10012, USA. cole@cs.nyu.edu. Supported in part by NSF grants CCR0105678 and CCF0515127.

†Institute of Informatics, Warsaw University, Banacha 2, 02-097, Warsaw, Poland and Max-Planck-Institute für Informatik, Stuhlsatzenhausweg 85, 66123 Saarbrücken, Germany. E-mail: kowalik@mimuw.edu.pl, phone: +49 681 9325 118, fax: +49 681 9325 199. Supported in part by KBN grant 4T11C04425.

given graph is Δ- or $(\Delta+1)$-colorable. We say that a graph is in Class 1 if it is Δ-colorable, and otherwise we say it is in Class 2.

Background Vizing's proof yields an $O(mn)$ time algorithm for $(\Delta+1)$-edge-coloring a graph with n vertices and m edges. It has been improved by Gabow *et al.* [4] to $O(\Delta m \log n)$ and $O(m(n \log n)^{1/2})$. However, it seems natural to look for more efficient or simpler algorithms for particular classes of graphs. For example, for the case of bipartite graphs there is an $O(m \log \Delta)$ algorithm due to Cole et al. [5] and a very simple $O(m \log m)$ algorithm due to Alon [6] (both algorithms use Δ colors). There is also a linear-time algorithm for 4-coloring cubic graphs due to Skulrattanakulchai [7].

Planar Graphs In this paper we investigate edge coloring of planar graphs. This problem is well studied. Vizing [8] showed that planar graphs with $\Delta \geq 8$ are in Class 1. He also noted that there are Class 2 planar graphs for $\Delta \in \{2,3,4,5\}$. Recently Sanders and Zhao [9] showed that planar graphs with $\Delta = 7$ are Δ-colorable. The $\Delta = 6$ case remains open. There are more cases to report when one considers algorithmic efficiency. The case $\Delta \leq 2$ is trivial. In 1878 Tait [10] showed that a cubic planar graph with no bridges is 3-edge-colorable if and only if it is 4-face colorable. It follows that optimally coloring graphs of maximum degree 3 is as hard as 4-coloring planar graphs for which the best algorithm known, due to Robertson *et al.* [11], takes $O(n^2)$ time. For $\Delta \in \{4,5\}$ Chrobak and Nishizeki [2] state that it has been conjectured that the problem of Δ-coloring is NP-hard. For $\Delta \geq 7$ we can color graphs with Δ colors. When $\Delta = 7$ we believe as does Sanders [12] that the proof of Sanders and Zhao could be turned into a polynomial time algorithm. When $\Delta \geq 8$ one can use the $O(n^2)$ algorithm of Gabow *et al.* [4]. For any $\Delta \geq 9$ the complexity decreases to $O(n \log n)$ due to Chrobak and Nishizeki [2]. Finally for $\Delta \geq 19$ Chrobak and Yung [1] gave an $O(n)$ algorithm. There is also an $O(n)$ algorithm for $\Delta \geq 33$ by He [13].

Our results Our main result is a linear-time algorithm for coloring planar graphs with maximum degree Δ with $\max\{\Delta, 9\}$ colors. Thus the coloring is optimal for graphs with maximum degree $\Delta \geq 9$. Moreover for $\Delta = 4,5,6$ we show linear-time algorithms that use $\Delta+2$ colors. Our results are presented in Table 1.

Our Approach Our approach combines two ideas. The first is the notion of reductions, which allow a suitable edge to be removed from a given planar graph, where the edge is chosen so that it can be colored in $O(1)$

Δ	number of colors	time	paper
2	optimal	$\mathcal{O}(n)$	easy
3	optimal	$\mathcal{O}(n^2)$	Robertson *et al.* [11]
3	$\Delta + 1$	$\mathcal{O}(n)$	Skulrattanakulchai [7]
$4, 5, 6, 7$	$\Delta + 1$	$\mathcal{O}(n \log n)$	Gabow *et al.* [4]
$4, 5, 6, 7$	$\Delta + 2$	$\mathcal{O}(n)$	This work
7	Δ	polynomial	Sanders and Zhao [9]
8	Δ	$\mathcal{O}(n^2)$	Gabow *et al.* [4]
8	$\Delta + 1$	$\mathcal{O}(n)$	This work
≥ 9	Δ	$\mathcal{O}(n)$	This work
≥ 19	Δ	$\mathcal{O}(n)$	Chrobak and Yung [1]

Table 1: Currently most efficient algorithms for edge-coloring planar graphs.

time following a recursive coloring of the reduced graph, possibly with some recoloring of the rest of the graph. The reductions are identified by means of a collection of configurations, constant size subgraphs, one of which is always present in a planar graph. The challenge is to identify configurations and to provide the corresponding constant time recoloring procedures. We illustrate this technique in Section 2, which gives a simple algorithm to color planar graphs using $\max\{\Delta, 12\}$ colors.

The second issue is to show that the collection of configurations suffices. This is done by means of discharging arguments. A charge is distributed to the faces and vertices of the graph, with negative total value. The charge is then redistributed in such a way that if none of the configurations are present, every face and vertex would have a nonnegative charge, a contradiction. The challenge is to find a suitable collection of configurations and the corresponding discharging argument. This technique is needed for all but the Section 2 algorithm.

The discharging technique as well as the idea of coloring planar graphs by providing a set of reducible configurations were originally developed to prove the Four-Color Theorem. While applying this approach for finding an efficient edge-coloring algorithm seems to be natural, the devising of suitable collections of configurations, the coloring procedures, and the discharging arguments is non-trivial.

Terminology We assume the reader is familiar with standard terminology and notation concerning graph theory and planar graphs in particular. Let us recall here some notions that are not so widely used. Let f be a face of a connected plane graph. A *facial walk* w corresponding to f is the shortest closed walk induced by all edges incident on f. Let $|w|$ denote the length of walk w and let $|f|$ denote the length of face f; we note that $|f| = |w|$. A k-path (k-cycle, k-face) refers to a path (cycle, face) of length k. Analogously, d-vertex refers to a vertex of degree d, and D-graph to a graph of maximum degree D. Let G be a graph and let $S \subseteq E(G)$ be a set of edges. Then $G-S$ denotes the graph $(V(G), E(G) - S)$.

Consider a partial coloring of edges of graph $G = (V, E)$. We say that color a is *free* at vertex x when there is no edge colored a incident on x. We say that color a is *free* at edge uv when it is free at both u and v. We say that color a is *used* by vertex x (resp. edge uv) when it is not free at x (resp. uv).

2 A Simple Algorithm

In this section we present our approach via a simple algorithm which colors planar graphs using $\max\{\Delta, 12\}$ colors. Let the *weight* of edge $e = uv$, denoted by $w(e)$, be the sum of the degrees of its ends, i.e. $w(e) = \deg_G(u) + \deg_G(v)$. We are inspired by the following result due to Borodin.

Theorem 2.1 (Borodin [14]). *Any simple planar graph with vertices of degree at least 3 contains an edge of weight at most* 13.

We need a slightly generalized version of the above theorem.

Theorem 2.2. *Let G be a simple planar graph with maximum degree Δ such that G contains no vertices of degree 0 or 1, each vertex of degree 2 is adjacent to two vertices of degree Δ, and each vertex of degree Δ is adjacent to at most one degree 2 vertex. Then G contains an edge of weight at most 13.*

Proof. We can assume that G contains at least one degree 2 vertex for otherwise we just apply Theorem 2.1. Further we can assume that $\Delta \geq 12$ for otherwise any edge incident on a 2-vertex has weight at most 13. Now consider graph G' obtained from G by replacing each path uxv such that $\deg(x) = 2$ by an edge joining u and v. Additionally, we replace double edges by single ones. Clearly G' is a simple planar graph with vertices of degree at least 3; further a vertex of degree d in G, $3 \leq d \leq \Delta - 1$, has degree d in G', while vertices of degree Δ in G may have degree $\Delta - 1$ or Δ in G'. By Theorem 2.1 G' contains an edge of weight at most 13. Consider any such edge e. Then each of e's endpoints has degree at most $10 \leq \Delta - 2$

in G' (as each endpoint has degree at least 3) and so each of e's endpoints has the same degree in G, i.e. e has weight at most 13 in G. $\square$

Clearly, edges of bounded weight are very useful in edge-coloring algorithms. Assume we want to color a graph using D colors (this notation will be used throughout the paper). When our algorithm finds an edge e of weight at most $D+1$ this edge is removed and the resulting graph is colored recursively. Since there are at most $D-1$ edges incident on e, these edges do not use all the colors and e can be colored with one of the remaining colors.

In the coloring algorithm we describe in this section we will use the following three types of edges of weight at most $D+1$ (recall that $D = \max\{\Delta, 12\}$), which will be called *reducible*:

- edges of weight 13,
- edges incident on a 1-vertex, and
- edges incident on a 2-vertex and a vertex of degree at most $\Delta - 1$.

By Theorem 2.2 to complete our algorithm we need to describe what to do when there are no reducible edges in the graph. Then, by Theorem 2.2, G contains a Δ-vertex v with two adjacent 2-vertices. Let us denote these 2-vertices by x and y. Let u and w be the other neighbors of x and y. Obviously, $\deg(u) = \deg(w) = \Delta$, for otherwise there is a reducible edge. There are two cases to consider: $u = w$ and $u \neq w$. In the first situation $uxvy$ is a cycle of length 4 with vertex degrees $\Delta, 2, \Delta, 2$, respectively, which we name configuration (A). In the second case $uxvyw$ is a 4-path with vertex degrees $\Delta, 2, \Delta, 2, \Delta$, respectively; we name this configuration (B).

Our algorithm handles configuration (A) as follows. It first removes edge xv and colors the remaining graph recursively. Let a denote the color of ux. If a is the color free at v the algorithm swaps the colors of ux and uy and colors xv with a. Otherwise it simply colors xv with the color free at v.

We call the above reasoning a *reduction*; we say that configuration (A) is *reducible*. More formally:

Definition. A configuration F is called D*-reducible* if, for every planar graph G which contains F, there is a corresponding configuration F' and a graph G' containing F' such that

(i) $|E(F')| < |E(F)|$,

(ii) $G - E(F)$ is isomorphic to $G' - E(F')$,

(iii) there is an algorithm which given any D-edge-coloring of G' finds a D-edge-coloring of G as an extension of the coloring of $G' - E(F')$.

As it is usually clear what is the number D of available colors, we will write that a configuration is reducible, instead of D-reducible. Reducibility of all configurations in this paper is proved in the following way. We consider a configuration F in graph G. We remove a suitable edge from F or in some other way reduce the number of edges in G. Then we color the resulting graph and show how to extend this coloring to the whole graph G (possibly recoloring some edges of F). To complete our description of the algorithm for coloring with $\max\{\Delta, 12\}$ colors it remains to show the following lemma.

Lemma 2.3. *Configuration* (B) *is reducible.*

Proof. Recall that u, x, v, y, w denote the successive vertices of the path in (B), where $\deg_G(x) = \deg_G(y) = 2$ and $\deg_G(u) = \deg_G(v) = \deg_G(w) = \Delta$.

Case 1. G contains neither uv nor vw. Then form graph G' by replacing the paths uxv and vyw by edges uv, vw respectively. Now color G'. Let a and b be the colors of uv and vw, respectively. G is colored as follows. Edges ux and vy are colored using a, and edges xv and yw using b; the remaining edges of G inherit the colors assigned in G'.

Case 2. G contains edge uv (or analogously, G contains vw). Edge vy is removed and the remaining graph is colored recursively. Then G is colored as follows. Let a denote the color of yw. We can assume that a is the free color at v for otherwise vy is simply colored with the non-a free color.

Case 2.1. ux is not colored with a. vy receives xv's color and xv is colored with a.

Case 2.2. ux is colored with a. vy receives uv's color and the colors of uv and ux are swapped. □

2.1 Implementation and Time Complexity

It remains to describe how to implement our algorithm efficiently.

All currently reducible edges are kept in queue Q_e, all current instances of configuration (A) in queue Q_A and all current instances of configuration (B) in queue Q_B. Our algorithm is recursive and works as follows.

Step 1. (i) Remove a reducible edge from Q_e, if any, and remove it from the graph G.

(ii) Otherwise remove an instance of configuration (A) from Q_A, if any. Remove the corresponding edge xv from G – see the discussion of how to handle configuration (A).

(iii) Otherwise remove an instance of configuration (B) from Q_B (there must me one).

  (a) Check if each of the relevant two pairs of Δ-vertices in the configuration are adjacent. In Section 2.1.2 we show that it takes $O(n)$ time over all such adjacency tests.

  (b) Depending on the adjacency tests, replace paths uxv and vyw by edges uv and vw, or remove edge uv or vw – see the proof of Lemma 2.3.

Step 2. Update Q_e, Q_A and Q_B to take account of the changes in G. We explain how this is done in Section 2.1.1.

Step 3. The recursive call.

Step 4. The edges or paths removed in Step 1 are reinserted into the graph, which takes constant time. The reinserted edges are colored, possibly along with recoloring some of the $O(1)$ edges in configuration (A) or (B) if this is the case being handled. How to do this in constant time is explained in Section 2.1.3.

2.1.1 Finding reducible edges and configurations

Finding reducible edges and configurations (A) and (B) can be done fast after linear-time preprocessing. To this end each vertex stores its current degree; also, a queue Q_e of reducible edges is kept. This information can easily be maintained in linear time over the course of the algorithm. Additionally, instances of configurations (A) and (B) are stored in two corresponding queues, $Q_{(A)}$ and $Q_{(B)}$, which are initialized in linear time with a maximal collection of edge-disjoint configurations.

A 2-vertex adjacent to two degree Δ vertices is called *extremal*. Observe that any degree Δ vertex, which is not a part of configuration (A) or (B), can be adjacent to at most one extremal 2-vertex. To enable fast update of queues $Q_{(A)}$ and $Q_{(B)}$ after an edge removal, each degree Δ vertex which is not part of a configuration (A) or (B) stores its sole neighboring extremal 2-vertex, if any.

Our algorithm performs two operations which modify the input graph. The first operation is replacing a 2-path with an edge (which occurs only if $\Delta \geq 12$). As the ends of the 2-path have degree Δ there is no queue to update. The second operation is removing an edge. Then the degrees of its ends are reduced and some queues may require updating, as follows.

If one of the degree Δ vertices in a configuration (A) or (B) instance is reduced to degree $\Delta-1$ then this is no longer an instance of configuration (A) or (B). In such situation the configuration is removed from the relevant queue in constant time (we assume that every edge that is part of a configuration stores a pointer to the corresponding entry in relevant queue).

Whenever the degree of a vertex is reduced to 1, the edge incident on this vertex is added to the queue Q_e. Each time the degree of a vertex is reduced to 2, any incident edges containing a vertex of degree $\Delta - 1$ are added to the queue Q_e. However, when both neighbors have degree Δ the 2-vertex is extreme and hence either an instance of a configuration is found (and stored in queue $Q_{(A)}$ or $Q_{(B)}$) or the degree 2 vertex is stored with each of its two neighbors. Similarly, whenever the degree of some vertex is reduced to 11 or less, incident edges which now have weight at most 13 are added to the queue Q_e. Each of these updates takes $O(1)$ worst-case time. There is one more update to describe. Each time the degree of some Δ-vertex is reduced to $\Delta - 1$ the algorithm adds to Q_e all the incident edges whose other endpoint has degree 2. Although one such update takes $O(\Delta)$ time, altogether they take $O(m)$ time, since there are only $O(m/\Delta)$ vertices originally of degree Δ in G.

2.1.2 Checking Adjacency of Δ-vertices

As we mentioned before, each time a configuration (B) is taken from queue $Q_{(B)}$ two pairs of Δ-vertices are tested for adjacency. Each test takes $\mathcal{O}(\Delta)$ time. Fortunately, Lemma 2.4 shows that the algorithm finds only $\mathcal{O}(n/\Delta)$ instances of configuration (B) and hence all the adjacency tests take only $O(n)$ time.

Lemma 2.4. *Our algorithm finds $\mathcal{O}(n/\Delta)$ instances of configuration* (B)*.*

Proof. It suffices to show that during the execution of the algorithm there are $\mathcal{O}(n/\Delta)$ degree 2 vertices adjacent to two degree Δ neighbors. Let n_Δ denote the number of degree Δ vertices in the initial graph G. Since $2|E(G)| = \sum_v \deg(v) \geq \Delta n_\Delta$ and $|E(G)| \leq 3n$ we see that $n_\Delta = \mathcal{O}(n/\Delta)$. Let $I^* = (V^*, E^*)$ be a graph such that V^* contains all degree Δ vertices of graph G. Two vertices u and v are adjacent in I^* if and only if at some moment of execution of the coloring algorithm they have a common neighbor of degree 2 in G. Observe that our goal is to show that $|E^*| = \mathcal{O}(|V^*|)$.

By the Four-Color Theorem the vertices of every planar graph can be colored using 4 colors in such a way that the ends of each edge are colored differently. Let us take an arbitrary 4-vertex-coloring of graph G. Then I^* can be partitioned into four edge-disjoint subgraphs: $I_1^*, I_2^*, I_3^*, I_4^*$ so that for each $j = 1, 2, 3, 4$ graph I_j^* contains edge uv when $uv \in E(I^*)$ and uv corresponds to a 2-path uxv in G with x colored j (if the edge corresponds to multiple 2-paths we choose one of them arbitrarily). Clearly, for each j, replacing each edge in I_j^* by a 2-path yields a subgraph of the initial graph G (observe that this need not be true for graph I^*). Thus the graphs I_j^* are planar and each of them has at most $3|V^*|$ edges so $|E^*| \leq 12|V^*| = \mathcal{O}(|V^*|)$. $\square$

2.1.3 Finding Free Colors

Here we describe how the algorithm finds a free color in constant time. For each vertex v we maintain a list $FreeSub(v)$ of the colors free at v among the colors $\{1, \ldots, \min\{\deg_G(v) + 1, \Delta\}\}$ – observe that as long as v has an uncolored incident edge this set always contains at least *one* color free at v and when $\deg_G(v) < \Delta$ at least *two* such colors. It follows that the lists $FreeSub(v)$ are sufficient for finding free colors when reducing configurations (A) and (B) or an edge incident on a 1- or 2-vertex. On the other hand, when the algorithm reduces an edge uv of weight at most 13 it may happen that lists $FreeSub(v)$ and $FreeSub(u)$ do not store a common color. Fortunately, in this case the algorithm can simply check the colors of all incident edges and find an unused color in constant time.

Additionally, each vertex v stores an array $Colors[1, \ldots, \deg_G(v)]$ where $Colors[c]$ is a pointer to color c in list $FreeSub(v)$. Observe that the initialization of lists $FreeSub$ and arrays $Colors$ in the preprocessing phase takes $O(\sum_v \deg(v)) = O(m) = O(n)$ time. Arrays $Colors$ are used to maintain lists $FreeSub(\cdot)$. More precisely, assume that the algorithm colors an edge uv with a certain color c. Then it verifies whether c is not greater than the degree of u in the initial graph (it proceeds similarly for the other endpoint v). If so, pointer $Colors[c]$ stored with u is used to remove c from $FreeSub(v)$ in constant time. Analogous changes occur when an edge is recolored (which can happen when handling configuration (A) or (B)).

Corollary 2.5. *Our algorithm colors every planar graph with maximum degree* Δ *using* $\max\{\Delta, 12\}$ *colors in linear time.*

3 General approach

The approach presented in Section 2 is used in all our coloring algorithms. In order to describe a D-coloring algorithm we need to:

- specify a set of configurations,
- show that the configurations are D-reducible,
- show unavoidability, i.e. that every planar graph contains a configuration.

Since the algorithm from Section 2 is optimal for graphs of degree at least 12, we can assume that the other algorithms are applied only to bounded degree graphs. Since all the configurations in this paper have bounded size it is straightforward to find configurations in constant time. To this end we maintain a queue which stores edge-disjoint configurations. Let d be the largest of the configuration diameters. Whenever the degree of any vertex v is changed the algorithm searches for configurations in the subgraph induced by the vertices at distance at most d from v. Since all vertices in the input graph have bounded degrees this subgraph has bounded size and the search takes only constant time. Each new configuration is added to the queue.

In the sequel we show a number of reducibility proofs. It can be easily verified that each of these proofs can be transformed into an algorithm which consists of bounded number of operations such as edge deletion, edge insertion, finding a free color and assigning a color. Each of these operations can easily be implemented to work in constant time provided that the input graph has bounded degree. Thus for bounded degree input graphs our configurations are reducible in constant time.

Hence we claim that the algorithms we present in the following sections work in linear time for bounded degree graphs.

4 Coloring Low Degree Graphs

4.1 6-Coloring Graphs with Maximum Degree $\Delta = 4$

As before, D denotes the number of available colors; here $D = 6$. As before, the basic reducible configuration is an edge of weight at most 7. Configuration (C_k) denotes a triangle with vertices of degrees at most $D - 1, 2 + k, D - k$ for any $k \geq 1$; it is reducible.

Lemma 4.1. *Configuration* (C_k) *is D-reducible for any $k \geq 1$ and $D-k \geq 2$.*

Proof. Let us denote the vertices of the triangle by x, y, w so that $\deg_G(x) \leq 2 + k$, $\deg_G(y) \leq D - k$ and $\deg_G(w) \leq D - 1$. We remove edge xy and color the remaining graph recursively. Now we show how to color G. Let a and b be the colors of xw and wy, respectively. Observe that x has at most $k+1$ used colors and y has at least $k+1$ free colors. We can assume that the colors used by x are free at y, for otherwise there is a free color at both x and y which can be used to color xy. There are two cases to consider. If one of the colors free at y is also free at w, then we color wy with this free color and xy with b. Otherwise, every free color at w is not free at y, and hence free at x; then we color wx with a color free at w and xy with a. $\square$

Clearly for $D = 6$ and $\Delta = 4$ any triangle is in configuration (C_2). In this case unavoidability is easy to show, as follows.

Theorem 4.2. *Any planar graph with maximum degree* 4 *contains a triangle or an edge of weight at most* 7.

Proof. Let $G = (V, E)$ be a planar graph with maximum degree 4. Assume that both ends of each edge in G have degree 4. It follows that G is 4-regular. Then $2|E| = \sum_v \deg(v) = 4|V|$; thus $|E| = 2|V|$. Now assume that G contains no triangles. Let F be the set of faces of graph G. Then $2|E| = \sum_{g \in F} |g| \geq 4 \cdot |F|$. Substituting into Euler's Formula $|V| - |E| + |F| = 2$ yields $\frac{|E|}{2} - |E| + \frac{|E|}{2} \geq 2$, which is a contradiction. $\square$

4.2 7-Coloring Graphs with Maximum Degree $\Delta = 5$

Now, $D = 7$. By configuration (P) we mean two triangles, xuy and zuy, sharing a common edge, with $\deg(u) \leq 5$, $\deg(x), \deg(y), \deg(z) \leq D - 2$.

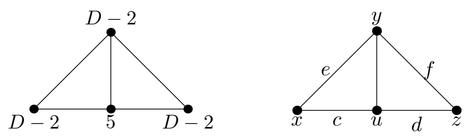

Figure 1: Configuration (P). Labels in the left picture denote upper bounds on degrees.

Lemma 4.3. *Configuration* (P) *is reducible.*

Proof. We name the vertices of the configuration as shown in Fig. 1. Assume that $\deg_G(u) = 5$ and $\deg_G(x) = \deg_G(y) = \deg_G(z) = D-2$ (if the degrees are smaller reducing is easier). We remove edge uy and color the remaining graph recursively. Now we show how to color G. We denote the edge colors as in Fig. 1. Additionally, let a and b be the colors of the two other edges incident on u. We can assume that $Free(y) \subseteq \{a, b, c, d\}$ for otherwise we simply color uy with a free color. We can also assume that both c and a are free in y (by symmetry). Next, we assume that $Free(x) \subseteq \{a, b, d\}$ for otherwise we can color uy with c and xu with a free color. Hence at least one of a, b is free at x.

Case 1. $e = b$. Then $Free(y) = \{a, c, d\}$ and $Free(x) = \{a, d\}$. We color yz and xu with d, uz with f and uy with c.

Case 2. $a \in Free(x)$ and $e \notin \{b, d\}$. Then color uy with e and xy with a.

Case 3. $a \in Free(x)$ and $e = d$. Then $Free(x) = \{a, b\}$ and $Free(y) = \{a, b, c\}$. If $Free(z)$ contains a or b or c color yz with this color and yu with f. Otherwise $Free(z)$ contains color h, $h \notin \{a, b, c\}$. Then color uz with h, uy with d and xy with a.

Case 4. $Free(x) = \{b, d\}$. Then b or d is free in xy and $e \neq d$. Color uy with e and xy with a free color. □

Besides configuration (P) we use also reducible edges (i.e. edges of weight at most $D+1$) and configuration (C_2), described before. The unavoidability proof is particularly interesting in this case, because it provides a gentle introduction to the next section, which gives the main result of this paper.

Lemma 4.4. *Any planar graph with maximum degree* 5 *contains an edge* uv *of weight at most 8 or one of configurations* (C_2) *or* (P) *for* $D = 7$.

Proof. We use the discharging technique. Let G be a planar graph with $\Delta(G) = 5$. Each vertex v of G receives a *charge* of $\deg_G(v) - 4$ units and each face q of G receives a charge of $|q| - 4$ units. Let n, m, f denote the number of vertices, edges and faces of graph G, respectively, and let V and F be the sets of vertices and faces of G, respectively. Using Euler's formula we can easily bound the total charge on G:

$$\sum_{v \in V}(\deg_G(v) - 4) + \sum_{q \in F}(|q| - 4) = 2m - 4n + 2m - 4f = -8 < 0.$$

We assume for a contradiction that G contains no edge of weight ≤ 8 and neither of the configurations (C_2), (P). Now we move charges in graph G so that it will be clear that the total charge in G is nonnegative, which is a contradiction. Specifically, degree 5 vertices send $\frac{1}{3}$ of a unit of charge to each incident triangle. We can assume that there are no triangles with a vertex of degree 4 for such a triangle would be a (C_2). Hence triangles end up with nonnegative charges. The other faces do not alter their allocated charge, which was already nonnegative. Clearly there are no vertices of degree lower than 4 since each such vertex would be an endpoint of an edge of weight at most 8. Degree 4 vertices do not alter their charge which was 0. Since configuration (P) is excluded, each 5-vertex is incident on only two triangles so it ends up with at least $\frac{1}{3}$ of a unit of charge. It follows that the total charge is nonnegative – a contradiction. □

4.3 8-Coloring Graphs with Maximum Degree $\boldsymbol{\Delta} = 6$

This case is similar to the preceding one. We use the same configurations and also (C_3). Below we give the unavoidability proof.

Lemma 4.5. *Any planar graph with maximum degree* 6 *contains an edge of weight at most 9 or one of configurations* (C_2)*,* (C_3) *or* (P) *for* $D = 8$.

Proof. We use discharging again and assign initial charges as before. Then degree 5 vertices send $\frac{1}{5}$ of a unit to each incident triangle. Degree 6 vertices send $\frac{2}{5}$ of a unit to each incident triangle with at least one 5-vertex and $\frac{1}{3}$ of a unit to every other incident triangle. We can assume that there are no triangles with a 4-vertex for such a triangle would be in configuration (C_2). Similarly, using (C_3), we note that there are no triangles with two 5-vertices. The other triangles have degrees sequence $5, 6, 6$ or $6, 6, 6$ and in both cases end up with nonnegative charges. Nontriangular faces do not change their allocated charge, which was nonnegative. Clearly there are no vertices of degree lower than 4 since each such vertex would be an endpoint of an edge of weight at most 9. Degree 4 vertices do not change their allocated charge which was 0. Degree 5 vertices send at most $5 \cdot \frac{1}{5} = 1$ unit and retain nonnegative charge.

Finally consider a 6-vertex v. It starts with 2 units of charge. We will show that it sends at most 2 units. First, suppose that v is incident on a nontriangular face. It follows that it sends charge to at most 5 triangles. Hence it sends at most $5 \cdot 2/5 = 2$ units as required. Now, suppose that v is incident on 6 triangles. Then v is not adjacent to a 5-vertex for otherwise there would be two incident triangles sharing a 5-vertex, i.e. configuration (P). Hence v is incident only on triangles with no 5-vertex. v gives each such triangle at most $1/3$ of a unit, so v sends at most $6 \cdot 1/3 = 2$ units as required.

It follows that the total charge is nonnegative – a contradiction. □

5 Coloring With $\max\{\Delta, 9\}$ Colors

In this section we show the main result of the paper – a linear time algorithm for coloring planar graphs with $D = \max\{\Delta, 9\}$ colors. Although it can be implemented to work in linear time for any planar graph, to simplify the presentation we use the algorithm from Section 2 for coloring graphs with maximum degree at least 12; thus we can assume that $\Delta \leq 11$ and hence it suffices to describe a set of reducible configurations and show that any planar graph contains one of them. Let us note that one can also use the algorithm of Chrobak and Yung [1] for coloring graphs with maximum degree at least 19 and the algorithm arising from this section for graphs with maximum degree from 9 to 18.

5.1 Discharging

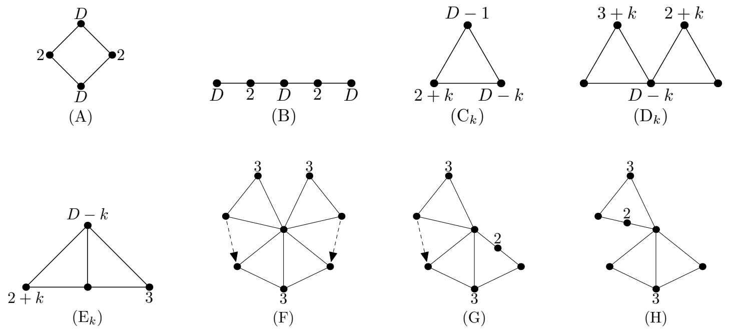

Figure 2: Reducible configurations. Labels denote upper bounds on vertices' degrees. No label denotes any degree. A dashed arrow means that that the designated pair of vertices may be a single vertex.

A *critical path* is a path abc such that $\deg(b) = 2$, $\deg(a) = \deg(c) = D$, and there is no edge joining a and c.

Theorem 5.1. *Let G be a simple planar Δ-graph and let $D = \max\{\Delta, 9\}$. Suppose that G contains neither a critical path nor any of configurations* (A) - (F)*. Then G contains a reducible edge, i.e., an edge of weight at most $D + 1$.*

Proof. We use discharging again and we assign charges in exactly the same way as in the proof of Lemma 4.4; here too the total charge is negative. We assume for a contradiction that G contains no reducible edge. Thus there is no edge with both endpoints of degree at most 5, and consequently each triangle has at most one vertex with degree 5 or less. Our goal is to move charges in graph G (using so-called *discharging rules*) in such a way that it will be clear that the total charge in G is nonnegative, which is a contradiction. Now we specify the discharging rules.

Rule 1. Each 5-vertex sends $\frac{1}{5}$ of a unit of charge to each incident triangle.

Rule 2. Each 6-vertex sends $\frac{1}{3}$ of a unit to each incident triangle.

Rule 3a. Each 7-vertex sends $\frac{1}{2}$ of a unit to each incident triangle containing a vertex of degree at most 4.

Rule 3b. Each 7-vertex sends $\frac{2}{5}$ of a unit to each incident triangle with all vertices of degree at least 5.

Rule 4a. Each vertex of degree at least 8 sends $\frac{1}{2}$ of a unit to each incident triangle containing a vertex of degree at most 5.

Rule 4b. Each vertex of degree at least 8 sends $\frac{1}{3}$ of a unit to each incident triangle with all vertices of degree at least 6.

Rule 5. Each vertex of degree at least $D-1$ sends $\frac{1}{3}$ of a unit of charge to each incident 3-vertex.

Rule 6. Each vertex of degree D sends 1 unit of charge to each incident 2-vertex.

A triangle with a vertex of degree at most 4 has two vertices of degree at least $D-2 \geq 7$, hence it gets 1 unit of charge by Rules 3a and 4a so it ends up with charge 0. Now consider a triangle with a 5-vertex. If the triangle contains a vertex x of degree at least 8 it receives $\frac{1}{5}$ from the 5-vertex by Rule 1, $\frac{1}{2}$ from x by Rule 4a and at least $\frac{1}{3}$ from the remaining vertex which is of degree at least 6 (see Rules 2, 3b, 4a and 4b). Hence triangles with a 5-vertex and a vertex of degree at least 8 have positive final charge. Triangles with a 5-vertex and with both remaining vertices of degree at most 7 do not contain a vertex of degree 6, as (C_3) is excluded. Hence each such triangle receives $\frac{1}{5} + 2 \cdot \frac{2}{5} = 1$ unit of charge by Rules 1 and 3b. Finally, a triangle with all vertices of degree at least 6 receives at least 1 unit of charge by Rules 2, 3b and 4b. Nontriangular faces do not alter their charge, which stays nonnegative. Thus all faces end up with nonnegative charge.

Let $c^*(v)$ denote the final charge at vertex v. Now we will show that for each vertex v, $c^*(v) \geq 0$. We can assume that there are no 1-vertices, since this would imply the existence of a reducible edge. Each 2-vertex is adjacent to two D-vertices so it receives 2 units by rule 6 and $c^*(v) = 0$. Similarly, each 3-vertex ends up with charge 0 by rule 5. Vertices of degree 4 do not alter their charge, which was 0. 5- and 6-vertices retain nonnegative charge. It suffices to examine vertices of degree at least 7.

Let v be a vertex of degree d, $7 \leq d \leq D-2$. Since configuration (D_2) is excluded, v is incident on at most two triangles with a 4-vertex. Then by

Rules 3a and 3b, $c^*(v) \geq d-4-2\cdot\frac{1}{2}-(d-2)\cdot\frac{2}{5}=\frac{3d-21}{5}\geq 0$.

Now let v be a vertex of degree $D-1$. Before moving charges, v had $D-5$ units of charge. We will show that it always sends at most $D-5$ units. As there are no reducible edges v is adjacent only to vertices of degree at least 3. First consider the situation where no 3-vertex is a part of a triangle containing v. Let k_3 be the number of 3-neighbors of v. Then $c^*(v) \geq D-1-4-k_3\cdot\frac{1}{3}-(D-1-k_3)\cdot\frac{1}{2}=\frac{D-9}{2}+\frac{k_3}{6}\geq 0$. Now assume that there is a triangle containing both a certain 3-vertex, x say, and v. Consider the k_3-1 degree 3 neighbors of v distinct from x. Since (D_1) and (E_1) are excluded each of them belongs to two nontriangular faces containing v. Hence v is incident on at least k_3 nontriangular faces. Then v sends charge only to the other $D-1-k_3$ faces. It follows that $c^*(v) \geq D-1-4-(D-1-k_3)\cdot\frac{1}{2}-k_3\cdot\frac{1}{3}=\frac{D-9}{2}+\frac{k_3}{6}\geq 0$.

Finally we consider a vertex v of degree D. Since G contains neither configuration (A) nor (B), v can have at most one neighbor of degree 2. Assume that v has such a neighbor. Since critical paths are excluded, this neighbor is incident on a triangle and a nontriangular face. Since (D_0) and (E_0) are excluded, each 3-neighbor of v is adjacent to two nontriangular faces containing v. Hence there are at least k_3+1 such faces. It follows that v sends at most $1+k_3\cdot\frac{1}{3}+[D-(k_3+1)]\cdot\frac{1}{2}\leq\frac{D+1}{2}$, which does not exceed $D-4$ for $D\geq 9$. This proves that v sends at most $D-4$ units of charge when there are 2-vertices among the neighbors of v.

It remains to show that v retains nonnegative charge when each of its neighbors has degree at least 3. Suppose that $k_3\leq 2$ and let q denote the number of nontriangular faces. Then $c^*(v)\geq D-4-\frac{2}{3}-\frac{D-q}{2}=\frac{3D+3q-28}{6}$. We see that $c^*(v)\geq 0$ when $D\geq 10$ or $D=9$ and $q\geq 1$. Consider the remaining case $D=9$, $q=0$. Then there are only triangles incident on v. Thus there is no pair of consecutive neighbors of degree at most 5, for such a pair would be joined by a reducible edge. Hence there are at most 4 neighbors of degree at most 5 and consequently there is a triangle with all vertices of degree at least 6. Such a triangle receives only $\frac{1}{3}$ from v by Rule 4b. Thus $c^*(v)\geq 5-\frac{2}{3}-\frac{1}{3}-\frac{8}{2}=0$.

We are left with the case $k_3\geq 3$. First assume that there is a 3-neighbor which belongs to two triangles incident on v. Then, since (F) is excluded, there is at most one more 3-neighbor which belongs to a triangle incident on v. Then there are k_3-2 degree 3 neighbors which belong to two nontriangular faces incident on v. It follows that v is incident on at least $(k_3-2)+1$ nontriangular faces. Then $c^*(v)\geq D-4-\frac{1}{3}\cdot k_3-[D-(k_3-1)]\cdot\frac{1}{2}=\frac{D-9}{2}+\frac{k_3}{6}\geq 0$.

Finally assume that each 3-neighbor belongs to at most one triangle incident on v. Let k_3' and k_3'' denote the number of 3-neighbors which respectively belong and do not belong to a triangle incident on v. Then there are at least $\frac{k_3'}{2} + k_3''$ nontriangular faces. It follows that $c^*(v) \geq D - 4 - (k_3' + k_3'') \cdot \frac{1}{3} - [D - (\frac{k_3'}{2} + k_3'')] \cdot \frac{1}{2} = \frac{D}{2} - 4 - \frac{k_3'}{12} + \frac{k_3''}{6}$. Since each 3-neighbor belongs to at most one triangle incident on v, among arbitrary three consecutive faces incident on v there are at most two triangles containing a 3-vertex. It follows that there are at most $\frac{2}{3} \cdot D$ such triangles. Consequently $k_3' \leq \frac{2}{3} \cdot D$ and $c^*(v) \geq \frac{D}{2} - 4 - \frac{2}{3} D \cdot \frac{1}{12} + \frac{k_3''}{6} = \frac{8D-72}{18} + \frac{k_3''}{6} \geq 0$. This settles the proof. □

Theorem 5.2. *Let G be any planar graph with maximum degree Δ and let $D = \max\{\Delta, 9\}$. If G contains none of the configurations* (A) - (H) *then G contains a reducible edge, i.e., an edge uv such that* $\deg(u)+\deg(v) \leq D+1$.

Proof. Let $\widehat{G}$ be the graph obtained from G by replacing each critical path joining u and v by a single edge uv. As (A) is excluded, $\widehat{G}$ is a simple planar graph with no critical paths and with maximum degree Δ. We show that $\widehat{G}$ does not contain any of configurations (A)-(F). If $\widehat{G}$ contains any of configurations (A), (B), (C_k) then G contains the same configuration, a contradiction. The same argument works for ($D_{\geq 1}$) and ($E_{\geq 1}$). If $\widehat{G}$ contains (D_0) then G contains (A) or (B) or (D_0), a contradiction. If $\widehat{G}$ contains (E_0) then G contains (E_0) or (A), again a contradiction. Finally, if $\widehat{G}$ contains (F) then G contains one of (F), (B), (G) or (H), a contradiction once again. Thus $\widehat{G}$ satisfies the conditions of Theorem 5.1. Hence $\widehat{G}$ contains a reducible edge. This edge cannot be one of those which appeared after substituting a path, because both ends of such edges have degree D. It follows that G also contains a reducible edge. □

5.2 Reducibility

We have already proved the reducibility of configurations (A), (B) and (C_k). In this section we give proofs for the other configurations.

Let $Free(x)$ (resp. $Free(uv)$) denote the set all of colors from $\{1, \ldots, D\}$ that are free at vertex x (resp. edge uv). Analogously we define sets $Used(x)$ and $Used(uv)$.

Lemma 5.3. *Configuration* (D_k) *is reducible for any $k \geq 0$.*

Proof. We name the vertices of the configuration as in Fig. 3. Recall that $\deg_G(x) \leq 3 + k$, $\deg_G(y) \leq 2 + k$ and $\deg_G(v) \leq D - k$. We remove edge vy and color the remaining graph recursively. Now we show how to color G. We denote the edge colors as in Fig. 3. Note that $|Free(v)| \geq k + 1$ and

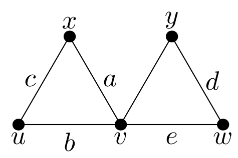

Figure 3: Labeling vertices in configuration (D_k).

$|Used(y)| \leq k+1$. We can assume that $Free(v) = Used(y)$ for otherwise we simply color vy with any color from $Free(v) - Used(y)$. Hence $d \in Free(v)$ and $a, b, e \in Free(y)$.

Case 1. Color e is free at x. Then we swap the colors of wy and wv, and color xv with e and vy with a.

Case 2. Color e is used by x. Then $|Used(x) - \{a, e\}| \leq k + 1$. We can assume that $Used(x) - \{a, e\} = Free(v)$ for otherwise we just color vy with a and vx with any color from $Free(v) - Used(x)$. Hence either $c = e$ or $c \in Free(v)$. We color vy with b and swap the colors of ux and uv. If uv is now colored with e (i.e. $c = e$) we additionally swap the colors of wy and wv. $\square$

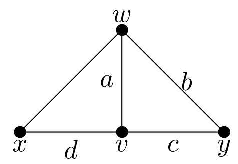

Figure 4: Configuration (E_k).

Lemma 5.4. *Configuration* (E_k) *is reducible for any* $k \geq 0$.

Proof. We name the vertices of the configuration as in Fig. 4. Recall that $\deg_G(x) \leq 2 + k$, $\deg_G(y) \leq 3$ and $\deg_G(w) \leq D - k$. We remove edge xw and color the remaining graph recursively. Now we show how to color G. We denote the edge colors as in Fig. 4. Note that $|Free(w)| \geq k + 1$ and $|Used(x)| \leq k+1$. We can assume that $Free(w) = Used(x)$ for otherwise we simply color xw with any color from $Free(w) - Used(x)$. Hence $d \in Free(w)$ and $a, b \in Free(x)$. Let e be the color used by y other than b, c, if any (otherwise let e be any color other than b, c).

Case 1. $e \neq d$. Color xw with b and wy with d.

Case 2. $e = d$. Swap the colors of vw and vx, color wy with a and xw with b. ☐

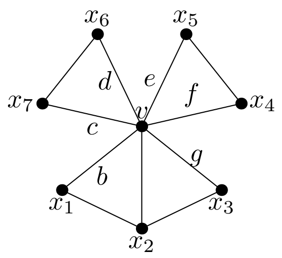

Figure 5: Configuration (F). Possibly $x_1 = x_7$ or $x_3 = x_4$. $\deg_G(x_2) = \deg_G(x_5) = \deg_G(x_6) = 3$.

Lemma 5.5. *Configuration* (F) *is reducible.*

Proof. We name the vertices of the configuration as in Fig. 5. Recall that $\deg_G(x_2) = \deg_G(x_5) = \deg_G(x_6) = 3$. We remove edge vx_2 and color the remaining graph. Let C be this coloring. Now we show how to color G. We denote edge colors as in Fig. 5 (note that $b = c$ if $x_1 = x_7$ and $f = g$ if $x_3 = x_4$). Let a be a color free at v. We can assume that $a \in Used(x_2)$ for otherwise we simply color vx_2 with a. By symmetry, w.l.o.g. we can assume that $C(x_1x_2) = a$.

Case 1. $C(x_2x_3) = b$. Then we can assume that $a \in Used(x_5)$ for otherwise we simply color vx_2 with e and vx_5 with a. Similarly, we can assume that $g \in Used(x_5)$ for otherwise we swap the colors of x_1v and x_1x_2, swap the colors of x_3v and x_3x_2, color vx_2 with e and vx_5 with g. Hence $a, g \in Used(x_5)$. By symmetry, $a, g \in Used(x_6)$.

Case 1.1. $C(x_6x_7) = a$. Then $x_1 \neq x_7$ for otherwise the coloring is not proper. We swap the colors of x_7x_6 and x_7v, and we color vx_2 with c.

Case 1.2. $C(x_6x_7) = g$.

Case 1.2.1. $x_1 \neq x_7$. Then $c \neq b$. We swap the colors of the following three pairs of edges: x_1v and x_1x_2, x_3v and x_3x_2, x_7x_6 and x_7v. Finally we color vx_2 with c.

Case 1.2.2. $x_1 = x_7$. Then $c = b$. We color x_1x_6 with b, x_1v with a, x_1x_2 with g, vx_2 with e, vx_5 with b.

Case 2. $C(x_2x_3) \neq b$.

Case 2.1. $C(x_2x_3) \notin \{e, f\}$. Then we can assume that $a \in Used(x_5)$ for otherwise we simply color vx_2 with e and vx_5 with a. Similarly, we can assume that $b \in Used(x_5)$ for otherwise we swap the colors of x_1v and x_1x_2, color vx_2 with e and vx_5 with b.

Then we color vx_2 with f and we swap the colors of x_4x_5 and x_4v. Then vx_4 is colored with a or b. In the latter case we additionally swap the colors of x_1v and x_1x_2.

Case 2.2. $C(x_2x_3) \in \{e, f\}$. Analogously to Case 2.1 we can assume that $a, b \in Used(x_6)$. Since $C(x_6x_7) \in \{a, b\}$, $x_1 \neq x_7$ and $c \neq b$. Hence we can proceed as in Case 2.1: we color vx_2 with c and we swap the colors of x_7x_6 and x_7v. Then vx_7 is colored with a or b. In the latter case we additionally swap the colors of x_1v and x_1x_2.

□

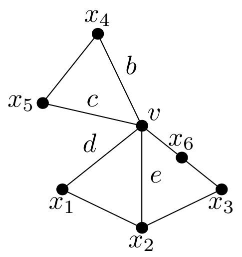

Figure 6: Configuration (G). Possibly $x_1 = x_5$.

Lemma 5.6. *Configuration* (G) *is reducible.*

Proof. We name the vertices of the configuration as in Fig. 6. Recall that $\deg_G(x_2) = \deg_G(x_4) = 3$ and $\deg_G(x_6) = 2$. We remove edge vx_6 and color the remaining graph. Let C be this coloring. Now we show how to color G. We denote edge colors as in Fig. 6 (observe that when $x_1 = x_5$ then $c = d$). Let a be a color free at v. We can assume that $C(x_3x_6) = a$ for otherwise we simply color vx_6 with a. Analogously we assume that $C(x_1x_2) = a$ for otherwise we color vx_6 with e and vx_2 with a. Finally we assume that $C(x_2x_3) = d$ for otherwise we swap the colors of x_1v and x_1x_2 and color vx_6 with d.

Case 1. a is free at x_4. Then we color vx_4 with a and vx_6 with b.

Case 2. d is free at x_4. Then we swap the colors of x_3x_6 and x_3x_2, swap the colors of x_1x_2 and x_1v, color vx_4 with d and vx_6 with b.

Case 3. Both a and d are used by x_4. Note that this is possible only when $x_1 \neq x_5$. Then we color vx_6 with c and we swap the colors of x_5x_4 and x_5v. If as a result $C(x_5v) = d$ we also swap the colors of x_3x_6 and x_3x_2, and swap the colors of x_1x_2 and x_1v. □

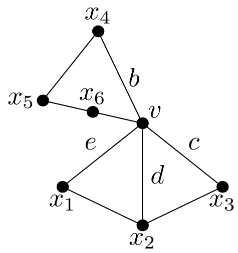

Figure 7: Configuration (H).

Lemma 5.7. *Configuration* (H) *is reducible.*

Proof. We name the vertices of the configuration as in Fig. 7. Recall that $\deg_G(x_2) = \deg_G(x_4) = 3$ and $\deg_G(x_6) = 2$. We remove edge vx_6 and color the remaining graph. Let C be this coloring. Now we show how to color G. We denote edge colors as in Fig. 7. Let a be a color free at v. We can assume that $C(x_5x_6) = a$ for otherwise we simply color vx_6 with a. Moreover, we can assume that x_1x_2 or x_2x_3 is colored with a for otherwise we color vx_6 with d and vx_2 with a. By symmetry we can assume that $C(x_2x_3) = a$.

Case 1. $C(x_1x_2) \neq c$. Then we color vx_6 with c and we swap the colors of x_3x_2 and x_3v.

Case 2. $C(x_1x_2) = c$. Then we swap the colors of x_1v and x_1x_2, swap the colors of x_3x_2 and x_3v, and we color vx_6 with e. □

6 Further Research

The most natural question is whether our approach can lead to a linear-time algorithm for coloring planar graphs with $\max\{\Delta, 8\}$ colors. We conjecture that this is possible, but we suspect that it would involve both a large number of and more elaborate configurations.

Acknowledgments

The authors would like to thank anonymous referees for their excellent work and many useful suggestions which helped us avoid several mistakes and also improved the presentation of the paper.

References

[1] Marek Chrobak and Moti Yung. Fast algorithms for edge-coloring planar graphs. *Journal of Algorithms*, 10:35–51, 1989.

[2] Marek Chrobak and Takao Nishizeki. Improved edge-coloring algorithms for planar graphs. *Journal of Algorithms*, 11:102–116, 1990.

[3] V. G. Vizing. On the estimate of the chromatic class of a p-graph. *Diskret. Analiz*, 3:25–30, 1964.

[4] Harold N. Gabow, Takao Nishizeki, Oded Kariv, D. Leven, and O.Terada. Algorithms for edge coloring graphs. Technical Report TR-41/85, Dept. of Computer Science, Tel Aviv University, 1985.

[5] Richard Cole, Kirstin Ost, and Stefan Schirra. Edge-coloring bipartite multigraphs in $O(E \log D)$ time. *Combinatorica*, 21:5–12, 2001.

[6] Noga Alon. A simple algorithm for edge-coloring bipartite multigraphs. *Information Processing Letters*, 85(6):301–302, 2003.

[7] San Skulrattanakulchai. 4-edge-coloring graphs of maximum degree 3 in linear time. *Information Processing Letters*, 81:191–195, 2002.

[8] V. G. Vizing. Critical graphs with a given chromatic number. *Diskret. Analiz*, 5:9–17, 1965.

[9] Daniel P. Sanders and Yue Zhao. Planar graphs of maximum degree 7 are class I. *Journal of Combinatorial Theory, Series B*, 83:201–212, 2001.

[10] Peter Guthrie Tait. On the coloring of maps. *Proc. Royal Soc. Edinburgh Sect. A*, 10:501–503, 1878.

[11] Neil Robertson, Daniel P. Sanders, Paul Seymour, and Robin Thomas. Efficiently four-coloring planar graphs. In *Proc. 28th Symposium on Theory of Computing*, pages 571–575. ACM, 1996.

[12] Daniel P. Sanders. Private communication. 2005.

[13] Xin He. An efficient algorithm for edge coloring planar graphs with Δ colors. *Theoretical Computer Science*, 74:299–312, 1990.

[14] Oleg V. Borodin. On the total coloring of planar graphs. *J. Reine Ange. Math.*, (394):180–185, 1989.