Approximating Maximum Edge 2-Coloring in Simple Graphs

Zhi-Zhong Chen, Sayuri Konno, and Yuki Matsushita

Department of Mathematical Sciences, Tokyo Denki University, Hatoyama, Saitama 350-0394, Japan
zzchen@mail.dendai.ac.jp

Abstract. We present a polynomial-time approximation algorithm for legally coloring as many edges of a given simple graph as possible using two colors. It achieves an approximation ratio of roughly 0.842 and runs in $O(n^3m)$ time, where n (respectively, m) is the number of vertices (respectively, edges) in the input graph. The previously best ratio achieved by a polynomial-time approximation algorithm was $\frac{5}{6} \approx 0.833$.

Keywords: Approximation algorithms, graph algorithms, edge coloring, NP-hardness.

1 Introduction

Given a graph G and a natural number t, the *maximum edge t-coloring problem* (called Max Edge t-Coloring for short) is to find a maximum-sized set F of edges in G such that F can be partitioned into at most t matchings of G. Motivated by call admittance issues in satellite based telecommunication networks, Feige et al. [3] introduced the problem and proved its APX-hardness. They also observed that Max Edge t-Coloring is a special case of the well-known maximum coverage problem (see [6]). Since the maximum coverage problem can be approximated by a greedy algorithm within a ratio of $1 - (1 - \frac{1}{t})^t$ [6], so can Max Edge t-Coloring. In particular, the greedy algorithm achieves an approximation ratio of $\frac{3}{4}$ for Max Edge 2-Coloring, which is the special case of Max Edge t-Coloring where the input number t is fixed to 2. Feige et al. [3] has improved the trivial ratio $\frac{3}{4} = 0.75$ to $\frac{10}{13} \approx 0.769$ by an LP approach.

The APX-hardness proof for Max Edge t-Coloring given by Feige et al. [3] indeed shows that the problem remains APX-hard even if we restrict the input graph to a simple graph and fix the input integer t to 2. We call this restriction (special case) of the problem Max Simple Edge 2-Coloring. Feige et al. [3] also pointed out that for Max Simple Edge 2-Coloring, an approximation ratio of $\frac{4}{5}$ can be achieved by the following *simple algorithm*: Given a simple graph G, first compute a maximum-sized subgraph H of G such that the degree of each vertex in H is at most 2 and there is no 3-cycle in H, and then remove one *arbitrary* edge from each odd cycle of H. This simple algorithm has been improved in [1,2,9]. The previously best ratio (namely, $\frac{5}{6}$) was given in [9]. In this

B. Chen (Ed.): AAIM 2010, LNCS 6124, pp. 78–89, 2010.

paper, we improve on both the algorithm in [1] and the algorithm in [9] to obtain a new approximation algorithm that achieves a ratio of roughly 0.842. Roughly speaking, our algorithm is based on local improvement, dynamic programming, and recursion. Its analysis is based on an intriguing charging scheme and certain structural properties of *caterpillar* graphs and *starlike* graphs (see Section 3 for definitions).

Kosowski et al. [10] also considered Max Simple Edge 2-Coloring. They presented an approximation algorithm that achieves a ratio of $\frac{28\Delta-12}{35\Delta-21}$, where Δ is the maximum degree of a vertex in the input simple graph. This ratio can be arbitrarily close to the trivial ratio $\frac{4}{5}$ because Δ can be very large. In particular, this ratio is worse than our new ratio 0.842 when $\Delta \geq 4$. Moreover, when $\Delta = 3$, our algorithm indeed achieves a ratio of $\frac{6}{7}$, which is equal to the ratio $\frac{28\Delta-12}{35\Delta-21}$ achieved by Kosowski et al.'s algorithm [10]. Note that Max Simple Edge 2-Coloring becomes trivial when $\Delta \leq 2$. Therefore, no matter what Δ is, our algorithm is better than or as good as all known approximation algorithms for Max Simple Edge 2-Coloring.

Kosowski et al. [10] showed that approximation algorithms for Max Simple Edge 2-Coloring can be used to obtain approximation algorithms for certain packing problems and fault-tolerant guarding problems. Combining their reductions and our improved approximation algorithm for Max Simple Edge 2-Coloring, we can obtain improved approximation algorithms for their packing problems and fault-tolerant guarding problems immediately.

2 Basic Definitions

Throughout the remainder of this paper, a graph means a simple undirected graph (i.e., it has neither parallel edges nor self-loops).

Let G be a graph. We denote the vertex set of G by $V(G)$, and denote the edge set of G by $E(G)$. The *degree* of a vertex v in G, denoted by $d_G(v)$, is the number of vertices adjacent to v in G. A vertex v of G with $d_G(v) = 0$ is called an *isolated vertex*. For a subset U of $V(G)$, let $G[U]$ denote the graph (U, E_U) where E_U consists of all edges $\{u, v\}$ of G with $u \in U$ and $v \in U$. We call $G[U]$ the *subgraph of* G *induced by* U. For a subset U of $V(G)$, we use $G-U$ to denote $G[V(G) - U]$. G is a *star* if G is connected, G has at least three vertices, and there is a vertex u (called the *center* of G) such that every edge of G is incident to u. Each vertex of a star other than the center is called a *satellite* of the star.

A *cycle* in G is a connected subgraph of G in which each vertex is of degree 2. A *path* in G is a connected subgraph of G in which exactly two vertices are of degree 1 and the others are of degree 2. Each vertex of degree 1 in a path P is called an *endpoint* of P, while each vertex of degree 2 in P is called an *inner vertex* of P. An edge $\{u, v\}$ of a path P is called an *inner edge* of P if both u and v are inner vertices of P. The *length* of a cycle or path C is the number of edges in C. A cycle of odd (respectively, even) length is called an *odd* (respectively, *even*) cycle.

A *path-cycle cover* of G is a subgraph H of G such that $V(H) = V(G)$ and $d_H(v) \leq 2$ for every $v \in V(H)$. Note that each connected component of a path-cycle cover of G is a single vertex, path, or cycle. A path-cycle cover $\mathcal{C}$ of G is *triangle-free* if $\mathcal{C}$ does not contain a cycle of length 3. A path-cycle cover C of G is *maximum-sized* if the number of edges in C is maximized over all path-cycle covers of G.

G is *edge-2-colorable* if each connected component of G is an isolated vertex, a path, or an even cycle. Note that MAX SIMPLE EDGE 2-COLORING is the problem of finding a maximum-sized edge-2-colorable subgraph in a given graph.

3 Two Crucial Lemmas and the Outline of Our Algorithm

We say that a graph $K = (V_K, E_K \cup F_K)$ is a *caterpillar graph* if it satisfies the following conditions:

- The graph (V_K, E_K) has $h+1$ connected components $C_0, \ldots, C_h$ with $h \geq 0$.
- C_0 is a path while C_1 through C_h are odd cycles of length at least 5.
- F_K is a matching consisting of h edges $\{u_1, v_1\}, \ldots, \{u_h, v_h\}$.
- For each $i \in \{1, \ldots, h\}$, u_i is an inner vertex of path C_0 while v_i is a vertex of C_i.

We call the edges of F_K the *leg edges* of K, call path C_0 the *spine path* of K, and call cycles C_1 through C_h the *foot cycles* of K.

We say that a graph $K = (V_K, E_K \cup F_K)$ is a *starlike graph* if it satisfies the following conditions:

- The graph (V_K, E_K) has $h+1$ connected components $C_0, \ldots, C_h$ with $h \geq 2$.
- C_0 is a cycle of length at least 4 while C_1 through C_h are odd cycles of length at least 5.
- F_K is a matching consisting of h edges $\{u_1, v_1\}, \ldots, \{u_h, v_h\}$.
- For each $i \in \{1, \ldots, h\}$, u_i is a vertex of C_0 while v_i is a vertex of C_i.

We call the edges of F_K the *bridge edges* of K, call C_0 the *central cycle* of K, and call C_1 through C_h the *satellite cycles* of K.

Let r be the root of the quadratic equation $23r^2 - 55r + 30 = 0$ that is smaller than 1. Note that $r = 0.84176 \ldots \approx 0.842$. The reason why we choose r in this way will become clear later in the proof of Lemma 7.

Lemma 1. *Suppose that K is a caterpillar graph such that each foot cycle of K is charged a penalty of $6 - 7r$. Let $p(K)$ be the total penalties charged to the foot cycles of K. Then, K has an edge-2-colorable subgraph K' such that $|E(K')| - p(K) \geq r|E_K|$, where E_K is the set of edges on the spine path or the foot cycles of K.*

Lemma 2. *Suppose that K is a starlike graph such that each satellite cycle of K is charged a penalty of $6 - 7r$. Let $p(K)$ be the total penalties charged to the satellite cycles of K. Then, K has an edge-2-colorable subgraph K' such that $|E(K')| - p(K) \geq r|E_K|$, where E_K is the set of edges on the central or satellite cycles of K.*

Based on Lemmas 1 and 2, we will design our algorithm roughly as follows: Given an input graph G, we will first construct a suitable maximum-sized triangle-free path-cycle cover $\mathcal{C}$ of G and compute a suitable set F of edges such that the endpoints of each edge in F fall into different connected components of $\mathcal{C}$ and each odd cycle of $\mathcal{C}$ has at least one vertex that is an endpoint of an edge in F. Note that $\mathcal{C}$ has at least as many edges as a maximum-sized edge-2-colorable subgraph of G. The edges in F will play the following role: we will break each odd cycle C in $\mathcal{C}$ by removing one edge of C incident to an edge of F and then this edge of F can possibly be added to $\mathcal{C}$ so that $\mathcal{C}$ becomes an edge-2-colorable subgraph of G. Unfortunately, not every edge of F can be added to $\mathcal{C}$ and we have to discard some edges from F, leaving some odd cycles of $\mathcal{C}$ *F-free* (i.e., having no vertex incident to an edge of F). Clearly, breaking an F-free odd cycle C of short length (namely, 5) by removing one edge from C results in a significant loss of edges from $\mathcal{C}$. We charge the loss to the non-F-free odd cycles (unevenly) as penalties. Fortunately, adding the edges of F to $\mathcal{C}$ will yield a graph whose connected components are caterpillar graphs, starlike graphs, or certain other kinds of graphs with good properties. Now, Lemmas 1 and 2 help us show that our algorithm achieves a ratio of r.

4 The Algorithm

Throughout this section, fix a graph G and a maximum-sized edge-2-colorable subgraph $\mathcal{B}$ (for "best") of G. Let n (respectively, m) be the number of vertices (respectively, edges) in G. Our algorithm starts by performing the following four steps:

1. If $|V(G)| \leq 2$, then output G itself and halt.
2. Compute a maximum-sized triangle-free path-cycle cover $\mathcal{C}$ of G. (*Comment:* This step can be done in $O(n^2m)$ time [5].)
3. While there is an edge $\{u, v\} \in E(G)-E(\mathcal{C})$ such that $d_{\mathcal{C}}(u) \leq 1$ and v is a vertex of some cycle C of $\mathcal{C}$, modify $\mathcal{C}$ by deleting one (arbitrary) edge of C incident to v and adding edge $\{u, v\}$.
4. Construct a graph $G_1 = (V(G), E_1)$, where E_1 is the set of all edges $\{u, v\} \in E(G) - E(\mathcal{C})$ such that u and v appear in different connected components of $\mathcal{C}$ and at least one of u and v appears on an odd cycle of $\mathcal{C}$.

Hereafter, $\mathcal{C}$ always means that we have finished modifying it in Step 3. We give several definitions related to the graphs G_1 and $\mathcal{C}$. Let S be a subgraph of G_1. S *saturates* an odd cycle C of $\mathcal{C}$ if at least one edge of S is incident to a vertex of C. The *weight* of S is the number of odd cycles of $\mathcal{C}$ saturated by S. For convenience, we say that two connected components C_1 and C_2 of $\mathcal{C}$ are *adjacent* in G if there is an edge $\{u_1, u_2\} \in E(G)$ such that $u_1 \in V(C_1)$ and $u_2 \in V(C_2)$.

Lemma 3. *We can compute a maximum-weighted path-cycle cover in G_1 in $O(nm \log n)$ time.*

Our algorithm then proceeds to performing the following four steps:

5. Compute a maximum-weight path-cycle cover M in G_1.
6. While there is an edge $e \in M$ such that the weight of $M - \{e\}$ is the same as that of M, delete e from M.
7. Construct a graph $G_2 = (V(G), E(\mathcal{C}) \cup M)$. (*Comment:* For each pair of connected components of $\mathcal{C}$, there is at most one edge between them in G_2 because of Step 6.)
8. Construct a graph G_3, where the vertices of G_3 one-to-one correspond to the connected components of $\mathcal{C}$ and two vertices are adjacent in G_3 if and only if the corresponding connected components of $\mathcal{C}$ are adjacent in G_2.

Fact 1. *Suppose that C' is a connected component of G_3. Then, the following statements hold:*

1. *C' is a vertex, an edge, or a star.*
2. *If C' is an edge, then at least one endpoint of C' corresponds to an odd cycle of $\mathcal{C}$.*
3. *If C' is a star, then every satellite of C' corresponds to an odd cycle of $\mathcal{C}$.*

An *isolated odd-cycle* of G_2 is an odd cycle of G_2 whose corresponding vertex in G_3 is isolated in G_3. Similarly, a *leaf odd-cycle* of G_2 is an odd cycle of G_2 whose corresponding vertex in G_3 is of degree 1 in G_3. Moreover, a *branching odd-cycle* of G_2 is an odd cycle of G_2 whose corresponding vertex in G_3 is of degree 2 or more in G_3.

Lemma 4. *Let I be the set of isolated odd-cycles in G_2. Then, $|E(\mathcal{B})| \le |E(\mathcal{C})| - |I|$.*

Proof. Let $C_1, \ldots, C_h$ be the odd cycles of $\mathcal{C}$ such that for each $i \in \{1, \ldots, h\}$, $\mathcal{B}$ contains no edge $\{u, v\}$ with $|\{u, v\} \cap V(C_i)| = 1$. Let $U_1 = \bigcup_{i=1}^{h} V(C_i)$ and $U_2 = V(G) - U_1$. For convenience, let $C_0 = G[U_2]$. Note that for each $e \in E(\mathcal{B})$, one of the graphs $C_0, C_1, \ldots, C_h$ contains both endpoints of e. So, $\mathcal{B}$ can be partitioned into $h+1$ disjoint subgraphs $\mathcal{B}_0, \ldots, \mathcal{B}_h$ such that $\mathcal{B}_i$ is a path-cycle cover of $G[V(C_i)]$ for every $i \in \{0, \ldots, h\}$. Since $\mathcal{C}[U_2]$ must be a maximum-sized path-cycle cover of C_0, $|E(\mathcal{C}[U_2])| \ge |E(\mathcal{B}_0)|$. The crucial point is that for every $i \in \{1, \ldots, h\}$, $|E(\mathcal{B}_i)| \le |V(C_i)| - 1 = |E(C_i)| - 1$ because $|V(C_i)|$ is odd. Thus, $|E(\mathcal{C})| = |E(\mathcal{C}[U_2])| + \sum_{i=1}^{h} |E(C_i)| \ge |E(\mathcal{B}_0)| + \sum_{i=1}^{h}(|E(\mathcal{B}_i)| + 1) = |E(\mathcal{B})| + h$.

Note that $(V(G), E(G_1) \cap E(\mathcal{B}))$ is a path-cycle cover in G_1 of weight $k - h$, where k is the number of odd cycles in $\mathcal{C}$. So, $k - h \le k - |I|$ because M is a maximum-weight path-cycle cover in G_1 of weight $k - |I|$. So, by the last inequality in the last paragraph, $|E(\mathcal{B})| \le |E(\mathcal{C})| - h \le |E(\mathcal{C})| - |I|$. ∎

Some definitions are in order (see Figure 1 for an example). A *bicycle* of G_2 is a connected component of G_2 that consists of two odd cycles and an edge between them. Note that a connected component of G_3 is an edge if it corresponds to a

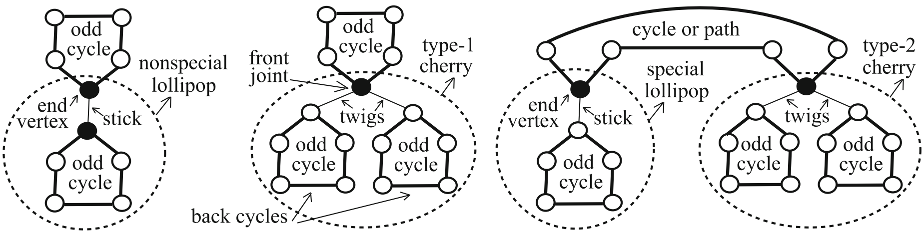

Fig. 1. An example of G_2, where the hollow vertices are free, the bold edges belong to $\mathcal{C}$, the left connected component is a bicycle, and the middle connected component is a tricycle

bicycle in G_2. A *tricycle* of G_2 is a connected component T of G_2 that consists of one branching odd-cycle C_1, two leaf odd-cycles C_2 and C_3, and two edges $\{u_1, u_2\}$ and $\{u_1, u_3\}$ such that $u_1 \in V(C_1)$, $u_2 \in V(C_2)$, and $u_3 \in V(C_3)$. For convenience, we call C_1 the *front cycle* of tricycle K, call C_2 and C_3 the *back cycles* of tricycle K, and call u_1 the *front joint* of tricycle K.

A *cherry* of G_2 is a subgraph Q of G_2 that consists of two leaf odd-cycles C_1 and C_2 of $\mathcal{C}$, a vertex $u \in V(G) - (V(C_1) \cup V(C_2))$, and two edges $\{u, v_1\}$ and $\{u, v_2\}$ such that $v_1 \in V(C_1)$ and $v_2 \in V(C_2)$. For convenience, we call edges $\{u, v_1\}$ and $\{u, v_2\}$ the *twigs* of cherry Q. By the construction of G_2, each pair of cherries are vertex-disjoint. Note that each odd cycle in a cherry of G_2 is a satellite of a star in G_3. We classify the cherries of G_2 into two types as follows. A cherry Q of G_2 is of *type-1* if Q is a subgraph of a tricycle of G_2. Note that the two odd cycles in a type-1 cherry of G_2 are the back cycles of a tricycle of G_2. A cherry of G_2 is of *type-2* if it is not of type-1. Further note that there is no edge $\{u, v\}$ in G such that u appears on an isolated odd-cycle of G_2 and v appears on an odd cycle in a cherry of G_2.

A *lollipop* of G_2 is a subgraph L of G_2 that consists of a leaf odd-cycle C of G_2, a vertex $u \notin V(C)$, and an edge $\{u, v\}$ with $v \in V(C)$. For convenience, we call edge $\{u, v\}$ the *stick* of lollipop L and call vertex u the *end vertex* of lollipop L. A lollipop of G_2 is *special* if it is neither a subgraph of a cherry of G_2 nor a subgraph of a bicycle of G_2. A vertex u of G_2 is *free* if no lollipop of G_2 has u as its end vertex. Because of Step 3, each vertex of degree at most 2 in G_2 is free.

We next define two types of operations that will be performed on G_2. An operation on G_2 is *robust* if it removes no edge of $\mathcal{C}$, creates no new odd cycle, and creates no new isolated odd-cycle of G_2.

Type 1: Suppose that C is an odd cycle of a cherry Q of G_2 and u is a free vertex of G_2 with $u \notin V(C)$ such that

- some vertex v of C is adjacent to u in G and
- if Q is a type-1 cherry of G_2, then u is not an endpoint of a twig of Q.

Then, a *type-1 operation* on G_2 using cherry Q and edge $\{u, v\}$ modifies G_2 by performing the following steps (see Figure 2 for example cases):

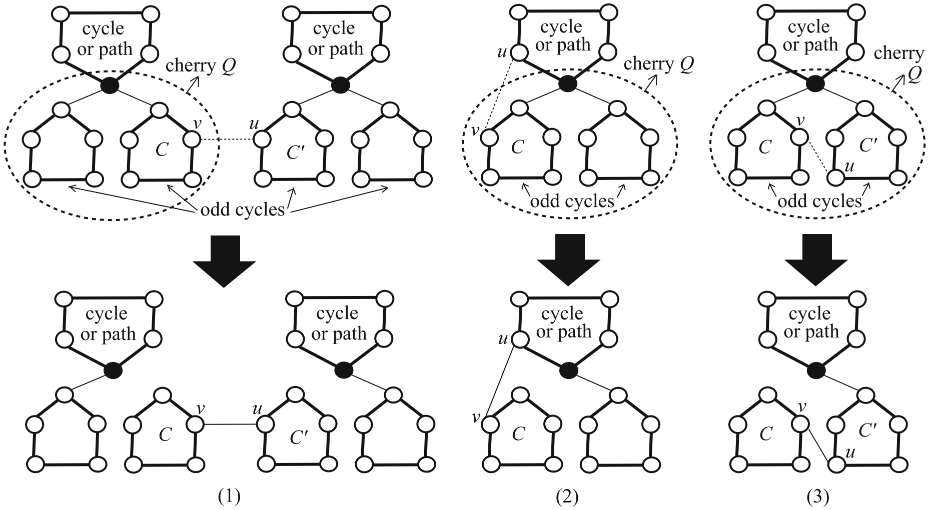

Fig. 2. Three example cases of a type-1 operation, where the dotted lines are edges in $E(G) - E(G_2)$

(1) If u appears on a leaf odd cycle C' of G_2 such that C' is not part of a bicycle of G_2 and Q is not a type-1 cherry of G_2 with $u \in V(Q)$, then delete the stick of the lollipop containing C' from G_2.
(2) Delete the twig of Q incident to a vertex of C from G_2.
(3) Add edge $\{u, v\}$ to G_2.
(*Comment:* A type-1 operation on G_2 is robust and destroys at least one cherry of G_2 without creating a new cherry in G_2.)

Type 2: Suppose that Q is a type-2 cherry of G_2, B is a bicycle of G_2, and $\{u, v\}$ is an edge in $E(G_1) - E(G_2)$ such that u appears on an odd cycle C of Q and v appears on an odd cycle of B. Then, a *type-2 operation* on G_2 using cherry Q, bicycle B, and edge $\{u, v\}$ modifies G_2 by deleting the twig of Q incident to a vertex of C and adding edge $\{u, v\}$.
(*Comment:* A type-2 operation on G_2 is robust. Moreover, when no type-1 operation on G_2 is possible, a type-2 operation on G_2 destroys a type-2 cherry of G_2 and creates a new type-1 cherry in G_2.)

Now, Step 9 of our algorithm is as follows.

9. While a type-1 or type-2 operation on G_2 is possible, perform the following step:
   (a) If a type-1 operation on G_2 is possible, perform a type-1 operation on G_2; otherwise, perform a type-2 operation on G_2.

Fact 2. *After Step 9, the following statements hold:*

1. *There is no edge $\{u, v\}$ in $E(G)$ such that u appears on an odd cycle in a type-2 cherry of G_2 and v appears on another odd cycle in a type-2 cherry of G_2.*
2. *If $\{u, v\}$ is an edge of G_1 such that u appears on an odd cycle of a type-2 cherry of G_2 and no type-2 cherry of G_2 contains v, then v is the end vertex of a special lollipop or the front joint of a tricycle of G_2.*

Hereafter, G_2 always means that we have finished modifying it in Step 9. Now, the final three steps of our algorithm are as follows:

10. Let U be the set of vertices that appear in type-2 cherries of G_2.
11. If $U = \emptyset$, then perform the following steps:
    (a) For each connected component K of G_2, compute a maximum-sized edge-2-colorable subgraph of K. (*Comment:* Because of the simple structure of K, this step can be done in linear time by a standard dynamic programming.)
    (b) Output the union of the edge-2-colorable subgraphs computed in Step 11a, and halt.
12. If $U \neq \emptyset$, then perform the following steps:
    (a) Obtain an edge-2-colorable subgraph R of $G - U$ by recursively calling the algorithm on $G - U$.
    (b) For each type-2 cherry Q of G_2, obtain an edge-2-colorable subgraph of Q by removing one edge from each odd cycle C of Q that shares an endpoint with a twig of Q.
    (c) Let $\mathcal{A}_1$ be the union of R and the edge-2-colorable subgraphs computed in Step 12b.
    (d) For each connected component K of G_2, compute a maximum-sized edge-2-colorable subgraph of K. (*Comment:* Because of the simple structure of K, this step can be done in linear time by a standard dynamic programming.)
    (e) Let $\mathcal{A}_2$ be the union of the edge-2-colorable subgraphs computed in Step 12d.
    (f) If $|E(\mathcal{A}_1)| \geq |E(\mathcal{A}_2)|$, output $\mathcal{A}_1$ and halt; otherwise, output $\mathcal{A}_2$ and halt.

Lemma 5. *Assume that G_2 has no type-2 cherry. Then, the edge-2-colorable subgraph of G output in Step 11b contains at least $r|E(\mathcal{B})|$ edges.*

Proof. Let $\mathcal{C}_2$ be the graph obtained from G_2 by removing one edge from each isolated odd-cycle of G_2. By Lemma 4, $|E(\mathcal{C}_2) \cap E(\mathcal{C})| \geq |E(\mathcal{B})|$. Consider an arbitrary connected component K of $\mathcal{C}_2$. To prove the lemma, it suffices to prove that K has an edge-2-colorable subgraph K' with $|E(K')| \geq r|E(K) \cap E(\mathcal{C})|$. We distinguish several cases as follows:

Case 1: K is a bicycle of $\mathcal{C}_2$. To obtain an edge-2-colorable subgraph K' of K, we remove one edge e from each odd cycle of K such that one endpoint of e is of degree 3 in K. Note that $|E(K')| = |E(K)| - 2 = |E(K) \cap E(\mathcal{C})| - 1$. Since $|E(K) \cap E(\mathcal{C})| \geq 10$, $|E(K')| \geq \frac{9}{10}|E(K) \cap E(\mathcal{C})| > r|E(K) \cap E(\mathcal{C})|$.

Case 2: K is a tricycle of $\mathcal{C}_2$. To obtain an edge-2-colorable subgraph K' of K, we first remove one edge e from each back odd-cycle of K such that one endpoint of e is of degree 3 in K, and then remove the two edges of the front odd-cycle incident to the vertex of degree 4 in K. Note that $|E(K')| = |E(K)| - 4 = |E(K) \cap E(\mathcal{C})| - 2$. Since $|E(K) \cap E(\mathcal{C})| \geq 15$, $|E(K')| \geq \frac{13}{15}|E(K) \cap E(\mathcal{C})| > r|E(K) \cap E(\mathcal{C})|$.

Case 3: K is neither a bicycle nor a tricycle of $\mathcal{C}_2$. If K contains no odd cycle of $\mathcal{C}$, then K itself is edge-2-colorable and hence we are done. So, assume that K contains at least one odd cycle of $\mathcal{C}$. Then, K is also a connected component of G_2. Moreover, the connected component K'' of G_3 corresponding to K is either an edge or a star.

Case 3.1: K'' is an edge. To obtain an edge-2-colorable subgraph K' of K, we start with K, delete the edge in $E(K) - E(\mathcal{C})$, and delete one edge from the unique odd cycle of K. Note that $|E(K')| = |E(K)| - 2 = |E(K) \cap E(\mathcal{C})| - 1$. Moreover, $|E(K) \cap E(\mathcal{C})| \geq 7$ because of Step 3 and the robustness of Type-1 or Type-2 operations. Hence, $|E(K')| \geq \frac{6}{7}|E(K) \cap E(\mathcal{C})| > r|E(K) \cap E(\mathcal{C})|$.

Case 3.2: K'' is a star. Let C_0 be the connected component of $\mathcal{C}$ corresponding to the center of K''. Let $C_1, \ldots, C_h$ be the odd cycles of $\mathcal{C}$ corresponding to the satellites of K''. If C_0 is a path, then K is a caterpillar graph and we are done by Lemma 1; otherwise, K is a starlike graph and we are done by Lemma 2. ∎

Corollary 1. *If the maximum degree Δ of a vertex in G is at most 3, then the ratio achieved by the algorithm is at least $\frac{6}{7}$.*

Proof. When $\Delta \leq 3$, G_2 has no cherry because of Step 3. Moreover, Lemmas 1, 2, and 5 still hold even when we replace the ratio r by $\frac{6}{7}$. ∎

In order to analyze the approximation ratio achieved by our algorithm when G_2 has at least one type-2 cherry after Step 9, we need to define several notations as follows:

- Let s be the number of special lollipops in G_2.
- Let t be the number of tricycles in G_2.
- Let c be the number of type-2 cherries in G_2.
- Let ℓ be the total number of vertices that appear on odd cycles in the type-2 cherries in G_2.

Lemma 6. *Let $E(\mathcal{B}_2)$ be the set of all edges $e \in E(\mathcal{B})$ such that at least one endpoint of e appears in a type-2 cherry of G_2. Then, $|E(\mathcal{B}_2)| \leq \ell + 2s + 2t$.*

Proof. $E(\mathcal{B}_2)$ can be partitioned into the following three subsets:

- $E(\mathcal{B}_{2,1})$ consists of those edges $e \in E(\mathcal{B})$ such that at least one endpoint of e is the vertex of a type-2 cherry of G_2 that is a common endpoint of the two twigs of the cherry.

– $E(\mathcal{B}_{2,2})$ consists of those edges $e \in E(\mathcal{B})$ such that each endpoint of e appears on an odd cycle of a type-2 cherry of G_2.
– $E(\mathcal{B}_{2,3})$ consists of those edges $\{u, v\} \in E(\mathcal{B})$ such that u appears on an odd cycle of a type-2 cherry of G_2 and no type-2 cherry of G_2 contains v.

Obviously, $|E(\mathcal{B}_{2,1})| \leq 2c$. By Statement 1 in Fact 2, $|E(\mathcal{B}_{2,2})| \leq \ell - 2c$ because for each odd cycle C, $\mathcal{B}_{2,2}$ can contain at most $|V(C)| - 1$ edges $\{u, v\}$ with $\{u, v\} \subseteq V(C)$. By Statement 2 in Fact 2, $|E(\mathcal{B}_{2,3})| \leq 2s + 2t$. So, $|E(\mathcal{B}_2)| \leq \ell + 2s + 2t$. ∎

Lemma 7. *The ratio achieved by the algorithm is at least* r.

Proof. By induction on $|V(G)|$, the number of vertices in the input graph G. If $|V(G)| \leq 2$, then our algorithm outputs a maximum-sized edge-2-colorable subgraph of G. So, assume that $|V(G)| \geq 3$. Then, after our algorithm finishes executing Step 10, the set U may be empty or not. If $U = \emptyset$, then by Lemma 5, the edge-2-colorable subgraph output by our algorithm has at least $r|E(\mathcal{B})|$ edges and we are done. So, suppose that $U \neq \emptyset$.

First consider the case where $s + t \leq \frac{1-r}{2r}\ell$. In this case, $\frac{\ell + r|E(\mathcal{B}_1)|}{\ell + 2s + 2t + |E(\mathcal{B}_1)|} \geq r$, where $\mathcal{B}_1$ is a maximum-sized edge-2-colorable subgraph of $G - U$. Moreover, by the inductive hypothesis, $|E(\mathcal{A}_1)| \geq \ell + r|E(\mathcal{B}_1)|$. Furthermore, by Lemma 6, $|E(\mathcal{B})| \leq \ell + 2s + 2t + |E(\mathcal{B}_1)|$. So, the lemma holds in this case.

Next consider the case where $s + t > \frac{1-r}{2r}\ell$. Let $\mathcal{C}_2$ be the graph obtained from G_2 by removing one edge from each isolated odd-cycle of G_2. By Lemma 4, $|E(\mathcal{C}_2) \cap E(\mathcal{C})| \geq |E(\mathcal{B})|$. Let $\mathcal{C}_3$ be the graph obtained from $\mathcal{C}_2$ by removing one twig from each type-2 cherry. Note that there are exactly c isolated odd-cycles in $\mathcal{C}_3$. Moreover, since the removed twig does not belong to $E(\mathcal{C})$, we have $|E(\mathcal{C}_3) \cap E(\mathcal{C})| \geq |E(\mathcal{B})|$. Consider an arbitrary connected component K of $\mathcal{C}_3$. To prove the lemma, we want to prove that K has an edge-2-colorable subgraph K' with $|E(K')| \geq r|E(K) \cap E(\mathcal{C})|$. This goal can be achieved because of Lemma 5, when K is not an isolated odd-cycle. On the other hand, this goal can not be achieved when K is an isolated odd-cycle (of length at least 5). Our idea behind the proof is to charge the deficit in the edge numbers of isolated odd-cycles of $\mathcal{C}_3$ to the other connected components of K because they have surplus in their edge numbers.

The deficit in the edge number of each isolated odd-cycle of $\mathcal{C}_3$ is at most $5r - 4$. So, the total deficit in the edge numbers of the isolated odd-cycles of $\mathcal{C}_3$ is at most $(5r - 4)c$. We charge a penalty of $6 - 7r$ to each non-isolated odd-cycle of $\mathcal{C}_3$ that is also an odd cycle in a type-2 cherry of G_2 or is also the odd cycle in a special lollipop of G_2. We also charge a penalty of $\frac{6-7r}{3}$ to each odd cycle of $\mathcal{C}_3$ that is part of a tricycle of G_2. Clearly, the total penalties are $(6 - 7r)c + (6 - 7r)(s + t) > (6 - 7r)c + \frac{(6-7r)(1-r)}{2r}\ell$. Note that $\ell \geq 10c$. The total penalties are thus at least $(6 - 7r)c + \frac{5(6-7r)(1-r)}{r}c = \frac{30-59r+28r^2}{r}c \geq (5r - 4)c$, where the last inequality follows from the equation $23r^2 - 55r + 30 = 0$. So, the total penalties are at least as large as the total deficit in the edge numbers of the isolated odd-cycles of $\mathcal{C}_3$. Therefore, to prove the lemma, it suffices to prove that for every connected component K of $\mathcal{C}_3$, we can compute an edge-2-colorable subgraph K' of K such that $|E(K')| - p(K) \geq r|E(K) \cap E(\mathcal{C})|$, where $p(K)$ is the total penalties of the odd cycles in K. As in the proof of Lemma 5, we distinguish several cases as follows:

Case 1: K is a bicycle of $\mathcal{C}_2$. In this case, $p(K) = 0$. Moreover, we can compute an edge-2-colorable subgraph K' of K such that $|E(K')| \geq \frac{9}{10}|E(K) \cap E(\mathcal{C})|$ (cf. Case 1 in the proof of Lemma 5). So, $|E(K')| - p(K) \geq r|E(K) \cap E(\mathcal{C})|$ because $r \leq \frac{9}{10}$.

Case 2: K is a tricycle of $\mathcal{C}_2$. In this case, $p(K) = 6 - 7r$. Moreover, we can compute an edge-2-colorable subgraph K' of K such that $|E(K')| = |E(K) \cap E(\mathcal{C})| - 2$ (cf. Case 2 in the proof of Lemma 5). So, $|E(K')| - p(K) \geq r|E(K) \cap E(\mathcal{C})|$ because $|E(K) \cap E(\mathcal{C})| \geq 15$ and $r \leq \frac{7}{8}$.

Case 3: K is neither a bicycle nor a tricycle of $\mathcal{C}_2$. We may assume that K contains at least one odd cycle of $\mathcal{C}$. Then, K is also a connected component of G_2. Moreover, the connected component K'' of G_3 corresponding to K is either an edge or a star.

Case 3.1: K'' is an edge. In this case, $p(K) \leq 6-7r$. Moreover, we can compute an edge-2-colorable subgraph K' of K such that $|E(K')| = |E(K) \cap E(\mathcal{C})| - 1$ (cf. Case 3.1 in the proof of Lemma 5). So, $|E(K')| - p(K) \geq r|E(K) \cap E(\mathcal{C})|$ because $|E(K) \cap E(\mathcal{C})| \geq 7$.

Case 3.2: K'' is a star. Let C_0 be the connected component of $\mathcal{C}$ corresponding to the center of K''. Let $C_1, \ldots, C_h$ be the odd cycles of $\mathcal{C}$ corresponding to the satellites of K''. If C_0 is a path, then K is a caterpillar graph and we are done by Lemma 1; otherwise, K is a starlike graph and we are done by Lemma 2. ∎

Clearly, each step of our algorithm except Step 12a can be implemented in $O(n^2m)$ time. Since the recursion depth of the algorithm is $O(n)$, it runs in $O(n^3m)$ total time. In summary, we have shown the following theorem:

Theorem 3. *There is an $O(n^3m)$-time approximation algorithm for* Max Simple Edge 2-Coloring *that achieves a ratio of roughly* 0.842.

5 An Application

Let G be a graph. An *edge cover* of G is a set F of edges of G such that each vertex of G is incident to at least one edge of F. For a natural number k, a [1,Δ]*-factor k-packing* of G is a collection of k disjoint edge covers of G. The *size* of a [1,Δ]-factor k-packing $\{F_1, \ldots, F_k\}$ of G is $|F_1| + \cdots + |F_k|$. The problem of deciding whether a given graph has a [1,Δ]-factor k-packing was considered in [7,8]. In [10], Kosowski *et al.* defined the *minimum* [1,Δ]*-factor k-packing problem* (Min-k-FP) as follows: Given a graph G, find a [1,Δ]-factor k-packing of G of minimum size or decide that G has no [1,Δ]-factor k-packing at all.

According to [10], Min-2-FP is of special interest because it can be used to solve a fault tolerant variant of the guards problem in grids (which is one of the art gallery problems [11,12]). Indeed, they proved the NP-hardness of Min-2-FP and the following lemma:

Lemma 8. *If* MAX SIMPLE EDGE 2-COLORING *admits an approximation algorithm A achieving a ratio of α, then* MIN-2-FP *admits an approximation algorithm B achieving a ratio of $2-\alpha$. Moreover, if the time complexity of A is $T(n)$, then the time complexity of B is $O(T(n))$.*

So, by Theorem 3, we have the following immediately:

Theorem 4. *There is an $O(n^3 m)$-time approximation algorithm for* MIN-2-FP *achieving a ratio of roughly* 1.158*, where n (respectively, m) is the number of vertices (respectively, edges) in the input graph.*

6 Open Problems

One obvious open question is to ask whether one can design a polynomial-time approximation algorithm for MAX SIMPLE EDGE 2-COLORING that achieves a ratio significantly better than 0.842. The APX-hardness of the problem implies an implicit lower bound of $1-\epsilon$ on the ratio achievable by a polynomial-time approximation algorithm. It seems interesting to prove an explicit lower bound significantly better than $1-\epsilon$.

References

1. Chen, Z.-Z., Tanahashi, R.: Approximating Maximum Edge 2-Coloring in Simple Graphs via Local Improvement. Theoretical Computer Science (special issue on AAIM 2008) 410, 4543–4553 (2009)
2. Chen, Z.-Z., Tanahashi, R., Wang, L.: An Improved Approximation Algorithm for Maximum Edge 2-Coloring in Simple Graphs. Journal of Discrete Algorithms 6, 205–215 (2008)
3. Feige, U., Ofek, E., Wieder, U.: Approximating Maximum Edge Coloring in Multigraphs. In: Jansen, K., Leonardi, S., Vazirani, V.V. (eds.) APPROX 2002. LNCS, vol. 2462, pp. 108–121. Springer, Heidelberg (2002)
4. Gabow, H.: An Efficient Reduction Technique for Degree-Constrained Subgraph and Bidirected Network Flow Problems. In: Proceedings of the 15th Annual ACM Symposium on Theory of Computing (STOC 1983), pp. 448–456. ACM, New York (1983)
5. Hartvigsen, D.: Extensions of Matching Theory. Ph.D. Thesis, Carnegie-Mellon University (1984)
6. Hochbaum, D.: Approximation Algorithms for NP-Hard Problems. PWS Publishing Company, Boston (1997)
7. Jacobs, D.P., Jamison, R.E.: Complexity of Recognizing Equal Unions in Families of Sets. Journal of Algorithms 37, 495–504 (2000)
8. Kawarabayashi, K., Matsuda, H., Oda, Y., Ota, K.: Path Factors in Cubic Graphs. Journal of Graph Theory 39, 188–193 (2002)
9. Kosowski, A.: Approximating the Maximum 2- and 3-Edge-Colorable Subgraph Problems. Discrete Applied Mathematics 157, 3593–3600 (2009)
10. Kosowski, A., Malafiejski, M., Zylinski, P.: Packing $[1, \Delta]$-Factors in Graphs of Small Degree. Journal of Combinatorial Optimization 14, 63–86 (2007)
11. O'Rourke, J.: Art Gallery Theorems and Algorithms. Oxford University Press, Oxford (1987)
12. Urrutia, J.: Art Gallery and Illumination Problems. In: Handbook on Computational Geometry. Elsevier Science, Amsterdam (2000)