The 7th International Symposium on Operations Research and Its Applications (ISORA'08)
Lijiang, China, October 31–Novemver 3, 2008

Approximate maximum edge coloring within factor 2: a further analysis*

Wangsen Feng   Ping Chen   Bei Zhang

Computing Center, Peking University, Beijing 100871, China
E-mail : {fengws,pchen,zb}@pku.edu.cn

Abstract

In [1], Feng et al. propose a polynomial time approximation algorithm for a novel maximum edge coloring problem which arises from the field of wireless mesh networks [2]. The problem is about coloring all the edges in a graph and finding a coloring solution which uses the maximum number of colors with the constraint, for every vertex in the graph, all the edges incident to it are colored with no more than $q(q \in \mathbb{Z},\ q \geq 2)$ colors. The case $q = 2$ is of great importance in practice. The algorithm is shown to achieve a factor of 2.5 for case $q = 2$ and $1 + \frac{4q-2}{3q^2-5q+2}$ for case $q > 2$ respectively. In this paper, we give a further analysis of the algorithm and improve the ratio from 2.5 to 2 for case $q = 2$. The ratio 2 is shown to be tight with a tight example. We also study maximum edge coloring in complete graphs and trees.

1 Introduction

Graph coloring problems occupy an important place in graph theory. Generally, there are two types of coloring: vertex coloring and edge coloring. For vertex coloring, Brooks [4] states that $\chi(G) \leq \Delta(G)$ for any graph G except complete graphs K_n and odd circles C_{2k+1}, where *chromatic number* $\chi(G)$ is the minimum number of colors needed in a vertex coloring of graph G. Karp [5] proves that to determine $\chi(G)$ is an NP-hard problem . If P $\neq$ NP holds, Garey and Johnson [6] point out that there is even no polynomial time approximation algorithm with ratio 2. However, Turner [7] designs an algorithm of complexity $O(|V| + |E|\log k)$ and with probability almost 1 to color a given k-colorable graph with k colors for the case that k is not too large relative to $|V|$. For edge coloring, Vizing [8] states that for any graph G, either $\chi^{'}(G) = \Delta(G)$ or $\chi^{'}(G) = \Delta(G) + 1$, where *chromatic index* $\chi^{'}(G)$ is the minimum number of colors needed in an edge coloring of G. Holyer [9] proves that it is also an NP-hard problem to determine $\chi^{'}(G)$. The proof of Vizing Theorem yields an approximation algorithm for this problem which finds an edge coloring solution using $\Delta(G) + 1$ colors within one of optimal. Recently, Uriel Feige et al. [10] have investigated the maximum edge t-coloring problem in multigraphs. The problem is to color as many edges as possible using t colors, such that no pairs of adjacent

*Supported by the National High-tech Research and Development Program (863) of China under Grant No. 2006AA01Z160.

1. **If** $(q = 2)$ **Then**
   Compute a maximum matching M in G;
   Else
   Compute a maximum $(q-1)$-matching M_{q-1} in G;
2. Assign a new color to each edge in $M(M_{q-1})$;
3. Delete the edges of $M(M_{q-1})$ from the original graph G and for each connected component of the residual graph $G^{'}$ which is not an isolated vertex, assign to it a new color;
4. Output each edge with the color assigned to it.

Figure 1: Algorithm 1

edges are colored with the same color. They show that the problem is NP-hard and design constant factor approximation algorithms for it.

The problems mentioned above are all traditional coloring ones, they obey the same rule: no two adjacent vertices(edges) are colored with the same color. However, in the maximum edge coloring problem proposed in [1], two adjacent edges are not necessary to be colored with different colors. It is defined as follows:

Maximum edge coloring problem: Given a connected undirected simple graph $G = (V, E)$ and a positive integer $q \geq 2$, color all the edges in E, with the constraint, for every vertex in V, all the edges incident to it are colored with no more than q colors, ask for a solution which uses maximum number of colors.

The problem arises from the field of wireless mesh networks. Because the mesh routers in a wireless mesh network often have two network interface cards, the case $q = 2$ is very important. For more details, readers are referred to [2, 3]. In [1], a polynomial time approximation algorithm (Algorithm 1) is designed for the problem (Figure 1). It achieves an approximation factor of 2.5 for case $q = 2$ and a factor of $1 + \frac{4q-2}{3q^2-5q+2}$ for case $q > 2$. In this paper, we prove that Algorithm 1 is a 2-approximation algorithm for case $q = 2$ and show the ratio 2 is tight. For complete graphs and trees, polynomial time accurate algorithms are found for them when $q = 2$.

In order to have a better understanding of Algorithm 1, let's review the maximum b-matching problem simply.

Maximum b-matching problem: Given an undirected graph $G = (V, E)$ and a function b: $V \rightarrow \mathbb{Z}^+$ specifying an upper bound for each vertex, the maximum b-matching problem asks for a maximum cardinality set $M \subseteq E$ such that $\forall\, v \in V,\ deg_M(v) \leq b(v)$.

The results on matchings are strongly self-refining. By applying splitting techniques to ordinary matchings, maximum b-matchings can be found in polynomial time too. Gabow [13] designed an algorithm of complexity $O(|V||E|\log|V|)$ for the problem in 1983.

Now, we introduce some notations frequently used below. $ALG(G)$ is used to denote the number of colors used in the solution given by Algorithm 1 on an input graph G; $OPT(G)$ to denote the number of colors used in an optimal coloring solution of G. For more knowledge on approximation algorithms, readers are referred to [11].

1.1 Previous Results

Given an arbitrary connected graph G, suppose the optimal solutions use m colors: $1, 2, \ldots, m$. Based on the color of each edge, the edge set can be divided into m subsets: $E_1, E_2, \ldots, E_m$. Each subset E_i denotes the set of edges colored with color i. If we choose one edge from each subset, the subgraph H induced by these m edges are called "character subgraph" of G.

Lemma 1: (Feng et al. [1]) *For a character subgraph H of a connected graph $G = (V, E)$, it satisfies:*
1)$\Delta(H) \leq q$;
2)If $q = 2$, then H consists of paths and cycles;
3)If $q = 2$, $OPT(G) \leq |V|$.

Lemma 2: (Feng et al. [1]) *Given a vertex cover V^* of a graph G with $|V^*| = k$, let H be the subgraph induced by V^* in G. Then:*
1)$OPT(G) \leq kq$;
2)If H has a matching of size m, then $OPT(G) \leq kq - m$;
3)If $q = 2$ and H is connected, then $OPT(G) \leq k + 1$;
4)If $q = 2$ and H has l connected components $(1 \leq l \leq k)$, then $OPT(G) \leq k + l$.

Theorem 1: (Feng et al. [1]) *For any connected graph G, Algorithm 1 achieves an approximation factor of 2.5 for case $q = 2$ and a factor of $(1 + \frac{4q-2}{3q^2-5q+2})$ for case $q > 2$.*

2 Further analysis of Algorithm 1 for case $q = 2$

Before discussing general graphs, let's see what will take place if input graphs are restricted to be bipartite graphs. In bipartite graphs, there exists the equation $max_{matching\ M}|M| = min_{vertex\ cover\ U}|U|$. Combined with Lemma 2, it is easy to see that

$$\frac{OPT(G)}{ALG(G)} \leq \frac{2|U_{min}|}{|M_{max}|} \leq 2 \tag{1}$$

In fact, for general graphs, we have the same result and this ratio is better than that in Theorem 1.

Theorem 2: *For any connected graph G, Algorithm 1 achieves an approximation factor of 2.*

Proof: Let $OPT(G) = m$ and H be a character subgraph of G. According to Lemma 1, H is a set of paths and cycles. The theorem is proved by two steps:
1) Construct a matching in G with size $\geq \lfloor \frac{m}{2} \rfloor$ based on H.
2) According to the result in 1), we can easily draw the conclusion:

$$\frac{OPT(G)}{ALG(G)} \leq 2 \tag{2}$$

Step 1): A path of odd(even) length is called an odd(even) path. Similarly, a cycle of odd(even) length is called an odd(even) cycle. Denote odd paths, even paths, odd cycles and even cycles in H by OP_i, EP_j, OC_s and EC_t respectively. Use $l(OP_i)$ $(0 \le i \le p_1)$, $l(EP_j)$ $(0 \le j \le p_2)$, $l(OC_s)$ $(0 \le s \le c_1)$ and $l(EC_t)$ $(0 \le t \le c_2)$ to denote the lengths of OP_i, EP_j, OC_s and EC_t respectively. Clearly, for the paths or cycles of even length $2k$, the size of their maximum matchings is k. For the paths of odd length $2k+1$, the size is $k+1$, and for cycles of odd length $2k+1$, the size is k. We can denote the number of edges in H, m, as follows:

$$m = \sum_{i=1}^{p_1} l(OP_i) + \sum_{j=1}^{p_2} l(EP_j) + \sum_{s=1}^{c_1} l(OC_s) + \sum_{t=1}^{c_2} l(EC_t) \tag{3}$$

And the size of a maximum matching M_H in H is:

$$|M_H| = \sum_{i=1}^{p_1} \frac{1}{2}[l(OP_i)+1] + \sum_{j=1}^{p_2} \frac{1}{2} l(EP_j) + \sum_{s=1}^{c_1} \frac{1}{2}[l(OC_s)-1] + \sum_{t=1}^{c_2} \frac{1}{2} l(EC_t) \tag{4}$$

Case 1: Clearly, if $c_1 = 0$, then $|M_H| \ge \lfloor \frac{m}{2} \rfloor$. M_H is the matching we want to construct.

Case 2: When $c_1 = 1$, there is one odd cycle, OC_1, in H. We can construct a matching M' with $|M'| \ge \lfloor \frac{m}{2} \rfloor$ as follows:

subcase 1): $G = H = OC_1$, which means the original graph is just an odd cycle, then we can let M' be a maximum matching of OC_1. Clearly, $|M'| \ge \lfloor \frac{m}{2} \rfloor$.

subcase 2): OC_1 is a real subgraph of G, which means there is at least one vertex $v \in V(G)$ and $v \notin V(OC_1)$, since there cannot be any other edge among the vertices of $V(OC_1)$ in G. Clearly, there is no edge in G among those vertices in H with $deg_H(v) = 2$. For each 1-degree vertex in a path of H, it cannot be adjacent to two 2-degree vertices in distinct connected components of H. Otherwise, it will contradict the fact that the optimal coloring solution is feasible. Because G is connected and $G \ne OC_1$, we can always find a vertex v_1 in an odd cycle in H and v_1 connects to an outside vertex v_2, which is not in the cycle. Based on the above analysis, v_2 must belong to one of the following three sets: V_1={the vertices not in H}; V_2={the 1-degree vertices in even paths in H}; V_3={the 1-degree vertices in odd paths in H}. Now, let's discuss how to construct M'.

1) If $v_2 \in V_1$, construct a maximum matching M_C of OC_1 leaving v_1 as an unsaturated vertex, let $M'_C = M_C \cup \{e = (v_1, v_2)\}$. Clearly, $|M'_C| = \frac{1}{2}[l(OC_1) - 1] + 1 > \frac{1}{2} l(OC_1)$.

2) If $v_2 \in V_2$, construct a maximum matching M_C of OC_1 leaving v_1 as an unsaturated vertex, find a maximum matching M_P of the even path EP_1 leaving v_2 as an unsaturated vertex, let $M'_C = M_C \cup M_P \cup \{e = (v_1, v_2)\}$. Clearly, $|M'_C| = \frac{1}{2}[l(OC_1) - 1] + \frac{1}{2} l(EP_1) + 1 > \frac{1}{2}[l(OC_1) + l(EP_1)]$.

3) If $v_2 \in V_3$, construct maximum matchings M_C, M_P of OC_1 and the odd path OP_1 respectively, let $M'_C = M_C \cup M_P$. Clearly, $|M'_C| = \frac{1}{2}[l(OC_1) - 1] + \frac{1}{2}[l(OP_1) + 1] = \frac{1}{2}[l(OC_1) + l(OP_1)]$.

For the rest connected components in H, find one maximum matching M_R in them, let $M^{'} = M_R \cup M^{'}_C$. Obviously, $|M^{'}| \geq \lfloor \frac{m}{2} \rfloor$, $M^{'}$ is the matching we want to construct.

Case 3: Now, let's discuss the case $c_1 > 1$. First a new graph G/H is constructed by contracting(shrinking) every connected component H_i of H into a new vertex v_i $(1 \leq i \leq p_1 + p_2 + c_1 + c_2)$. Clearly, G/H has vertex set $(V(G)/V(H)) \cup \{v_1, v_2 ..., v_{p_1+p_2+c_1+c_2}\}$, and for each edge e in G, an edge of G/H is obtained from e by replacing any end point in H_i by the new vertex v_i. (Here we ignore loops and multiple edges that may arise.) Obviously, G/H is also connected.

When there is an edge in G/H between an original vertex v, which is not in H but in G, and a new vertex coming from H_i, it means there is an edge in G between v and a vertex in H_i. When there is an edge in G/H between a new vertex from H_i and another new vertex from H_j, it means there is an edge in G between a vertex in H_i and a vertex in H_j. Clearly, there is no edges among the new vertices from cycle components in G/H and each such vertex only connects to vertices which are either new vertices from a path component or original vertices. For convenience, new vertices from path components and original vertices are called as *compatible vertices*. For each compatible vertex, it can be adjacent to two new vertices from cycle components at most. For one new vertex from a path component, if it connects to two new vertices from cycle components in G/H, then it must be that each of its two 1-degree end points connects to a vertex in one of the two cycle components in G respectively.

Denote by $U = \{u_1, u_2, ..., u_{c_1}\}$ the set of new vertices from odd cycles. Then we introduce a procedure to extract a set of compatible vertices from G/H which can dominate U. The graph output by the procedure is called matching graph B. (See Figure 2)

1. $B = \emptyset$;
2. **while** $(U \neq \emptyset)$

{
1)Take an element u from U, scan its neighbors in G/H;
2)**if** (u is adjacent to a compatible vertex v by edge e and v doesn't connect to any other new vertex from odd cycle)
then
$\{$ Add u, v and e into B, $U = U - \{u\}$; $\}$
else if (u is adjacent to a compatible vertex v by edge e and v also connects to another new vertex from odd cycle which has been added into B)
then
$\{$ Add u, v and e into B, $U = U - \{u\}$; $\}$
else (in this case, u must only connect to those compatible vertices which connect to two elements which are still in U at this time)
$\{$ Suppose u is adjacent to a compatible vertex v by edge e_1 and v also connects to another new vertex u' in U by edge e_2.
Add u, u', v and e_1, e_2 into B, $U = U - \{u, u'\}$. $\}$
}

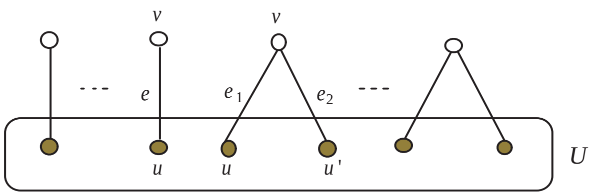

Figure 2: Matching graph B: the filled vertices are new vertices from odd cycles, the empty ones correspond to compatible vertices.

3. output B.

For u_i in a path of length 1 in B, the case is similar to the case $c_1 = 1$. We emphasize the case of u_i in a path of length 2 in B. When u_i is in a path of length 2, it means two new vertices from odd cycles connect to the same compatible vertex. Denote by OC_1, OC_2 the two odd cycles. Now, let's discuss how to construct $M^{'}$.

1) If the compatible vertex is an original vertex, say v, then we can always find v_1 in OC_1, v_2 in OC_2, which connect to v in G. Construct a maximum matching M_C of OC_1 and OC_2 leaving v_1 as an unsaturated vertex, let $M_C^{'} = M_C \cup \{e = (v, v_1)\}$. Clearly, $|M_C^{'}| = \frac{1}{2}[(l(OC_1) - 1) + (l(OC_2) - 1)] + 1 = \frac{1}{2}[l(OC_1) + l(OC_2)]$.

2) If the compatible vertex is a new vertex from an even path EP_1. Then we can always find v_1 in OC_1, v_2 in OC_2, which connect to the two 1-degree nodes, v_3, v_4, in EP_1 in G respectively. Construct a maximum matching M_C of OC_1 and OC_2 leaving v_1, v_2 as unsaturated vertices, find the maximum matching M_P in EP_1 leaving v_3 as a saturated vertex, let $M_C^{'} = M_C \cup M_P \cup \{e_1 = (v_1, v_3)\}$. Clearly, $|M_C^{'}| = \frac{1}{2}[(l(OC_1) - 1) + (l(OC_2) - 1)] + \frac{1}{2}l(EP_1) + 1 = \frac{1}{2}[l(OC_1) + l(OC_2) + l(EP_1)]$.

3) If the compatible vertex is a new vertex from an odd path OP_1. Then we can always find v_1 in OC_1, v_2 in OC_2, which connect to the two 1-degree nodes, v_3, v_4, in OP_1 in G respectively. Construct a maximum matching M_C of OC_1 and OC_2 leaving v_1, v_2 as unsaturated vertices, find the maximal matching M_P in OP_1 leaving v_3, v_4 as unsaturated vertices, let $M_C^{'} = M_C \cup M_P \cup \{e_1 = (v_1, v_3), e_2 = (v_2, v_4)\}$. Clearly, $|M_C^{'}| = \frac{1}{2}[(l(OC_1) - 1) + (l(OC_2) - 1)] + \frac{1}{2}[l(OP_1) - 1] + 2 > \frac{1}{2}[l(OC_1) + l(OC_2) + l(OP_1)]$.

Thus we can always construct a matching $M^{'}$ of G with size $\geq \lfloor \frac{m}{2} \rfloor$ as follows:

1. Induce the subgraph H;

2. **if** ($c_1 = 0$) **then** { let $M^{'} = M_H$; }
   else if (G is an odd cycle)
   then { let $M^{'}$ be a maximum matching of G; }
   else {
   1) shrink G into G/H;
   2) extract the matching graph B from G/H;

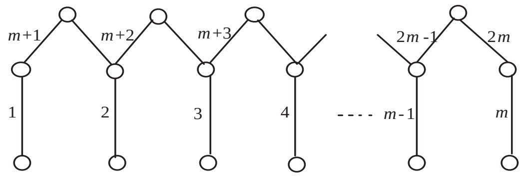

Figure 3: tight example for Algorithm 1

3) for each connected component in B, construct $M_C^{'}$ as above;
4) for the rest connected components in H, which is not in B, construct one maximum matching M_R in them;
5) let $M^{'} = (\ \bigcup M_C^{'}\ ) \cup M_R$;
}

Step 2): Since M is a maximum matching of G, it is easy to see:

$$\frac{OPT(G)}{ALG(G)} \leq \frac{m}{|M|+1} \leq \frac{m}{|M'|+1} \leq \frac{m}{\lfloor \frac{m}{2} \rfloor + 1} \leq \frac{m}{m/2} = 2 \tag{5}$$

Here, we assume that the residual graph $G^{'} = G - M$ has at least one edge. Because if $G^{'}$ has no edge, $M = G$, thus $\Delta(G) < 2$. This case is trivial: $ALG(G) = OPT(G) = |E|$, Theorem 2 follows immediately. □

The following graph gives a tight example for Algorithm 1.

Example 1: In the graph shown in Figure 3, the set of vertical edges is a maximum matching of G; on the other hand, G can be colored with $2m$ colors at most. Thus, $ALG(G) = m+1$, $OPT(G) = 2m$.

3 Maximum edge coloring in complete graphs and trees

For complete graphs and trees, we can get an accurate solution when $q = 2$. Obviously, $OPT(K_3) = 3$. For $K_n (n \geq 4)$, Theorem 3 stands.

Theorem 3: *For a complete graph K_n $(n \geq 4)$, $OPT(K_n) = \lfloor \frac{n}{2} \rfloor + 1$.*

A vertex in a tree is called an internal vertex, if and only if it is of degree at least two. If a tree is just an edge, then there is no internal vertex in it.

Theorem 4: *For any tree T, $OPT(T) = |V_{in}| + 1$, where V_{in} is the set of internal vertices in T.*

References

[1] Wangsen Feng, Li'ang Zhang, Wanling Qu and Hanpin Wang: Approximation algorithms for maximum edge coloring problem. *TAMC* 2007, LNCS 4484: 646-658.

[2] Ashish Raniwala, Tzi-cker Chiueh: Architecture and algorithms for an IEEE 802.11-based multi-channel wireless mesh network. *INFOCOM* 2005: 2223-2234.

[3] Ashish Raniwala, Kartik Gopalan, Tzi-cker Chiueh: Centralized channel assignment and routing algorithms for multi-channel wireless mesh networks. *Mobile Computing and Communications Review* 8(2): 50-65; 2004.

[4] Brooks, R.L. On colouring the nodes of a network. *Proc. Cambridge Phil. Soc.* 37:194-197, 1941.

[5] Karp, R.M. Reducibility among combinatorial problems. In: *Complexity of computer computations* (Eds. R.E.Miller and J.W.Thatcher.) Plenum Press, New York, 1972: 85-103.

[6] Garey, M.R. and Johnson, D.S. The complexity of near optimal graph coloring. *J. ACM* 23: 43-49 1976.

[7] Turner, J.S. Almost all k-colorable graphs are easy to color. *J. Algor.* 9: 63-82, 1988.

[8] Vizing V.G. On an estimate of the chromatic class of a p-graph. (in Russian) *Diskret. Analiz.* 3: 25-30, 1964.

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

[10] Uriel Feige, Eran Ofek and Udi Wieder: Approximating maximum edge coloring in multigraphs. *APPROX* 2002: 108-121.

[11] Vijay V. Vazirani. Approximation algorithms. Springer-Verlag, Berlin, 2001.

[12] S. Micali, Vijay V. Vazirani. An $O(|V|^{\frac{1}{2}}|E|)$ algorithm for finding maximum matching in general graphs. *Proc.* 21^{st} *IEEE FOCS*, 1980: 17–27.

[13] H.N. Gabow. An efficient reduction technique for degree-constrained subgraph and bidirected network flow problems. *Proc.* 15^{th} *ACM STOC*, 1983: 448-456.