Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 12, no. 4, pp. 401–417 (2008)

Fast edge colorings with fixed number of colors to minimize imbalance

Gruia Călinescu[1] *Michael J. Pelsmajer*[2]

[1]Department of Computer Science,
Illinois Institute of Technology, Chicago, USA
[2] Department of Applied Mathematics,
Illinois Institute of Technology, Chicago, USA

Abstract

We study the following optimization problem: the input is a multigraph $G = (V, E)$ and an integer parameter g. A feasible solution consists of a (not necessarily proper) coloring of E with colors $1, 2, \ldots, g$. Denote by $d(v, i)$ the number of edges colored i incident to v. The objective is to minimize $\sum_{v \in V} \max_i d(v, i)$, which roughly corresponds to the "imbalance" of the edge coloring. This problem was proposed by Berry and Modiano (INFOCOM 2004), with the goal of optimizing the deployment of tunable ports in optical networks. Following them we call the optimization problem MTPS - Minimum Tunable Port with Symmetric Assignments.

Among other results, they give a reduction from Edge Coloring showing that MTPS is NP-Hard and then implicitly give a 2-approximation algorithm. We give a (3/2)-approximation algorithm. Key to this problem is the following question: given a multigraph $G = (V, E)$ of maximum degree g, what fraction of the vertices can be properly edge-colored in a coloring with g colors, where a vertex is properly edge-colored if the edges incident to it have different colors? Our main lemma states that there is such a coloring with half of the vertices properly edge-colored. For $g \leq 4$, two thirds of vertices can be made properly edge-colored.

Our algorithm is based on g Maximum Matching computations (total running time $O(gm\sqrt{n + m/g})$, where $n = |V|$ and $m = |E|$) and a local optimization procedure, which by itself gives a 2-approximation. An interesting analysis gives an expected $O((gn + m)\log(gn + m))$ running time for the local optimization procedure.

Submitted:	Reviewed:	Revised:	Accepted:	Final:
June 2007	July 2008	September 2008	September 2008	November 2008

Published:
December 2008

Article type: Regular paper
Communicated by: S. Albers

E-mail addresses: calinescu@iit.edu (Gruia Călinescu) pelsmajer@iit.edu (Michael J. Pelsmajer)

1 Introduction

Berry and Modiano [2] study the benefits of using tunable electronic ports in WDM/TDM Optical Networks. They provide formulations for the "tunable" optimization problems of reducing the number of tunable electronic ports. These ports are very expensive and optimal placement is very desirable.

They introduce two optimization problem. In this paper we concentrate on the Minimum Tunable Port with Symmetric Assignments (MTPS) problem: The input is a multigraph $G = (V, E)$ and an integer parameter g. A feasible solution consists of a (not necessarily proper) coloring (called a *g-edge coloring*) of E with colors $1, 2, \ldots, g$. Denote by $d(v, i)$ the number of edges colored i incident to v. The objective is to minimize $\sum_{v \in V} \max_i d(v, i)$.

Actually, the MTPS problem as described in [2, 3] has a different description, with colors corresponding to wavelengths/time-slots, and $\max_i d(v, i)$ being the number of ports required at vertex v. They give a non-trivial equivalence reduction to the formulation above, which they use for proving NP-Completeness and for approximation algorithms. For $g = 3$, they show the problem is NP-Complete by an easy reduction from Edge Coloring in cubic graphs [8]. Indeed, one can see that a proper 3-edge coloring (that is, a coloring with 3 colors where no two edges incident to a vertex have the same color) of a cubic graph is the only way a 3-edge coloring can have objective function in MTPS equal to $|V|$. The result of [8] can be used (though we do not prove this here) to show that MTPS is APX-Hard: that is no $(1 + \epsilon)$-approximation algorithm exists unless P=NP [1].

An edge coloring is called *equitable* [6] if for all vertices v and colors i, j, we have $d(v, i) \leq d(v, j) + 1$. It is not hard to see that if an equitable exists, then it minimizes the objective function [3]. Certain classes of graphs, for example simple graphs where no vertex has degree multiple of g, are known to have equitable g-edge colorings [6].

Berry and Modiano [3] implicitly give a conceptually simple 2-approximation algorithm, which we describe later. We give a (3/2)-approximation algorithm. Key to the MTPS problem is the following question: given a graph $G = (V, E)$ of maximum degree g, what fraction of the vertices can be *properly edge-colored* in a g-edge coloring, where a vertex is properly edge-colored (or just *proper*) if the edges incident to it have different colors? Our main lemma states that there is a g-edge coloring with half of the vertices properly edge-colored. For $g \leq 4$, two thirds of vertices can be made properly edge-colored; this bound is tight. We leave as an open question the problem of finding tight bounds for larger values of g.

Our algorithm for $g > 4$ is based on g Maximum Matching computations (total running time $O(gm\sqrt{n + m/g})$, where $n = |V|$ and $m = |E|$) and a local optimization procedure, which by itself gives a 2-approximation. By carefully implementing this local optimization procedure (that is, removing the freedom inherent in local optimization thus obtaining a specific algorithm) , we prove it has an expected $O((gn + m) \log(gn + m))$ running time. This implementation is needed to ensure the overall running time is not dominated by local optimization. Local optimization would be a top choice of a practitioner, and depending on the size of the instance, it may be important to have a fast implementation. For $g = 3$ and $g = 4$ we obtain a 4/3-approximation algorithm with running time of $O((n+m)\log n)$ and $O(n^2 + m^2)$, respectively.

A related problem was considered by Feige et. al. [5] In Maximum Edge Coloring, given a multigraph $G = (V, E)$ and a parameter g, one seeks a subgraph with maximum number of edges which can be properly edge-colored with g colors. They show that Maximum Edge Coloring is Max-SNP Hard, even for $g = 2$, and give constant approximation algorithms.

Our paper is organized as follow. The next section presents preliminaries, the case $g = 2$, introduces the local optimization procedure, and gives two simple 2-approximation algorithms. Section 3 gives the approximation ratio of the algorithms, ignoring implementation details and the analysis of the running time of the algorithms, which appear in Section 4.

2 Preliminaries

All our graphs are multigraphs, unless stated otherwise. For $F \subseteq E(G)$, we use $G \setminus F$ to denote the graph $(V(G), E(G) \setminus F)$. For $A \subseteq V(G)$, we use $G \setminus A$ to denote the subgraph of G induced by $V(G) \setminus A$.

The obvious lower bound for the optimum is $L = \sum_{v \in V} \lceil \frac{d(v)}{g} \rceil$, where $d(v)$ is the degree of vertex v. Berry and Modiano [2] use L as a lower bound, and we will do the same.

We call a vertex v *unbalanced* in a g-edge coloring if there are colors i and j with $d(v,i) > d(v,j) + 1$, and *grossly unbalanced* if there are colors i and j with $d(v,i) > d(v,j) + 2$. A vertex that is unbalanced but not grossly unbalanced contributes at most $\lceil \frac{d(v)}{g} \rceil + 1$ to the objective function. Vertices which are not unbalanced are called *balanced*.

The paper [6] describes how to compute the optimum for MTPS when $g = 2$. We include their method for completeness. We may consider each graph component separately, so let us assume that G is connected. If G is Eulerian with even number of edges, then following an Eulerian tour and coloring edges with alternate colors gives us an equitable 2-edge coloring of G. If G is Eulerian with an odd number of edges, [6] shows that no equitable 2-edge coloring exists; following the Eulerian tour and using alternate colors results in a coloring with one vertex unbalanced but not grossly unbalanced and a 2-edge coloring with objective $L+1$. If G is not Eulerian we add extra edges between vertices having odd degree to make it Eulerian, then use alternate colors on the Eulerian tour, starting with an extra edge. Removing the extra edges results in a 2-edge coloring with objective L.

Next we describe a local optimization procedure, called *quasibalancing*, that can be used to improve an edge coloring to ensure that no grossly unbalanced vertices exist, without creating additional unbalanced vertices. Suppose that $d(v,i) > d(v,j) + 2$ for some vertex v. Consider the subgraph induced by colors i and j and let H be the component containing v. Use the $g = 2$ procedure

```
1 replace each $u \in V$ by $\lceil \frac{d(v)}{g} \rceil$ copies $u_i$
2 **for** each edge $e \in E(G)$ with endpoints $u, v \in V$
3       replace the endpoints of $e$ with $u_i$, $v_j$,
        picking $i, j$ such that $d(u_i) \leq g$ and $d(v_j) \leq g$
4 **endfor**
5 color the edges with $g$ colors
```

Figure 1: Pseudocode of the reduction from MTPS to $\overline{MTPS}$

to recolor H such that no vertex of H remains unbalanced in i and j except for possibly v. It is easy to check that this procedure reduces the quantity $\sum_{u\in V} \sum_{1\leq k<l\leq g} |d(u,k) - d(u,l)|$ and it does not make balanced vertices unbalanced or create any new grossly unbalanced vertices. The repeated application of the procedure results in a g-edge coloring without grossly unbalanced vertices. At the end, $\max_i d(v,i) \leq 1 + \lceil \frac{d(v)}{g} \rceil$, so $\sum_{v\in V} \max_i d(v,i) \leq L + n \leq 2L$, which shows that we have a 2-approximation algorithm. Pseudocode for one local improvement step appears later in this paper (Figure 3, in Section 4), when we analyze the running time of our algorithms.

We define the $\overline{MTPS}$ problem to be the restriction of MTPS to instances where every vertex has degree at most g. If one uses L as a lower bound (as we do), $\overline{MTPS}$ is equivalent from approximation point of view: we give a simple reduction (also used in [2]) from MTPS to $\overline{MTPS}$. Starting with an instance G of MTPS we construct an instance G' of $\overline{MTPS}$ as follows. For every vertex $v \in V(G)$ add to $V(G')$ vertices v_i, for $1 \leq i \leq \lceil \frac{d(v)}{g} \rceil$. Edges of G are processed one by one, and for edge $e \in E(G)$ with endpoints u and v, we choose i, j smallest such that $d(u_i) < g$ and $d(v_i) < g$, then add to G' the edge e' with endpoints u_i and v_j. Since $d(v) \leq g\lceil \frac{d(v)}{g} \rceil$, all edges incident to v can be assigned to vertices v_i in this way. Then $L = L' = |V(G')|$. Any g-edge coloring of G' translates back to G without an increase in the objective function. Pseudocode appears in Figure 1.

Berry and Modiano's [3] implicit 2-approximation algorithm works as follows. Given a multigraph G, replace each vertex v by $\lceil 3d(v)/(2(g-1)) \rceil$ copies (their conference version [2] uses $\lceil 3d(v)/2g \rceil$ copies, which is not enough when $d = 800$ and $g = 4$, as pointed out by a referee of this paper!) and distribute the endpoints of edges that had been incident to v evenly among its copies, so that each copy has degree at most $(2/3)g$. Then Shannon's bound [13] gives a proper g-edge-coloring of the modified graph; transferring the colors to edges of G yields $d(v,i) \leq 2\lceil d(v)/g \rceil$ for each vertex v and color i. Thus for $g \geq 4$ the objective is at most $2L$, as claimed. For $g = 3$ one uses $\lceil d(v)/2 \rceil$ copies for each v and colors the resulting graphs of degree 2 with 3 colors.

One may ask why one cannot use newer results on edge coloring such as [7, 11, 12] instead of the sharp bound of [13]. The newer results use (besides the maximum degree) the following lower bound on the chromatic index of a graph:

$\max_{H\subseteq V}|E(H)|/\lfloor|H|/2\rfloor$, where $E(H)$ is the set of edges in the subgraph of G induced by H. This lower bound does not relate at all to our lower bound L. We leave open the search for lower bounds better than L.

Both the 2-approximation obtained by [2, 3] and the one obtained by quasi-balancing suggest the hardest instances for MTPS are g-regular graphs. Since we are using L as a lower bound, we restrict the rest of this paper to the $\overline{MTPS}$ problem. Note that for $\overline{MTPS}$ instances a vertex is properly colored if and only if it is balanced.

3 The approximation ratio

The main result of this section is:

Lemma 1 *Given a graph $G = (V, E)$ of maximum degree g, there is a g-edge coloring with at most $\lfloor\frac{|V|-1}{2}\rfloor$ unbalanced vertices. Such a coloring can be obtained in polynomial time.*

Proof: The proof is by induction on $n+g$, where $n = |V|$. The base cases $n = 1$ and $n = 2$ are trivial. If G is not connected, let $G_1 = (V_1, E_1)$ be one connected component and $G_2 = (V_2, E_2)$ be the remaining graph. Induction gives a g-edge coloring with at most

$$\lfloor\frac{|V_1|-1}{2}\rfloor + \lfloor\frac{|V_2|-1}{2}\rfloor \leq \lfloor\frac{|V_1|+|V_2|-1}{2}\rfloor$$

unbalanced vertices. Thus we assume G is connected.

We wish to make use of a maximum matching of G, so we apply the Edmonds-Gallai decomposition theorem to the underlying simple graph of G. The statement of this theorem, as in, for example, [9, p94], is described in the next sentence. In polynomial time, using Edmonds' Maximum Matching Algorithm [4], we obtain a set $A \subseteq V$ such that $G \setminus A$ has components $B_1, B_2, \ldots, B_k$ and $D_1, D_2, \ldots D_j$ such that:

- for each $1 \leq i \leq j$ D_i has a perfect matching,
- for each $1 \leq i \leq k$ and each vertex $v \in B_i$, $B_i \setminus \{v\}$ has a perfect matching.
- any maximum matching of G matches the vertices of A to vertices in distinct B_i; moreover, the matchings above can be quickly found and extended to a maximum matching M of G.

If $|V|$ is even and $A = \emptyset$, then M is a perfect matching; we color the edges of M with color g, remove them from G, and apply induction. Any vertex of $G \setminus M$ that is properly edge-colored with $(g-1)$ colors will remains properly edge-colored after adding the edges of M colored g.

If $|V|$ is odd and $A = \emptyset$, then there is a matching M that leaves exactly one vertex (say, v) unmatched. The graph $(G \setminus M) \setminus \{v\}$ has maximum degree at most $g-1$ and thus, by induction, has a $(g-1)$-edge coloring with at most

$\lfloor\frac{|V|-2}{2}\rfloor$ unbalanced vertices. We use the color g for the edges of M. Then for edges of the form uv with $u \in V \setminus \{v\}$, we use colors from $\{1, 2, \ldots, g-1\}$ such that, if u is proper in the $(g-1)$-edge coloring, it remains proper; a color for uv is always available since the degree of u in G does not exceed g. The vertex v could also be unbalanced, and so the number of unbalanced vertices in the g-coloring of G is at most $\lfloor\frac{|V|-2}{2}\rfloor + 1 = \lfloor\frac{|V|-1}{2}\rfloor$.

Now we may assume that $A \neq \emptyset$. Select in each B_i a vertex v_i adjacent to some vertex of A to specify M so that M restricted to $B_i \setminus \{v_i\}$ is a perfect matching. Consider $(G\setminus M)\setminus A$. It has at least $k+j$ components, and maximum degree at most $g-1$: some vertex degrees drop due to being matched in M, and the remaining vertices are adjacent to vertices in A. Let each B_i have $2b_i + 1$ vertices and each D_i have $2d_i$ vertices. We apply recursion to each component, obtaining a $(g-1)$-edge coloring with at most $\sum_{i=1}^k b_i + \sum_{i=1}^j (d_i - 1)$ unbalanced vertices. We use color g for the edges of M, and for edges of type uv with $u \notin A$ and $v \in A$ we use colors from $\{1, 2, \ldots, g\}$ such that, if u is proper in the $(g-1)$-edge coloring, it remains proper; a color for uv is always available since the degree of u in G does not exceed g. For edges with both endpoints in A, we use arbitrary colors. In the resulting g-edge coloring, the number of unbalanced vertices is at most $|A| + \sum_{i=1}^k b_i + \sum_{i=1}^j (d_i - 1)$. Note that G has $|A| + \sum_{i=1}^k (2b_i + 1) + \sum_{i=1}^j 2d_i$ vertices.

Since M is not a perfect matching we have $|A| \leq k - 1$, and therefore

$$\begin{aligned}\lfloor\frac{|A| + \sum_{i=1}^k (2b_i+1) + \sum_{i=1}^j 2d_i - 1}{2}\rfloor &= \lfloor\frac{|A| + k - 1}{2}\rfloor + \sum_{i=1}^k b_i + \sum_{i=1}^j d_i \\ &\geq |A| + \sum_{i=1}^k b_i + \sum_{i=1}^j d_i.\end{aligned}$$

Thus the number of unbalanced vertices does not exceed $\lfloor\frac{|V|-1}{2}\rfloor$. □

Using the procedure of the previous lemma followed by quasibalancing we obtain an algorithm that produces a g-edge coloring of a graph of maximum degree g with at most $n/2$ unbalanced vertices and no grossly unbalanced vertex. The objective is then at most $n + n/2 \leq 3n/2 = 3L/2$, proving the following theorem:

Theorem 1 *There is a* $(3/2)$*-approximation algorithm for MTPS.*

3.1 Small g

In the case $g \leq 4$, we use a second local optimization procedure to improve Lemma 1. Consider a vertex v with $d(v, i) \geq 2$ and $d(v, j) = 0$, and let u_1 and u_2 be the two vertices with vu_1 and vu_2 colored i. If $u_1 = u_2$ then, unless $d(u_1, j) = 2$, recoloring vu_1 with color j improves the quantity $\sum_{u\in V}\sum_{1\leq k<l\leq g} |d(u, k) - d(u, l)|$ without creating any new unbalanced vertices. If $d(u_1, j) = 2$, using the fact that $d(v, i) \geq 2$ and $d(u_1, i) = 2$, there is a color

j' with $d(u_1, j') = 0$ and $d(v, j') < 2$. Recoloring vu_1 with color j' improves the quantity $\sum_{u \in V} \sum_{1 \leq k < l \leq g} |d(u,k) - d(u,l)|$without creating any new unbalanced vertices. Thus in the following we assume that $u_1 \neq u_2$. We say in this case that v *can be fixed by* u_1 and v *can be fixed by* u_2.

The second local optimization procedure applies when an unbalanced vertex can be fixed by another unbalanced vertex or if two unbalanced vertices can be fixed by the same vertex. To describe the procedure we define the following terminology: We call a vertex v *simply unbalanced* if there is only one i with $d(v, i) = 2$ and *doubly unbalanced* if there are two such i. (Since $g \leq 4$, there are at most two such i.) Quasibalancing eliminates grossly unbalanced vertices without making balanced vertices unbalanced and without making simply unbalanced vertices doubly unbalanced. We have the following four cases.

1. Suppose that v is a simply unbalanced vertex that can be fixed by an unbalanced vertex u. Then we fix v by changing the color of uv. In case u becomes grossly unbalanced, we apply quasibalancing. We reduce the number of unbalanced nodes.

2. Suppose that v is a simply unbalanced vertex that can be fixed by a balanced vertex u, and $v_1 \neq v$ is an unbalanced vertex that can also be fixed by u. Then we fix v by changing the color of uv, and reach either the previous (if v_1 is simply unbalanced - v_1 goes instead of v) or the next case (if v_1 is doubly unbalanced - again v_1 goes instead of v).

3. Suppose that v is doubly unbalanced and can be fixed by an unbalanced vertex u. By symmetry we assume that v has two edges colored 1, among them uv, and two edges colored 3. If u does not have incident edges of color 2 (or 4), then we recolor the edge uv with color 2 (or 4) and make v simply unbalanced. Thus we assume u has incident edges of color 2 and 4. Recall that u is unbalanced; it has two edges colored 1, 2, or 4. If u has two edges colored 4, then the subgraph induced by colors 1 and 4 has a vertex of odd degree (u), and quasibalancing can recolor it to make every vertex balanced in the pair of colors $(1, 4)$: this recoloring will make v simply unbalanced. The case of u having two edges of color 2 is symmetric by interchanging 2 with 4, and the case of u having two edges of color 1 can be tackled by the same argument.

4. Suppose that v is doubly unbalanced and can be fixed by a balanced vertex u, and v_2 is another unbalanced vertex which can be fixed by u. By symmetry we assume that v has two edges colored 1, among them uv, and two edges colored 3. In a first case, v_2u has color 2 (the case when v_2u has color 4 is symmetric by interchanging 2 and 4 below). We make uv of color 2 and v_2u of color 1. This keeps u balanced, makes v simply unbalanced, and either improves v_2 (simply unbalanced to balanced, doubly unbalanced to simply unbalanced; here we use the fact that v_2 has another edge colored 2), or leaves v_2's status unchanged, or makes v_2 grossly

unbalanced. In the last case, quasibalancing is then applied to make v_2 not grossly unbalanced: in fact it makes v_2 simply unbalanced.

We are left with the case when v_2u has color 3. Make uv of color 3 and v_2u of color 1. This keeps u balanced, makes v grossly unbalanced, and either improves v_2 (simply unbalanced to balanced, doubly unbalanced to simply unbalanced), or leaves v_2's status unchanged, or makes v_2 grossly unbalanced. Quasibalancing is applied to make v not grossly unbalanced: in fact it makes v simply unbalanced. If needed, quasibalancing is applied again to make v_2 not grossly unbalanced: in fact it makes v_2 simply unbalanced.

Whenever a (either simply or doubly) unbalanced vertex can be fixed by another unbalanced vertex, Cases 1 or 3 apply, and if two unbalanced vertices can be fixed by the same vertex, Cases 2 or 4 apply. In each case, without creating grossly unbalanced nodes, the second local optimization procedure either reduces the number of unbalanced nodes, or the number of doubly unbalanced nodes without creating unbalanced nodes. Thus, at the end of the second local optimization procedure each unbalanced vertex v has two private (not shared with other unbalanced vertices) balanced vertices u_1 and u_2: the two vertices v can be fixed by. This implies:

Lemma 2 *Assume $g \leq 4$. Given a graph $G = (V, E)$ of maximum degree g, there is a g-edge coloring with at most $\lfloor \frac{|V|}{3} \rfloor$ unbalanced vertices. Such a coloring can be obtained in polynomial time.*

The lemma is tight: for $g = 2$ consider a triangle $v_1v_2v_3$, while for $g = 3$ or $g = 4$ add one or two parallel edges between v_1 and v_2. Using the procedure of the previous lemma followed by quasibalancing we obtain an algorithm that produces, for $g \leq 4$, a g-edge coloring of a graph of maximum degree g with at most $n/3$ unbalanced vertices and no grossly unbalanced vertex. The objective is then at most $n + n/3 \leq 4n/3 = 4L/3$, and we have:

Theorem 2 *For $g \leq 4$ there is a $(4/3)$-approximation algorithm for MTPS.*

4 Implementation and Running Time Analysis

We start with an equivalent version of the algorithm of Lemma 1, given in Figure 2. The next paragraph discusses the equivalence.

In the proof of Lemma 1, the third case considered ($A \neq \emptyset$) is always followed by applying to non-trivial components B_j or D_j either the first case ($A = \emptyset$ and $|V(G)|$ even) or the second case ($A = \emptyset$ and $|V(G)|$ odd). Trivial components, (one vertex only, whose only neighbors are in A) give each a proper vertex immediately. When the algorithm, as described in Figure 2, encounters the third case, it merges this next application into the same step, removing in Step 5 one vertex from each non-trivial component with odd number of vertices (such a vertex joins A in the set of vertices we gave up on being proper).

```
**Input:** Positive integer parameter $g$ and graph $G = (V, E)$ of max degree $g$
**Output:** Coloring of $E(G)$ with colors $1, 2, \ldots, g$

1   If $g = 1$, color all the edges 1 and **return**
2   Compute a maximum matching $M \subseteq E$ in $G$
3   Construct, as in Edmonds' algorithm, alternating forests,
    implicitly shrinking the blossoms
4   Let $Q$ be the set of unmatched vertices, $A$ be the set of inner vertices
    in the alternating forest, and $J$ be the vertices matched by $M$ to some
    vertex of $A$. Let $L$ be the subset of $Q \cup J$ consisting of vertices who
    have a neighbor outside $A$.
5   Recurse on $(G \setminus (A \cup L)) \setminus M$ with parameter $g - 1$
6   Assign color $g$ to the edges of $M$
7   **for** each edge $e$ incident to some vertex $u \in A \cup L$
8       **if** $v$, the other endpoint of $e$, is in $V \setminus (A \cup L)$
9           color $e$ such that, if $v$ is proper in $(G \setminus A) \setminus L$, then $v$ stays proper
10      **else** color $e$ arbitrarily
11  **return**
```

Figure 2: The algorithm of Lemma 1. The set A above is the set of A of the proof of the lemma, and we mention that each shrunk blossom becomes an outer vertex; thus the final inner vertices are vertices of the original graph. L is designed to have exactly one vertex from each non-trivial odd component of the subgraph of G induced by $V \setminus A$.

For the running time of the algorithm in Figure 2, we first note that g maximum matchings are computed in the graph $G' = (V', E')$ which was found when reducing MTPS to $\overline{MTPS}$. G' has at most $n' := n + 2m/g$ vertices and $m' := m$ edges, where n and m are the number of vertices and edges of the original graph. Using the algorithm of Micali and Vazirani [10], we obtain a total running time of $O(gm\sqrt{n + m/g})$. We need to elaborate a bit on steps 7-10. To ensure that proper vertices remain proper, we keep for each vertex v an array M_v of size g indicating which color is already used by edges incident to v. In addition we keep for v an integer j_v (initially 0) such that colors $1, \ldots, j_v$ are used. An unused color for an edge incident to v is found by increasing j_v and testing (and eventually updating) the array M_v. Since j only increases, the total time spent for vertex v on finding unused colors incident to v is $O(g(n+m/g)) = O(gn+m)$.

We move to quasibalancing, which we apply directly to the output of the algorithm in Figure 2. Thus the input consists of G', a graph of maximum degree at most g, and whose edges are colored with colors $1, 2, \ldots, g$. Applying quasibalancing to G', rather than the original graph, does not affect the 1.5 approximation ratio and makes the running time easier to analyze.

As described in Section 2, quasibalancing is clearly polynomial: in $O(m + n)$ time we reduce the quantity $\sum_{u \in V} \sum_{1 \le k < l \le g} |d(u,k) - d(u,l)|$by one, and the initial $\sum_{u \in V} \sum_{1 \le k < l \le g} |d(u,k) - d(u,l)| \le ng^3$.We can do a much better analysis if we carefully describe the procedure, changing it a bit and adding randomization. We describe this specific implementation below. But first, some intuition.

The goal, roughly speaking, is to bound the number of times Euler tours are constructed, as in Section 2 or later below. A natural way to obtain such a bound is to have a potential function which decreases fast whenever we do an Euler tour construction and recoloring; each such Euler tour comes from edges colored with only two colors, say i and j. It can reasonably be hoped that picking i and j such as to decrease the potential the most would be good. But to show a good pair of colors exists it is natural to compute the average decrease in potential over pairs i, j. Then we will not even have to pick the best i, j: we save time by picking them randomly. Such an approach, with potential function $\sum_v \sum_i (d(v,i) - 1)^2$ would have worked and give the same bound we give on the running time, provided the recoloring is "perfect" in the sense that all vertices are left balanced in colors i, j. But one vertex v can be left unbalanced in each Euler tour; in particular, when $d(v,i) = 3$ and $d(v,j) = 1$ no progress may be done. There exist graphs on which the "random pair of colors" (and also "best pair of colors") approach fails. So, instead we use randomization in a more complicated way.

Recall that $d(v,j)$ is the number of edges incident to v with color j. Thus $\sum_{1 \le j \le g} d(v,j) \le g$, and $\sum_{j,v} d(v,j) = 2m$. We also think of these $d(v,j)$ as being values in a matrix M where rows are indexed by colors, and columns by vertices. Our goal is to ensure no grossly unbalanced vertex remains, which in our graph of maximum degree bounded by g means that we must reach that $d(v,i) \le 2$ for all $v \in V'$ and $i \in \{1, 2, \ldots, g\}$.

```
1   Let $H$ be the subgraph of $G'$ induced by the edges colored 1 and $j$
2   Apply the following to each connected component $C$ of $H$
3   If $C$ is not Eulerian, add “fake” edges between vertices of odd degree
    to obtain graph $C'$
4   Compute $T$, an Eulerian tour of $C'$
5   **if** there is a fake edge
6       start $T$ with a fake edge, colored arbitrarily
7   **else**
8       **if** there is $v \in V(C)$ with $d(v,1) = 2$ and $d(v,j) = 0$,
        or with $d(v,1) = 0$ and $d(v,j) = 2$
9           start $T$ with an edge incident with $v$, colored $j$
10      **else**
11          **if** there is $v \in V(C)$ with $d(v,1) \neq d(v,j)$
12              start $T$ with an edge incident with $v$, colored 1
13          **else**
14              start $T$ with an arbitrary edge, colored arbitrarily
15  Follow $T$, assigning colors to its edges alternating 1 and $j$
16  Remove the fake edge, if any
```

Figure 3: Balancing colors 1 and j

The basic move is to pick two colors and “balance” them. For this an Euler tour is produced; it is important for the analysis that a certain edge is picked as the first edge of the tour, and a color is assigned to this edge. Pseudocode appears in Figure 3, with one of the colors being 1. To analyze the total running time of quasibalancing, we give a special role (to be described later) to color 1.

The running time of the procedure is $O(n'+|E(H)|) = O(\sum_{v\in V'}(1+d(v,1)+d(v,j)))$. The procedure results in $|d(v,1) - d(v,j)| \leq 1$ at every vertex of the component C except perhaps one vertex v (where the Eulerian tour starts and ends), in which case $|d(v,1) - d(v,j)| = 2$. Here we choose which of $d(v,1)$ or $d(v,j)$ is larger: If the two values are 0 and 2, we make $d(v,1) = 0$ and $d(v,j) = 2$; otherwise we make $d(v,1) > d(v,j)$.

For each vertex v, let $||v||_j = [\max\{d(v,j) - 2, 0\}]^2$ and let $||v|| = \sum_j ||v||_j$. $||v||$ is defined to measure, roughly speaking, the progress of a balancing: when $||v|| = 0$, we have $d(v,j) \leq 2$ for all j. More formally, we define the *progress*(v) of a balancing as $||v|| - ||v'||$, where $||v'||$ is $||v||$ after balancing.

Lemma 3 *For any balancing of two rows i and j,* progress(v) *is a non-negative integer. Moreover, if we balance rows with values $d(v,i)$ and $d(v,j)$ being M, m, with $m \leq M$, and after the balancing $d(v,i)$ and $d(v,j)$ are (not necessarily in this order) M', m', with $m' \leq M'$, then* progress$(v) \geq 1$ *unless $M = m$, $M = m+1$, $M = M' = m+2 = m'+2$, or $M \leq 2$.*

Proof: Clearly *progress*(v) is an integer.

If $M = m+1$ then there is a "fake" edge added at v, which eventually yields $M' = m'+1$. If $M = m$ then the Eulerian tour through v either has even length or it does not start at v; this yields $M' = m'$. In each case *progress*$(v) = 0$, so we may assume that $M \geq m+2$.

If m, m', M, M' are each at least 2, then *progress*$(v) = (m-2)^2 + (M-2)^2 - [(m'-2)^2 + (M'-2)^2]$. According to the algorithm in Figure 3 we get $M' \leq m'+2$. Since $M+m = M'+m'$, the average of the two values does not change. Thus we have $m \leq m' \leq \frac{M+m}{2} \leq M' \leq M$. It follows that $(m-2)^2 + (M-2)^2 - [(m'-2)^2 + (M'-2)^2]$ is non-negative, with equality if and only if $m = m'$ and $M' = M$.

If $M \geq 4$ and m is 0 or 1, *progress*(v) is at least

$$\begin{aligned}
& (M-2)^2 - [(m'-2)^2 + (M'-2)^2] \\
\geq\ & (M-2)^2 - [(\frac{M+m}{2} - 3)^2 + (\frac{M+m}{2} - 1)^2] \\
=\ & (M-2)^2 - [2(\frac{M+m}{2} - 2)^2 + 2] \\
\geq\ & (M-2)^2 - 2(\frac{M+1}{2} - 2)^2 - 2 \\
=\ & \frac{1}{2}M^2 - M - \frac{5}{2} \geq 0.
\end{aligned}$$

If $M = 3$ and $M' \leq 2$ then $m, m' \leq 2$ and *progress*$(v) = 1$. Otherwise $M = M'$ or $M \leq 2$ and *progress*$(v) = 0$. $\square$

Consider the subsequence of rows $r = \sigma(1) < \sigma(2) < \ldots < \sigma(k)$ for which $d(v,r) \neq 1$.

Claim 3 *Suppose that $d(v,\sigma(j)) \geq 3$, $d(v,\sigma(j+1)) = 0$, and either $d(v,\sigma(j-1)) = 0$ or $j = 1$. If $\sigma(j) \neq 1$ then balancing row 1 with rows $\sigma(j-1),\ldots,\sigma(j+1)$ yields* progress$(v) \geq ||v||_{\sigma(j)}/80$. *The same progress is achieved when $\sigma(j) = 1$ by balancing row 1 with rows $2,\ldots,\sigma(2)$.*

Proof: Let $y = d(v,\sigma(j))$. If $\sigma(j) > 1$, let b be the value of $d(v,1)$ just after balancing row $\sigma(j)$ and row 1. Since the value of $d(v,1)$ before this balancing was non-negative, $b \geq \frac{y}{2} - 1$. If $\sigma(j) = 1$ (in which case $j = 1$ and $y = d(v,1)$), then we let $b = y$. Suppose that $b \geq 4$.

From the hypothesis, $d(v,\sigma(j+1)) = 0$, and if $\sigma(j+1) > \sigma(j)+1$, then by the definition of the σ's we have $d(v,\sigma(j)+1) = 1$. Thus, there a 0 or 1 in row $\sigma(j)+1$, and therefore after balancing rows 1 and $\sigma(j)+1$, the new entries are either $\frac{b+1}{2} \pm \epsilon$ or $\frac{b}{2} \pm \epsilon$, integers with $\epsilon \in \{0, \frac{1}{2}, 1\}$. Thus $||v||_1 + ||v||_j$ becomes at most $(\frac{b+1}{2} + \epsilon - 2)^2 + (\frac{b+1}{2} - \epsilon - 2)^2$ or $(\frac{b}{2} + \epsilon - 2)^2 + (\frac{b}{2} - \epsilon - 2)^2$. The largest this could be is $(\frac{b+1}{2}-1)^2+(\frac{b+1}{2}-3)^2$, so *progress*$(v)$ is at least

$$\begin{aligned} & (b-2)^2-[(\frac{b+1}{2}-1)^2+(\frac{b+1}{2}-3)^2] \\ = \quad & (b-2)^2-[2(\frac{b+1}{2}-2)^2-2] \\ = \quad & \frac{1}{2}b^2-b-\frac{5}{2}. \end{aligned}$$

Since $\frac{9}{20}b^2-b-\frac{5}{2}>0$ when $b\geq 4$, *progress*$(v)\geq\frac{1}{20}b^2\geq\frac{1}{80}(y-2)^2$, which suffices.

If $b\leq 3$ then $(y-2)^2/80\leq 36/80<1$, so *progress*$(v)\geq 1$ would suffice.

If $\sigma(j)=1$ (and $j=1$) then $d(v,1)\geq 3$ at first, and is balanced only with 1s until row $\sigma(j+1)$. Each balancing with a 1 that doesn't yield positive progress must begin and end with $d(v,1)=3$. Thus, if there has been no progress until row 1 is balanced with row $\sigma(j+1)$, then at that point 0 is balanced with a 3 (or $d(v,1)\geq 3$ if $\sigma(j+1)=\sigma(1)+1$), for positive progress.

If $j=1$ and $\sigma(j)\neq 1$, then $d(v,1)$ is 0 or 1 until row 1 is balanced with row $\sigma(j)$, which yields positive progress or makes $d(v,1)=3$. In the latter case, the situation is identical to the previous case, and thus will later yield positive progress.

If $j>1$ and no progress is made balancing row 1 and row $\sigma(j-1)$, then $d(v,1)$ becomes 0 or 1, after which it follows the previous case. □

Each pass of the algorithm randomly orders the colors (redefining the rows), then balances row 1 with rows $2,\ldots,g$ in sequence. See Figure 4.

Lemma 4 *One pass takes $O(n'g)$ time.*

Proof: Recall that balancing rows 1 and j takes time $O(\sum_{v\in V'}(1+d(v,1)+d(v,j)))$. Let $t(v,j)=1+d(v,1)+d(v,j)$ and $t(v)=\sum_{j=2}^{g}t(v,j)$; note however that $d(v,1)$ changes after each balancing. It is enough to show that $t(v)=O(g)$ for all $v\in V'$.

We employ a credit scheme for the proof. Row j starts with $2d(v,j)+3$ credits. We maintain the invariant that row 1 has at least $2d(v,1)$ credits. Consider the rebalancing of row 1 with row j. Let $d'(v,1)$ be the number of edges colored 1 incident to v after the rebalancing. Row j brings $2d(v,j)+3$ credits, and we have $2d(v,1)$ credits in row 1. Of these, $1+d(v,1)+d(v,j)$ go toward $t(v,j)$, and we are left with $d(v,1)+d(v,j)+2$ credits. Since $d'(v,1)\leq 1+(d(v,1)+d(v,j))/2$, the credit invariant is maintained. Thus $t(v)\leq\sum_{j=1}^{g}(3+2d(v,j))\leq 3g+2g=O(g)$, completing the proof. □

We continue with estimating the total number of passes. Consider any vertex v with $\max_j d(v,j)\geq 3$, and let $X_q=X_q(v)$ be the value of $||v||$ after q passes. Each X_q is a random variable, and X_0 is a constant bounded by the maximum possible value of $||v||$, which is clearly $(g-2)^2$. We begin by showing that $E[X_q|X_0,\ldots,X_{q-1}]\leq(1279/1280)X_{q-1}$.

```
1 Randomly reorder colors $1, \ldots, g$, redefining the rows
2 for $j = 2, \ldots, g$
3     balance row 1 and row $j$ according to the algorithm in Figure 3
4 endfor
```

Figure 4: One pass of the algorithm through the colors

As above, we consider the subsequence of rows $r = \sigma(1) < \sigma(2) < \ldots < \sigma(k)$ for which $d(v, r) \neq 1$. Since $\sum_{j=1}^{g} d(v, j) \leq g$, at least half of these $d(v, r)$ are 0s. Each color which has degree 3 or more is now in a row $r = \sigma(j)$ for some $1 \leq j \leq k$. We are *happy* with the color in row $\sigma(j)$ if the entry in row $\sigma(j+1)$ is 0 (and $j + 1 \leq k$) and the entries in rows $\sigma(j-1)$ and $\sigma(j-2)$ are either 0 or undefined (in case j is 1 or 2). By Claim 3, any happy row $r = \sigma(j)$ with $1 \leq j < k$ yields progress at least $||v||_r/80$ from balancing row 1 with rows $\sigma(j), \ldots, \sigma(j+1)$ (rows $2, \ldots, \sigma(j+1)$ if $\sigma(j) = 1$). Observe that there are at least two rows between happy rows in the subsequence $\sigma(1), \ldots, \sigma(k)$, so the progress counted for one happy row does not overlap with progress counted for another happy row.

Suppose that $k \geq 5$. Each color with $d(v, j) \geq 3$ is placed in a row j with $j \neq k$ with probability $\frac{k-1}{k}$. The probability that such a color is happy is at least $(\frac{k/2}{k-1})(\frac{k/2-1}{k-2})(\frac{k/2-2}{k-3}) = \frac{1}{8}\frac{k(k-4)}{(k-1)(k-3)} \geq \frac{1}{16}\frac{k}{k-1}$. Thus the expected progress for one pass at v is at least the sum of $\frac{1}{16}(||v||_r/80)$ over all r such that $d_r(v) \geq 3$. Since $||v||_r = 0$ whenever $d_r(v) \leq 2$, also $\sum_r \frac{1}{1280}||v||_r = \frac{1}{1280}||v||$. Therefore $E[X_q|X_0, \ldots, X_{q-1}] \leq (1279/1280)X_{q-1}$ as desired.

If k is 3 or 4, using $\max_j d(v, j) \geq 3$, it follows that exactly one row of the subsequence has a nonzero entry. Hence the color with $d(v, j) \geq 3$ is happy with probability 1/3 or 1/4. As this is greater than $\frac{1}{16}$, in this case we still have $E[X_q|X_0, \ldots, X_{q-1}] \leq (1279/1280)X_{q-1}$ in this case.

Let Y_q be the value of $\sum_v ||v||$ after q passes. Then Y_q is the sum of $X_q(v)$ over all vertices v for which $\max_j d(v, j) \geq 3$. By linearity of expectation, we have $E[Y_q|Y_0, \ldots, Y_{q-1}] \leq (1279/1280)Y_{q-1}$. It immediately follows that $E[Y_q] = E[Y_q|Y_0] \leq (1279/1280)^q Y_0$. By Markov's inequality, $Pr(Y_q > 0) < (1279/1280)^q Y_0$. Let $q = 2\log_{1279/1280}(n'(g-2)^2)$. Then $(1279/1280)^q Y_0 \leq (n'(g-2)^2)^{-2+1} < 1/n$, so $Y_q = 0$ with high probability. That is, after $O(\log(n'g))$ passes, $||v||$ is zero w.h.p. Since each pass takes time $O(n'g)$, the total time needed is $O(n'g \log n'g)$.

With $n' \leq n + 2m/g$, the main result of this paper is:

Theorem 4 *There is a randomized $O(gm\sqrt{n + m/g})$ algorithm that gives a* 1.5 *approximation to MTPS.*

4.1 Running time for $g = 3$ and $g = 4$

The algorithms are described in Subsection 3.1. First, consider $g = 3$, for which we have a faster algorithm. In this case, only cases 1 and 2 apply in Subsection 3.1. Each time one of these cases applies to a vertex, this vertex becomes balanced in $O(1)$ time, possibly making some other vertex grossly unbalanced.

Quasibalancing can make sure that no grossly unbalanced vertices exist. If we look at the algorithm, we see that the remaining unbalanced vertices are of two types: *content* vertices have two private (not shared with other unbalanced vertices) balanced vertices, while *unfinished* vertices do not have these two balanced vertices. We get better running time if we alternate the following two steps:

1. Make sure no grossly unbalanced vertices exist.
2. Make a list of unbalanced vertices. If there are more than $n/3$ elements in this list, traverse the list once ignoring the content vertices and processing the unfinished ones. Any grossly unbalanced vertex produced by this processing is put aside.

Looking at one pass as described in Figure 4, since a column corresponding to a grossly unbalanced vertex has one 3 and two 0s, a grossly unbalanced vertex cannot remain grossly unbalanced. Thus Step 1 above takes $O(n')$ time.

Let q be the number of elements in the list at the beginning of Step 2, of which c are content and u are unfinished. Using the private balanced vertices of content unbalanced vertices, we obtain $3c + u \leq n'$. Using $3c + 3u = 3q$, we obtain $u \geq (3q - n')/2$. At least half of the unfinished vertices are processed (processing one unfinished vertex can make another grossly unbalanced). Processing a vertex takes $O(1)$ of time, and makes the vertex balanced. Thus Step 2 takes in total $O(n')$ time, and reduces the quantity $(3q - n')$ by a constant factor. As an aside, note that content vertices stay content in Step 2, but can become unfinished in Step 1 - this can happen when the edges incident to such an unbalanced content vertex v change colors during Step 1, and the two vertices which can fix v are not the same as before Step 1. Hence we instead measure progress by the decrease in $(3q - n')$, and do not plan to eliminate all unfinished vertices.

Thus after repeating Steps 1 and 2 $O(\log n)$ times, we have at most $n/3$ unbalanced vertices. Therefore:

Theorem 5 *There is a $O((n + m) \log n)$ algorithm giving a $4/3$-approximation for MTPS when $g = 3$.*

For $g = 4$, we follow exactly the proof from Subsection 3.1. It takes $O(n')$ time to find an unfinished vertex, if there is one. In each case from the proof we process this unfinished vertex without creating other unbalanced vertices but possibly making another unbalanced vertex grossly unbalanced. One or two balancing thus might follow this processing. Thus a processing takes $O(n')$ time. There are at most n' processing of Case 1, since each processing balances a vertex. Note that at most one other vertex can become doubly unbalanced in each processing. Cases 3 and 4 reduce the number of doubly unbalanced vertices, giving in total at most $2n'$, since at most n' vertices can start doubly unbalanced, and only Case 1 can increase the number of doubly unbalanced vertices, by one for each Case 1 processing. Finally, each Case 2 is followed by Case 1 or Case 3, so there are at most $3n$ of Case 2 processing. Since $n' = n + 2m/4$, we obtain:

Theorem 6 *There is a $O(n^2 + m^2)$ algorithm giving a $4/3$-approximation for MTPS when $g = 4$.*

Acknowledgements

The research of the first author supported in part by NSF grant CCF-0515088. Second author supported in part by NSA grant BH98230-08-1-0043. This manuscript is submitted for publication with the understanding that the United States Government is authorized to reproduce and distribute reprints.

Extended abstract in FSTTCS '06. Also, results from Sections 1-3 were presented at the 2005 Workshop on Combinatorial and Algorithmic Aspects of Networking. The workshop did not have proceedings, only a book of abstracts.

References

[1] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and hardness of approximation problems. *Journal of ACM*, 45(3):501–555, 1998.

[2] R. A. Berry and E. Modiano. On the benefit of tunability in reducing electronic port counts in WDM/TDM networks. In *INFOCOM*, volume 2, pages 1340–1351, 2004.

[3] R. A. Berry and E. Modiano. Optimal transceiver scheduling in WDM/TDM networks. *IEEE Journal on Selected Areas in Communications*, 23(8):1471–1495, 2005.

[4] J. Edmonds. Paths, trees, and flowers. *Canadian J. Math*, 17:449–467, 1965.

[5] U. Feige, E. Ofek, and U. Wieder. Approximating maximum edge coloring in multigraphs. In *APPROX, volume 2462 of LNCS*, pages 108–121, 2002.

[6] A. J. W. Hilton and D. de Werra. A sufficient condition for equitable edge-colourings of simple graphs. *Discrete Mathematics*, 128:179–201, 1994.

[7] D. S. Hochbaum, T. Nishizeki, and D. B. Shmoys. A better than best possible algorithm to edge color multigraphs. *J. Algorithms*, 7(1):79–104, 1986.

[8] I. Holyer. The NP-completeness of edge-coloring. *SIAM J. Comput.*, 10(4):718–720, 1981.

[9] L. Lovász and M. D. Plummer. *Matching Theory.* Elsevier Science, 1986.

[10] S. Micali and V. V. Vazirani. An $O(\sqrt{|V|}|e|)$ algorithm for finding maximum matching in general graphs. In *FOCS*, pages 17–27, 1980.

[11] T. Nishizeki and K. Kashiwagi. On the 1.1 edge-coloring of multigraphs. *SIAM Journal on Discrete Mathematics*, 3(3):391–410, 1990.

[12] P. Sanders and D. Steurer. An asymptotic approximation scheme for multigraph edge coloring. In *SODA '05: Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 897–906, Philadelphia, PA, USA, 2005. Society for Industrial and Applied Mathematics.

[13] C. E. Shannon. A theorem on coloring the lines of a network. *J. Math. Phys*, 28:148–151, 1949.