Approximating Maximum Edge Coloring in Multigraphs

Uriel Feige Eran Ofek Udi Wieder

Department of Computer Science
Weizmann Institute, Rehovot 76100, Israel
{feige,erano,uwieder}@wisdom.weizmann.ac.il

November 5, 2003

Abstract

We study the complexity of the following problem that we call *Max edge t-coloring*: given a multigraph G and a parameter t, color as many edges as possible using t colors, such that no two adjacent edges are colored with the same color. (Equivalently, find the largest edge induced subgraph of G that has chromatic index at most t). We show that for every fixed $t \geq 2$ there is some $\epsilon > 0$ such that it is NP-hard to approximate *Max edge t-coloring* within a ratio better than $1 - \epsilon$. We design approximation algorithms for the problem with constant factor approximation ratios. An interesting feature of our algorithms is that they allow us to estimate the value of the optimum solution up to a multiplicative factor that tends to 1 as t grows. Our study was motivated by call admittance issues in satellite based telecommunication networks.

1 Introduction

A *multigraph* is a graph that may contain multiple edges between any two vertices, but can not contain self loops. Given a multigraph $G = (V, E)$, a *edge t-coloring* of the graph is an assignment $c : E \mapsto [t]$, where adjacent edges are mapped to distinct colors. The smallest t for which there exists an edge t-coloring, is called the *chromatic index* of the graph and is denoted as $\chi'(G)$. We study a related problem:

Problem 1.1. Given a multigraph G and a number t, legally color as many edges as possible using t colors. We call this problem *Max edge t-coloring*.

1.1 Motivation

Our motivation for investigating the *Max edge t-coloring* problem comes from questions which arise when designing a large scale satellite based communication network. Such a network serves clients spread sparsely over large distances. Clients are connected to the network via ground stations. A ground station serves the clients that are geographically close to it. The satellite serves as a relay for connection between ground stations. The traffic in the network is based on a Time Division Multiplexing combined with Frequency Division Multiplexing. In this method the time is sliced into t time slots which repeat in a cyclic manner, and the frequency spectrum is divided into f different frequencies. Whenever a client wishes to establish a connection with a client from a different ground station, one of the ground stations requests the satellite to establish a session between them. Then the satellite allocates for this session a *time slot-frequency* pair. In order to avoid collisions each session must have a unique pair of time slot and frequency. There may be parallel sessions running simultaneously between two ground stations. We assume for simplicity that all ground stations are identical and can handle at most one session per time slot. A ground station is allowed to use the same frequency at two different time slots. This means that the satellite must first find a time slot unused by *both* ground stations, and then match to it a frequency unused in *this* time slot. The choices the satellite makes may affect the blocking of future session requests.

1.1.1 Modeling:

In this work we concentrate on the offline problem: we assume that all requests for sessions are given *in advance*. Furthermore once a session is established it remains forever, in other words people never 'hang up'. This offline problem can be translated into the following edge coloring problem:

Problem 1.2. Given a multigraph G and numbers f, t, legally color as many edges as possible using t colors such that every color class contains at most f edges. This is called the *Max edge ft-coloring* problem.

The interpretation is as follows: each node represents a ground station, and each edge represents a request for a session. Multiple edges are allowed since there may be multiple session requests between the same two ground stations. Each color class represents a time slot, and it may contain at most f edges which correspond to the f available frequencies. Once the assignment of edges to time slots is given, the f frequencies are assigned to the edges in a time slot arbitrarily. Edges that are not colored represent requests that were not satisfied. Each session has a unique time slot-frequency pair, and each ground station uses at most one frequency per time slot. The following proposition shows that the offline versions of the *Max edge ft-coloring* problem and the *Max edge t-coloring* problem are similar.

Proposition 1.3. *If there exists an efficient algorithm A which α approximates the Max edge t-coloring problem then there exists an efficient algorithm A' which α approximates the Max edge ft-coloring problem.*

Define a *balanced edge t-coloring* of a graph to be an edge t-coloring in which the cardinality of any two color classes differ by at most one.

Lemma 1.4. *For a multigraph G, let T be a t-edge coloring of G. Then there exists a balanced t-edge coloring of G, denoted by T', that can be derived from T in polynomial time.*

The proof of lemma 1.4 appears in appendix A. Proposition 1.3 follows quite easily from lemma 1.4. The proposition proof is also given in appendix A.

1.2 Online *Max edge t-coloring*

The online version of *Max edge t-coloring* is the one in which the edges of the graph are given to the algorithm one by one. For each edge the algorithm must either color the edge with one of the t colors, or reject it, before seeing the next edge. Once an edge has been colored the color cannot be altered and a rejected edge cannot be colored later. The aim is to color as many edges as possible. This problem was investigated in [FN00]. They showed that *any* algorithm for online *Max edge t-coloring* is at most $\frac{4}{7}-$competitive. Assume the t colors are numbered by $\{1 \dots t\}$. The *First Fit* algorithm upon receiving an edge, colors it with the lowest ranking color available. It is shown that *First Fit* is $0.48-$competitive.

2 Our Results

The original motivation for our research leads to problem 1.2. However in view of proposition 1.3 our results concentrate on problem 1.1 which is the *Max edge t-coloring* problem.

1. Hardness results: we show that for every $t \geq 2$ there exists an $\epsilon > 0$ such that it is NP-hard to approximate the *Max edge t-coloring* problem within a ratio better than $1 - \epsilon$ (section 4).
2. We observe that a simple greedy algorithm achieves an approximation ratio of $1 - (1 - \frac{1}{t})^t$ (section 5). The bulk of our work is concerned with improving over this ratio.
3. For $t = 2$ we give an improved algorithm which uses an LP relaxation and has an approximation ratio of at least $\frac{10}{13} \approx 0.77$ (section 6.2).
4. The main result of this paper appears in sections 7, 8, 9.
   a. Denote by α the best approximation ratio for the chromatic index in multigraphs. We show an algorithm for the *Max edge t-coloring* whose approximation ratio tends to $\frac{1}{\alpha}$ as $t \to \infty$.
   b. We show an estimatation for optimum value of *Max edge t-coloring* (without actually finding a coloring). The ratio between the estimation and the optimum tends to 1 as $t \to \infty$.

3 Related Work

3.1 Edge-coloring

Max edge t-coloring is closely related to the problem of finding the chromatic index of a graph. Approximating the chromatic index has received a large amount of attention.

3.1.1 Lower Bounds on the Chromatic Index

Clearly the maximum degree of a multigraph which is denoted by Δ is a lower bound. Another lower bound is the **odd density of a graph**. Let $S \subseteq V$ be a subset of the vertices of the graph. Every color class in a legal edge coloring of the edges inside S forms a matching, thus it's size is bounded by $\lfloor \frac{|S|}{2} \rfloor$. It follows that the number of colors needed to color the subgraph induced by S is at least $\left\lceil \frac{|E(S)|}{\lfloor \frac{|S|}{2} \rfloor} \right\rceil$. This number can be bigger than Δ if $|S|$ is odd. The *odd density* of a multigraph G is defined as: $\rho(G) \stackrel{\triangle}{=} \max_{\substack{S \subseteq V, \\ |S|=2k+1}} \left\lceil \frac{|E(S)|}{k} \right\rceil$. We conclude that $\chi'(G) \geq \max\{\rho(G), \Delta(G)\}$. In general it is not known how to efficiently compute $\rho(G)$, though the value of $\max\{\rho(G), \Delta(G)\}$ can be computed in polynomial time.

The fractional chromatic index: the chromatic index can be viewed as the minimum number of matchings required to cover the graph. A natural relaxation would be to assign fractional weights to matchings such that each edge is covered by matchings with total weight of at least 1. The *fractional chromatic index* is the minimum total weight of matchings required to cover the graph. It is denoted by $\chi'^*(G)$. Clearly $\chi'^*(G) \leq \chi'(G)$. Edmonds proved that $\chi'^*(G) = \max\{\Delta(G), \rho(G)\}$ (see for example [LP86]). The value of $\chi'^*(G)$ together with a fractional coloring of G can be computed efficiently (this is proved implicitly in section 6).

3.1.2 Upper Bounds for the Chromatic Index

The *multiplicity* of a multigraph is the maximum number of edges between any specified two vertices.

Theorem 3.1 (Vizing [Viz64]). *Let G be a multigraph with multiplicity d, then:*

$$\Delta(G) \leq \chi'(G) \leq \Delta(G) + d$$

In particular for simple graphs, the chromatic index is at most $\Delta + 1$. Vizing's proof is constructive and yields an efficient algorithm. The current best bound which also yields an efficient algorithm is:

Theorem 3.2 (Nishizeki and Kashiwagi [NK90]). $\chi'(G) \leq \max\{\lfloor 1.1\Delta(G) + 0.8 \rfloor, \rho(G)\}$

The additive factor was later improved to 0.7 by [CR98]. The term $(1.1\Delta(G) + 0.7)$ is tight for graphs such as the Petersen graph (see Figure 5) for which $\Delta(G) = \rho(G) = 3$, but $\chi'(G) = 4$. The current state of affairs is that it is possible to approximate the chromatic index up to an *additive* factor of 1 in simple graphs (which is a tight result), and some *multiplicative* factor in multigraphs. It seems likely that a generalization of Vizing's theorem to multigraphs should hold:

Conjecture 3.3. *For any multigraph G, $\chi'(G) \leq \chi'^*(G) + 1$.*

This was proposed by Goldberg [Gol73], Anderson [And77] , and again by Seymour [Sey79] in the stronger form: $\chi'(G) \leq \max\{\Delta(G) + 1, \rho(G)\}$. While conjecture 3.3 remains unproven, an "asymptotic" version of it was proven:

Theorem 3.4 (Kahn [Kah96]). *For every $\epsilon > 0$ there exists $D(\epsilon)$ so that for any multigraph G with $\chi'^*(G) \geq D(\epsilon)$ we have $\chi'(G) \leq (1 + \epsilon)\chi'^*(G)$.*

Kahn's proof is not constructive. It is unknown whether there is a (probabilistic) polynomial time algorithm with performance guarantee that matches theorem 3.4.
For some families of graphs it is possible to compute the chromatic index exactly. These include *bipartite graphs* for which the chromatic index always equals the maximum degree (see for instance [Die96]) and simple *planar graphs* with maximal degree larger than 8. See [NZ99] for an overview.

3.2 Matchings and the Matching Polytope

As noted, when coloring the edges of a graph, each color class forms a *matching*. There is a known efficient algorithm that finds a matching with maximum cardinality in a graph. See [Edm65b] and [Edm65a]. Given a graph G with edge set E, any matching M can be associated with a point $x \in \{0,1\}^{|E|}$, where $x_e = 1$ iff $e \in M$. The *matching polytope* of G is the convex hull of all the points in $\mathbb{R}^{|E|}$ which represent a matching in G. This polytope can be described as an intersection of half-spaces, each half space is represented by a constraint on the set of variables x_e. For a set of vertices $S \subseteq V$, we denote by $X(S)$ the total sum of edge variables in the subgraph induced by S.

Theorem 3.5 (Edmonds [Edm65a]). *The following linear constraints define the matching polytope of a graph:*

The matching LP

$\forall v \in V$	$\sum_{e \mid v \in e} x_e \leq 1$	*Degree constraint*	(1)
$\forall S \subseteq V, \lvert S \rvert$ *is odd*	$X(S) \leq \left\lfloor \frac{\lvert S \rvert}{2} \right\rfloor$	*Blossom constraint*	(2)
$\forall e \in E$	$0 \leq x_e \leq 1$	*Capacity constraint*	(3)

A few remarks:

1. If instead of the capacity constraint, we had demanded that $x_e \in \{0,1\}$, then constraint 1 would suffice to define the matching polytope. When the integrality constraint is relaxed to a fractional one, the blossom constraints become necessary.
2. The number of blossom constraints is exponentially large in the number of variables. In order to solve maximum weighted matching in an LP approach (using for instance the ellipsoid method) one needs a separation oracle for the blossom constraints. Padberg and Rao in [PR82] showed a polynomial time algorithm which finds a violated blossom constraint in the special case that degree constraints are not violated. This suffices for the purpose of solving the LP.

3.2.1 Maximum b-matching

In the maximum b-matching problem the input is a multigraph G, and a vector $b \in \mathbb{N}^{|V|}$. The goal is to find the maximum (with respect to the number of edges) subgraph H, for which the degree of a vertex v in H is at most b_v. The name for this problem might be misleading: even if $b_v = t$ for all $v \in V$ this problem is different from the *Max edge t-coloring* problem (in which one has to find a maximum edge subgraph which can be covered by t matchings). For example a triangle is a graph with all degrees bounded by 2, however it can not be covered by 2 matchings. In the special case that $b_v = 1$ for all $v \in V$ the $b-$matching problem is simply a maximum matching problem. A more natural name for this problem can be *the maximum degree constrained subgraph*, however we use the name b-matching for historical reasons. Edmonds and Johnson gave a generalization of the blossom constraints (see appendix B.1 constraint 7) which together with degree constraints induce a polytope with integer vertices; the vertices of this polytope are exactly all the edge subgraphs of G which obey the degree constraints. A separation oracle for finding a violated constraint of this LP appears at [PR82]. This implies a polynomial time algorithm for solving the b-matching problem. More details can be found at [LP86] and [GLS88].

4 Hardness of *Max edge t-coloring*

Theorem 4.1. *For every $t \geq 2$ there exists an ϵ_t such that it is NP-Hard to approximate Max edge t-coloring within a factor better than $1 - \epsilon_t$.*

Holyer showed [Hol81] that deciding whether an input graph is 3 edge colorable is NP-hard. A similar result for the problem of deciding whether an input multigraph is t edge colorable (for every $t > 3$) was given by Leven and Galil [LG83]. This implies that for every $t \geq 3$ the *Max edge t-coloring* problem is NP-hard. The reductions used by Holyer, Leven and Galil are from MAX 3 SAT. If the domain is limited to MAX 3SAT-3 instances (3SAT formulas in which every variable appears in at most 3 clauses) then these reductions are in fact L reductions (in the terms of [PY91]). Therefore the reduction preserves the inapproximability of MAX 3SAT-3. Details are omitted from this manuscript. The case of *Max edge 2-coloring* is different since deciding whether the chromatic index of a graph is 2 can be done in polynomial time. The proof of theorem 4.1 for $t = 2$ uses a modified version of a reduction attributed to Papadimitriou in [CP80]. The details appear in appendix C.

5 The Greedy Approach

A simple greedy algorithm is based on the online approach in which the edges are colored one after the other (for instance *First Fit*). The following offline algorithm **Greedy** has a better performance guarantee: in each iteration (out of t) a maximum matching is selected, colored and removed from the graph.

The *Max edge t-coloring* problem is a special case of the *maximum coverage problem*. We wish to cover the maximum number of edges of G with t sets, where each set is a matching. The following theorem is well known (see for example [Hoc97] theorem 3.8):

Theorem 5.1. *The greedy algorithm for the maximum coverage problem yields a $1 - \left(1 - \frac{1}{t}\right)^t$ approximation.*

It follows that **Greedy** has an approximation ratio of at least $1 - \left(1 - \frac{1}{t}\right)^t$. We could not design for every value of t examples that show that the approximation ratio of **Greedy** is no better than $1 - \left(1 - \frac{1}{t}\right)^t$. However, a simple example shows that for every even t, the approximation ratio is no better than $\frac{3}{4}$. Consider a line of four edges, each with multiplicity of $\frac{t}{2}$. All edges can be colored with t colors. However, **Greedy** might take the first $\frac{t}{2}$ matchings to consist of the first and fourth edges, and then it will cover only $\frac{3}{4}$ of the edges.

6 A Linear Program

An optimal solution to *Max edge t-coloring* has a maximum degree and odd density bounded by t. Therefore a natural approach for solving *Max edge t-coloring* may be to solve the following problem as an intermediate step:

Problem 6.1. Given a multigraph G and a parameter t, Find a largest (in edges) subgraph of G whose degree and odd density are bounded by t. We call this problem *Max Sparse t-Matching*.

The following is an LP relaxation of *Sparse t-Matching*:

$$\max \sum x_e \qquad \textit{subject to}$$

$$\sum_{e|v\in e} x_e \leq t \qquad \forall v \in V \qquad (4)$$

$$X(S) \leq t \cdot \left\lfloor \frac{|S|}{2} \right\rfloor \qquad \forall S \subseteq V,\ |S| \textit{is odd} \qquad (5)$$

$$0 \leq x_e \leq 1 \qquad \forall e \in E \qquad (6)$$

Since every feasible solution for *Max edge t-coloring* is also feasible for *Sparse t-Matching*, the *Sparse t-Matching* LP is a relaxation also for *Max edge t-coloring*. We will show that fractional solutions of *Sparse t-Matching* LP can be rounded to give a solutions of *Max edge t-coloring*, of value at least a $1 - \left(1 - \frac{1}{t}\right)^t$ fraction of the value of the LP. This is the same approximation ratio that the greedy approach yields. It uses some ideas and intuition that will be useful later.
In order to solve the linear program using the ellipsoid method, we need an oracle for finding violated constraints of the LP. Dividing all the constraints by t and multiplying the objective function by t we get an equivalent LP. This linear program is similar (though not identical) to the *matching LP* (see theorem 3.5), and can be solved using the same techniques (using the separation oracle given by Padberg and Rao at [PR82]).

6.1 Rounding the Linear Program Solution

The vector $\frac{1}{t}\vec{x}$ is valid for the matching LP and therefore according to theorem 3.5 is a convex combination of a set of matchings. We will now describe how to find such a convex combination. Let $\mathcal{M}$ denote the set of all matchings, and define a variable λ_M for each $M \in \mathcal{M}$. Solve the following LP:

$$\forall e \in E \qquad \sum_{M|e\in M} \lambda_M = \frac{x_e}{t}$$

$$\sum_{M\in\mathcal{M}} \lambda_M = 1$$

$$\forall M \in \mathcal{M} \qquad \lambda_M \geq 0$$

This LP has an exponential number of *variables* and a polynomial number of *constraints*. We sketch how to solve it: the dual of this LP can be solved using the ellipsoid method. By doing so we identify a polynomial number of constraints that are tight with respect to the solution. Moving back to the primal results in an LP that has a polynomial number of variables thus can be solved efficiently. See [Wie01] for more details. Now we sample t matchings. Each sample is done such that matching M has probability of λ_M to be chosen. Note that only a *polynomial* number of matchings have positive probability, therefore the sampling algorithm is efficient. We output as a solution the union of those t matchings (denote it by S).

6.1.1 Analysis

Claim 6.2. $\Pr\{\ e \in S\} \geq \left(1 - \left(1 - \frac{1}{t}\right)^t\right) \cdot x_e$

Proof: Each time a matching is sampled, edge e enters S with probability $\frac{x_e}{t}$. Each sample is independent from the rest. Therefore: $\Pr\{e \notin S\} \leq \left(1 - \frac{x_e}{t}\right)^t$

$$\Rightarrow \Pr\{e \in S\} \geq \left(1 - \left(1 - \frac{x_e}{t}\right)^t\right) \geq \left(1 - \left(1 - \frac{1}{t}\right)^t\right) \cdot x_e.$$

where the last inequality follows from the concavity of $(1-\left(1-\frac{x_e}{t}\right)^t)$ in $[0,1]$. ■

From linearity of expectation it follows that $E(|S|) \geq (1-\left(1-\frac{1}{t}\right)^t) \cdot x^*$. Since x^* is an upper bound on the optimum integral solution we conclude that the algorithm yields (on expectation) a $(1-\left(1-\frac{1}{t}\right)^t)$ approximation.

6.2 The Case t=2

In an interesting special case we are to find the 2-edge colorable subgraph with a maximum number of edges. In this case the greedy algorithm and the rounding procedure described in section 6.1 both yield an approximation ratio of 0.75. We present an algorithm that achieves an approximation ratio of $\frac{10}{13} \approx 0.77$. Denote by OPT the size of the optimal subgraph. Set a variable x_e for each edge. Solve the *Sparse t-Matching* LP presented previously with $t = 2$. Denote by x^* it's optimal value. The idea of the algorithm is to choose between two different kinds of roundings. If a large portion of the weight is in 'heavy' variables, then a threshold rounding is used. Otherwise the random rounding of section 6.1 is used.

Algorithm for t=2:

1. Solve the linear program presented above. Let x^* denote it's value.
2. If $\sum\{x_e \mid x_e > \frac{2}{3}\} \geq \frac{10}{13}x^*$ then remove all edges e with $x_e \leq \frac{2}{3}$, remove one edge from every odd cycle that remains, and output the remaining edges.
   Otherwise perform the random rounding of section 6.1.

6.2.1 Analysis:

We will use the term 'light' edge for an edge whose weight is at most $\frac{2}{3}$, otherwise the edge is considered 'heavy'. Denote by α the fraction of weight (from x^*) of all the heavy edges; i.e. $\alpha \stackrel{\triangle}{=} \frac{\sum\{x_e \mid x_e > \frac{2}{3}\}}{x^*}$.

Lemma 6.3. *There exists a 2-edge-colorable subgraph with at least αx^* edges.*

Proof: Let H denote the subgraph induced by all the heavy edges. Constraint 4 implies that the maximum degree of H is 2. Therefore H consists of paths and cycles. We find a two edge colorable subgraph, by omitting one edge from each odd cycle. It remains to show that the solution built has at least αx^* edges. Let C be a component of H. If C is a path or an even length cycle, then the number of edges C has contributed to the solution is $|C|$ while $\sum_{e \in C} x_e \leq |C|$. If C is an odd length cycle, then the number of edges C has contributed to the solution is $|C|-1$, while constraint 5 implies that $\sum_{e \in C} x_e \leq |C|-1$. We conclude that the cardinality of the solution is at least $\sum_{e \in H} x_e \geq \alpha x^*$. ■

Lemma 6.4. *There exists a 2-edge-colorable subgraph with at least $\left(\frac{5}{6} - \frac{\alpha}{12}\right) x^*$ edges.*

Proof: Perform a random rounding as in section 6.1. The expected contribution of an edge e to the solution is $(1-(1-\frac{x_e}{2})^2) = x_e(1-\frac{x_e}{4})$. As before let H be the subgraph induced by the heavy edges, and let $\overline{H}$ denote the subgraph induced by the light edges. The expected contribution of H to the solution is at least $\frac{3}{4}(\alpha \cdot x^*)$, and that of $\overline{H}$ is at least $(1-\frac{2}{3}\cdot\frac{1}{4})(1-\alpha)x^*$. Linearity of expectation implies that the expected cardinality of the solution is at least $(\frac{5}{6}-\frac{\alpha}{12})x^*$. ■

Corollary 6.5. *The break even point between the two lemmas is $\alpha = \frac{10}{13}$, therefore we have a $\frac{10}{13}$ approximation for the problem.*

The LP approach as presented here can not yield an approximation better than 0.9 as some graphs have an integrality ratio of 0.9 (see section B.1.1). David Hartvigsen [Har84] designed an algorithm that given a *simple* graph, finds a maximum triangle free subgraph with degree bounded by 2. This algorithm can be used to yield a $\frac{4}{5}$ approximation for simple graphs in the following way: apply the algorithm and delete from every odd cycle one edge. A 2-matching without odd cycles can be decomposed into two disjoint matchings. Since the output of the algorithm is triangle free, at most $\frac{1}{5}$ of the edges are deleted.

7 The *Sparse t-Matching* Scheme

Conjecture 3.3 states that the chromatic index of a multigraph is very close to the fractional chromatic index. This motivates us to try to deal with the *Max Sparse t-Matching* problem (problem 6.1) as an intermediate problem, using an approximation for it to approximate *Max edge t-coloring*. In case $t = 2$ *Max Sparse t-Matching* is equivalent to *Max edge t-coloring* and therefore is known to be NP-Hard and does not admit a PTAS. In case $t \geq 3$ it is unknown whether the problem is hard, yet we believe it is. An example for an integrality gap of the *Sparse t-Matching* LP can be found at the appendix.
The general algorithmic scheme we will use in this section is as follows:

1. Find a large subgraph with degree and odd density bounded by $t + o(t)$. Let H be the subgraph returned.
2. Use an edge coloring algorithm to color H.
3. Output the t largest color classes.

Note that in stage 2 the edge coloring algorithm is used as a black box. We will analyze our algorithm with respect to a general edge coloring algorithm, and then plug in the performance of the known algorithms.

7.1 A Few Simple Cases:

In the two following examples we simplify step 1. The simplified step 1 is to find the maximum (in edges) subgraph with maximum degree bounded by t. This can be done by invoking Edmond's $b-$matching algorithm with $b_v = t$ for all $v \in V$.

If a multigraph G has a multiplicity of d, then *Max edge t-coloring* can be approximated by a polynomial time algorithm up to a factor of $1 - \frac{d}{t+d}$. To see this let OPT denote the maximum number of edges that can be colored by t colors and follow the algorithmic scheme presented above. Clearly $|E(H)| \geq OPT$. Vizing's theorem states that H can be colored using $t+d$ colors. Therefore in step 3, we discard at most a $\frac{d}{t+d}$ fraction of the edges, which yields the approximation ratio.

For bipartite multigraphs, it is possible to solve *Max edge t-coloring* in polynomial time. It is well known that for bipartite multigraphs $\chi'(G) = \Delta(G)$ (see for example [Die96]), therefore replacing step 2 of the previous algorithm, by an algorithm that colors a bipartite graph H with $\Delta(H)$ colors (for example [COS01]) yields an exact solution.

8 Approximating the *Sparse t-Matching* Problem

We present an approximation algorithm for the *Sparse t-Matching* problem. Our approximation however will not be in the usual sense. Given a multigraph G and a parameter t, let OPT denote the number of edges in the optimal subgraph. Our algorithm will output a subgraph H such that $|E(H)| \geq OPT$, albeit $\Delta(H) \leq t + 1$ and $\rho(H) \leq t + \sqrt{t+1} + 2$ (rather than $\rho(h), \Delta(H) \leq t$). The first stage of the algorithm for approximating the *Sparse t-Matching* problem is to solve the

Sparse t-Matching LP from section 6.1. This can be done in polynomial time and gives a fractional solution. The second stage is to round the fractional solution returned by the LP and is explained in section 8.1. The integrality ratio of the *Sparse t-Matching* LP is at least $1-\Omega(\frac{1}{t})$; details appear in appendix B.

8.1 Rounding the Fractional Solution:

The rounding technique consists of three stages. We hold a subset of the solution that increasingly grows as we proceed through the stages. Let H be the set of edges that represents the solution. Initially $H = \phi$. In each stage of the algorithm edges are added to H, until finally H is returned as the rounded solution. For each node v define the variable d_v. As we proceed through the algorithm, d_v will hold the number of edges adjacent to v that can be added to H. Initially $d_v = t$. Every edge (u, v) added to H decreases d_u and d_v by one.

Stage 1: First we get rid of multiple edges. Let (u, v) be a pair of vertices that are connected by multiple edges. If two parallel edges have positive fractional value, shift values between them until one of the edges either reaches the value of 0 or the value of 1. If an edge reached the value of 1 then add it to H and decrease d_u, d_v by one. Continue until the edges which have a positive fractional value form a *simple* graph. Denote this graph by G'. Clearly The edges of H satisfy constraints (4), (5), (6).

Stage 2: We concentrate on G' and get rid of even length cycles:

1. Find a cycle (not necessarily simple) in G' with an even number of edges, and mark them *even*, *odd* alternately. An algorithm to do so appears in appendix D.
2. Shift weights from the even edges to the odd edges until one of the two events happens:
   - An odd edge reached the weight of 1. In this case discard the edge from G', add it to H and modify the d-values of it's vertices.
   - An even edge reached the weight of 0. In this case discard it from G'
3. If there are even length cycles left in the graph then return to step one.

When this stage is done, it is evident that the edges of H satisfy constraints (4), (6) though not necessarily constraint (5). The proof of claim 8.1 can be found in appendix D.

Claim 8.1. *Let G be a graph without even cycles. G has the following properties:*

1. *All odd cycles are simple cycles.*
2. *All odd cycles are disjoint in vertices.*

From claim 8.1 we deduce that the graph at the end of stage (2) looks like a tree in which each node might represent an odd cycle. We use this restricted topology to complete the rounding.

Stage 3: Now we round the values of the edges in the 'tree' from the leaves up. A leaf of the remaining graph may be a node or an odd cycle. First we deal with the case of an odd cycle. Assume that the cycle $C = (v_1, v_2, \ldots, v_l)$ is an odd length cycle which forms a leaf of the 'tree'. Let the edge $w = (u, v_1)$ connect C to the rest of the graph. Until this stage all that was done is value shifting, and the d-values were always updated correctly, so it holds that $d_{v_i} \geq 1$ for every node in the cycle. There are two cases:

case 1: There exists a node $v_i \in C$ for which $d_{v_i} = 1$. Add to H the edges of the cycle and discard the edge w from the graph (figure 1). Since $d_{v_i} = 1$, it follows that the total value of edges in the cycle is bounded by $l - 1$. If we add the value of the edge w the total value is bounded by l. l edges of the cycle were entered to H so no value was lost.

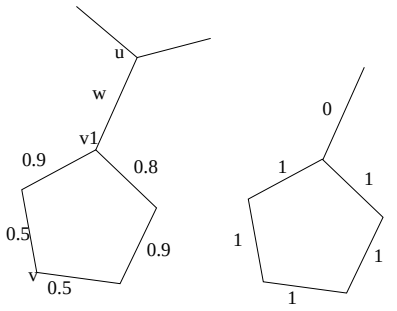

Figure 1: *Case 1,* vertex v has $d_v = 1$, so the cycle is taken and w is discarded.

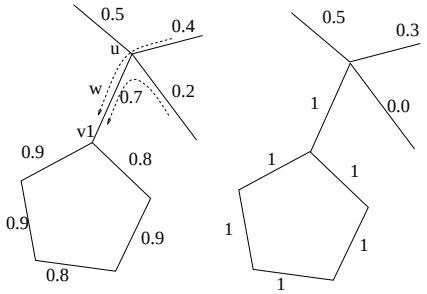

Figure 2: *Case 2,* $\forall v\ d_v \geq 2$. The cycle and w are added to H and d_u is decreased.

case 2: For every node $v \in C$ we have $d_v \geq 2$. Shift values from other edges that contain u into w, add the cycle *and* w into H, and decrease d_u by 1. The value shift did not change the degree of u, though the edges of u may reach the value of 0 and therefore be discarded from the graph. See figure 2. Since all the $l+1$ edges were taken, no value was lost in this case.

The case that a leaf of the tree is a node is simple and can be solved by a value shift downwards towards the node, i.e. the same as case 2. We iterate through stage 3 traversing the tree bottom up, until all edges are either discarded from the graph or added to H. Finally output H.

8.2 Analysis of the *Sparse t-Matching* Algorithm

Let OPT denote the maximum size of a subgraph with degree and odd density bounded by t. From the description of the rounding it is clear that $|H| \geq OPT$. The only place in the rounding that the degree of a vertex v might exceed t, is in stage 3 when v belongs to a cycle. Note that when $d_v = 1$, at most two edges containing v were added, and when $d_v \geq 2$ at most 3 edges were added. We conclude that $\Delta(H) \leq t+1$. We show that the odd density of H does not exceed t by much.

Theorem 8.2. $\rho(H) \leq t + \sqrt{t+1}\ + 2$

Proof:

Let $S \subseteq V$ such that $|S| = 2k+1$ for some integer k. Each $v \in S$ has degree at most $t+1$ in H. The number of edges of S is at most $\frac{(2k+1)\cdot(t+1)}{2}$ therefore:

$$\rho_H(S) \leq \left\lceil \frac{(2k+1)(t+1)}{2k} \right\rceil \leq t + \left\lceil \frac{t+1}{2k} \right\rceil + 1 \leq t + \frac{t+1}{2k} + 2.$$

This bound is good for large sets as it tends to $t+2$ when k tends to infinity. We will give another bound which is good for small sets:

Let $\vec{x}$ denote the fractional solution. Constraint 5 implies that $X(S) \leq k \cdot t$. Some mass was left for the manipulations of stage 2, thus at the beginning of stage 2 the number of edges in S that

were added to H is at most $k \cdot t - 1$. Since at stage 2 we deal with a simple graph, the number of edges added to S in stage 2 is at most $\binom{2k+1}{2} = (2k+1)k$. Therefore:

$$\rho_H(S) \leq \left\lceil \frac{(2k+1)k + k \cdot t - 1}{k} \right\rceil \leq t + 2k + 1.$$

The break even point between the two bounds is when $k = \frac{1}{4} + \frac{1}{2}\sqrt{t + \frac{5}{4}}$. Setting k to this value yields the desired result. ■

We remark here that the loss of 1 in Δ which is caused by the rounding is unavoidable, if the rounding procedure returns a solution with value at least as large as the value of the fractional solution (see appendix B).

9 Analysis of the *Max edge t-coloring* Algorithm

In Stage 2 of the algorithmic scheme presented in section 7 we use an edge coloring algorithm as a black box. Naturally the quality of the result depends heavily on the performance of the algorithm used. Assume we have an efficient algorithm A for the chromatic index problem, which edge colors a multigraph G with $max\{\alpha\Delta(G) + \beta\ ,\ \rho(G)\}$ colors. The algorithm A can be used to color the subgraph H found by the *Sparse t-Matching* algorithm with $max\{\alpha \cdot (t+1) + \beta\ ,\ t + \sqrt{t+1}\ + 2\}$ colors. Let L denote this maximum. Our algorithm yields then a $\frac{t}{L}$ approximation. If t is large enough then the maximum would be achieved by $\alpha \cdot (t+1) + \beta$. Taking the largest t color classes results in a $(\frac{t}{\alpha(t+1)+\beta})$-approximation for the *Max edge t-coloring* problem. This value tends to α^{-1} as $t \to \infty$.
Currently the best approximation algorithm for the chromatic index problem is due to [NK90] where $\alpha = 1.1$. If conjecture 3.3 is true (with a suitable efficient algorithm), then there exists an efficient edge coloring algorithm for which $\alpha = 1$. This means that the larger t is, the better is the approximation for the *Max edge t-coloring*. This phenomena combined with Kahn's result [Kah96] that $\frac{\chi'(G)}{\chi'^*(G)} \to 1$ as $\chi'^*(G) \to \infty$ implies the following:

Corollary 9.1. *The value of the Sparse t-Matching LP is an estimation to the optimal value of Max edge t-coloring, with an approximation ratio that tends to* 1 *as* $t \to \infty$.

Acknowledgment

This work was supported in part by a grant of the Israeli Ministry of Industry and Commerce through the consortium on Large Scale Rural Telephony. Some of the details that are omitted from this manuscript can be found in the M.Sc thesis of Eran Ofek [Ofe01] and in the M.Sc thesis of Ehud Wieder [Wie01], both done at the Weizmann Institute of Science.

References

[And77] L. D. Andersen. On edge-colourings of graphs. *Math. Scand.*, 40:161–175, 1977.

[COS01] R. Cole, K. Ost, and S. Schirra. Edge-coloring bipartite multigraphs in O(E log D) time. *Combinatorica*, 21(1), 2001.

[CP80] Gerard Cornuejols and William Pulleyblank. A matching problem with side conditions. *Discrete Mathematics*, 29:135–159, 1980.

[CR98] Alberto Caprara and Romeo Rizzi. Improving a family of approximation algorithms to edge color multigraphs. *Information Processing Letters*, 68(11-15), 1998.

[Die96] Reinhard Diestel. *Graph Theory*. Springer, 1996.

[Edm65a] J. Edmonds. Maximum matching and a polyhedron with 0, 1 vertices. *Journal of Research National Bureau of Standards*, 69B:125–130, 1965.

[Edm65b] J. Edmonds. Paths, trees, and flowers. *Canadian Journal of Mathematics*, 17:449–467, 1965.

[EJ70] Jack Edmonds and Ellis L. Johnson. Matching: A well-solved class of integer linear programs. In R. K. Guy, H. Hanani, N. Sauer, and J. Schonheim, editors, *Proceedings of the Calgary International Conference on Combinatorial Structures and their Applications*, pages 89–92. Gordon and Breach, New York, London, Paris, 1970.

[FN00] Lene M. Favrholdt and Morten N. Nielsen. On-line edge-coloring with a fixed number of colors. *FSTTCS: Foundations of Software Technology and Theoretical Computer Science*, 20, 2000.

[GLS88] M. Grotschel, L. Lovasz, and A. Schrijver. *Geometric algorithms and combinatorial optimization*. Springer-Verlag, Berlin, 1988.

[Gol73] M.K. Goldberg. On multigraphs of almost maximal choratic class (in russion). *Diskret. Analiz*, 1973.

[Har84] D. Hartvigsen. *Extensions of Matching Theory*. PhD thesis, Carnegie-Mellon University, 1984.

[Hoc97] Dorit Hochbaum. *Approximation Algorithms For NP-Hard Problems*. PWS Publishing Company, Boston, 1997.

[Hol81] Ian Holyer. The NP-completeness of edge-coloring. *SIAM Journal on Computing*, 10(4):718–720, 1981.

[Kah96] J. Kahn. Asymptotics of the chromatic index for multigraphs. *JCTB: Journal of Combinatorial Theory, Series B*, 68, 1996.

[LG83] Daniel Leven and Zvi Galil. NP completeness of finding the chromatic index of regular graphs. *Journal of Algorithms*, 4(1):35–44, March 1983.

[LP86] L. Lovasz and M.D. Plummer. Matching theory, north-holland. *AMSTERDAM*, 88:1986, 1986.

[NK90] Takao Nishizeki and Kenichi Kashiwagi. On the 1.1 edge-coloring of multigraphs. *SIAM Journal on Discrete Mathematics*, 3(3):391–410, August 1990.

[NZ99] T. Nishizeki and X. Zhou. Edge-coloring and f-coloring for various classes of graphs. *Journal of Graph Algorithms and Applications*, 3(1), 1999.

[Ofe01] Eran Ofek. Maximum edge coloring with a bounded number of colors. Master's thesis, Weizmann Institute of Science, Rehovot, Israel, November 2001.

[PR82] M. Padberg and M. R. Rao. Odd minimum cut-sets and b-matchings. *Mathematics of Operations Research*, 7:67–80, 1982.

[PY91] Christos H. Papadimitriou and Mihalis Yannakakis. Optimization, approximation, and complexity classes. *Journal of Computer and System Sciences*, 43(3):425–440, December 1991.

[Sey79] P. Seymour. Some unsolved problems on one-factorizations of graphs. In J. A. Bondy and U. S. R. Murty, editors, *Graph Theory and Related Topics*, pages 367–368. Academic Press, 1979.

[Viz64] V. G. Vizing. On an estimate of the chromatic class of a p-graph (in russian). *Diskret. Analiz*, 3:23–30, 1964.

[Wie01] Ehud Wieder. Offline satellite resource allocation or maximum edge coloring with a fixed number of colors. Master's thesis, Weizmann Institute of Science, Rehovot, Israel, November 2001.

A *Max edge ft-coloring* Reduces to *Max edge t-coloring*

Proof of lemma 1.4: Given T we show how to construct an appropriate T'. Denote by m the number of edges in G. If T is not a balanced coloring then there exists two colors x, y, for which the cardinality of color class x is bigger than the cardinality of color class y by at least 2. Denote by $H(x, y)$ the subgraph induced by edges colored x or y. As in the proof of Vizing [Viz64] we note:

1. The degree of each vertex in $H(x, y)$ is at most 2. Every connected component of $H(x, y)$ is either a path or an even length cycle.
2. Every connected component of $H(x, y)$ is colored alternately by x and y. If for some component we change the coloring by assigning x to edges that were colored y, and assigning y to edges that were colored x, then the new coloring is a legal coloring of G.

Since more edges are colored x than are colored y, we must have in $H(x, y)$ a path of odd length in which the number of edges colored x is one more than those colored y. If we change the coloring of that component by substituting x for y and y for x, we are left with a legal coloring in which x colors one edge less and y colors one more. It takes at most m iterations of this process to bring the cardinality of the color classes to their proper sizes ($\llcorner \frac{m}{t} \lrcorner, \ulcorner \frac{m}{t} \urcorner$). This yields the desired T'. ∎

Proof of proposition 1.3: Invoke A to get a subgraph $H \subseteq G$ and a t-coloring T of H. Now use lemma 1.4 to transform T into a balanced coloring T' of H. Let S_i be the set of edges colored i. Since T' is balanced $\forall i, j \ |S_i| - |S_j| \leq 1$. Let $k = \max_i\{|S_i|\}$. There are two cases:

$k \leq f$**:** keep all edges and the claim follows.

$k > f$**:** remove edges from classes whose cardinality is bigger than f until $\forall i \ \ |S_i| = f$. In this case the solution is of size $t \cdot f$, thus it is the optimum. ∎

B The *Sparse t-Matching* Polytope

When dealing with an algorithm based on linear programming, a natural question one should ask is how far can the integral and the fractional solutions of the LP be.

Fact B.1. *There are instances for which the Sparse t-Matching LP has an integrality ratio of* $\frac{3t-2}{3t-1.5}$.

Figure 3 demonstrates such an integrality gap. We use $[i]$ to denote the value given by the LP solution to an edge. The graph in figure 3 meets constraints 4 to 6, and has a total weight of $3t - 1.5$. The optimal integral solution however takes all the $3t - 3$ edges going out of the triangle and *one* triangle edge. So the optimal integral solution has the weight of $3t - 2$. This yields an integrality gap of at least $\frac{3t-2}{3t-1.5}$. For large t this ratio is about $1 - \frac{1}{6t}$. Our algorithm gives an approximation ratio of about $1 - \frac{1}{\sqrt{t}}$ (for large t). Notice that this example in fact demonstrates the integrality gap for the b-matching problem if one uses an LP only with degree constraints (without general blossom constraints). This is true since the fractional solution obeys the degree constraints, however the largest edge subgraph with degree t contains only $3t - 2$ edges. This example also shows that if the rounding procedure returns a solution whose value is at least the value of the fractional solution, then (in this case) it must violate the degree constraints. This explains why our rounding procedure (section 8.1) returns a subgraph with $\Delta \leq t + 1$ rather then $\Delta \leq t$.

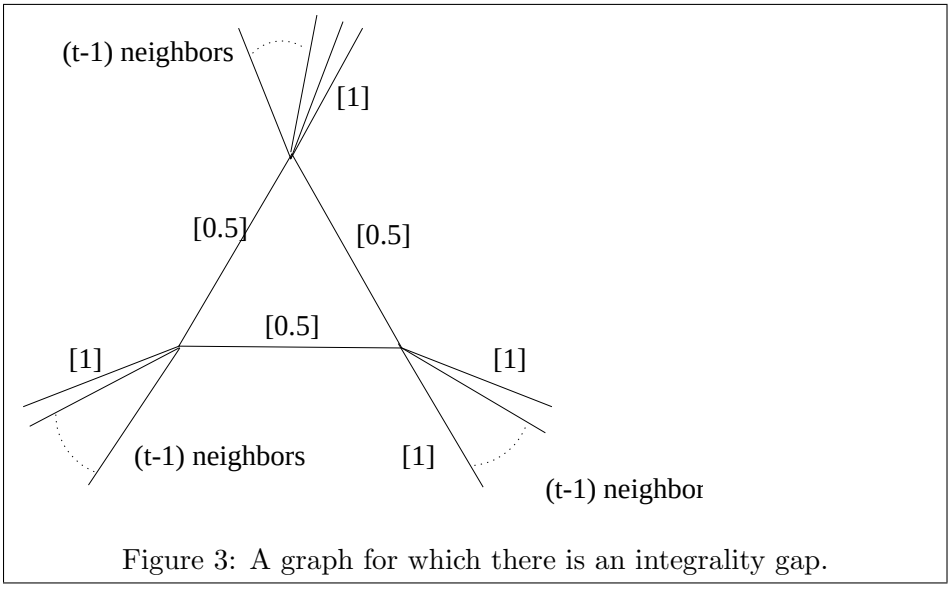

Figure 3: A graph for which there is an integrality gap.

B.1 A Strengthened LP Formulation for *Sparse t-Matching*

Edmonds and Johnson showed in [EJ70] a sufficient characterization of the b-matching polytope. They did this by generalizing the matching *blossom constraints* for b-matching. We can try to strengthen our LP formulation for *Sparse t-Matching* by further demanding that the solution be inside the $b-$matching polytope where $b_v = t$ for all v. In other words we add the generalized blossom constraints. First some notation: for $S \subseteq V$ define $X(S)$ to be the sum of values in the subgraph induced by S. Similarly for $T \subseteq E$ define $X(T) \stackrel{\triangle}{=} \sum_{e \in T} x_e$. For $S \subseteq V$ we denote by $\delta(S)$ the cut induced by S, i.e. all the edges that have exactly one vertex in S. We also define $b(S)$ to be the sum of bounds on the degrees of the vertices of S. In our case $b(S) = t \cdot |S|$. We now state the generalized blossom constraint:

$$\forall S \subseteq V, T \subseteq \delta(S) \text{ such that } b(S) + |T| \text{ is odd} \qquad X(S) + X(T) \leq \frac{1}{2}(b(S) + |T| - 1) \tag{7}$$

Note that if we choose $T = \phi$ and $t = 1$, this is the usual blossom constraint. Figure 3 shows that adding the blossom constraints strengthens the LP. Indeed the largest edge subgraph has at most $3t - 2$ edges, thus ruling out the fractional solution indicated in Figure 3.
Adding the general blossom constraints to the LP enabled us to come up with a rounding procedure

(different from the one in section 8.1) which returns a subgraph with $\Delta \leq t$ and $\rho \leq t + \sqrt{7t} + 8$. Details omitted.

Fact B.2. *There are instances for which the integrality ratio of the Sparse t-Matching LP with blossom constraints is* $\frac{3t+1}{3t+1.5}$

We use (i) to denote i parallel edges. A construction for an integrality gap in the strengthened LP can be seen in figure 4, where we assume t is even. (A similar example holds if t is odd). Let $\overline{x}$ denote the value assignment depicted in figure 4. The total values of edges in the graph is $3t + 1.5$. The optimal solution takes t edges from each outer triangle and one edge from the inner triangle adding up to $3t+1$. It is easy to see that the values of the edges satisfy the odd density constraints. It remains to show that the values satisfy the blossom constraints. Edmond's theorem implies that it is enough to show that the value assignment is a convex combination of $t-$matchings (the odd density of these matchings is not necessarily bounded by t). This convex combination is as follows: Let $\overline{z}$ be an assignment that gives the value 1 to all the edges of the outer triangles, and the value 0 to the edges of the inner triangle. Note that $\overline{z}$ is a $t-$matching because all degrees are bounded by t, yet it's odd density is $t + 1$. Let $\overline{y_i}$ $(i = \{1, 2, \ldots, \frac{t}{2}\})$ be an assignment that gives the value of 1 to the inner triangle edges and to the outer edges of the outer triangles. Each edge going out of the inner triangle has multiplicity of $\frac{t}{2}$. The assignment $\overline{y_i}$ gives the value 1 to all those edges except the i'th edge. Note that $\overline{y_i}$ is also a $t-$matching. Now it holds that

$$\overline{x} = \frac{1}{2}\overline{z} + \sum_{i=1}^{\frac{t}{2}} \frac{1}{t}\overline{y_i}$$

Hence $\overline{x}$ is within the $t-$matching polytope and therefore satisfies the blossom constraints.

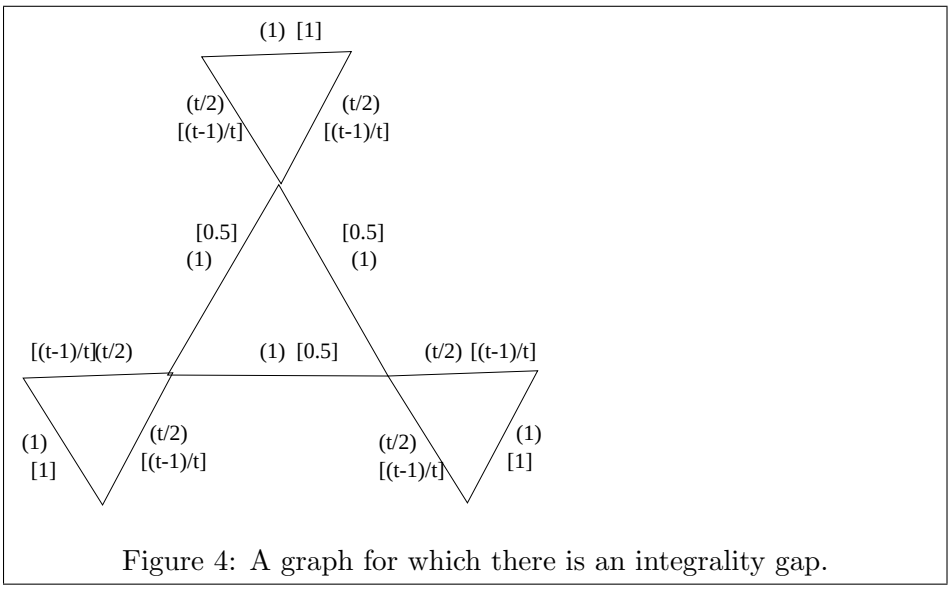

Figure 4: A graph for which there is an integrality gap.

B.1.1 The Case t=2

In case $t = 2$ we show that the integrality gap is at most 0.9. Consider the Petersen graph in figure 5. The optimal solution for *Sparse t-Matching* is to take 9 edges. Yet if we assign the value of $\frac{2}{3}$ to each edge then the sum of values is 10, and the assignment satisfies the degree, odd density and blossom constraints. This construction does not seem to generalize to larger t's.

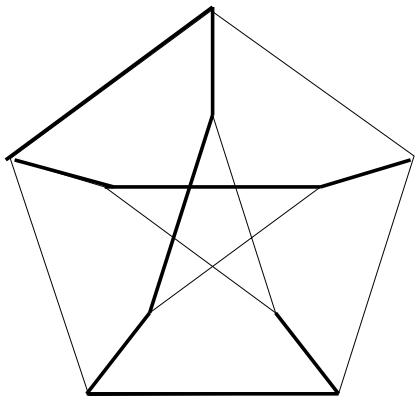

Figure 5: The Petersen graph. The bold edges depict a possible solution for $t = 2$.

C Hardness Result for Max edge 2-coloring

A solution for the *Max edge 2-coloring* problem is two (edge) disjoint matchings, thus it is a subgraph of G with bounded degree 2 and no odd cycles. This gives us another definition for the *Max edge 2-coloring* problem:

Problem C.1. Given an input multigraph G, find a largest (in edges) subgraph of G with degree bounded by 2 and no odd cycles.

Notice that the restriction of no odd cycles is equivalent to the restriction that the odd density of the output subgraph is bounded by 2: odd length cycles have odd density of 3, but even length cycles have odd density of 2. It follows that the problems *Max edge 2-coloring* and *max sparse 2-matching* are the same. This equivalence is unique for the parameter $t = 2$. A problem similar to C.1 was proven by Papadimitriou to be NP-hard [CP80]. Before we introduce the problem, we have to define the term 2-factor. Let G be a multigraph, a 2-*factor* of G is an edge induced subgraph of G, in which every vertex $v \in V(G)$ has degree of exactly 2. Papadimitriou showed that it is NP-hard to decide whether an input graph has a 2-factor without cycles of length 5 or less. We will use the basic ideas and some gadgets from Papadimitriou's reduction.
The reduction will be from MAX 3 SAT-3. We will use the term *good* 2-factor to denote a 2-factor without odd cycles. Given an instance with clauses $C_1, C_2, ..., C_m$ and variables $x_1, x_2, ..., x_n$ we build a graph which has a good 2-factor iff the instance is satisfiable. This will imply that *Max edge 2-coloring* is NP-hard.

C.1 The Reduction

To each variable x_l we associate a component γ_l (figure 6). Note that this component has exactly two good 2-factors: one contains e_{l_0} and the other contains e_{l_1}. This corresponds to a true/false assignment for x_l. To each clause C_i we associate a component Γ_i (figure 7). The edges $j_{i_1}, j_{i_2}, j_{i_3}$ correspond to the literals $y_{i_1}, y_{i_2}, y_{i_3}$ of C_i. Intuitively the edge j_{i_k} is chosen by a good 2-factor iff literal y_{i_k} is true in the corresponding assignment. The component Γ_i has the property that every good 2-factor of it must contain a none empty subset of edges $A \subseteq \{j_{i_1}, j_{i_2}, j_{i_3}\}$. Moreover for every such non empty A there exists a good 2-factor whose intersection with $\{j_{i_1}, j_{i_2}, j_{i_3}\}$ is exactly A. This will ensure that every good 2-factor of Γ_i must satisfy one of the literals of C_i. It remains to ensure the consistency between clauses and variables. This is done by using a *connector*

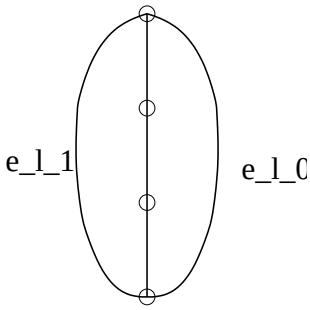

Figure 6: variable component γ_l

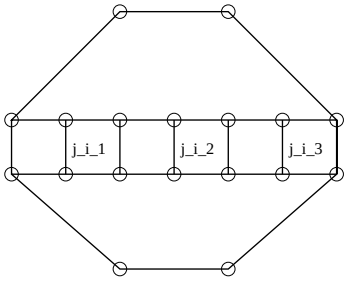

Figure 7: clause component Γ_i

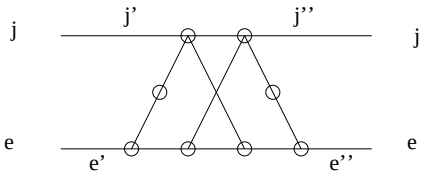

Figure 8: connector

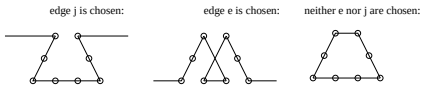

Figure 9: All possible configurations for a connector

component (figure 8) the connector has the following property: every good 2-factor of it contains j' iff it contains j'', similarly it contains e' iff it contains e''. Furthermore it can not contain all four edges j', j'', e', e''. It follows that in the context of 2-factor the edges j', j'' can be thought of as one *imaginary* edge j, similarly e', e'' can be thought of as one edge e. It is guaranteed that a good 2-factor does not contain both e, j.

We use the connector component in the following way: consider a variable x_l which has a positive literal in C_i e.g. $y_{i_1} = x_l$. We want to prevent the situation in which both e_{l_0} and j_{i_1} are chosen (since it can not be that x_l is false but it's positive literal has a true value). We prevent this by connecting the edges e_{l_0} and j_{i_1} using a connector. Since variable x_l may have up to 3 appearances and there is only one e_{l_0} edge, we chain copies of e_{l_0} as needed (see figure 10). The connector properties ensures that all the edges in the side of e_{l_0} behave as one edge; i.e. every good 2-factor contains one of them iff it contains all of them.

Cycles length parity does not change: notice that after we connect the components of the graph, edge j_{i_1} becomes *imaginary*. This is due to the fact that it is replaced by two connector edges j'_{i_1}, j''_{i_1} which behave the same. An important fact is that if the edges j'_{i_1}, j''_{i_1} are contained in a good 2-factor then they are a part of an odd path which goes in and out of the corresponding connector (as illustrated in figure 9). Since the length of this path is odd, it has the same parity of a single edge. It follows that all the properties of the clause component (which depend on the parity of cycles) are still valid. A similar thing happens with the imaginary edge e_{l_0}. It follows that the resulting graph has a good 2-factor iff the MAX 3 SAT-3 instance has a satisfying assignment.

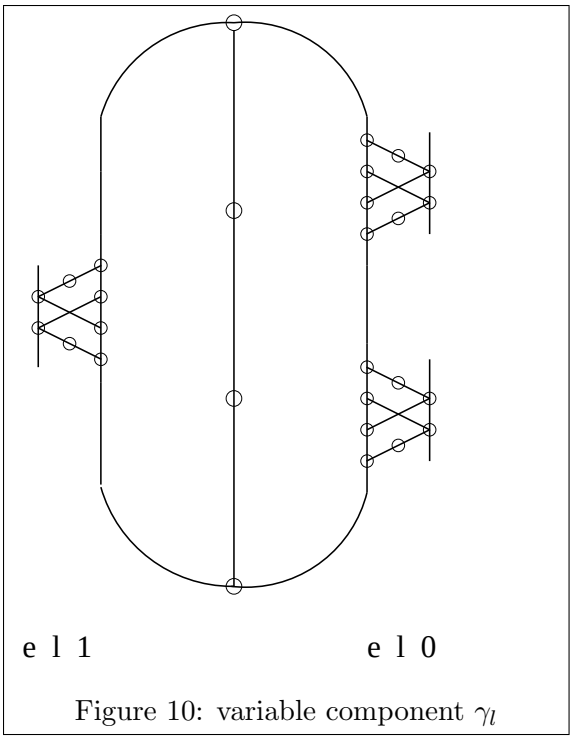

Figure 10: variable component γ_l

C.2 It is NP-hard to Approximate *Max edge 2-coloring* Better Than $1-\epsilon$

The reduction maps a MAX 3 SAT-3 instance A into a graph G. A vertex $v \in V(G)$ is called *perfect* with respect to a solution of G if it has degree 2 in this solution. Denote by OPT(A) the fraction of clauses satisfied by an optimal assignment, denote by OPT(G) the fraction of edges in an optimal solution for G out of $|V(G)|$ ($|V(G)|$ is the best possible). Since MAX 3 SAT-3 is APX-Complete then there exists $\delta < 1$ such that distinguishing between instances which have $OPT = 1$ and instances which have $OPT \leq \delta$ is NP-hard. We will prove the following connection between OPT(A) and OPT(G): there exists an $\epsilon_1 < 1$ such that for every instance A there exists $\epsilon \leq \epsilon_1$ (dependent on A and can be easily computed from A) s.t. $OPT(G) = \epsilon + (1-\epsilon) \cdot OPT(A)$. It follows that if $OPT(A) \leq \delta$ then $OPT(G) \leq \epsilon_1 + (1-\epsilon_1)\delta$, if $OPT(A) = 1$ then $OPT(G) = 1$. Thus it is NP-hard to approximate the new problem better then $\epsilon_1 + (1-\epsilon_1)\delta$ which is strictly smaller then one. The number of clauses and variables in an instance A are linearly related. For each variable/clause there is a bounded number of edges in G, so both the instance size and the optimum value (as an integer rather then as a fraction out of $|V(G)|$) grow only linearly by the reduction. To show the existence of such $\epsilon_1 < 1$ we will show that for every unsatisfied clause in an optimal solution of an instance A we have to pay in exactly two unperfect vertices in an optimal solution for G (i.e. one edge less). This follows from the following facts that can be verified by inspecting the gadgets of section C.1:

- An optimal assignment for the MAX 3 SAT-3 instance can be translated into a solution for G in which all variable components are perfect, all satisfied clause components are perfect and only clause components of unsatisfied clauses are not perfect. They have exactly two vertices which are not perfect and all other vertices are perfect.
- We may assume without loss of generality that in an optimal solution of G the only vertices with degree less than 2 are inside clause components. Also if a clause component is not perfect

(i.e. contains a vertex which is not perfect) then it contains exactly 2 vertices which are not perfect and all the other vertices in the component are perfect.

- If an optimum solution for the MAX 3 SAT-3 instance does not satisfy m' clauses then an optimum solution for G will have exactly $2m'$ vertices with degree 1 and all other vertices with degree of 2.

D Even Cycles

We present an algorithm that given a multigraph G finds an even length cycle (not necessarily simple), or asserts that there isn't one.

Algorithm find-even-cycles

1. Initially all the vertices of the graph are marked as *simple*. During the run of the algorithm the vertices will be marked either *simple* or *shrunk*.
2. Find a cycle in the graph. This can be done using the *DFS* algorithm. Call the cycle found C. If there are no cycles in the graph return: *NO CYCLE*.
3. If C is of even length *or* C is not a simple cycle *or* C has a chord *or* C contains a vertex which is shrunk, then return C and stop.
4. Shrink the vertices of C into one vertex v_C and set this vertex as *shrunk*. The shrinking process may create parallel edges. Return to step 2.

Lemma D.1. *Algorithm* ***find-even-cycles*** *returns a cycle C iff there exists an even length cycle in the graph.*

Proof: Assume algorithm **find-even-cycles** returns a cycle C. If C contains only simple vertices then all the vertices of C belong to G as well. Therefore according to claim 8.1 C contains an even length cycle in G. Assume C contains a shrunk vertex v_D. v_D can be blown up again to be the original cycle D. Since D is odd, we can augment the cycle C by adding to it either an odd or an even path of D. Thus all shrunk vertices of C can be blown up to create an even length cycle in G. Assume algorithm **find-even-cycles** returns *NO CYCLE*. This means that when the algorithm terminates, it holds a tree composed of simple and shrunk vertices. If we blow up the shrunk vertices we see that G looked like a tree when some of it's vertices are in fact simple odd cycles. Hence G has no even cycles. ■

Proof of claim 8.1 a: If an odd cycle is not simple, it can be divided into two cycles, one of which is odd length and one of which is even length, contradicting the assumption that there are no even length cycles in the graph.
b: Let C_1 and C_2 be two non disjoint odd cycles. We show that $C_1 \cup C_2$ contain an even length cycle. Let T be the set of edges that appear in both cycles. Let H be the graph formed by the symmetric difference of C_1, C_2; i.e. $E(H) = (E(C_1) \cup E(C_2)) \setminus T$. Note that $|E(H)| = |E(C_1)| + |E(C_2)| - 2|T|$ which is even. Furthermore all vertices in H have an even degree. We deduce that every component of H is an Euler cycle. If H has a component with an even number of edges then we are done. Otherwise every component of H is an odd cycle and there is an even number of components.
$C_1 \cup C_2$ form a graph without bridges. It follows that there exists a cycle in $C_1 \cup C_2$ that connects components of H. Such a cycle is depicted in figure 11; the bold edges represent edges of T. Each component of H can be decomposed to an odd path and an even path. We conclude that an even cycle can be formed. ■

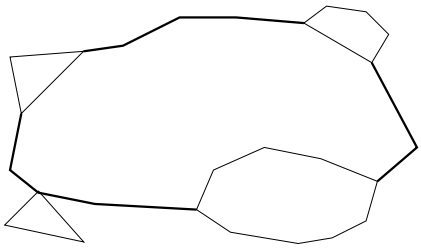

Figure 11: A cycle formed by components from H