Athens University of Economics and Business
Department of Informatics
Graduate Program in Computer Science

Scheduling in Computer and Communication Systems and Generalized Graph Coloring Problems

Ph.D. Thesis

by

Giorgio Lucarelli

Athens, October 2009

Athens University of Economics and Business
Department of Informatics
Graduate Program in Computer Science

Scheduling in Computer and Communication Systems and Generalized Graph Coloring Problems

Ph.D. Thesis by Giorgio Lucarelli

Advisory committee: Ioannis Milis (Supervisor)
Evangelos Magirou
Martha Sideri

Approved on the 2nd October 2009 by

Elias Koutsoupias, Professor
Department of Informatics and Telecommunications
National and Kapodistrian University of Athens

Evangelos Magirou, Professor
Department of Informatics
Athens University of Economics and Business

Ioannis Milis, Associate Professor
Department of Informatics
Athens University of Economics and Business

Vangelis Th. Paschos, Professor
LAMSADE
Université Paris-Dauphine

Martha Sideri, Associate Professor
Department of Informatics,
Athens University of Economics and Business

Stathis Zachos, Professor
School of Electrical and Computer Engineering
National Technical University of Athens

Vassilis Zissimopoulos, Professor
Department of Informatics and Telecommunications
National and Kapodistrian University of Athens

Acknowledgements

Before beginning, I would like to thank my supervisor Ioannis Milis for his support and mentoring over these years. I would also like to thank the members of my advisory committee, Evangelos Magirou and Martha Sideri, as well as, the members of the evaluation committee, Elias Koutsoupias, Vangelis Th. Paschos, Stathis Zachos and Vassilis Zissimopoulos, for their observations and comments.

During my doctorate, I worked with Evripidis Bampis, Nicolas Bourgeois, Alexander Kononov, Ioannis Milis and Vangelis Th. Paschos. I want to thank them all for the collaboration.

My graduate studies were funded by the project PENED 2003. The project is cofinanced 75% of public expenditure through EC–European Social Fund, 25% of public expenditure through Ministry of Development–General Secretariat of Research and Technology of Greece and through private sector, under measure 8.3 of Operational Programme "Competitiveness" in the 3rd Community Support Programme.

Many thanks to my friends Danae, Michael and Pantelis for going out and having a great time. Finally, special thanks to Katerina for her patience and support for more than ten years.

Contents

Summary

The thesis deals with weighted generalizations of the classical graph vertex/edge-coloring problems motivated by scheduling arising in computer and communication systems.

The most general problems we attack are called bounded max-vertex/edge-coloring problems and take as input a vertex/edge weighted graph and a bound b. In these problems each color class is of cardinality at most b and of weight equal to that of the heaviest vertex/edge in this class. The objective is to find a proper coloring of the input graph minimizing the sum of all color classes' weights. For unit weights these problems reduce to the known bounded coloring problems, while in the absence of the cardinality bound we get the (unbounded) max-coloring problems.

The max-coloring problems have been well motivated and studied in the literature. Max-vertex-coloring problems arise in the management of dedicated memories, organized as buffer pools, which is the case for wireless protocol stacks like `GPRS` or `3G`. Max-edge-coloring problems arise in switch based communication systems, like `SS/TDMA`, where messages are to be transmitted through direct connections established by an underlying network. Moreover, max-coloring problems correspond to scheduling jobs with conflicts in multiprocessor or batch scheduling environments. However, in all practical applications there exist physical constraints on the number of entities (corresponding to vertices/edges of a graph) assigned the same resource (color), which motivate the study of the bounded max-coloring problems.

In the first part of the thesis we present new complexity and approximation results for several variants of the max-edge-coloring problem with respect to the class of the underlying graph. In particular, we present polynomial algorithms for special graph classes (bounded degree trees, stars of chains) and approximation results for NP-complete variants. For bipartite graphs we present a series of four approximation algorithms; the last of them achieves an 1.74 approximation ratio and improves substantially the known ratio of 2. For trees we give a 3/2-approximation algorithm, two parameterized approximation algorithms and finally a PTAS. We also prove that the problem is NP-complete for complete bi-valued graphs and we present an asymptotic 4/3-approximation algorithm for general bi-valued graphs.

In the second part of the thesis we give the first known results for the bounded max-coloring problems. For the bounded max-edge-coloring problem, we prove approximation factors of at most 3 for general and bipartite graphs and 2 for trees. Furthermore, we show that this bounded problem is NP-complete for trees. This is the first complexity result for any max-coloring problem on trees. For the bounded

max-vertex-coloring problem we present a generic scheme which becomes a 17/11-approximation algorithm for bipartite graphs, a PTAS for bipartite graphs when b is fixed and also a PTAS for trees even if b is part of the problem's instance.

Περίληψη

Η ερευνητική εργασία που παρουσιάζεται στη διατριβή επικεντρώνεται στην αντιμετώπιση ανοικτών ερωτημάτων που αφορούν την πολυπλοκότητα και την προσεγγισιμότητα γενικευμένων προβλημάτων χρωματισμού (coloring) των κόμβων/ακμών ενός γράφου, τα οποία προκύπτουν ως προβλήματα χρονοπρογραμματισμού σε συστήματα υπολογιστών και επικοινωνιών.

Τα γενικότερα από τα προβλήματα που μελετώνται ονομάζονται προβλήματα *φραγμένου μέγιστου χρωματισμού κόμβων/ακμών* (bounded max-vertex/edge-coloring). Σε αυτά τα προβλήματα δεδομένου ενός γράφου με βάρη στους κόμβους/ακμές και ενός ακεραίου b, αναζητούμε ένα χρωματισμό των κόμβων/ακμών του γράφου όπου κάθε χρώμα χρησιμοποιείται το πολύ b φορές και το βάρος του ισούται με το βάρος του μεγαλύτερου κόμβου/ακμής που περιέχει. Στόχος είναι η εύρεση ενός χρωματισμού ο οποίος ελαχιστοποιεί το άθροισμα των βαρών όλων των χρωμάτων. Στην περίπτωση όπου τα βάρη είναι όλα ίδια, τα προβλήματα αυτά είναι γνωστά ως προβλήματα *φραγμένου χρωματισμού κόμβων/ακμών* (bounded coloring), ενώ αν δεν υπάρχει περιορισμός στην εμφάνιση των χρωμάτων προκύπτουν τα προβλήματα *μέγιστου χρωματισμού κόμβων/ακμών* (max-coloring).

Τα προβλήματα μέγιστου χρωματισμού έχουν μελετηθεί στη βιβλιογραφία τα τελευταία χρόνια και για καθένα από αυτά έχουν παρουσιαστεί πρακτικές εφαρμογές του σε προβλήματα χρονοπρογραμματισμού σε συστήματα υπολογιστών και επικοινωνιών. Το πρόβλημα μέγιστου χρωματισμού κόμβων εμφανίζεται σε συστήματα διαχείρισης μνήμης οργανωμένης σε δεξαμενές, όπως για παράδειγμα συμβαίνει με τη διαχείριση της στοίβας πρωτοκόλλων στα σύγχρονα συστήματα κινητής τηλεφωνίας (GPRS και 3G). Το πρόβλημα μέγιστου χρωματισμού ακμών εμφανίζεται σε συστήματα μεταγωγής μηνυμάτων, π.χ. στα δορυφορικά συστήματα επικοινωνιών (SS/TDMA), όπου τα μηνύματα πρέπει να μεταφερθούν μέσω απευθείας συνδέσεων που εγκαθιδρύονται από ένα υποκείμενο δίκτυο επικοινωνιών. Επίσης, τα προβλήματα μέγιστου χρωματισμού αντιστοιχούν στο χρονοπρογραμματισμό αμοιβαίως αποκλειόμενων εργασιών σε περιβάλλοντα πολυεπεξεργαστών ή επεξεργασίας δεσμών εργασιών. Σε όλες τις παραπάνω εφαρμογές, υπάρχουν στην πράξη φυσικοί περιορισμοί στον αριθμό των οντοτήτων (που αντιστοιχούν στους κόμβους/ακμές ενός γράφου) που είναι δυνατόν να ανατεθούν στον ίδιο φυσικό πόρο (χρώμα). Οι περιορισμοί οδηγούν στα προβλήματα φραγμένου μέγιστου χρωματισμού που μελετώνται για πρώτη φορά.

Στο πρώτο μέρος αυτής της διατριβής παρουσιάζονται αποτελέσματα πολυπλοκότητας και προσεγγισιμότητας του προβλήματος *μέγιστου χρωματισμού ακμών* για διάφορες κλάσεις γράφων. Πιο συγκεκριμένα, παρουσιάζονται πολυωνυμικοί αλγόριθμοι για ει-

δικές κλάσεις γράφων (δέντρα φραγμένου βαθμού και αστέρια αλυσίδων) καθώς και προσεγγιστικοί αλγόριθμοι για κλάσεις γράφων όπου το πρόβλημα είναι NP-πλήρες. Για διμερείς γράφους δίνεται μία σειρά από τέσσερις προσεγγιστικούς αλγορίθμους, από τους οποίους ο τελευταίος βελτιώνει τον λόγο προσέγγισης από 2 σε 1.74. Για δέντρα παρουσιάζεται ένας αλγόριθμος με λόγο προσέγγισης 3/2, δύο παραμετρικοί αλγόριθμοι και τέλος ένα προσεγγιστικό σχήμα. Επίσης, αποδεικνύεται ότι το πρόβλημα είναι NP-πλήρες σε πλήρεις γράφους με μόνο δύο διαφορετικά βάρη στις ακμές τους και δίνεται ένας αλγόριθμος με ασυμπτωτικό λόγο προσέγγισης 4/3 για γενικούς γράφους με δύο βάρη.

Στο δεύτερο μέρος της διατριβής παρουσιάζονται τα πρώτα αποτελέσματα για προβλήματα *φραγμένου μέγιστου χρωματισμού*. Για το πρόβλημα φραγμένου μέγιστου χρωματισμού ακμών, δίνονται αλγόριθμοι με λόγο προσέγγισης 3 για γενικούς και διμερείς γράφους και 2 για δέντρα. Επιπροσθέτως, αποδεικνύεται ότι το πρόβλημα είναι NP-πλήρες για δέντρα, το οποίο είναι το πρώτο αποτέλεσμα πολυπλοκότητας για όλα τα προβλήματα (φραγμένου) μέγιστου χρωματισμού σε δέντρα. Για το πρόβλημα φραγμένου μέγιστου χρωματισμού κόμβων παρουσιάζεται ένα γενικευμένο σχήμα, μέσω του οποίου επιτυγχάνεται (i) λόγος προσέγγισης 17/11 για διμερείς γράφους, (ii) ένα προσεγγιστικό σχήμα για διμερείς γράφους αν το b είναι φραγμένο και (iii) ένα προσεγγιστικό σχήμα για δέντρα ακόμα και αν το b αποτελεί μέρος του στιγμιότυπου του προβλήματος.

Chapter 1

Introduction

In this first chapter of the thesis, we introduce weighted generalizations of the classical vertex and edge coloring problems, which correspond to scheduling problems arising in computer and communication systems. We also give an outline of the next chapters of the thesis and a preview of our results.

1.1 Generalized graph coloring problems

A *vertex-* (resp. *edge-*) *coloring* of a graph $G = (V, E)$ is a partition $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$ of its vertex (resp. edge) set into color classes such that each class C_i constitutes an independent set (resp. matching) of G. Such a coloring is called a proper coloring of G. The classical vertex- (resp. edge-) coloring problem asks for a proper coloring of a graph G such that the number, k, of colors is minimized. This minimum number of colors required to color the vertices (resp. edges) of a graph G is also known as the *chromatic number* $\chi(G)$ (resp. *chromatic index* $\chi'(G)$) of G.

In the *bounded vertex-coloring* [4] (resp. *bounded edge-coloring* [2]) problem we are, in addition, given a positive integer b and we ask for a proper coloring where each color appears at most b times, i.e., $|C_i| \leq b$, $1 \leq i \leq k$. The goal in both bounded coloring problems is also to minimize the number, k, of colors.

In several application domains the following weighted generalizations of graph coloring problems arise: Each vertex u (resp. edge e) of G is associated with a positive integer weight $w(u)$ (resp. $w(e)$) and we ask again for a partition $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$ of the vertex (resp. edge) set of G into color classes, each one of weight $w_i = \max\{w(v) \mid v \in C_i\}$, (resp. $w_i = \max\{w(e) \mid e \in C_i\}$), such that the total weight of the partition $W = \sum_{i=1}^{k} w_i$ is minimized. These coloring problems are known as *max-(vertex-)coloring* [52] and *max-edge-coloring*.

The presence of a bound b to the cardinality of each color leads to the *bounded max-vertex-coloring* and *bounded max-edge-coloring* problems, respectively.

In this thesis we shall denote the above coloring problems as follows:

- VC/EC: vertex/edge-coloring problem

- VC(b)/EC(b): bounded vertex/edge-coloring problem
- VC(w)/EC(w): max-vertex/edge-coloring problem
- VC(w, b)/EC(w, b): bounded max-vertex/edge-coloring problem

where VC and EC are used to denote the vertex and edge versions of the classical coloring problem, respectively, b indicates the existence of a cardinality bound to colors, and w shows the existence of weights on the vertices or the edges of the input graph.

Clearly, both VC(b) and VC(w) reduce to the classical VC problem, if $b = |V|$ and $w(u) = 1,\ \forall u \in V$, respectively. Moreover, VC(w, b) reduces to all VC, VC(b) and VC(w) problems. Analogous reductions exist between the EC, EC(b), EC(w) and EC(w, b) problems. These reductions are shown in Figure 1.1, where each directed edge indicates that the source problem reduces to the destination one.

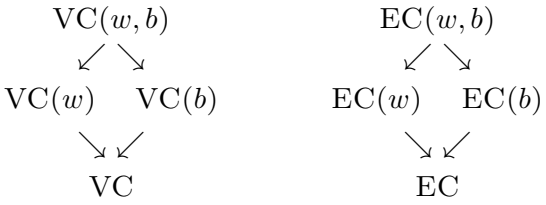

Figure 1.1: Reductions between coloring problems.

Remark that any generalization of the edge-coloring problem on a general graph G is equivalent to the corresponding vertex-coloring problem on the line graph, $L(G)$, of G. Thus, the results for any vertex-coloring problem on a graph G apply also to the corresponding edge-coloring one on the graph $L(G)$ and *vice versa*, if both G and $L(G)$ are in the same graph class. Note, however, that this is true for general graphs and chains, but not for the most other special graph classes, including bipartite graphs and tree, since they are not closed under line graph transformation (e.g., the line graph of a bipartite graph is not anymore a bipartite one).

1.2 Motivation: Scheduling in computer and communication systems

In this section we motivate both vertex and edge variants of our coloring problems as scheduling problems arising in computer and communication systems.

In many applications with strict memory constraints a dedicated memory allocation manager is often used in order to optimize the memory utilization and the performance of the system. For example, in wireless communications [51, 52], where mobile devices have to manage a protocol stack (e.g., `GPRS` or `3G`) with stringent memory requirements, a dedicated memory manager is the natural choice. A common approach in the design of such a memory manager is the use of an amount of

the total memory as a segregated buffer pool which consists of a set of buffers of different sizes. As a memory request arrives it is scheduled to a free buffer of large enough size. Memory requests in different time intervals can use the same buffer which, however, should be of size greater than the largest request allocated to this buffer. The aim of such a memory manager is to minimize the total size of the buffers used to service a set of requests and thus the amount of the total memory which is used as a segregated buffer pool.

The above problem correspond directly to the VC(w) problem, with memory requests corresponding to the set of vertices of a graph G and their weights to the size of the memory requests. An edge between two vertices of G exists if the corresponding requests have to be served in overlapping time intervals. So, a partition of G into independent sets corresponds to an allocation of the requests to buffers, with each set corresponding to a set of requests that can be allocated to the same buffer. The weight of a set corresponds to the buffer size which should be at least equal to the largest memory request in this set. Thus, the total size of the buffers needed to service a given set of requests is equal to the total weight of the partition of G.

On the other hand, the EC(w) problem arises in single hop communication systems, like `SS/TDMA` [35, 46], where messages are to be transmitted directly from senders to receivers through connections established by an underlying switching network. Any node of such a system cannot participate in more than one transmissions at the same time, while messages between different pairs of senders and receivers can be transmitted simultaneously. The scheduler of such a system establishes successive configurations of the switching network, each one routing a non-conflicting subset of the messages from senders to receivers. Given the transmission time of each message, the transmission time of each configuration equals to the longest message transmitted. Moreover, in practice, there is a non negligible setup delay to establish each configuration. The aim is to find a sequence of configurations such that all the messages are transmitted and the total transmission time (including setup delays) is minimized.

It is easy to see that the above situation corresponds directly to the EC(w) problem: senders and/or receivers correspond to the vertices of the graph G, (transmission times of) messages correspond to (weights of) edges of G and configurations correspond to colors. Although the graph G obtained is originally a weighted directed multi-graph it can be considered as an undirected one, since the directions of its edges do not play any role in the objective function we study here.

The presence of the setup delay in the instance of the EC(w) problem, can be easily handled: by adding d to the weight of all edges of G, the weight of each color class will be also increased by d, incorporating its set up delay. Furthermore, a standard idea to decrease the completion time of a schedule is to allow preemption [1, 17, 35], i.e., interrupt the service of a (set of) scheduled activity(ies) and complete it (them) latter. It is obvious that allowing preemption in the EC(w) problem will result in increasing the number of colors in a solution. In this case, the presence of a set up delay d plays a crucial role in the hardness of the (preemptive) EC(w) problem.

In all applications mentioned above, context-related entities require their service by physical resources for a time interval. However, there exists in practice a natural constraint on the number of entities assigned the same resource or different resources at the same time. Indeed, the number of memory requests assigned the same buffer is determined by strict deadlines on their completion times, while the number of messages assigned at the same time to different channels is bounded by the number of the available resources. The existence of such a constraint motivates the bounded max-coloring problems VC(w, b) and EC(w, b).

The VC(b) and VC(w, b) problems are also equivalent with scheduling problems where jobs correspond to the vertices of a graph $G = (V, E), |V| = n$, which describes incompatibilities between them. If all jobs have unit processing times, then the VC(b) problem is equivalent to the $Pb \mid graph,\ p_j = 1 \mid C_{max}$ scheduling problem, where the goal is to minimize the makespan of the schedule of n jobs on b processors under the constraint that adjacent jobs cannot be scheduled at the same time. This problem is also known as Mutual Exclusion Scheduling (MES) [4]. If jobs have different processing times, then the VC(w, b) problem is equivalent to the $1 \mid p - batch,\ graph,\ b < n \mid C_{max}$ parallel batch scheduling problem, where the goal is to minimize the makespan of the batch schedule of n jobs on one machine under the constraints that adjacent jobs cannot be scheduled at the same batch and each batch is of cardinality at most b [22, 26].

Analogous scheduling definitions can be given for both the EC(b) and EC(w, b) problems. For these problems the jobs correspond to the edges of the graph, while adjacent jobs cannot be scheduled at the same time.

1.3 Organization of the thesis

In this thesis we present complexity and approximation results for the coloring problems introduced in the previous section, which correspond to scheduling problems arising in computer and communication systems.

In Chapter 2, we review the known results for the coloring problems, with respect to the class of the underlying graph.

In Chapter 3, we exploit the relation between our coloring problems and two other well studied problems, and we give two preliminary results. First we use a relation with the *list coloring* problem and we obtain that all VC(w), VC(w, b), EC(w) and EC(w, b) problems are polynomial on trees, if the number, k, of colors is fixed. Next, through a transformation to the *set cover* problem we show an H_b-approximation algorithm for all VC(b), VC(w, b), EC(b), EC(w, b) problems on general graphs, if the cardinality bound, b, is fixed. Furthermore, we present two more observations on the approximability of our coloring problems. The first shows the existence of an $(e \cdot \rho)$-approximation algorithm for any max-coloring problem on a hereditary class of graphs, where ρ is a known approximation ratio for the corresponding unweighted coloring problem on the same class. The second observation shows a $b/2$ approximation ratio for all the bounded coloring problems. We close this chapter with the notation that we shall use in this thesis.

In Chapter 4, we deal with the EC(w) problem on general, bipartite and bi-valued graphs. We first improve the analysis of a well known 2-approximation algorithm presented in [46], and using this analysis we derive even better approximation ratios for general and bipartite graphs. Next, we present four approximation algorithms for the EC(w) problem on bipartite graphs, each one improving the previous ones for some values of the maximum degree of the input graph. The first three algorithms achieve approximation ratios increasing with the maximum degree of the graph, while the forth one improves the known 2 approximation ratio to 1.74 for bipartite graphs. Finally, we prove that the EC(w) problem is NP-complete for complete bi-valued graphs and we present an asymptotic 4/3-approximation algorithm for general bi-valued graphs.

In Chapter 5, we present results for the EC(w) problem on trees. We first show that the EC(w) problem is polynomial for stars of chains (also known as spiders). Next, we present a 2-approximation algorithm for trees. Combining this algorithm and our analysis of the known 2-approximation one [46], we obtain a 3/2 approximation ratio for the EC(w) problem on trees. Next, we propose two moderately exponential approximation algorithms for trees that improve the 3/2 ratio with running time much better than that needed for the computation of an optimal solution. More interestingly, we have also finally succeeded to derive a PTAS for the EC(w) problem on trees.

In Chapter 6, we give complexity and approximation results for the EC(w, b) problem. We first prove lower and upper bounds to the number of colors of any solution for the EC(w, b) problem. Using these bounds, we present an approximation algorithm of ratios $3 - \frac{2}{\sqrt{2b}}$ and $3 - \frac{2}{\sqrt{b}}$ for general and bipartite graphs, respectively. Furthermore, we show that the EC(w, b) problem is NP-complete for trees, which is the first complexity result for any max-coloring problems on trees. For this last problem we also give a 2-approximation algorithm.

In Chapter 7, we present a 2-approximation algorithm for the VC(w, b) problem on bipartite graphs. This algorithm reduces to a 4/3-approximation algorithm for the VC(b) problem on bipartite graphs, closing the approximability question for this problem. Then, we use this 2-approximation algorithm to obtain a generic scheme that leads to the following results for the VC(w, b) problem: (i) a 17/11-approximation algorithm for bipartite graphs, (ii) a PTAS for bipartite graphs, when b is fixed, and (iii) a PTAS for trees, even if b is part of the problem's instance.

In Chapter 8 we summarize our results and we discuss questions that remain still open.

For the reader to have a chart preview of the problems we study and the current state of the art on their complexity and approximability, we include here Table 1.1, where our results (in bold) are summarized together with known ones that are reviewed in the next chapter.

Problem	General graphs		Bipartite graphs		Trees	
	Lower Bound	Upper Bound	Lower Bound	Upper Bound	Lower Bound	Upper Bound
VC(b)	$\|V\|^{1-\epsilon}$ [58]	$\min\left\{\begin{array}{l}\lceil\frac{\mathbf{b}}{\mathbf{2}}\rceil\\ \mathbf{H_b}^{(1)}\end{array}\right.$	4/3 [6]	**4/3**	OPT [42]	
VC(w)		$O\left(\frac{\|V\|}{\log\|V\|}\right)$ [22]	8/7 [20, 22, 51]		open$^{(2)}$	$PTAS$ [25, 51]
VC(w,b)		$\min\left\{\begin{array}{l}\lceil\frac{\mathbf{b}}{\mathbf{2}}\rceil\\ \mathbf{H_b}^{(1)}\end{array}\right.$	4/3 [6]	**17/11**	open$^{(2)}$	**PTAS**
EC(b)	4/3 [41]	4/3 [2]	OPT [8]		OPT [8]	
EC(w)		2 [46]	7/6 [20]	**1.74**	open$^{(2)}$	**PTAS**
EC(w,b)		$\min\left\{\begin{array}{l}\mathbf{3}-\frac{\mathbf{2}}{\sqrt{\mathbf{2b}}}\\ \lceil\frac{\mathbf{b}}{\mathbf{2}}\rceil\\ \mathbf{H_b}^{(1)}\end{array}\right.$	7/6 [20]	$\min\left\{\begin{array}{l}e\\ \mathbf{3}-\frac{\mathbf{2}}{\sqrt{\mathbf{b}}}\\ \lceil\frac{\mathbf{b}}{\mathbf{2}}\rceil\\ \mathbf{H_b}^{(1)}\end{array}\right.$	**NP-complete**	**2**

Table 1.1: Known and ours (in bold) approximability results for bounded and/or max coloring problems.
$^{(1)}$The ratio H_b holds only if b is fixed.
$^{(2)}$Even the complexity of the problem is unknown.

Chapter 2

Related work

In this chapter we review the known results for the coloring problems introduced in the previous chapter, with respect to the class of the underlying graph. Although our results concern general graphs, bipartite graphs and trees, for the sake of completeness we present here results for several other interesting classes of graphs.

2.1 Vertex/Edge-Coloring

The **vertex coloring** (or chromatic number) problem on general graphs is one of Karp's 21 NP-complete problems [45]. It is also known to be NP-complete on planar graphs even for three colors [32], although four colors suffice to color any planar graph (see for example, [54]), as well as on circular arc graphs [31]. On the other hand, the VC problem can be solved in polynomial time for several other classes of graphs, including perfect graphs [37], chordal and interval graphs [33], split graphs, comparability graphs and cographs [34]. For bipartite graphs, trees, cliques and chains the VC problem is trivially polynomial.

In [58] it is shown that it is NP-hard to approximate the VC problem on general graphs within a factor of $|V|^{1-\epsilon}$, for all $\epsilon > 0$, while an algorithm of approximation ratio $O\left(|V|\frac{(\log\log|V|)^2}{(\log|V|)^3}\right)$ has been presented in [38]. For planar graphs, the approximability question for the VC problem is closed, since the NP-completeness proof [32] and the four color theorem [54] lead to a 4/3 inapproximability result and approximation algorithm, respectively. Moreover, a 3/2-approximation algorithm has been proposed in [44] for the VC problem on circular arc graphs.

For the **edge coloring** (or chromatic index) problem, it is well known that its optimal solution consists of either Δ or $\Delta + 1$ colors [56]. However, it is NP-hard to decide between these two values even on cubic graphs [41] and on comparability graphs [11] (and hence on perfect graphs). The first result implies also a 4/3 inapproximability bound for the problem. On the other hand, the EC problem is solvable in polynomial time on bipartite graphs [47] (and hence on trees), and cliques [27].

There are many classes of graphs for which the VC problem is polynomial, while

the complexity of the EC problem still remains open. Examples of such classes are chordal, split and interval graphs and cographs. Moreover, the complexity of the EC problem is open for planar graphs, while partial results are known for split [12], interval [7] and planar graphs [55, 57].

As mentioned above, any graph has a $(\Delta+1)$-edge-coloring and this coloring can be found in $O\left(\min\{|V|\Delta\log|V|, |E|\sqrt{|V|\log|V|}\}\right)$ time [29]. Using such a $(\Delta+1)$-edge-coloring, a 4/3-approximation algorithm is obtained for the EC problem on general graphs of $\Delta \geq 3$; recall that the EC problem is trivially polynomial for graphs of maximum degree two.

Table 2.1 summarizes known complexity results for the classical coloring problems on several well-studied strong classes. Results for more specific classes of graphs can be found in [43, 49].

Graph	Vertex-Coloring	Edge-Coloring
general	NP-complete [45] $\omega \leq \chi \leq \Delta + 1$	NP-complete [41] $\chi' = \Delta$ or $\Delta + 1$ [56]
perfect	$\chi = \omega$ [37]	NP-complete
chordal	$\chi = \omega$ [33]	open
split	$\chi = \omega$ [34]	$\chi' = \begin{cases} \Delta+1, & \text{if } \Delta \text{ is odd [12]} \\ \text{open}, & \text{if } \Delta \text{ is even} \end{cases}$
comparability	$\chi = \omega$ [34]	NP-complete [11]
bipartite	$\chi = \omega = 2$	$\chi' = \Delta$ [47]
trees	$\chi = \omega = 2$	$\chi' = \Delta$
cograph	$\chi = \omega$ [34]	open
chains	$\chi = \omega = 2$	$\chi' = \Delta = 2$
circular arc	NP-complete [31]	
interval	$\chi = \omega$ [33]	$\chi' = \begin{cases} \Delta, & \text{if } \Delta \text{ is odd [7]} \\ \text{open}, & \text{if } \Delta \text{ is even} \end{cases}$
clique	$\chi = \omega = \lvert V\rvert$	$\chi' = \begin{cases} \Delta, & \text{if } \Delta \text{ is odd} \\ \Delta+1, & \text{if } \Delta \text{ is even} \end{cases}$ [27]
planar	NP-complete [32] $\chi \leq 4$ [54]	$\chi' = \begin{cases} \Delta, & \text{for } \Delta \geq 7 \text{ [55, 57]} \\ \text{open}, & \text{otherwise} \end{cases}$

Table 2.1: Complexity results for the classical coloring problems.

2.2 Bounded Vertex/Edge-Coloring

The **bounded vertex-coloring** (or Mutual Exclusion Scheduling [4]) problem is NP-complete for general, circular arc and planar graphs, as a generalization of the VC problem. The complexity of the VC(b) problem has been also extensively studied on special graph classes. It is NP-complete for cographs [6], interval graphs [6] (and hence for chordal and perfect graphs) and bipartite graphs even for three

colors [6]. This last result implies also a 4/3 inapproximability bound for the VC(b) problem on bipartite graphs as well as the NP-completeness of the VC(b) problem on comparability graphs. On the other hand, the VC(b) problem is polynomial for split graphs [6], trees [42] (and hence for chains) and cliques [19]. Note, finally, that for the number of colors, k^*, in an optimal solution of the VC(b) problem it holds that $\max\left\{\left\lceil\frac{|V|}{b}\right\rceil, \chi\right\} \leq k^* \leq \chi + \left\lfloor\frac{|V|-\chi}{b}\right\rfloor$ [40].

Table 2.2 summarizes known complexity results for the VC(b) problem. For further results the readers referred to [30] and the references therein.

Graph	Complexity
general	NP-complete$^{(1)}$
perfect	NP-complete
chordal	NP-complete
split	polynomial [6]
comparability	NP-complete
bipartite	NP-complete [6]
trees	polynomial [42]
cograph	NP-complete [6]
chains	polynomial
circular arc	NP-complete$^{(1)}$
interval	NP-complete [6]
clique	polynomial [19]
planar	NP-complete$^{(1)}$

Table 2.2: Complexity results for the bounded vertex-coloring problem. $^{(1)}$The NP-completeness comes from the VC problem.

The complexity of the **bounded edge-coloring** problem is related substantially to the complexity of the EC problem by the following proposition.

Proposition 1 (de Werra [18])**.** *For any $k \geq \Delta$, a bipartite multigraph has a decomposition into k colors such that for all i and j, $1 \leq i,j \leq k$, it holds that $||C_i| - |C_j|| \leq 1$.*

The proof of this proposition is based on the fact that the graph which is induced by the edges of any two colors, C_i and C_j, is a collection of chains and even cycles. Without loss of generality, assume that $|C_i| - |C_j| > 2$. Thus, there is at least one subchain in $C_i \cup C_j$ whose the number of edges from C_i is greater by one than the number of edges from C_j. A swap of the edges of C_i and C_j of such a subchain decreases by one the cardinality of C_i and increases by one the cardinality of C_j. Therefore, for any $k \geq \Delta$ we can create by successive swaps a solution of k colors such that $||C_i| - |C_j|| \leq 1$.

Clearly, if we replace the condition $k \geq \Delta$ by $k \geq \chi'$ then Proposition 1 holds also for general graphs. Note also that an optimal solution for the EC(b) problem

consists of at least $\left\lceil \frac{|E|}{b} \right\rceil$ colors. Thus, if $\left\lceil \frac{|E|}{b} \right\rceil > \Delta$ then an optimal solution for the EC(b) problem on general graphs can be found using Proposition 1. Otherwise, the EC(b) problem is as hard as the EC problem.

Concluding, the EC(b) problem is 4/3-inapproximable on general graphs, while a 4/3-approximation algorithm is obtained by Vizing's theorem [56] and the discussion above. On the other hand, the EC(b) problem is polynomial on bipartite graphs by Proposition 1, a result that has been also proved independently in [8] in matrix decomposition context. For complexity results for other classes of graphs see the edge-coloring column of Table 2.2, as well as [43, 49].

Finally, Alon in [2] proved that the EC(b) problem is polynomially solvable on general graphs if the cardinality bound b is fixed. To prove this, he based on the fact that "if $\chi' = \Delta + 1$ then $|E| \geq \frac{1}{8}\left(3\Delta^2 + 6\Delta - 1\right)$" [28].

2.3 Max-Vertex/Edge-Coloring

The **max-vertex-coloring** problem is strongly NP-hard even for (i) bipartite graphs and edge weights $w(e) \in \{1, 2, 3\}$ [22, 51] (and hence for comparability and perfect graphs), (ii) split graphs and edge weights $w(e) \in \{1, 2\}$ [22] (and hence for chordal graphs), (iii) planar bipartite graphs [20], and (iv) interval graphs [25, 52] (and hence for circular arc graphs).

Moreover, it has been shown that the VC(w) problem on bipartite graphs with edge weights $w(e) \in \{1, 2, 3\}$ [22, 51] and planar bipartite graphs [20] cannot be approximated within a ratio less than 8/7. This bound has been attained for general bipartite graphs [20, 51], while an $O(|V|/\log|V|)$-approximation algorithm for general graphs is known [22]. Although the complexity of the problem on trees is an open question, a PTAS for this case has been presented in [25, 51]. In addition, a 4-approximation algorithm has been presented in [52] for perfect graphs; this ratio has been improved to e in [24], using randomization/derandomization techniques. For k-colorable graphs, an approximation algorithm of ratio $\frac{k^3}{3k^2-3k+1}$ has been proposed in [25], leading to a 64/37-approximation algorithm for planar graphs. In addition, approximation algorithms of ratio 2 and 3 for interval and circular arc graphs, respectively, have been presented in [52], exploiting the relation between the VC(w) problem and online coloring. Furthermore, a PTAS for split graphs is known [20].

Moreover, the VC(w) problem is known to be polynomial on bipartite graphs and edge weights $w(e) \in \{1, 2\}$ [22], cographs [22], and chains [25]. In fact, the algorithm for chains can be also extended for graphs of $\Delta = 2$. Finally, a new algorithm for the VC(w) problem on chains which improves the complexity for this case from $O(|V|^2)$ to $O(|V| \cdot \log|V|)$ has been presented in [39].

Table 2.3 summarizes the above results for the VC(w) problem. Several results for even more restricted classes of graphs can be found in [20].

The **max-edge-coloring** problem is strongly NP-hard even for (i) complete balanced bipartite graphs [53], (ii) bipartite graphs of maximum degree three and edge weights $w(e) \in \{1, 2, 3\}$ [35, 46], (iii) cubic bipartite graphs [22], and (iv)

Graph	Lower Bound	Upper Bound
general	$\|V\|^{1-\epsilon(1)}$	$O\left(\frac{\|V\|}{\log\|V\|}\right)$ [22]
perfect	8/7	e [24]
chordal	NP-complete	e
split	NP-complete [22]	$PTAS$ [20]
comparability	8/7	e
bipartite	8/7 [20, 22, 51]	
bipartite, $w \in \{a, b\}$	OPT [22]	
trees	open$^{(2)}$	$PTAS$ [25, 51]
cograph	OPT [22]	
chains	OPT [25, 39]	
circular arc	NP-complete	3 [52]
interval	NP-complete [25, 52]	2 [52]
clique	OPT	
planar	$4/3^{(1)}$	64/37 [25]

Table 2.3: Known approximability results for the max-vertex-coloring problem. $^{(1)}$This result comes from the VC problem. $^{(2)}$Even the complexity of the problem is unknown.

cubic planar bipartite graphs with edge weights $w(e) \in \{1, 2, 3\}$ [20]. Moreover, it has been shown that the EC(w) problem on r-regular bipartite graphs cannot be approximated within a ratio less than $\frac{2^r}{2^r-1}$, which for $r = 3$ becomes 8/7 [22]. This inapproximability result has been improved to 7/6 for cubic planar bipartite graphs [20].

On the other hand, a simple greedy 2-approximation algorithm has been presented in [46] for bipartite graphs (in fact, the same algorithm applies also for general graphs). In addition, a $\frac{2\Delta-1}{3}$-approximation algorithm, for bipartite graphs of maximum degree Δ, has been presented in [22], which gives an approximation ratio of 5/3 for $\Delta = 3$. A new algorithm for bipartite graphs with $\Delta = 3$ has been presented in [20]; it achieves an approximation ratio of 7/6 which attains, for this case, the known 7/6 inapproximability bound. Finally, an algorithm that achieves approximation ratio $\rho_\Delta = \dfrac{\Delta}{\sum_{i=1}^{\Delta}\prod_{j=i}^{\Delta-1}(1-\frac{\rho_j}{\Delta})}$, for graphs of maximum degree Δ has been proposed in [25]. This ratio is smaller than 2 only for bipartite graphs of maximum degree $\Delta \leq 7$.

Moreover, the EC(w) problem is known to be polynomial for a few very special cases including complete balanced bipartite graphs and edge weights $w(e) \in \{1, 2\}$ [53], general bipartite graphs and edge weights $w(e) \in \{1, 2\}$ [22], and chains [25]. In fact, the last result was presented for the VC(w) problem, and holds also for the EC(w) problem, since chains are closed under the linegraph transformation.

Finally, the preemptive-EC(w) problem, without setup delays, for bipartite

graphs is equivalent to the preemptive open shop scheduling problem which can be solved optimally in polynomial time [48]. However, with the presence of a setup delay, d, required to establish each color, the preemptive-EC problem on bipartite graphs becomes strongly NP-hard [35] and non approximable within a factor less than 7/6 [17]. Approximation algorithms for this problem of factors 2 and $2 - \frac{1}{d+1}$ have been presented in [17] and [1], respectively.

2.4 Bounded Max-Vertex/Edge-Coloring

Clearly, any negative result for the VC(b)/VC(w) and EC(b)/EC(w) problems holds also for the VC(w, b) and EC(w, b) problems, respectively.

Known results for the VC(w, b) problem have appeared in the context of batch scheduling jobs with compatibilities (see e.g. [26]). In this problem the goal is to decompose the graph into a set of cliques (instead of colors/independent-sets). Thus, results for this problem on special graph classes lead to analogous results for the VC(w, b) problem on the complements of these classes. An interesting result that has been shown in this context is that the VC(w, b) problem is NP-complete for split graphs and $b = 3$ [9], since the complement of a split graph remains in the same class.

Furthermore, a polynomial algorithm for general graphs and $b = 2$ has been presented for the scheduling problem with compatibilities [10]. This algorithm can be used to obtain an analogous result for both VC(w, b) and EC(w, b) problems on general graphs, since general graphs are closed under complement and linegraph operations.

Finally, a 8/3-approximation algorithm is known for the preemptive-EC(w, b) problem on bipartite graphs [15].

Chapter 3

Preliminaries and Notation

In this chapter we first relate our coloring problems with two well studied problems, namely list coloring and set cover. Using these relations we obtain preliminary results for the (bounded) max-coloring problems when either the number, k, of colors or the cardinality bound, b, are fixed.

Next, we present two preliminary approximation results for the bounded max-coloring problems. The first one follows from a known general framework, which allows to convert a ρ-approximation algorithm for a coloring problem to an $e \cdot \rho$-approximation one for the corresponding max-coloring problem. The second approximation result follows by using a solution to a bounded coloring problem with cardinality bound $b = 2$ to approximate a solution for an arbitrary bound b.

We close this chapter by the notation that we shall use throughout this thesis.

3.1 Preliminaries

List coloring and fixed number of colors

In several algorithms that we shall present in the next chapters, the following decision problem has to be answered (we present here the bounded vertex version of this problem; unbounded and/or edge versions are defined similarly):

FEASIBLE-VC(w, b)
INSTANCE: A vertex weighted graph $G = (V, E)$, a sequence of k weights, $w_1 \geq w_2 \geq \ldots \geq w_k$, and an integer b.
QUESTION: Is there a a feasible solution $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$ to the VC(w, b) problem on G such that $\max_{v \in C_i} w(v) \leq w_i$ and $|C_i| \leq b$, $1 \leq i \leq k$?

The FEASIBLE-VC(w, b) problem is equivalent to the next well known variant of the vertex-coloring problem:

Bounded List Vertex-Coloring problem (VC(ϕ, b_i))
Instance: A graph $G = (V, E)$, a set of colors $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$, a list of colors $\phi(u) \subseteq \mathcal{C}$ for each $u \in V$, and integers b_i, $1 \leq i \leq k$.
Question: Is there a k-coloring of G such that each vertex u is assigned a color in its list $\phi(u)$ and every color C_i is used at most b_i times?

Indeed, an instance of the Feasible-VC(w, b) problem on a graph G, where we are given k weights $w_1 \geq w_2 \geq \ldots \geq w_k$ and an integer b, can be easily transformed to the next equivalent instance of the VC(ϕ, b_i) problem: is there a k-coloring of G where each vertex $u \in V$ is assigned a color in $\phi(u) = \{C_i : w_i \geq w(u),\ 1 \leq i \leq k\}$ and every color C_i is used at most $b_i = b$ times? A "yes" answer to this instance of the VC(ϕ, b_i) problem corresponds to the existence of a feasible solution $C = \{C_1, C_2, \ldots, C_k\}$ for the VC(w, b) problem of weight $W = \sum_{i=1}^{k} w_i$.

In an analogous way, we can use the List Vertex-Coloring (VC(ϕ)), Bounded List Edge-Coloring (EC(ϕ, b_i)) and List Edge-Coloring (EC(ϕ)) problems to answer to the Feasible-VC(w), Feasible-EC(w, b) and Feasible-EC(w) problems, respectively.

Clearly, the (bounded) list vertex and edge coloring problems generalize the (bounded) vertex and edge coloring problems. List coloring problems have been studied extensively in the literature, for many classes of underlying graphs. In the next theorem we summarize some of these results which we shall use in this thesis.

Theorem 1.

(i) Both VC(ϕ, b_i) *and* EC(ϕ, b_i) *problems are polynomial on trees if the number, k, of colors is fixed* [21, 36].

(ii) The VC(ϕ, b_i) *problem is polynomial on general graphs if* $k = 2$ [36].

(iii) The VC(ϕ, b_i) *problem is NP-complete even for chains,* $|\phi(u)| \leq 2$*, for all* $u \in V$*, and* $b_i \leq 5$*,* $1 \leq i \leq k$ [23].

By exhaustively searching for the weights of an optimal solution of k colors for a max-coloring problem, and answering to the obtained feasible coloring problem through the corresponding list coloring problem, we get the following theorem.

Theorem 2.

(i) For a fixed number of colors k, both VC(w, b) *and* EC(w, b) *problems, and hence* VC(w) *and* EC(w) *problems, are polynomial on trees.*

(ii) For two colors, the VC(w, b) *problem is polynomial on general graphs.*

Proof. For (i), we consider all $O(|V|^k)$ (resp. $O(|E|^k)$) combinations of k color weights and for each combination, $w_1, w_2, \ldots, w_k$, we have to answer to the Feasible-VC(w, b) (resp. Feasible-EC(w, b)) problem. This can be done using the relation to the VC(ϕ, b_i) (resp. EC(ϕ, b_i)) problem described above and the results of Theorem 1(i). An optimal solution to the VC(w, b) (resp. EC(w, b)) problem corresponds to the combination where a feasible solution exists and the total weight W is minimized.

In a similar way, we can prove (ii), using Theorem 1(ii). □

Set cover and fixed cardinality bounds

The VC(w, b) and EC(w, b) problems are also related to the well known set cover problem, where we are given a universe U of elements, and a collection, $\mathcal{S} = \{S_1, S_2, \ldots, S_m\}$, of subsets of U, each one of a positive cost c_i, $1 \leq i \leq m$, and we ask for a minimum cost subset of $\mathcal{S}$ that covers all elements of U.

For an instance of the VC(w, b) problem on a graph $G = (V, E)$, let $U = V$ and consider $\mathcal{S}$ consisting of all the subsets of V, but those containing adjacent vertices, of cardinalities $j = 1, 2, \ldots, b$; for each such subset $S_i \in \mathcal{S}$ set $c_i = \max\{w(u)|u \in S_i\}$. Clearly, a solution to the set cover problem constructed corresponds to a solution of the VC(w, b) problem and a quite analogous transformation applies for the EC(w, b) problem. The cardinality of $\mathcal{S}$ is $O(b \cdot |V|^b)$, and as an H_b-approximation algorithm is known for the set cover problem [14], the next theorem follows.

Theorem 3. *For a fixed bound b, there is an H_b-approximation algorithm for both* VC(w, b) *and* EC(w, b) *problems on general graphs.*

Max-coloring vs Coloring problems

In [24], a general framework has been presented, which allows us to convert any ρ-approximation algorithm for the classical vertex coloring problem into an $e \cdot \rho$-approximation one for the VC(w) problem, on hereditary classes of graphs. The main idea of this framework is to select, in a random way, two parameters in order to round down the weight of each vertex. Hence, we can consider the input graph partitioned into subgraphs, where the vertices in each subgraph have the same weight. For each one of these subgraphs a ρ-approximation solution for the classical VC problem is obtained, and by concatenating the solutions found for all subgraphs we get a solution for the whole graph. Finally, this procedure is derandomized by choosing appropriate values for the two random parameters and a deterministic approximation algorithm is obtained.

This framework is used in [24] to derive an e-approximation algorithm for the VC(w, b) problem on perfect graphs. It is easy to see that it can be also applied for conversions from the VC(b), EC and EC(b) problems to the VC(w, b), EC(w) and EC(w, b) problems, respectively. For example, an e-approximation algorithm for the EC(w, b) problem on bipartite graphs is obtained using such a conversion, as the EC(b) problem is polynomial for bipartite graphs. However, this framework does not give an improvement to the approximation ratio of any other (bounded) max-coloring problem on the classes of graphs we study in this thesis, because a better ratio either is known or is presented in the next chapters.

Theorem 4. *There is an e-approximation algorithm for the* EC(w, b) *problem on bipartite graphs.*

Arbitrary b vs $b = 2$

Another approximation result for both VC(w, b) and EC(w, b) problems on general graphs can be obtained by relating the values OPT_b and OPT_2 of the optimal solution when the bound is b and 2, respectively. Let W be the weight of solution obtained by splitting each color in OPT_b into $\lceil \frac{b}{2} \rceil$ colors, all of cardinality at most 2. Obviously, $W \leq \lceil \frac{b}{2} \rceil OPT_b$. Moreover, OPT_2 is the optimal solution when all colors have cardinality at most 2, and hence $OPT_2 \leq W$. Thus, we have $OPT_2 \leq \lceil \frac{b}{2} \rceil OPT_b$, and since both VC$(w, b)$ and EC(w, b) problems are polynomial for general graphs and $b = 2$ [10], the following theorem follows.

Theorem 5. *There is a $\lceil \frac{b}{2} \rceil$-approximation algorithm for both* VC(w, b) *and* EC(w, b) *problems on general graphs.*

3.2 Notation

In the following, we consider the coloring problems defined in Chapter 1 on a graph $G = (V, E)$, where $|V| = n$ and $|E| = m$. For the (bounded) max-vertex-coloring (resp. (bounded) max-edge-coloring) problem, a positive integer weight $w(u)$ (resp. $w(e)$) is associated with each vertex $u \in V$ (resp. edge $e \in E$). For the bounded vertex/edge-coloring problems, we are also given a cardinality bound, b, on the number of vertices/edges allowed to appear in each color.

We denote by $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$ a proper vertex- (resp. edge-) coloring of G of weight $W = \sum_{i=1}^{k} w_i$, where $w_i = \max\{w(u) | u \in C_i\}$ (resp. $w_i = \max\{w(e) | e \in C_i\}$), $1 \leq i \leq k$. By $\mathcal{C}^* = \{C_1^*, C_2^*, \ldots, C_{k^*}^*\}$ we denote an optimal solution of weight $OPT = \sum_{i=1}^{k^*} w_i^*$, where w_i^*, $1 \leq i \leq k^*$, is the weight of the i-th color class.

By $d_G(u)$ (or simply $d(u)$) we denote the degree of vertex $u \in V$ and by $\Delta(G)$ (or simply Δ) the maximum degree of the graph G. We define the degree of each edge $e = (u, v) \in E$ as $d(u, v) = d(u) + d(v)$, while $\Delta'(G)$ (or simply Δ') denotes the maximum edge degree. For a subset of edges of G, $E' \subseteq E$, we denote by $G[E']$ the subgraph of G induced by the edges in E'.

The following ordering and partition of the elements of a set accordingly to their weights will be used to present and analyze most of our algorithms. Given a set S and a positive integer weight $w(s)$ for every element $s \in S$, we denote by $\langle S \rangle = \langle s_1, s_2, \ldots, s_{|S|} \rangle$ an ordering of S such that $w(s_1) \geq w(s_2) \geq \ldots \geq w(s_{|S|})$. For such an ordering of S and a positive integer b, let $k_S = \left\lceil \frac{|S|}{b} \right\rceil$. We define the *ordered b-partition* of S, denoted by $\mathcal{P}_S = \{S_1, S_2, \ldots, S_{k_S}\}$, to be the partition of S into k_S subsets, such that $S_i = \{s_j, s_{j+1}, \ldots, s_{\min\{j+b-1, |S|\}}\}$, $i = 1, 2, \ldots, k_S$, $j = (i-1)b + 1$. In other words, S_1 contains the b heaviest elements of S, S_2 contains the next b heaviest elements of S and so on; clearly, S_{k_S} contains the $|S|$ mod b lightest elements of S.

Chapter 4

Max-Edge-Coloring on general and bipartite graphs

In this chapter we present approximation results for the EC(w) problem on general and, mainly, on bipartite graphs. We, first, slightly improve the ratio of the known 2-approximation algorithm proposed by Kesselman and Kogan in [46] (Algorithm KK). Our analysis of this algorithm is based on lower and upper bounds on the number of colors of any reasonable solution to the EC(w) problem. Next, we give a simple algorithm that returns the best among two solutions: the solution found by Algorithm KK and the one obtained by an edge-coloring of the input graph. The ratio of this simple algorithm already beats the known ratios for general and bipartite graphs.

Next, we explore an idea used in [20, 25] to derive approximation ratios less than 2 for the EC(w) problem on bipartite graphs of $\Delta \leq 7$. The same ideas has been also used in [20, 51] to derive an 8/7-approximation algorithm for the VC(w) problem for bipartite graphs. In general, we find a number of solutions for a bipartite graph G by concatenating partial solutions for disjoint edge induced subgraphs of G and we select the best among them. Using this idea we present a series of four algorithms of different approximation ratios for the EC(w) problem on bipartite graphs. The approximation ratios of our three first algorithms depend on the maximum degree of the input graph. The last of our algorithms achieves an 1.74 approximation ratio for this problem and it is the first one that improves the known ratio of 2.

Finally, in Section 4.4 we prove that the EC(w) problem is NP-complete even on bi-valued complete graphs. Moreover, we present an asymptotic 4/3-approximation algorithm for general bi-valued graphs.

4.1 Preliminaries

The EC(w) problem is polynomial for graphs of maximum degree $\Delta = 2$. This result follows from the same variant of the VC(w) problem. In fact, an $O(|V|^2)$ algorithm for the VC(w) problem on chains has been presented in [25], which can be easily adapted to graphs of maximum degree $\Delta = 2$ (that are collections of chains

and cycles). If G is a graph of maximum degree $\Delta(G) = 2$, then its line graph $L(G)$ is also a graph with $\Delta(L(G)) = 2$ and the following theorem holds.

Theorem 6. *An optimal solution to the* EC(w) *problem for graphs of maximum degree* $\Delta = 2$ *can be found in* $O(|E|^2)$ *time.*

To bound the number of colors in any solution to the EC(w) problem we can restrict only on a specific subset of them. We shall call a solution $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$ to the EC(w) problem *nice* if: (i) $w_1 \geq w_2 \geq \ldots \geq w_k$, and (ii) each color C_i is maximal in the subgraph $G[\bigcup_{j=i}^{k} C_j]$. Due to the next proposition we consider, w.l.o.g., any, suboptimal or optimal, solution to the EC(w) problem to be a nice one.

Proposition 2. *Any solution to the* EC(w) *problem can be transformed into a nice one, without increasing its total weight. For the number of colors,* k*, in such a solution it holds that* $\Delta \leq k \leq \Delta' - 1 \leq 2\Delta - 1$.

Proof. Obviously, any solution to the EC(w) problem consists of at least Δ colors, since there is at least one vertex with exactly Δ adjacent edges, and these Δ edges belong to different colors.

Assume that an optimal solution consists of Δ' or more colors. Consider those colors sorted in non-increasing order of their weights. Each edge of G has at most $\Delta' - 2$ neighbor edges. So, for each edge e in any color C_i, $i \geq \Delta'$, there is at least one color, C_j, $j < \Delta'$, such that edge e can be moved to C_j without increasing C_j's weight.

The last part of the inequality follows directly by the definition of Δ'. □

The most interesting and general result for the EC(w) problem is due to Kesselman and Kogan [46] who proposed the following greedy algorithm:

Algorithm KK

1: Let $\langle E \rangle = \langle e_1, e_2, \ldots, e_m \rangle$;
2: **for** $i = 1$ to m **do**
3: Insert e_i into the first color not containing other edges adjacent to e_i;
4: **end for**

In [46], it has been shown that Algorithm KK is a 2-approximation one and an example has been presented yielding an approximation ratio of $2 - \frac{1}{\Delta}$. By a slightly tighter analysis using Proposition 2 we prove here the next lemma.

Lemma 1. Algorithm KK *achieves an approximation ratio of* $\min\{2 - \frac{w_1^*}{OPT}, 2 - \frac{1}{\Delta}\}$ *for the* EC(w) *problem.*

Proof. The solution, $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$, that Algorithm KK returns is, by its construction, a nice one. Let e be the first edge that the algorithm inserts into color C_i; then it holds that $w_i = w(e)$. Let E_i be the set of edges preceding e in $\langle E \rangle$ and edge e itself, i.e., $E_i = \{e_1, e_2, \ldots, e_{i-1}\}$, and Δ_i be the maximum degree

of the subgraph $G[E_i]$. The optimal solution for the EC(w) problem on the graph $G[E_i]$ contains $i^* \geq \Delta_i$ colors each one of weight at least w_i, that is $w_i \leq w^*_{i^*}$. By Proposition 2, the colors constructed by ALGORITHM KK for the graph $G[E_i]$ are $i \leq 2\Delta_i - 1 \leq 2i^* - 1$, that is $i^* \geq \lceil \frac{i+1}{2} \rceil$. Hence, $w_i \leq w^*_{i^*} \leq w^*_{\lceil \frac{i+1}{2} \rceil}$.

Summing up the above bounds for all w_i's, $1 \leq i \leq k \leq 2\Delta - 1$, we obtain $W \leq \sum_{i=1}^{2\Delta-1} w_i \leq w^*_1 + 2\left(\sum_{i=2}^{\Delta} w^*_i\right) = 2\left(\sum_{i=1}^{\Delta} w^*_i\right) - w^*_1$. As $k^* \geq \Delta$, it follows that $\sum_{i=1}^{\Delta} w^*_i \leq OPT$. Therefore, $\frac{W}{OPT} \leq 2 - \frac{w^*_1}{OPT}$ and also $\frac{W}{OPT} \leq \frac{2\sum_{i=1}^{\Delta} w^*_i - w^*_1}{\sum_{i=1}^{\Delta} w^*_i} \leq 2 - \frac{w^*_1}{\sum_{i=1}^{\Delta} w^*_i} \leq 2 - \frac{w^*_1}{\Delta \cdot w^*_1} = 2 - \frac{1}{\Delta}$. □

It is well known that a $(\Delta+1)$-coloring of a general graph and a Δ-coloring of a bipartite graph can be found in polynomial time. In fact, a $(\Delta+1)$-coloring of a general graph can be found in $O\left(\min\{|V|\Delta \log |V|, |E|\sqrt{|V| \log |V|}\}\right)$ time [29], while a Δ-coloring of a bipartite graph in $O(|E| \log \Delta)$ time [16]. Such a coloring yields a feasible, but in general not optimal, solution for the EC(w) problem.

Intuitively, a solution obtained this way will be close to an optimal one when the edge weights are close to each other, while ALGORITHM KK performs better in the opposite case. Next theorem follows by selecting the best among the two solutions found by ALGORITHM KK and a $(\Delta+1)-$ or Δ-coloring of the input graph.

Theorem 7. *There is an approximation algorithm for the* EC(w) *problem of ratio* $2 - \frac{2}{\Delta+2}$ *for general graphs and* $2 - \frac{2}{\Delta+1}$ *for bipartite graphs.*

Proof. By Lemma 1, a solution found by ALGORITHM KK is of weight $W \leq 2OPT - w^*_1$. Any Δ-coloring of a bipartite graph yields a solution for the EC(w) problem of weight $W \leq \Delta \cdot w^*_1$, since w^*_1 equals to the weight of the heaviest edge of the graph, and hence to the weight of the heaviest color of any solution created for this graph. Multiplying both sides of the second inequality with $1/\Delta$ and adding this to the first one we obtain: $\left(1 + \frac{1}{\Delta}\right) W \leq 2OPT$, that is $W \leq \left(2 - \frac{2}{\Delta+1}\right) OPT$.

For general graphs we simply consider a $(\Delta+1)$-coloring, instead of a Δ-coloring, of the input graph. □

For the tightness of our analysis for bipartite graphs, consider the instance of the EC(w) problem shown in Figure 4.1(a). The weight of an optimal solution to this instance is $2+\epsilon$ (Figure 4.1(b)), the weight of the solution of ALGORITHM KK is 3 (Figure 4.1(c)), and the weight of a solution found by a Δ-coloring of the input graph (Figure 4.1(d)) is also 3. By selecting either solution a ratio of $\frac{3}{2+\epsilon} \simeq \frac{3}{2} = 2 - \frac{2}{\Delta+1}$ is attained.

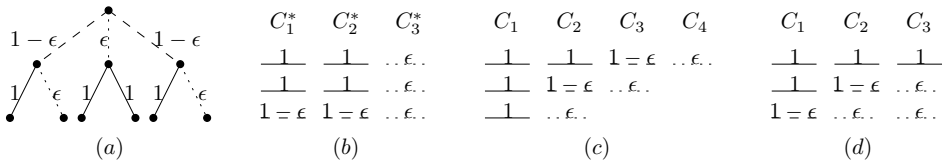

Figure 4.1: A tight example for the $\left(2-\frac{2}{\Delta+1}\right)$-approximation ratio of Theorem 7 for bipartite graphs ($\Delta = 3$, $\epsilon << 1$).

Figure 4.2(a) shows an analogous example for the approximation ratio of Theorem 7 for general graphs. The solution created by ALGORITHM KK (Figure 4.2(c)) has weight $4-3\epsilon$, while the solution obtained by a $(\Delta+1)$-coloring (Figure 4.2(d)) has weight 4. By selecting the first solution a ratio of $\dfrac{4-3\epsilon}{\frac{5}{2}} \simeq \dfrac{8}{5} = 2 - \dfrac{2}{\Delta+2}$ is attained, since the optimal solution (Figure 4.2(b)) has weight 5/2.

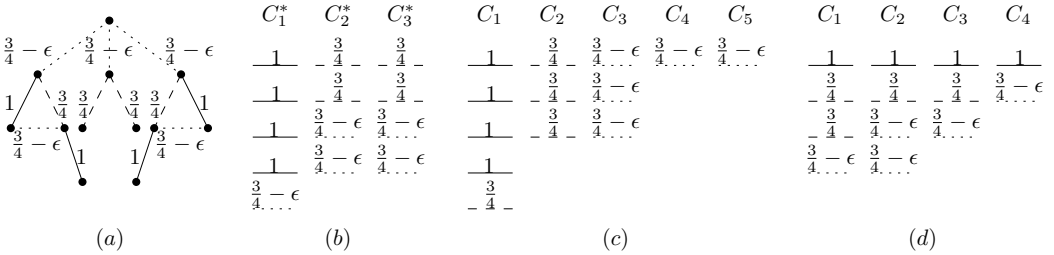

Figure 4.2: A tight example for the $\left(2-\frac{2}{\Delta+2}\right)$-approximation ratio of Theorem 7 for general graphs ($\Delta = 3$, $\epsilon << 1$).

The ratios of Theorem 7 are better than $2-\dfrac{1}{\Delta}$ for any $\Delta \geq 3$. More interestingly, the ratios for bipartite graphs are better than the $(2\Delta-1)/3$ approximation ratio proposed in [22], for any Δ, as well as than the ratios of the algorithm presented in [25], for $\Delta \geq 4$.

4.2 $f(\Delta)$-approximation algorithms for bipartite graphs

The known approximation algorithms [20, 25] of ratios less than 2 for the EC(w) problem on a bipartite graph $G = (V, E)$ are based on the following general idea: Consider an ordering $\langle E \rangle = \langle e_1, e_2, \ldots, e_m \rangle$ of the edges of G, and let $E_{p,q} = \{e_p, e_{p+1}, \ldots, e_q\}$. Repeatedly, partition the graph G into three edge induced subgraphs: the graph $G[E_{1,p}]$ induced by the p heaviest edges of G, the graph $G[E_{p+1,q}]$, induced by the next $q-p$ edges of G, and the graph $G[E_{q+1,m}]$, induced by the $m-q$ lightest edges of G. Find a solution for the whole graph G by considering the EC(w) problem on these three subgraphs and return the best among the solutions

found. Depending on how the problem is handled for each subgraph and the analysis followed, this general idea leads to different algorithms and approximation ratios. Notice that the same approach is employed by the 8/7-approximation algorithm for the VC(w) problem on bipartite graphs [20, 51].

In this section we further explore this idea, and we present a series of three $f(\Delta)$-approximation algorithms, each one improving the ratios of the previous ones. Our first algorithm gives approximation ratios better than the previous known for bipartite graphs of maximum degree $4 \leq \Delta \leq 12$, but its ratio becomes greater than 2 for $\Delta \geq 13$. The approximation ratios of our second and third algorithms are smaller than 2 for any Δ. Our third algorithm achieves the best ratios for bipartite graphs of maximum degree $\Delta \geq 7$. However, both these algorithms give ratios that tend asymptotically to 2 as Δ increases.

To describe our algorithms, let us introduce some additional notation. We denote by (p, q), $0 \leq p < q \leq m$, a partition of G into subgraphs $G[E_{1,p}]$, $G[E_{p+1,q}]$ and $G[E_{q+1,m}]$; by convention, we define $E_{1,0} = \emptyset$, $E_{m+1,m} = \emptyset$ and $E_{0,q} = E_{1,q}$. By $\Delta_{p,q}$ we denote the maximum degree of the subgraph $G[E_{p,q}]$, and by $d_{p,q}(u)$ the degree of $u \in V$ in $G[E_{p,q}]$. We denote by p_δ the maximum index such that $\Delta_{1,p_\delta} = \delta$. It is clear that $p_1 < p_2 < \ldots < p_\Delta = m$.

Finally, henceforth in this section, the following proposition will be useful.

Proposition 3. *Given a graph $G = (V, E)$ and a subset $A \subseteq V$, we can determine if there is a matching M in G saturating all vertices in A in $O(|V|^{2.5})$ time.*

Proof. Consider the graph $G' = (X, Y)$ constructed by adding to G an additional vertex, if $|V|$ is odd, and all the missing edges between the vertices $X - A$ (i.e., the vertices $X - A$ induce a clique in G'). If there exists a perfect matching in G', then there exists a matching in G saturating all vertices in A, since no edges adjacent to A have been added in G'.

Conversely, if there exists a matching M in G saturating all vertices in A, then there exists a perfect matching in G', consisting of the edges of M plus the edges of a perfect matching in the complete subgraph of G' induced by its vertices that are not saturated by M.

Therefore, in order to determine if there exists a matching M in G it is enough to check if there exists a perfect matching in G'. It is well known that this last check can be done in $O(|V|^{2.5})$ time [50]. □

Bipartite graphs of small maximum degree

In this section, we present a first approximation algorithm for the EC(w) problem on bipartite graphs, exploiting the idea of splitting the input bipartite graph into edge induced subgraphs. In fact, our algorithm generalizes the 7/6-approximation algorithm for bipartite graphs of $\Delta = 3$, proposed in [20], such that: (a) it remains polynomial for general bipartite graphs, and (b) it achieves a substantial improvement of the best known approximation ratios for the EC(w) problem on bipartite graphs of maximum degree $4 \leq \Delta \leq 12$.

In general, for each partition (p, q), $p = 1, 2, \ldots, p_{\Delta-1}$, $q = p+1, \ldots, m$, our algorithm checks the existence of two different specific sets of edges in graph $G[E_{p+1,q}]$ and for each one of them, if there exist, it computes a solution to the EC(w) problem on graph G. The algorithm returns the best among all the solutions found.

Algorithm Bipartite-1(G)

1: **for** $p = 1$ to $p_{\Delta-1}$ **do**
2: **for** $q = p + 1$ to m **do**
3: **if** there is a matching M in $G[E_{p+1,q}]$ saturating all vertices of $G[E_{1,q}]$ with degree Δ **then**
4: Create a solution for $G[E_{1,q}]$ by concatenating a $(\Delta-1)$-coloring solution for $G[E_{1,q} \setminus M]$ and the matching M;
5: Complete greedily this solution with the edges in $E_{q+1,m}$;
6: **end if**
7: **if** $p \leq p_2$ **then**
8: Find an optimal solution $\mathcal{C}_{1,p}$ for $G[E_{1,p}]$ by Theorem 6;
9: **else**
10: Find a solution $\mathcal{C}_{1,p}$ for $G[E_{1,p}]$ by Algorithm Bipartite-1($G[E_{1,p}]$);
11: **end if**
12: **if** there is a set of edges E' in $G[E_{p+1,q}]$ saturating any vertex of $G[E_{p+1,q}]$ with degree Δ and E' fits in $\mathcal{C}_{1,p}$ **then**
13: Find a $(\Delta-1)$-coloring solution $\mathcal{C}_{p+1,q}$ for $G[E_{p+1,q} \setminus E']$;
14: Concatenate $\mathcal{C}_{1,p}$ and $\mathcal{C}_{p+1,q}$ and complete greedily this solution with the edges of $G[E_{q+1,m}]$;
15: **end if**
16: **end for**
17: **end for**
18: Return the best among the solutions found in Lines 5 and 14;

In Line 3 the algorithm checks the existence of a matching M in $G[E_{p+1,q}]$ saturating all vertices of degree Δ in $G[E_{1,q}]$. It holds that $\Delta_{1,p} \leq \Delta - 1$, since $1 \leq p \leq p_{\Delta-1}$, and therefore the vertices of degree Δ in subgraphs $G[E_{1,q}]$ and $G[E_{p+1,q}]$ are the same. Hence, the existence of M can be checked by applying Proposition 3 on the graph $G[E_{p+1,q}]$ with A being the set of vertices of degree Δ in $G[E_{1,q}]$.

In Line 12 the algorithm checks the existence of a set of edges E' in $G[E_{p+1,q}]$ saturating all vertices of degree Δ in $G[E_{p+1,q}]$ and, moreover, fitting the solution $\mathcal{C}_{1,p}$. Consider the subgraph H of $G[E_{p+1,q}]$ induced by its vertices of degree $d_{1,p}(u) \leq \Delta_{1,p} - 1$. Note that, by construction, each edge in H fits in a color of the solution $\mathcal{C}_{1,p}$. Let A be the subset of vertices of H of degree $d_{p+1,q}(u) = \Delta$, i.e. the set of vertices which we want to saturate, and B the subset of vertices in A of degree $d_H(u) = 1$. For each vertex $u \in B$ we can clearly insert the single edge (u, v) in E'. Let H' be the subgraph of H induced by its vertices but those in B and $A' \subseteq A$ be the subset of vertices of A that are not saturated by the edges already in E'. It is now enough to find a matching on H' that saturates each vertex in A', using

Proposition 3. Adding the edges of this matching in E' we get a set that saturates each vertex of degree Δ in graph $G[E_{p+1,q}]$.

In Lines 5 and 14 the algorithm completes a partial solution by examining the remaining lightest edges one by one and assigning them to the first color they fit in. If such a color does not exist, then a new one is created. As both partial solutions consist of at most $2\Delta - 1$ colors, the complete solutions obtained will consist also of at most $2\Delta - 1$ colors, according to the arguments in the proof of Proposition 2.

The following lemma provides bounds to the weight W of the solution obtained by Algorithm Bipartite-1. We denote by ϱ_Δ the approximation ratio of our algorithm for a graph of maximum degree Δ. By definition, $\varrho_1 = \varrho_2 = 1$, since, by Theorem 6, the EC(w) problem for graphs of maximum degree 1 or 2 can be solved in polynomial time. Recall also that we consider the colors of an optimal solution in non-increasing order with respect to their weights, i.e., $w_1^* \geq w_2^* \geq \cdots \geq w_{k^*}^*$.

Lemma 2. Algorithm Bipartite-1 *returns a solution of weight:*

$$W \leq \min\Big\{(\Delta-1)\cdot w_1^* + w_\Delta^* + (\Delta-1)\cdot w_{\Delta+1}^*, \min_{1\leq\delta\leq\Delta-1}\left\{\varrho_\delta \cdot \sum_{i=1}^{\delta} w_i^* + (\Delta-1)\cdot w_{\delta+1}^* + (\Delta-\delta)\cdot w_{\Delta+\delta}^*\right\}\Big\}$$

Proof. For the first term in the righthand side of the lemma's inequality, we consider the solution obtained in Lines 3 to 6 of Algorithm Bipartite-1(G) in the iteration (p, q) where $w(e_{p+1}) = w_\Delta^*$ and $w(e_{q+1}) = w_{\Delta+1}^*$ (note that if $q = m$, then $k^* = \Delta$ and $w_i^* = 0$, for each $i \geq \Delta + 1$). In this iteration the matching M exists, since in the optimal solution the edges in $E_{1,q}$ belong to at most Δ colors and the Δ-th color contains edges of weight at most $w(e_{p+1})$. The weight of M is at most w_Δ^*, while the weight of the solution found for $G[E_{1,q} \setminus M]$ is bounded by $(\Delta - 1) \cdot w_1^*$. The greedy step in Line 5 creates at most $\Delta - 1$ colors, each one of weight at most $w_{\Delta+1}^*$. Therefore, $W \leq (\Delta - 1) \cdot w_1^* + w_\Delta^* + (\Delta - 1) \cdot w_{\Delta+1}^*$.

For the second term in the righthand side of the lemma's inequality, consider the solutions obtained in Lines 7 to 15 of Algorithm Bipartite-1(G) for $\Delta - 1$ different iterations (p, q). For $p \leq p_\delta$, consider the iteration q where $w(e_{p+1}) = w_{\delta+1}^*$ and $w(e_{q+1}) = w_{\Delta+\delta}^*$. In this iteration, the set of edges E' exists, since in the optimal solution the edges in $E_{p+1,q}$ belong to at most $\Delta - 1$ colors.

If $\delta = 1$ or 2, the algorithm creates an optimal solution for $G[E_{1,p}]$ using Theorem 6, since $\Delta_{1,p} \leq 2$.

If $q \geq 3$, the algorithm creates an approximation solution for $G[E_{1,p}]$.

In both cases, the edges in $E_{1,p}$ are a subset of the edges of the δ heaviest colors in the optimal solution. Thus, it holds that $\mathcal{C}_{1,p} \leq \varrho_\delta \cdot \sum_{i=1}^{\delta} w_i^*$, where $\varrho_1 = \varrho_2 = 1$.

The weight of the solution found for $G[E_{p+1,q} \setminus E']$ is bounded by $(\Delta - 1) \cdot w_{\delta+1}^*$, since $\Delta\left(G[E_{p+1,q} \setminus E']\right) \leq \Delta - 1$ and for each edge $e \in E_{p+1,q} \setminus E'$ it holds that $w(e) \leq w_{\delta+1}^*$.

The greedy step in Line 14 creates at most $\Delta - \delta$ colors, each one of weight at most $w_{\Delta+\delta}^*$.

Summing up the weights of these three partial solutions the lemma follows. □

Algorithm Bipartite-1(G) performs $O(|E|^2)$ iterations. In Line 10 of (some of) those iterations the algorithm is called recursively for the graph $G[E_{1,p}]$. In this call, for each combination of p', q', $1 \leq p' < q' \leq p$, the graph $G_{1,p}$ is partitioned into the three edge induced subgraphs $G_{1,p'}$, $G_{p'+1,q'}$ and $G_{q'+1,p}$.

The solution for $G[E_{1,p'}]$ has been already computed in a previous iteration of Algorithm Bipartite-1(G) in which $p = p'$. By storing this solution we avoid its recursive recomputation. Thus, Algorithm Bipartite-1$(G[E_{1,p}])$ performs in total also $O(|E|^2)$ iterations. By Proposition 3, the matching M and the set E' can be computed in polynomial time, and this derives a polynomial overall complexity.

To obtain the ratio of Algorithm Bipartite-1 we consider the Δ bounds on W and we combine them in an optimal way. For instance, for $\Delta = 4$, Algorithm Bipartite-1 returns a solution of weight W, for which holds that:

$$
\begin{aligned}
W &\leq 3 \cdot w_1^* + w_4^* + 3 \cdot w_5^* \\
W &\leq w_1^* + 3 \cdot w_2^* + 3 \cdot w_5^* \\
W &\leq w_1^* + w_2^* + 3 \cdot w_3^* + 2 \cdot w_6^* \\
W &\leq 7/6 \cdot (w_1^* + w_2^* + w_3^*) + 3 \cdot w_4^* + w_7^*
\end{aligned}
$$

Multiplying these inequalities by 44/458, 66/458, 99/458 and 138/458, respectively, and adding them we obtain a ratio $\varrho_4 = 458/347 \simeq 1.32$ for the EC(w) problem on bipartite graphs with maximum degree $\Delta = 4$, that dominates the 1.61 ratio by [25]. In the same way, we can compute the ratio ϱ_Δ for different values of Δ.

Table 4.1 summarizes the approximation ratios achieved by our algorithm, as well as the previous best known ratios for $\Delta = 3, 4, \ldots, 12$.

Δ	Best known ratio	Our ratio
3	1.17 [20]	1.17
4	1.61 [25]	1.32
5	1.75 [25]	1.45
6	1.86 [25]	1.56
7	1.95 [25]	1.65
8	2 [46]	1.74
9	2 [46]	1.81
10	2 [46]	1.87
11	2 [46]	1.93
12	2 [46]	1.98

Table 4.1: Summary of the approximation ratios achieved by Algorithm Bipartite-1 vs. previous best known ratios for $\Delta = 3, \ldots, 12$.

Algorithm Bipartite-1 achieves better results than the approximation ratio $(2\Delta-1)/3$ proposed in [22], for any Δ, as well as than the 2-approximation algorithm given in [46], for bipartite graphs with $\Delta \leq 12$. Furthermore, for bipartite graphs of $4 \leq \Delta \leq 7$ our algorithm improves the best known results given in [25]. For

bipartite graphs of maximum degree $\Delta = 3$, Algorithm Bipartite-1 reduces to the 7/6-approximation algorithm proposed in [20].

Bipartite graphs of arbitrary maximum degree

In this section we present a new algorithm for the EC(w) problem on bipartite graphs. It also produces $O(|E|^2)$ different solutions by splitting the input graph into edge induced subgraphs and chooses the best of them. The new algorithm beats the previous ratios for bipartite graphs for any $\Delta \geq 9$ and it is the first one of this kind yielding approximation ratios that tends asymptotically to 2 as Δ increases.

In general, for each $p = 1, 2, \ldots, p_2$, our algorithm examines a partition of the graph G into two edge induced subgraphs: $G[E_{1,p}]$ of $\Delta_{1,p} \leq 2$ and $G[E_{p+1,m}]$. For each one of these partitions, the algorithm computes a solution to the EC(w) problem on graph G.

Moreover, for each partition (p, q), $p = 1, 2, \ldots, p_2$, $q = p + 1, \ldots, m$, our algorithm checks the existence of a set of edges in graph $G[E_{p+1,q}]$, just the same to the set required to Algorithm Bipartite-1. If such a set of edges exists the algorithm computes a solution to the EC(w) problem on graph G.

The algorithm computes one more solution by finding a Δ-coloring of the original graph G and returns the best among all the solutions found.

Algorithm Bipartite-2

1: Find a Δ-coloring solution $\mathcal{C}^0_{1,m}$ for G;
2: **for** $p = 1$ to p_2 **do**
3: Find an optimal solution $\mathcal{C}^1_{1,p}$ for $G[E_{1,p}]$;
4: Find a Δ-coloring solution $\mathcal{C}^1_{p+1,m}$ for $G[E_{p+1,m}]$;
5: Concatenate $\mathcal{C}^1_{1,p}$ and $\mathcal{C}^1_{p+1,m}$;
6: **for** $q = p + 1$ to m **do**
7: Find an optimal solution $\mathcal{C}^2_{1,p}$ for $G[E_{1,p}]$;
8: **if** there is a set of edges E' in $G[E_{p+1,q}]$ saturating any vertex of $G[E_{p+1,q}]$ with degree $\Delta_{1,q}$ and E' fits in $\mathcal{C}^2_{1,p}$ **then**
9: Find a solution $\mathcal{C}^2_{p+1,q}$ by a $(\Delta_{1,q} - 1)$-coloring of $G[E_{p+1,q} \setminus E']$;
10: Find a solution $\mathcal{C}^2_{q+1,m}$ by a Δ-coloring of $G[E_{q+1,m}]$;
11: Concatenate $\mathcal{C}^2_{1,p}$, $\mathcal{C}^2_{p+1,q}$ and $\mathcal{C}^2_{q+1,m}$;
12: **end if**
13: **end for**
14: **end for**
15: Return the best among the solutions found in Lines 1, 5 and 11;

Note that the main difference between the new algorithm and earlier ones, like the algorithm proposed in [25] as well as Algorithm Bipartite-1, is that it keeps always the maximum degree $\Delta_{1,p}$ of the subgraph $G[E_{1,p}]$ at most two and thus it always computes an optimal solution for $G[E_{1,p}]$. Recall that in the earlier algorithm, $\Delta_{1,p}$ is allowed to take values form 1 to Δ, and thus there are cases where only an

approximate solution can be found for $G[E_{1,p}]$. Due to this fact, the ratio of those algorithms can increase to $O(\Delta)$.

The existence of the set of edges E' in Line 8 of the algorithm can be determined in polynomial time in the same way as the same set in Algorithm Bipartite-1.

Theorem 8. Algorithm Bipartite-2 *is a* $(\frac{2\Delta^3}{\Delta^3+\Delta^2+\Delta-1})$*-approximation one for the* EC(w) *problem on bipartite graphs.*

Proof. The solution obtained by a Δ-coloring of the input graph computed in Line 1 of the algorithm is of weight $W \leq \mathcal{C}^0_{1,m} \leq \Delta \cdot w^*_1$, since w^*_1 equals to the heaviest edge of the graph.

In Lines 3–5, consider the solutions obtained in the iterations where $w(e_{p+1}) = w^*_z$, for $z = 2, 3$. In both cases it holds that $\Delta_{1,p} \leq 2$. An optimal solution is computed for $G_{1,p}$ of weight $\mathcal{C}^1_{1,p} \leq \sum_{i=1}^{z-1} w^*_i$, since the edges in $E_{1,p}$ are a subset of the edges that appear in the $z-1$ heaviest colors of the optimal solution. Moreover, a Δ-coloring is built for $G[E_{p+1,m}]$ of weight $\mathcal{C}^1_{p+1,m} \leq \Delta \cdot w^*_z$, since e_{p+1} is the heaviest edge of this subgraph. Therefore,

$$W \leq \textstyle\sum_{i=1}^{z-1} w^*_i + \Delta \cdot w^*_z, \text{ for } z = 2, 3.$$

In Lines 7–12, consider the solutions obtained in the iterations (p, q) where $w(e_{p+1}) = w^*_3$ and $w(e_{q+1}) = w^*_z$, for $4 \leq z \leq \Delta$. In these iterations the set of edges E' exists, since in the optimal solution the edges in $E_{1,q}$ belong in at most $\Delta_{1,q} \leq z - 1$ colors. The edges of E' are lighter than the edges in $E_{1,p}$, and thus it is possible to add them in $\mathcal{C}^2_{1,p}$ without increasing its weight. Thus, using the same arguments as for the weight of $\mathcal{C}^1_{1,p}$, it holds that $\mathcal{C}^2_{1,p} \leq w^*_1 + w^*_2$. The heaviest edges in $G[E_{p+1,q} \setminus E']$ and $G[E_{q+1,m}]$ are equal to w^*_3 and w^*_z, respectively. Hence, we have that $\mathcal{C}^2_{p+1,q} \leq (\Delta_{1,q} - 1) \cdot w^*_3 \leq (z-2) \cdot w^*_3$ and $\mathcal{C}^2_{q+1,m} \leq \Delta \cdot w^*_z$. Therefore,

$$W \leq w^*_1 + w^*_2 + (z-2) \cdot w^*_3 + \Delta \cdot w^*_z, \text{ for } 4 \leq z \leq \Delta.$$

As the algorithm returns the best among the solutions found, we have Δ bounds on the weight W of this best solution, i.e.,

$$\begin{aligned}
&W \leq \Delta \cdot w^*_1 \\
&W \leq \textstyle\sum_{i=1}^{z-1} w^*_i + \Delta \cdot w^*_z, \text{ if } z = 2 \text{ or } 3, \text{ and} \\
&W \leq w^*_1 + w^*_2 + (z-2) \cdot w^*_3 + \Delta \cdot w^*_z, \text{ if } 4 \leq z \leq \Delta.
\end{aligned}$$

To derive our ratio we denote by $c_{jz}, 1 \leq z, j \leq \Delta$, the coefficient of the weight w^*_j in the z-th bound on W and we find the solution of the system of linear equations $\mathbf{C} \cdot \mathbf{x}^T = \mathbf{1}^T$, that is

$$x_z = \begin{cases} \dfrac{\Delta^2 - 1}{2\Delta^3}, & \text{if} \quad z = 1 \\ \dfrac{\Delta + 1}{2\Delta^2}, & \text{if} \quad z = 2 \\ \dfrac{5 - \Delta}{2\Delta}, & \text{if} \quad z = 3 \\ \dfrac{1}{\Delta}, & \text{if} \quad 4 \leq z \leq \Delta. \end{cases}$$

By multiplying both sides of the i-th bound on W by x_i and adding all of them we have $\sum_{z=1}^{\Delta} x_z \cdot W \leq w_1^* + w_2^* + \ldots + w_\Delta^* \leq OPT$. Hence,

$$\begin{aligned} \frac{W}{OPT} &\leq \frac{1}{\sum_{z=1}^{\Delta} x_z} \leq \frac{2\Delta^3}{(\Delta^2-1)+\Delta(\Delta+1)+\Delta^2(5-\Delta)+2\Delta^2(\Delta-3)} \\ &= \frac{2\Delta^3}{\Delta^3+\Delta^2+\Delta-1}. \end{aligned}$$

□

The complexity of Algorithm Bipartite-2 is dominated by the check in Line 8, which by Proposition 3 can be done in polynomial time. This check runs for $O(|E|^2)$ different combinations of weights.

Improvement for bipartite graphs of arbitrary maximum degree

In this section we further explore the limitations of the same idea of repeatedly partitioning the graph into three edge induced subgraphs of heaviest, medium and lightest edges. We present a new algorithm for the EC(w) problem on bipartite graphs, which improves all the previous ratios for $\Delta \geq 7$.

For a partition (p, q) of G, we call *critical matching* a matching $M \subseteq E_{p+1,q}$ which saturates all the vertices of $G[E_{1,q}]$ of degree $\Delta_{1,q}$. The proposed algorithm relies on the existence of such a critical matching M: a solution for the subgraph $G[E_{1,q}]$ is found by concatenating a $(\Delta_{1,q}-1)$-coloring solution for the subgraph $G[E_{1,q} \setminus M]$ and the matching M, if exists, and by a $\Delta_{1,q}$-coloring of the subgraph $G[E_{1,q}]$, otherwise. For each partition (p, q), the algorithm computes a solution for the input graph G by concatenating a solution for $G[E_{1,q}]$ and a Δ-coloring solution for $G[E_{q+1,m}]$. The algorithm computes also a Δ-coloring solution for the input graph and returns the best among them.

Algorithm Bipartite-3

1: Find a Δ-coloring solution for G;
2: Let $\langle E \rangle = \langle e_1, e_2, \ldots, e_m \rangle$;
3: **for** $p = 0$ to $m-1$ **do**
4: **for** $q = p+1$ to m **do**
5: Find, if any, a critical matching M in $G[E_{p+1,q}]$;
6: **if** M exists **then**
7: Find a $(\Delta_{1,q}-1)$-coloring solution for $G[E_{1,q} \setminus M]$;
8: **else**
9: Find a $\Delta_{1,q}$-coloring solution for $G[E_{1,q}]$;
10: **end if**
11: Find a Δ-coloring solution for $G[E_{q+1,m}]$;
12: Find a solution for G by concatenating the solutions found in Lines 6–10 and 11 and matching M, if exists;
13: **end for**
14: **end for**
15: Return the best among the solutions found in Lines 1 and 12;

To find, if any, the critical matching M in Line 5 of the algorithm, we use Proposition 3. In fact, we ask for a matching $M \subseteq E_{p+1,q}$ which saturates all the vertices of $G[E_{1,q}]$ of degree $\Delta_{1,q}$.

Theorem 9. Algorithm Bipartite-3 *achieves an approximation ratio of* $\frac{2(\Delta+1)^3}{\Delta^3+5\Delta^2+5\Delta+3-2(-1/\Delta)^\Delta}$ *for the* EC(w) *problem on bipartite graphs.*

Proof. The solution obtained by a Δ-coloring of the input graph computed in Line 1 of the algorithm is of weight $W_1 \leq \Delta \cdot w_1^*$.

Consider the partition (p, q) of G where $w(e_{p+1}) = w_{i-1}^*$ and $w(e_{q+1}) = w_i^*$, for $2 \leq i \leq \Delta$ (recall that $w_1^* \geq w_2^* \geq \ldots \geq w_{k^*}^*$ and $k^* \geq \Delta$). In such an iteration, all the edges in $E_{1,q}$ belong to $i-1 \geq \Delta_{1,q}$ colors of an optimal solution $\mathcal{C}^*$.

If $\Delta_{1,q} < i-1$, then an $(i-2)$-coloring of $G[E_{1,q}]$ yields a solution of weight at most $(i-2) \cdot w_1^*$ for this subgraph.

If $\Delta_{1,q} = i-1$ then a critical matching M exists. Indeed, in this case the $(i-1)$-th color of $\mathcal{C}^*$ always contains some edges from $E_{p+1,q}$, for otherwise all the edges in $E_{1,q}$ belong to $i-2$ colors of $\mathcal{C}^*$, a contradiction; these edges of $E_{p+1,q}$ could be a critical matching M for the partition (p, q). Thus, a $(i-2)$-coloring solution of $G[E_{1,q} \setminus M]$ and critical matching M yield a solution for the subgraph $G[E_{1,q}]$ of weight at most $(i-2) \cdot w_1^* + w_{i-1}^*$. Finally, a Δ-coloring solution for $G[E_{q+1,m}]$ is of cost at most $\Delta \cdot w_i^*$.

Hence, for such a partition (p, q) the algorithm finds a solution for the whole input graph of weight

$$W_i \leq (i-2) \cdot w_1^* + w_{i-1}^* + \Delta \cdot w_i^*, \ 2 \leq i \leq \Delta.$$

As the algorithm returns the best among the solutions found, we have Δ bounds on the weight W of this best solution, i.e.,

$$\begin{array}{l} W \leq \Delta \cdot w_1^*, \text{ if } i = 1, \text{ and} \\ W \leq (i-2) \cdot w_1^* + w_{i-1}^* + \Delta \cdot w_i^*, \text{ if } \ 2 \leq i \leq \Delta. \end{array}$$

To derive our ratio we perform as in the previous section, solving a system of linear equations, and we get the following multipliers:

$$x_i = \begin{cases} \dfrac{1}{\Delta}, & \text{if} \quad i = \Delta \\ \dfrac{1}{\Delta+1}\left(1 - \left(\dfrac{-1}{\Delta}\right)^{\Delta-i+1}\right), & \text{if} \quad \Delta - 1 \geq i \geq 2 \\ \dfrac{1}{\Delta} - \displaystyle\sum_{j=0}^{\Delta-3}\left(\frac{\Delta-(j+2)}{\Delta} x_{\Delta-j}\right) - \frac{1}{\Delta}x_2, & \text{if} \quad i = 1. \end{cases}$$

By multiplying both sides of the i-th bound on W by x_i and adding all of them we have $\displaystyle\sum_{i=1}^{\Delta} x_i \cdot W \leq w_1^* + w_2^* + \ldots + w_\Delta^* \leq OPT$. Hence, $\dfrac{W}{OPT} \leq \dfrac{1}{\sum_{i=1}^{\Delta} x_i}$, which after some algebra becomes

$$\frac{W}{OPT} \leq \frac{(\Delta+1)}{\frac{\Delta^3+3\Delta^2+\Delta-3}{2(\Delta^2-1)} - \frac{\left(\Delta^{2+(\Delta \bmod 2)}+(-1)^{\Delta}(\Delta-1)\right)}{(\Delta^2-1)\Delta^{\Delta}} - (\Delta-1)\sum_{i=1}^{\lfloor \Delta/2 \rfloor} \frac{2i}{\Delta^{2i}}}.$$

By differentiating both sides of the formula $\sum_{i=0}^{\lfloor \Delta/2 \rfloor} \left(\frac{1}{x^2}\right)^i = \frac{1-(x^{-2})^{\lfloor \Delta/2 \rfloor+1}}{1-x^{-2}}$ for the sum of geometric series we get

$$-\frac{1}{x}\sum_{i=1}^{\lfloor \Delta/2 \rfloor} \left(\frac{2i}{x^{2i}}\right) = \frac{-2x+2x^{-2\lfloor \Delta/2 \rfloor+1}+2\lfloor \Delta/2 \rfloor(x^2-1)x^{-2\lfloor \Delta/2 \rfloor-1}}{(x^2-1)^2}$$

and by using this last expression for $x = \Delta$ we finally get

$$\frac{W}{OPT} \leq \frac{2(\Delta+1)^3}{\Delta^3+5\Delta^2+5\Delta+3-2(-1/\Delta)^{\Delta}}. \qquad \square$$

Lines 5–12 of the algorithm are repeated $O(|E|^2)$ times. Finding a critical matching in Line 5, takes, by Proposition 3, $O(|V|^{2.5})$ time, while finding the colorings of the bipartite subgraphs of G in Lines 6–10 and 11, takes $O(|E| \log \Delta)$ time [16].

In Table 4.2 we compare the approximation ratios achieved by our three algorithms, as Δ increases, with the best known ones.

Δ	Best known ratio		Our ratio	
3	1.17	[20]	1.42	Algorithm Bipartite-1
4	1.61	[25]	1.50	Algorithm Bipartite-1
5	1.75	[25]	1.55	Algorithm Bipartite-1
6	1.86	[25]	1.60	Algorithm Bipartite-1
7	1.95	[25]	1.64	Algorithm Bipartite-3
8	2	[46]	1.67	Algorithm Bipartite-3
9	2	[46]	1.69	Algorithm Bipartite-3
10	2	[46]	1.71	Algorithm Bipartite-3
11	2	[46]	1.73	Algorithm Bipartite-3
12	2	[46]	1.75	Algorithm Bipartite-3
13	2	[46]	1.76	Algorithm Bipartite-3
20	2	[46]	1.83	Algorithm Bipartite-3
50	2	[46]	1.93	Algorithm Bipartite-3

Table 4.2: Approximation ratios for bipartite graphs

4.3 An 1.74-approximation algorithm for bipartite graphs

All the above algorithms for the EC(w) problem give good approximation guarantees for bipartite graphs of low degree. However, if the maximum degree is arbitrary large the achieved ratios tend or exceed two. In this section we exploit the same idea and

we attain an 1.74 approximation ratio that improves substantially the known 2 approximation one for the EC(w) problem on bipartite graphs of maximum degree Δ.

For a partition (p, q) of G, we call a set of edges $F \subseteq E_{p+1,q}$ *critical* if each vertex $u \in V$ of degree $d_{1,q}(u) > \Delta_{1,p}$, has degree in $G[F]$ such that $d_{1,q}(u) - \Delta_{1,p} \leq d_{G[F]}(u) \leq \Delta_{1,q} - \Delta_{1,p}$. The proposed algorithm relies on the existence of such a critical set of edges F: a solution for the subgraph $G[E_{1,q}]$ is found by concatenating a $\Delta_{1,p}$-coloring solution for the subgraph $G[E_{1,q} \setminus F]$ and a $(\Delta_{1,q} - \Delta_{1,p})$-coloring solution for the subgraph $G[F]$, if F exists, and by a $\Delta_{1,q}$-coloring of the subgraph $G[E_{1,q}]$, otherwise. For each partition (p, q), the algorithm computes a solution for the input graph G by concatenating a solution for $G[E_{1,q}]$ and a Δ-coloring solution for $G[E_{q+1,m}]$. The algorithm computes also a Δ-coloring solution for the input graph and returns the best among them.

Algorithm Bipartite-4

1: Find a Δ-coloring solution for G;
2: **for** $p = 0$ to $m - 1$ **do**
3: **for** $q = p + 1$ to m **do**
4: Find, if any, a critical set of edges F in $G[E_{p+1,q}]$;
5: **if** F exists **then**
6: Find a $\Delta_{1,p}$-coloring solution for $G[E_{1,q} \setminus F]$;
7: Find a $(\Delta_{1,q} - \Delta_{1,p})$-coloring solution for $G[F]$;
8: **else**
9: Find a $\Delta_{1,q}$-coloring solution for $G[E_{1,q}]$;
10: **end if**
11: Find a Δ-coloring solution for $G[E_{q+1,m}]$;
12: Find a solution for G by concatenating the solutions found in Lines 5–10 and 11;
13: **end for**
14: **end for**
15: Return the best among the solutions found in Lines 1 and 12;

The following proposition shows that the check in Line 4 of Algorithm Bipartite-4 can be done in polynomial time. This proposition is a generalization of Proposition 3 and plays a crucial role to achieve the approximation ratio of Algorithm Bipartite-4.

Proposition 4. *For a partition (p, q) of a graph $G = (V, E)$, a critical set of edges F, if any, can be found in $O(|V|^3)$ time.*

Proof. A (g, f)-factor of a graph G is a spanning subgraph H such that $g(u) \leq d_H(u) \leq f(u)$, for all $u \in V$.

Recall that $F \subseteq E_{p+1,q}$ and consider the subgraph $G[E_{p+1,q}]$. For each vertex u of $G[E_{p+1,q}]$ we define $g(u) = \max\{0, d_{1,q}(u) - \Delta_{1,p}\}$ and $f(u) = \Delta_{1,q} - \Delta_{1,p}$. Then, there exists a critical set of edges $F \subseteq E_{p+1,q}$ if and only if there exists a (g, f)-factor

in $G[E_{p+1,q}]$. It is known that such a factor, if any, can be found in $O(|V|^3)$ time [3]. □

Theorem 10. Algorithm Bipartite-4 *achieves an* 1.74*-approximation ratio for the* EC(w) *problem on bipartite graphs.*

Proof. The solution obtained by a Δ-coloring of the input graph computed in Line 1 of the algorithm is of weight $W_1 \leq \Delta \cdot w_1^*$.

Consider the partition (p, q) of G where $w(e_{p+1}) = w^*_{\lceil \frac{i}{2} \rceil}$ and $w(e_{q+1}) = w_i^*$, for $2 \leq i \leq \Delta$ (recall that $w_1^* \geq w_2^* \geq \cdots \geq w_{k^*}^*$ and $k^* \geq \Delta$). In such an iteration, all the edges in $E_{1,p}$ belong to $\lceil \frac{i}{2} \rceil - 1 \geq \Delta_{1,p}$ colors of an optimal solution $\mathcal{C}^*$, and all the edges in $E_{1,q}$ belong to $i - 1 \geq \Delta_{1,q}$ colors of an optimal solution $\mathcal{C}^*$.

If $\Delta_{1,q} = \Delta_{1,p}$ then the set F does not exist. Hence, a $\Delta_{1,q}$-coloring of $G[E_{1,q}]$ yields a solution of weight at most $\left(\lceil \frac{i}{2} \rceil - 1\right) \cdot w_1^*$ for this subgraph.

If $\Delta_{1,q} > \Delta_{1,p}$ then a critical set of edges F exists. Indeed, in this case the colors $C^*_{\lceil \frac{i}{2} \rceil}, C^*_{\lceil \frac{i}{2} \rceil + 1}, \ldots, C^*_{i-1}$ of $\mathcal{C}^*$ always contain some edges from $E_{p+1,q}$, for otherwise all the edges in $E_{1,q}$ belong to $\lceil \frac{i}{2} \rceil - 1$ colors of $\mathcal{C}^*$, a contradiction; these edges of $E_{p+1,q}$ could be a critical set of edges F for the partition (p, q). Thus, a $\Delta_{1,p}$-coloring solution of $G[E_{1,q} \setminus F]$ and a $(\Delta_{1,q} - \Delta_{1,p})$-coloring solution for $G[F]$ yield a solution for the subgraph $G[E_{1,q}]$ of weight at most $\Delta_{1,p} \cdot w_1^* + (\Delta_{1,q} - \Delta_{1,p}) \cdot w^*_{\lceil \frac{i}{2} \rceil} \leq \left(\lceil \frac{i}{2} \rceil - 1\right) \cdot w_1^* + \lfloor \frac{i}{2} \rfloor \cdot w^*_{\lceil \frac{i}{2} \rceil}$, since $\Delta_{1,p} \leq \lceil \frac{i}{2} \rceil - 1$, $\Delta_{1,q} \leq i - 1$ and $w_1^* \geq w^*_{\lceil \frac{i}{2} \rceil}$. Finally, a Δ-coloring solution for $G[E_{q+1,m}]$ is of weight at most $\Delta \cdot w_i^*$.

Hence, for such a partition (p, q) the algorithm finds a solution for the whole input graph of weight

$$W_i \leq \left(\left\lceil \frac{i}{2} \right\rceil - 1 \right) \cdot w_1^* + \left\lfloor \frac{i}{2} \right\rfloor \cdot w^*_{\lceil \frac{i}{2} \rceil} + \Delta \cdot w_i^*, \ 2 \leq i \leq \Delta.$$

As the algorithm returns the best among the solutions found, we have Δ bounds on the weight W of this best solution, i.e.,

$$\begin{array}{l} W \leq \Delta \cdot w_1^*, \text{ if } i = 1, \text{ and} \\ W_i \leq \left(\left\lceil \frac{i}{2} \right\rceil - 1 \right) \cdot w_1^* + \left\lfloor \frac{i}{2} \right\rfloor \cdot w^*_{\lceil \frac{i}{2} \rceil} + \Delta \cdot w_i^*, \text{ if } \ 2 \leq i \leq \Delta. \end{array}$$

To derive our ratio, we search again for appropriate multipliers, which can be found by solving a system of linear equations, as in Section 4.2. For the case where the maximum degree of the graph is a power of 2, we get the following multipliers:

$$x_i = \begin{cases} \displaystyle\sum_{j=0}^{\lfloor \log \frac{\Delta}{i} \rfloor} \left(- \left(\frac{-1}{\Delta} \right)^{j+1} \sum_{y=1}^{2^j} \left(\prod_{z=1}^{j} \left(2^{z-1}(i-1) + \left\lceil \frac{y}{2^{j-z+1}} - \frac{1}{2} \right\rceil \right) \right) \right), & \text{if} \quad \Delta \geq i \geq 2 \\ \displaystyle\frac{1}{\Delta} \left(1 - x_2 - \sum_{j=3}^{\Delta} \left(\left\lceil \frac{j}{2} \right\rceil - 1 \right) x_j \right), & \text{if} \quad i = 1. \end{cases}$$

These multipliers become more complicated in the case where the maximum degree of the input graph is not a power of 2. In this case, for $2 \leq i \leq \Delta$ we get

$$x_i = \sum_{j=0}^{\lfloor \log \frac{\Delta}{i} \rfloor} \left(- \left(\frac{-1}{\Delta} \right)^{j+1} \sum_{y=1}^{2^j} \left(\prod_{z=1}^{j} \left(2^{z-1}(i-1) + \left\lceil \frac{y}{2^{j-z+1}} - \frac{1}{2} \right\rceil \right) \right) \right) -$$
$$\left(\frac{-1}{\Delta} \right)^{\lfloor \log \frac{\Delta}{i} \rfloor + 2} \sum_{y=1}^{\left(\Delta - i + 1 - \sum_{r=0}^{\lfloor \log \frac{\Delta}{i} \rfloor} ((i-1)2^r) \right)} \left(\prod_{z=1}^{\lfloor \log \frac{\Delta}{i} \rfloor + 1} \left(2^{z-1}(i-1) + \left\lceil \frac{y}{2^{\lfloor \log \frac{\Delta}{i} \rfloor + 2 - z}} - \frac{1}{2} \right\rceil \right) \right)$$

while x_1 is the same as in the previous case.

By multiplying both sides of the i-th bound on W by x_i and adding all of them we have $\sum_{i=1}^{\Delta} x_i \cdot W \leq w_1^* + w_2^* + \ldots + w_\Delta^* \leq OPT$, and hence, $\frac{W}{OPT} \leq \frac{1}{\sum_{i=1}^{\Delta} x_i}$.

Using `Mathematica`, we compute the above ratio for quite large values of Δ, and it is found to tend to 1.74. Table 4.3 shows the values of the ratio as Δ increases. However, an interesting question is whether we can compute a close formula for this ratio.

Δ	Our ratio
2^2	1.48837
2^3	1.60188
2^4	1.66582
2^5	1.70023
2^6	1.71809
2^7	1.72719
2^8	1.73178
2^9	1.73409
2^{10}	1.73525
2^{11}	1.73583
2^{12}	1.73612
2^{13}	1.73626
2^{14}	1.73634
2^{15}	1.73637
2^{16}	1.73639
2^{17}	1.73640

Table 4.3: Approximation ratios achieved by Algorithm Bipartite-4 for different values of Δ.

□

4.4 Bi-valued graphs

In this section we show first that the EC(w) problem is NP-complete for complete graphs with bi-valued edge weights. Recall that the EC(w) problem is polynomial for bi-valued bipartite graphs [22], while for general bi-valued graphs it generalizes the classical edge-coloring problem, which is known to be NP-complete even for cubic graphs [41]. In the next theorem we give a reduction from this latter problem.

Theorem 11. *The* EC(w) *problem is NP-complete for complete graphs even with edge weights* $w(e) \in \{1, 2\}$.

Proof. The edge-coloring problem for cubic graphs takes as input a graph $G = (V, E)$, $|V| = n$, with $d(u) = 3$, for each $u \in V$, and asks for the existence of a proper 3-coloring of G. Notice that any cubic graph has an even number, n, of vertices.

From such an instance we construct a complete weighted graph K_n with edge weights $w(e) = 2$, for each $e \in E$, and $w(e) = 1$, otherwise, and we show that there is a 3-coloring of G iff there is a solution $\mathcal{C}$ for the EC(w) problem on K_n of weight at most $n + 2$.

Assume, first, that there is a 3-coloring of G. Then, there are three colors of K_n each one of weight equal to 2, which include all the edges of K_n of weight 2. Let $K_n - G$ be the graph induced by the remaining edges of K_n (those of weight 1). The graph $K_n - G$ is $(n-4)$-colorable as a $(n-4)$-regular graph of even order [13]. Therefore, there is a solution $\mathcal{C}$ for the EC(w) problem on K_n of weight at most $3 \cdot 2 + (n-4) \cdot 1 = n + 2$.

Conversely, consider that there is a solution $\mathcal{C}$ to the EC(w) problem on K_n of weight at most $n+2$. This solution contains $k \geq n-1$ colors, since a complete graph of even order can be colored with at least $n-1$ colors [27]. Moreover, $\mathcal{C}$ contains at least three colors of weight equal to 2, since, by its construction, K_n has exactly three edges of weight 2 adjacent to each vertex. Assume that there is a fourth color in $\mathcal{C}$ of weight equal to 2. Then, $\mathcal{C}$ will be of weight at least $4 \cdot 2 + (k-4) \cdot 1 \geq n+3$, a contradiction. Therefore, $\mathcal{C}$ contains exactly 3 colors of weight equal to 2, which imply a 3-coloring for G. □

Theorem 11 implies that the EC(w) problem is NP-complete in all superclasses of complete graphs, including split and interval graphs. Note also that the complexity of the classical edge-coloring problem on interval graphs of even maximum degree remains an open question [7]. Next corollary follows also from Theorem 11.

Corollary 1. *It is NP-hard to approximate the* EC(w) *problem within a ratio less than* $\frac{n+3}{n+2}$ *in complete graphs.*

In what follows, we present an approximation algorithm for general graphs with two different edge weights. Assume that the edges of the graph $G = (V, E)$ have weights either 1 or $t \geq 2$. Let $G[E_1]$, of maximum degree Δ_1, and $G[E_t]$, of maximum degree Δ_t, be the subgraphs of G induced by its edges of weights 1 and t, respectively.

Algorithm Bi-valued

1: Find a $(\Delta + 1)$-coloring solution for G;
2: Find a $(\Delta_1 + 1)$-coloring solution for $G[E_1]$, a $(\Delta_t + 1)$-coloring solution for $G[E_t]$ and concatenate them;
3: Return the best among the two solutions found in Lines 1 and 2;

Theorem 12. Algorithm Bi-valued *achieves a* $\frac{4}{3}$*-approximation ratio for the* EC(w) *problem on general graphs of arbitrarily large* Δ *and edge weights* $w(e) \in \{1, t\}$.

Proof. An optimal solution contains at least Δ colors and at least Δ_t of them are of weight equal to t. Therefore, a lower bound to the weight of an optimal solution is $OPT \geq \Delta_t \cdot t + (\Delta - \Delta_t)$.

A $(\Delta + 1)$-coloring of G in Line 1 of the algorithm yields a solution for the EC(w) problem of weight $W \leq (\Delta + 1) \cdot t$, while a $(\Delta_1 + 1)$-coloring of $G[E_1]$ and a $(\Delta_t + 1)$-coloring of $G[E_t]$ in Line 2 yield another solution of weight $W \leq (\Delta_t + 1) \cdot t + (\Delta_1 + 1) \cdot 1 \leq (\Delta_t + 1) \cdot t + (\Delta + 1)$. By multiplying both sides of the first inequality with $\frac{\Delta_t^2 + 2\Delta_t - \Delta}{(\Delta+1)^2}$, both sides of the second one with $\frac{\Delta - \Delta_t}{\Delta + 1}$ and adding them, we get $\frac{\Delta^2 + \Delta_t^2 - \Delta \cdot \Delta_t + \Delta_t}{(\Delta+1)^2} \cdot W \leq \Delta_t \cdot t + (\Delta - \Delta_t) \leq OPT$, that is $\frac{W}{OPT} \leq \frac{(\Delta+1)^2}{(\Delta - \Delta_t)^2 + \Delta_t(\Delta+1)}$. This ratio is maximized when $\Delta_t = \frac{\Delta - 1}{2}$, and therefore $\frac{W}{OPT} \leq \frac{4(\Delta+1)}{(\Delta+1)+2(\Delta-1)} = \frac{4\Delta+4}{3\Delta-1} = \frac{4}{3} + \frac{16}{9\Delta-3}$. □

Chapter 5

Max-Edge-Coloring on trees

In this chapter we describe approximation results for the EC(w) problem on trees, as well as polynomial algorithms for subclasses of trees.

In Section 5.1 we present a polynomial algorithm for a special class of trees, namely stars of chains (or spiders). To obtain this algorithm we need the following theorem for trees of bounded maximum degree.

Theorem 13. *The* EC(w) *problem can be solved in polynomial time on trees of bounded maximum degree.*

Proof. By Theorem 2(i), the EC(w) problem is polynomial on trees when the number, k, of colors is fixed. Moreover, by Proposition 2 it holds that $k \leq 2\Delta - 1$, and hence the theorem follows. □

Although the complexity question for the EC(w) problem on trees remains open, we give, in Section 5.2, a 3/2-approximation algorithm. Next, we present two approximation algorithms of low exponential complexity. The complexity of the first algorithm is exponential to the maximum degree of the tree, while the second one is exponential to the number of edges. Finally, a PTAS is presented for the EC(w) problem on trees in Section 5.4, reaching the approximability of the VC(w) problem on the same class of graphs.

5.1 Stars of chains

A star consists of m edges, $e_1, e_2, \ldots, e_m$, sharing a common endpoint. Obviously, the optimal solution to the EC(w) problem for such a weighted star is of weight $OPT = \sum_{j=1}^{m} w(e_j)$ and consists of exactly $\Delta = m$ colors.

A star of chains consists of $\Delta \geq 3$ chains, $T_1, T_2, \ldots, T_\Delta$, all starting from a common vertex, say u. We consider each chain T_j, $1 \leq j \leq \Delta$, starting from u with an edge e_j^u which we call start edge. We also assume, w.l.o.g., that $w(e_1^u) \geq w(e_2^u) \geq \ldots \geq w(e_\Delta^u)$.

Lemma 3. *For an optimal solution* $\mathcal{C}^* = \{C_1^*, C_2^*, \ldots, C_{k^*}^*\}$ *of the* EC(w) *problem on a star of chains the following hold:*

(i) The number of colors k^* equals to either Δ or $\Delta + 1$.

(ii) Only $p \leq 3$ colors have cardinality $|C_i^*| > 1$.

(iii) At least the $p - 1$ heaviest start edges appear in these p colors.

Proof. For item (i), according to Proposition 2, $\Delta \leq k^* \leq \Delta' - 1$. Here, it holds that $\Delta' = \Delta + 2$.

In order to prove item (ii), assume that an optimal solution has more than three colors of cardinality $|C_i^*| > 1$. Consider those colors sorted in non-increasing order with respect to their weights. Each non start edge e has at most two neighbor edges. So, such an edge e can be moved in one of the three first heaviest colors, since its neighbor edges can belong in at most two different colors.

Finally, for item (iii), consider first that $p = 2$. Assume that in the optimal solution the heaviest start edge e_1^u does not belong to neither of two colors of cardinality $|C_i^*| > 1$. Then e_1^u can be either inserted into one of those two colors (if this does not contain another start edge), or e_1^u can replace an existing start edge. In both cases the weight of the optimal solution does not increase.

Assume next that $p = 3$. As in the previous case, e_1^u can be inserted into one of the three colors of cardinality $|C_i^*| > 1$. Similarly, e_2^u can be inserted in one of the remaining two of those colors. □

In what follows, we distinguish between two cases according to possible number of colors in an optimal solution, i.e., $\Delta + 1$ or Δ.

If an optimal solution consists of $\Delta + 1$ colors, then it contains exactly one color without any start edge. Algorithm Star-of-Chains($\Delta+1$) finds such an optimal solution with $\Delta + 1$ colors.

Algorithm Star-of-Chains($\Delta + 1$)

1: Remove from the star the $\Delta - 2$ lightest start edges;
   (this creates a graph H consisting of $\Delta - 1$ chains)
2: Find an optimal solution $S^*(H)$ for the graph H, using Theorem 6;
3: **if** there are 3 non empty colors in $S^*(H)$ **then**
4: Return the solution consisting of these 3 colors of $S^*(H)$ plus $\Delta - 2$ colors each one containing one of the removed $\Delta - 2$ lightest start edges;
5: **end if**

Note that Algorithm Star-of-Chains($\Delta + 1$) it is possible to return $\Delta - 1$ colors of $|C_i| = 1$, in the case where one of the three colors of $S^*(H)$ found in Line 2 consists of a single edge. Taking into account Lemma 3, it follows that Algorithm Star-of-Chains($\Delta + 1$) returns an optimal solution of $\Delta + 1$ colors since: (i) the $\Delta - (p - 1)$ colors of cardinality $|C_i| = 1$ contain the $\Delta - (p - 1)$ lightest start edges (one per each color), and (ii) the weight of p colors is optimal.

The complexity of Algorithm Star-of-Chains($\Delta + 1$) is dominated by Line 2 and by Theorem 6 it is $O(|E| \log |E|)$.

If an optimal solution consists of Δ colors, then each of them contains a start edge. Algorithm Star-of-Chains(Δ) returns such an optimal solution with Δ colors.

Algorithm Star-of-Chains(Δ)

1: **for** $j = 3$ to Δ **do**
2: Remove $\Delta - 3$ start edges $e_3^u, e_4^u, \ldots, e_{j-1}^u, e_{j+1}^u, \ldots, e_\Delta^u$;
(this creates a star T of 3 chains and a graph H of $\Delta - 3$ chains)
3: Find the optimal solution $S^*(T)$ using Theorem 13;
4: **if** there are exactly 3 colors in $S^*(T)$ **then**
5: Find the optimal solution $S^*(H)$ using Theorem 6;
6: Combine the solutions $S^*(T)$ and $S^*(H)$ into exactly 3 colors;
7: Find a solution for the initial star consisting of these 3 colors plus $\Delta - 3$ colors each one containing one of the removed $\Delta - 3$ start edges;
8: **end if**
9: **end for**
10: Return the best solution found;

The star T, obtained after Line 2 of the above algorithm, is a tree of maximum degree three and therefore an optimal solution $S^*(T)$, in Line 3, can be found in polynomial time, by Theorem 13. Notice that such a solution consists of at least three colors. Moreover, the optimal solution $S^*(H)$ built in Line 5, consists of at most three colors, by Proposition 2.

In Line 6, an optimal solution of three colors for the edges in T and H can be obtained by considering the colors in both solutions in non-increasing order with respect to their weights and merging the colors of each solution having the same rank, since T and H are vertex-disjoint. The optimality of the solution that Algorithm Star-of-Chains(Δ) returns, follows from Lemma 3 using the same arguments as for Algorithm Star-of-Chains($\Delta + 1$).

The complexity of the algorithm is dominated by Line 3 which takes polynomial time and is executed $\Delta - 2$ times.

By Lemma 3 the optimal solution to the EC(w) problem on stars of chains is the best between the solution found by Algorithm Star-of-Chains($\Delta + 1$) (optimal with $\Delta + 1$ colors) and the one found by Algorithm Star-of-Chains(Δ) (optimal with Δ colors). Thus, the following theorem holds.

Theorem 14. *The* EC(w) *problem is polynomial on stars of chains.*

5.2 A 3/2 approximation algorithm

In this section, we first present an $(1 + \frac{w_1^* - w_\Delta^*}{OPT})$-approximation algorithm for the EC(w) problem on trees. Then, combining this algorithm with Algorithm KK we derive a 3/2 approximation ratio.

For our first algorithm we consider the tree rooted in an arbitrary vertex and we denote by E^u the edges of the tree adjacent to a vertex u. The algorithm traverses the vertices of the tree in pre-order and for each vertex u assigns the edges in E^u to colors as follows.

ALGORITHM TREES

1: Root the tree in an arbitrary vertex r;
2: **for** each vertex u in a pre-order traversal of the tree **do**
3: Let $\langle E^u \rangle = \langle e^u_1, e^u_2, \ldots, e^u_{d(u)} \rangle$, and e^u_j, $1 \leq j \leq d(u)$, be the edge between $u, u \neq r$, and its parent;
4: **for** $i = 1$ to $d(u)$, $i \neq j$, **do**
5: Insert edge e^u_i into the first color not containing other edge in E^u;
6: **end for**
7: **end for**

To analyze our algorithm we define y_i, $1 \leq i \leq \Delta$, to be the weight of the heaviest edge between those ranked i in each ordering $\langle E^u \rangle$, $u \in V$, i.e., $y_i = \max_{u \in V} \{w(e^u_i)\}$. It is clear that $y_1 \geq y_2 \geq \ldots \geq y_\Delta$. Next two propositions use these values for bounding the weights of the colors of both an optimal solution and a solution found by ALGORITHM TREES. Recall that an optimal solution to the EC(w) problem consists of at least Δ colors.

Proposition 5. *For all $1 \leq i \leq \Delta$, it holds that $w^*_i \geq y_i$.*

Proof. Let $e = (u, v)$ be the heaviest edge with rank equal to i, i.e., $y_i = w(e)$. W.l.o.g., assume that e is ranked i in E^u. Then, there exist i edges in E^u of weight at least y_i and as they belong into i different colors in an optimal solution, it follows that $w^*_i \geq y_i$. □

Proposition 6. ALGORITHM TREES *constructs a solution of exactly Δ colors. For the weight, w_i, of the i-th, $2 \leq i \leq \Delta$, color it holds that $w_i \leq y_{i-1}$.*

Proof. For a vertex $u \neq r$ of the tree let e be the edge between u and its parent and j be its rank in E^u, i.e., $e = e^u_j$. In the iteration processing the vertex u the edge e has already been inserted by the algorithm into a color, say C_p.

The algorithm inserts the edges in E^r into $d(r) \leq \Delta$ colors. For any other vertex u, the algorithm inserts the edges in $E^u \setminus \{e\}$ into $d(u) - 1 \leq \Delta - 1$ colors different than C_p. Therefore, the algorithm finds a solution $\mathcal{C} = \{C_1, C_2, \ldots, C_\Delta\}$ of exactly Δ colors.

We prove the bounds on the color's weights by induction on the vertices in the order they are processed by the algorithm. We consider all colors in $\mathcal{C}$ of an initial weight $w_i = 0$, $1 \leq i \leq \Delta$.

For the root vertex r, the algorithm inserts each edge e^r_i into color C_i, $1 \leq i \leq d(r)$. Clearly, $w_i = w(e^r_i) \leq y_i \leq y_{i-1}$, $2 \leq i \leq \Delta$.

Assume that before the iteration processing a vertex $u \neq r$, it holds that $w_i \leq y_{i-1}$, $2 \leq i \leq \Delta$, and let w'_i be the weight of the color C_i, $2 \leq i \leq \Delta$, after processing

the vertex u. We prove that $w'_i \leq y_{i-1}$, $2 \leq i \leq \Delta$, by distinguishing among three cases depending on the values of p and j:

(i) $p = j$: Each edge e_i^u belongs to color C_i, $1 \leq i \leq d(u)$. Since $w_i \leq y_{i-1}$ and $w(e_i^u) \leq y_i$, it follows that $w'_i = \max\{w_i, w(e_i^u)\} \leq \max\{y_{i-1}, y_i\} = y_{i-1}$, $2 \leq i \leq \Delta$.

(ii) $p > j$: For $1 \leq i \leq j-1$ and $p+1 \leq i \leq d(u)$ each edge e_i^u belongs to color C_i and we conclude as in Case (i). For $j+1 \leq i \leq p$ each edge e_i^u belongs to color C_{i-1}, that is $w'_i = \max\{w_i, w(e_{i+1}^u)\} \leq \max\{y_{i-1}, y_{i+1}\} = y_{i-1}$.

(iii) $p < j$: For $1 \leq i \leq p-1$ and $j+1 \leq i \leq d(u)$ each edge e_i^u belongs to color C_i and we conclude as in Case (i). For $p \leq i \leq j-1$ each edge e_i^v belongs to color C_{i+1}, that is $w'_i = \max\{w_i, w(e_{i-1}^u)\} \leq \max\{y_{i-1}, y_{i-1}\} = y_{i-1}$. □

Using the bounds established in Propositions 5 and 6 we obtain the next lemma.

Lemma 4. Algorithm Trees *achieves an approximation ratio of* $1 + \frac{w_1^* - w_\Delta^*}{OPT} < 2$ *for the* EC(w) *problem on trees.*

Proof. For the weight of the first color obtained by Algorithm Trees it holds that $w_1 \leq y_1 = w_1^*$, since both y_1 and w_1^* are equal to the weight of the heaviest edge of the tree. By Proposition 6 it holds that $w_i \leq y_{i-1}$, $2 \leq i \leq \Delta$ and by Proposition 5 it holds that $y_i \leq w_i^*$, $1 \leq i \leq \Delta$. Therefore, the weight of the solution obtained by Algorithm Trees is $W = \sum_{i=1}^{\Delta} w_i \leq y_1 + \sum_{i=2}^{\Delta} y_{i-1} = y_1 + \sum_{i=1}^{\Delta-1} y_i \leq w_1^* + \sum_{i=1}^{\Delta-1} w_i^* \leq w_1^* + OPT - w_\Delta^*$, that is $\frac{W}{OPT} \leq 1 + \frac{w_1^* - w_\Delta^*}{OPT} < 2$. □

The example illustrated in Figure 5.1(a) shows that the ratio of our algorithm can be arbitrarily close to 2. For this instance $OPT = 1 + 2\epsilon$ (Figure 5.1(b)), the weight of the solution found by Algorithm Trees is $W = 2 + \epsilon$ (Figure 5.1(c)) and the approximation ratio becomes $\frac{2+\epsilon}{1+2\epsilon}$.

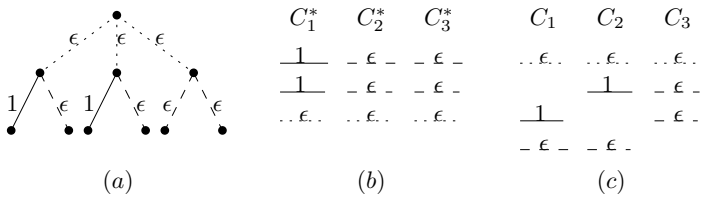

Figure 5.1: A tight example for the 2 approximation ratio of Algorithm Trees ($\Delta = 3$, $\epsilon << 1$).

To derive the 3/2 approximation ratio we simply select the best among the solutions found by Algorithm KK and Algorithm Trees.

Theorem 15. *There is a $\frac{3}{2}$-approximation algorithm for the* EC(w) *problem on trees.*

Proof. Let W be the weight of the best among the solutions found by Algorithm KK and Algorithm Trees. By Lemma 1 it holds that $\frac{W}{OPT} \leq 2 - \frac{w_1^*}{OPT}$ and by Lemma 4 that $\frac{W}{OPT} \leq 1 + \frac{w_1^* - w_\Delta^*}{OPT}$. As the first bound is increasing and the second one is decreasing with respect to OPT, it follows that the ratio $\frac{W}{OPT}$ is maximized when $2 - \frac{w_1^*}{OPT} = 1 + \frac{w_1^* - w_\Delta^*}{OPT}$, that is $OPT = 2 \cdot w_1^* - w_\Delta^*$. Therefore, $\frac{W}{OPT} \leq 2 - \frac{w_1^*}{OPT} = 2 - \frac{w_1^*}{2 \cdot w_1^* - w_\Delta^*} \leq 2 - \frac{w_1^*}{2 \cdot w_1^*} = \frac{3}{2}$. □

For the tightness of the analysis in Theorem 15 consider the instance given in Figure 5.2(a). For this instance $OPT = 2 + 2\epsilon$ (Figure 5.2(b)) and the weights of the solutions found by Algorithm Trees and Algorithm KK are 3 (Figure 5.2(c)) and $3 - \epsilon$ (Figure 5.2(d)), respectively. Our algorithm selects the solution found by Algorithm KK and the approximation ratio becomes $\frac{3-\epsilon}{2+2\epsilon}$.

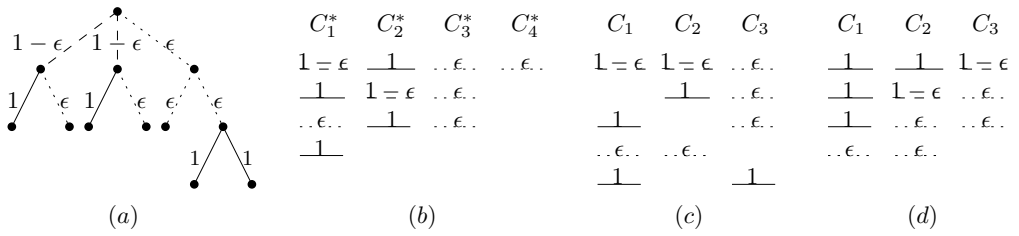

Figure 5.2: A tight example for the 3/2-approximation algorithm for trees ($\Delta = 3$, $\epsilon << 1$).

5.3 Moderately exponential approximation algorithms

In this section, we present two approximation algorithms for trees that improve the 3/2 ratio of Theorem 15 within exponential running time much better than that needed for the computation of an optimal solution.

The idea employed by the algorithms is to find an approximate solution to the EC(w) problem on a tree $T = (V, E)$ by searching exhaustively for the weights of a number of colors of an optimal solution $\mathcal{C}^*$. A parameter z, given as input to the algorithms, determines the (maximum) number of colors of $\mathcal{C}^*$ that we search

exhaustively and, hence, the complexity and the approximation ratio of the algorithms. In such an exhaustive search, each step of the proposed algorithms has to answer to an instance of the FEASIBLE-EC(w) problem, which can be done through the EC(ϕ) problem (see Section 3.1).

The first algorithm proposed in Section 5.3 is exponential to the maximum degree, Δ, of the input tree and achieves a ρ approximation ratio in $O^*(m^{f(\rho)\cdot\Delta})$ time, where $f(\rho) = \dfrac{9-\rho}{4\rho}$. The second algorithm presented in Section 5.3 is exponential to the number of edges, m, of the input tree and achieves a ratio of ρ in $O^*(g(\rho)^m)$ time, where $g(\rho) = \dfrac{(2\rho-1)^2+1}{(2\rho-1)^{2(2\rho-1)^2/((2\rho-1)^2+1)}}$. Some values of $\rho \leq 3/2$, $f(\rho)$ and $g(\rho)$ are summarized in Table 5.1.

Complexity	ρ	OPT	1.1	1.2	1.3	1.4	1.5
$O^*(m^{f(\rho)\cdot\Delta})$	$f(\rho)$	2	1.795	1.625	1.481	1.357	1.250
$O^*(g(\rho)^m)$	$g(\rho)$	2	1.968	1.896	1.811	1.727	1.649

Table 5.1: Approximation ratios vs. complexities for trees

An exponential to Δ algorithm

This algorithm depends on a parameter z taking integer values in $[1, 2\Delta - 1]$ and iterates z times, for $j = 1, 2, \ldots z$. In each iteration the algorithm considers all the combinations of j edge weights as the weights of the j heaviest colors of an optimal solution. For each combination of weights, $w_1 \geq w_2 \geq \ldots \geq w_j$, the algorithm has to answer to an instance of the FEASIBLE-EC(w) problem on the input tree T. In order a "yes" answer to this FEASIBLE-EC(w) problem to be probable for all values of j we extend the combination of weights $w_1 \geq w_2 \geq \ldots \geq w_j$ to a sequence $w_1 \geq w_2 \geq \ldots \geq w_j = w_{j+1} = w_{j+2} = \ldots = w_k$ by adding $k - j$ new weights all equal to w_j. In fact, this extended sequence consists of $k = j - 1 + \Delta$ weights, if $j \leq \Delta$ (this way the T's edges of weights $w(e) \leq w_j$ can be assigned into the Δ colors of weight w_j) and $k = 2\Delta - 1$, otherwise (since by Proposition 2 any solution to the EC(w) problem consists of at most $2\Delta - 1$ colors). Hence, $k = \min\{j - 1 + \Delta, 2\Delta - 1\}$. This instance of the FEASIBLE-EC(w) problem has answer "yes" if and only if the edges of weight $w(e) > w_j$ can be assigned to (colors of) weights greater than w_j (see the proof of Theorem 16). In this case the algorithm finds a feasible solution for the EC(w) problem and it returns the best among all feasible solutions found.

ALGORITHM TREES-$\Delta(z)$

1: **for** $j = 1$ to z **do**
2: **for** each combination of j edge weights, $w_1 \geq w_2 \geq \ldots \geq w_j$, **do**
3: Answer to the FEASIBLE-EC(w) problem with input T and $k = \min\{j - 1 + \Delta, 2\Delta - 1\}$ weights: $w_1 \geq w_2 \geq \ldots \geq w_j = w_{j+1} = w_{j+2} = \ldots = w_k$;
4: **if** the answer is "yes" **then**
5: A feasible solution to the EC(w) problem is found;
6: **end if**
7: **end for**
8: **end for**
9: Return the best among the feasible solutions found;

Theorem 16. *For any $\rho \geq 1$,* ALGORITHM TREES-$\Delta(z)$ *achieves a ρ approximation ratio for the* EC(w) *problem on trees, in polynomial space and running time $O^*(m^{f(\rho)\Delta})$, where $f(\rho) = \frac{9-\rho}{4\rho}$.*

Proof. Consider the j-th iteration of the algorithm and in this iteration the combination of j edge weights which coincide with the weights $w_1^* \geq w_2^* \geq \ldots \geq w_j^*$, of the j heaviest colors of an optimal solution $\mathcal{C}^*$. In this step the algorithm answers to the instance of the FEASIBLE-EC(w) problem with input T and weights $w_i \geq w_i^*$, $1 \leq i \leq k$. We claim that this FEASIBLE-EC(w) problem has always a "yes" answer. Indeed, if $k = 2\Delta - 1$, then the claim follows since $k^* \leq 2\Delta - 1$ and $w_i \geq w_i^*$, $1 \leq i \leq k^*$. If $k = j - 1 + \Delta < 2\Delta - 1$, then the edges of weights $w(e) > w_j^*$ can be assigned (belong) to the $j - 1$ heaviest weights (colors of $\mathcal{C}^*$). Moreover, there are Δ weights equal to w_j^* and the edges of weights $w(e) \leq w_j^*$ can be assigned to them. Hence, a feasible solution for the EC(w) problem on T is found of weight

$$W_j = w_1^* + w_2^* + \ldots + w_{j-1}^* + (k - j + 1) \cdot w_j^*.$$

The algorithm finds such a feasible solution in each iteration j and as it returns the best among them we obtain Δ bounds on the weight of this best solution, that is $W \leq w_1^* + w_2^* + \ldots + w_{j-1}^* + (k-j+1) \cdot w_j^*$, $1 \leq j \leq z$. Proceeding as in the proof of Theorem 8 we find z multipliers

$$x_j = \begin{cases} \dfrac{2\Delta - 1 - z}{\Delta^2} \left(\dfrac{\Delta - 1}{\Delta}\right)^{\Delta - 1 - j}, & \text{if} \quad 1 \leq j \leq \Delta \\ \dfrac{2\Delta - 1 - z}{(2\Delta - j)(2\Delta - 1 - j)}, & \text{if} \quad \Delta + 1 \leq j \leq z \end{cases}$$

such that $\dfrac{W}{OPT} \leq \dfrac{1}{\sum_{i=1}^{z} x_i} = \dfrac{1}{1 - \frac{2\Delta-1-z}{\Delta} \cdot (\frac{\Delta-1}{\Delta})^{\Delta-1}}$.

The EC(w) problem is polynomial for graphs of $\Delta = 2$ and as for $\Delta \geq 3$ it holds that $\left(\dfrac{\Delta - 1}{\Delta}\right)^{\Delta - 1} > \dfrac{4}{9}$ we get $\dfrac{W}{OPT} \leq \dfrac{1}{1 - \frac{4}{9} \cdot \frac{2\Delta-1-z}{\Delta}} = \rho$. Hence, an approximation ratio ρ is derived for $z = \dfrac{9 - \rho}{4\rho} \cdot \Delta - 1 = f(\rho) \cdot \Delta - 1$, where $f(\rho) = \dfrac{9 - \rho}{4\rho}$.

The complexity of Algorithm Trees-$\Delta(z)$ is exponential in z. In Line 2 the algorithm examines $\binom{m}{j}$ combinations of weights. Thus, for all iterations $\sum_{j=1}^{z}\binom{m}{j} = O(z \cdot m^z)$ combinations of weights are examined. For each one of these combinations, it takes $O(m \cdot \Delta^{3.5})$ time to answer to the instance of the Feasible-EC(w) in Line 3. Since z and Δ are $O(m)$, the complexity of Algorithm Trees-$\Delta(z)$ is $O^*(m^z)$, that is $O^*\left(m^{f(\rho)\Delta}\right)$. Moreover, the algorithm needs polynomial space, since Line 3 is executed independently for each combination of weights. □

Notice that for $z = 2\Delta - 1$ the Algorithm Trees-$\Delta(z)$ finds an optimal solution within $O^*\left(m^{2\Delta}\right)$ time.

An exponential to m algorithm

This algorithm depends on a parameter z taking integer values in $[1, \lfloor \frac{m}{2} \rfloor]$ and iterates $2z$ times, for $k = 1, 2, \ldots, z, m-z, \ldots, m$. In each iteration, the algorithm exhaustively considers k edge weights, $w_1, w_2, \ldots, w_k$, as the weights of the k heaviest colors of an optimal solution $\mathcal{C}^*$, and answers to the instance of the Feasible-EC(w) problem, with input T and $w_1 \geq w_2 \geq \ldots \geq w_k$. This way an optimal solution is found when $k^* \leq z$ or $k^* \geq m - z$. In order to derive an approximate solution when $z < k^* < m - z$, the algorithm, in the iteration where $k = z$, answers also to instances of the Feasible-EC(w) problem with input T and weights $w_1 \geq w_2 \geq \ldots \geq w_z = w_{z+1} = \ldots = w_{k'}$, for $k' = z+1, z+2, \ldots, m-z-1$. The algorithm returns the best among the feasible solutions found.

Algorithm Trees-$E(z)$

1: **for** $k = 1$ to z and $k = m - z$ to m **do**
2: **for** each combination of k edge weights, $w_1 \geq w_2 \geq \ldots \geq w_k$, **do**
3: Answer to the Feasible-EC(w) with input T and weights $w_1 \geq w_2 \geq \ldots \geq w_k$;
4: **if** the answer is "yes" **then**
5: A feasible solution to the EC(w) problem is found;
6: **end if**
7: **if** $k = z$ **then**
8: **for** $k' = z + 1$ to $m - z - 1$ **do**
9: Answer to the Feasible-EC(w) with input T and k' weights: $w_1 \geq w_2 \geq \ldots \geq w_z = w_{z+1} = w_{z+2} = \ldots = w_{k'}$;
10: **if** the answer is "yes" **then**
11: A feasible solution to the EC(w) problem is found;
12: **end if**
13: **end for**
14: **end if**
15: **end for**
16: **end for**
17: Return the best among the feasible solutions found;

Theorem 17. *For any $\rho \geq 1$,* Algorithm Trees-$E(z)$ *achieves a ρ approximation ratio for the* EC(w) *problem on trees, in polynomial space and running time $O^*(g(\rho)^m)$, where* $g(\rho) = \dfrac{(2\rho-1)^2+1}{(2\rho-1)^{2(2\rho-1)^2/((2\rho-1)^2+1)}}$.

Proof. If $k^* \leq z$ or $k^* \geq m-z$ then the algorithm in an iteration of Lines 3–6 finds an optimal solution.

If $z < k^* < m-z$ then we consider the following two solutions found by the algorithm:

(i) In the iteration where $k = m-z$, for a combination $w_1 \geq w_2 \geq \ldots \geq w_k$ of weights, it holds that $w_i = w_i^*$, $1 \leq i \leq k^*$. Hence, for this combination there is a feasible solution of weight at most $w_1^* + w_2^* + \ldots + w_{k^*}^* + (m-z-k^*)w_{k^*}^* = OPT + (m-z-k^*)w_{k^*}^*$.

(ii) In the iteration where $k = z$ and $k' = k^*$, for a combination $w_1 \geq w_2 \geq \ldots \geq w_k$ of weights, it holds that $w_i = w_i^*$, $1 \leq i \leq z$. Hence, for this combination there is a feasible solution of weight at most $w_1^* + w_2^* + \ldots + w_z^* + (k^*-z)w_z^* = OPT - \sum_{i=z+1}^{k^*} w_i^* + (k^*-z)w_z^*$.

Thus, it holds that

$$\begin{aligned}\frac{W}{OPT} &\leq \min\left\{\frac{OPT + (m-z-k^*)w_{k^*}^*}{OPT}, \frac{OPT - \sum_{i=z+1}^{k^*} w_i^* + (k^*-z)w_z^*}{OPT}\right\} \\ &\leq \min\left\{1 + \frac{(m-z-k^*)w_{k^*}^*}{zw_z^* + (k^*-z)w_{k^*}^*}, 1 + \frac{(k^*-z)(w_z^* - w_{k^*}^*)}{zw_z^* + (k^*-z)w_{k^*}^*}\right\}.\end{aligned}$$

As the first value is increasing with $w_{k^*}^*$ and the second one is decreasing, this quantity is maximized when $(k^*-z)w_z^* = (m-2z)w_{k^*}^*$. Therefore, we have

$$\frac{W}{OPT} \leq 1 + \frac{(m-z-k^*)w_{k^*}^*}{\frac{z(m-2z)}{k^*-z}w_{k^*}^* + (k^*-z)w_{k^*}^*} = \frac{k^*(m-2z)}{z(m-2z) + (k^*-z)^2},$$

which is maximized for $k^* = \sqrt{z(m-z)}$. Hence,

$$\frac{W}{OPT} \leq \frac{\sqrt{z(m-z)}(m-2z)}{z(m-2z) + (\sqrt{z(m-z)} - z)^2} = \frac{m-2z}{2\sqrt{z(m-z)} - 2z}.$$

By setting $z = \lambda m$, where $0 < \lambda \leq \frac{1}{2}$, we get

$$\frac{W}{OPT} \leq \frac{m - 2\lambda m}{2\sqrt{\lambda m(m-\lambda m)} - 2\lambda m} = \frac{1-2\lambda}{2\sqrt{\lambda(1-\lambda)} - 2\lambda} = \rho.$$

Therefore, in order to achieve a ρ approximation ratio we choose $\lambda = \dfrac{1}{(2\rho-1)^2+1}$, that is $z = \dfrac{m}{(2\rho-1)^2+1}$.

The algorithm needs polynomial space, since Lines 3–14 are executed independently for each combination of weights. As the FEASIBLE-EC(w) problem is polynomial for trees, the complexity of the algorithm is, within a polynomial factor, $O(T(m))$, where $T(m)$ is the number of combinations generated. For this number it holds that

$$\begin{aligned} T(m) &\leq \sum_{i=1}^{z}\binom{m}{i} + \sum_{i=m-z}^{m}\binom{m}{i} = 2\sum_{i=1}^{z}\binom{m}{i} \leq 2z\binom{m}{z} \leq m\binom{m}{\lambda m} \\ &\leq m\left(\left(\frac{1}{\lambda}\right)^{\lambda}\left(\frac{1}{1-\lambda}\right)^{1-\lambda}\right)^{m} = m\left(\frac{(2\rho-1)^2+1}{(2\rho-1)^{2(2\rho-1)^2/((2\rho-1)^2+1)}}\right)^{m} \\ &= m\cdot g(\rho)^m. \end{aligned}$$

Hence, the complexity of ALGORITHM TREES-$E(z)$ becomes $O^*(g(\rho)^m)$, where $g(\rho) = \dfrac{(2\rho-1)^2+1}{(2\rho-1)^{2(2\rho-1)^2/((2\rho-1)^2+1)}}$. □

Note that for $z = \lfloor \frac{m}{2} \rfloor$ ALGORITHM TREES-$E(z)$ computes an optimal solution for the EC(w) problem on trees in $O^*(2^m)$ time and polynomial space.

In [5], an algorithm has been presented with running time and space $O^*(2^n)$, which, for any k, computes the number of all proper k-vertex-colorings of a graph, and moreover enumerates these colorings. This algorithm can be used to find an optimal solution for the VC(w) problem on a general graph, by running it for $1 \leq k \leq n$. Considering the line graph $L(G)$ of the input graph G of the EC(w) problem, we derive that the EC(w) problem on general graphs can be optimally solved with running time and space $O^*(2^m)$.

Next proposition shows that if $\Delta = o(m)$ then the running time of ALGORITHM TREES-$E(z)$ for computing an optimal solution is improved.

Proposition 7. *If $\Delta = o(m)$, then* ALGORITHM TREES-$E(z)$ *requires subexponential running time $2^{o(m)}$ in order to compute an exact solution for trees.*

Proof. By Proposition 2, the number k^* of colors in any optimal solution to the EC(w) problem is at most $2\Delta - 1$. Thus, the number of combinations of weights needed to be generated by the algorithm becomes

$$\begin{aligned} T(m) &\leq \binom{m}{2\Delta} \leq \frac{m^m}{(2\Delta)^{2\Delta}(m-2\Delta)^{m-2\Delta}} \\ &\leq 2^{m\log m - 2\Delta\log(2\Delta) - (m-2\Delta)\log(m-2\Delta)} \\ &\leq 2^{m\log(1+2\Delta/(m-2\Delta)) + 2\Delta\log(m/2\Delta-1)} \end{aligned}$$

Notice first that $2\Delta/(m-2\Delta)$ tends to 0 for $m \to \infty$, since $\Delta = o(m)$, and thus $m\log\left(1+\dfrac{2\Delta}{(m-2\Delta)}\right) \to 0$. Moreover, note that $2\Delta\log\left(\dfrac{m}{2\Delta}-1\right) = o(m)$, since $\dfrac{2\Delta\log\left(\frac{m}{2\Delta}-1\right)}{m}$ tends to 0 as m increases. Combining the two observations

above, we get that $T(m) = 2^{o(m)}$ and, hence, the running time of Algorithm Trees-$E\left(\frac{m}{2}\right)$ is $O^*\left(2^{o(m)}\right)$. □

Notice that Algorithm Trees-$E\left(\left\lfloor\frac{m}{2}\right\rfloor\right)$ and Algorithm Trees-$\Delta(2\Delta-1)$ coincide and both return an optimal solution to the EC(w) problem on trees. Thus the last proposition holds for both algorithms.

5.4 Polynomial Time Approximation Scheme

In this section we combine the idea of iteratively splitting the input graph with Algorithm Trees in order to derive a PTAS for the EC(w) problem on trees.

In fact, our algorithm splits a tree $G = (V, E)$, into subgraphs $G[E_{1,j}]$ and $G[E_{j+1,m}]$ induced by the j heaviest and the $n - j$ lightest edges of G, respectively (by convention, we consider $G[E_{1,0}]$ as an empty subgraph). The algorithm depends on a parameter p such that all the edges of G of weights $w_1^*, w_2^*, \ldots, w_{p-1}^*$ are in a subgraph $G[E_{1,j}]$. We obtain a solution for the whole graph by concatenating an optimal solution of at most $p - 1$ colors for $G_{1,j}$, if there is one, and the solution obtained by Algorithm Trees for $G[E_{j+1,m}]$.

Algorithm Trees-Scheme(p)
1: Let $\langle E\rangle = \langle e_1, e_2, \ldots e_m\rangle$;
2: **for** $j = 0$ to m **do**
3: Split the graph into two edge induced subgraphs:
- $G[E_{1,j}]$ induced by edges $e_1, e_2, \ldots, e_j$
- $G[E_{j+1,n}]$ induced by edges $e_{j+1}, e_{j+2}, \ldots, e_m$
4: **if** there is a solution for $G[E_{1,j}]$ with at most $p - 1$ colors **then**
5: Find an optimal solution for $G[E_{1,j}]$ with at most $p - 1$ colors;
6: Run Algorithm Trees for $G[E_{j+1,m}]$;
7: Concatenate the two solutions found in Lines 5 and 6;
8: **end if**
9: **end for**
10: Return the best solution found;

Theorem 18. Algorithm Trees-Scheme(p) *is a PTAS for the* EC(w) *problem on trees.*

Proof. Consider the iteration j, $j \leq m$, of the algorithm where the weight of the heaviest edge in $G[E_{j+1,m}]$ equals to the weight of the i-th color of an optimal solution, i.e. $w(e_{j+1}) = w_i^*$, $1 \leq i \leq p$.

The edges of $G[E_{1,j}]$ are a subset of those appeared in the $i - 1$ heaviest colors of the optimal solution. Thus, an optimal solution for $G[E_{1,j}]$ is of weight

$$OPT_{1,j} \leq w_1^* + w_2^* + \ldots + w_{i-1}^*.$$

The edges of $G[E_{j+1,m}]$ are a superset of those appeared in the $k^*-(i-1)$ lightest colors of the optimal solution. The extra edges of $G[E_{j+1,m}]$ are of weight at most w_i^* and appear in an optimal solution into at most $i-1$ colors. Thus, an optimal solution for $G[E_{j+1,m}]$ is of weight

$$OPT_{j+1,m} \leq w_i^* + w_{i+1}^* + \ldots + w_{k^*}^* + (i-1) \cdot w_i^* = i \cdot w_i^* + w_{i+1}^* + \ldots + w_{k^*}^*.$$

By Lemma 4, Algorithm Trees returns a solution for $G[E_{j+1,m}]$ of weight

$$\begin{aligned} W_{j+1,m} &\leq OPT_{j+1,m} + w_i^* - w_\Delta^* \\ &\leq i \cdot w_i^* + w_{i+1}^* + \ldots + w_{k^*}^* + w_i^* \\ &\leq (i+1) \cdot w_i^* + w_{i+1}^* + \ldots + w_{k^*}^*. \end{aligned}$$

Therefore, the solution found in this iteration j for the whole graph G is of weight

$$W_i = OPT_{1,j} + W_{j+1,m} \leq w_1^* + w_2^* + \ldots + w_{i-1}^* + (i+1) \cdot w_i^* + w_{i+1}^* + \ldots + w_{k^*}^*.$$

In all the iterations of the algorithm we obtain p such inequalities for W. By multiplying the i-th, $1 \leq i \leq p$, inequality by $\dfrac{1}{i \cdot (H_p+1)}$ and adding up all of them, we have $\left(\displaystyle\sum_{i=1}^{p} \frac{1}{i \cdot (H_p+1)} \right) \cdot W \leq OPT$, that is $\dfrac{W}{OPT} \leq \dfrac{H_p+1}{H_p} = 1 + \dfrac{1}{H_p}$.

Algorithm Trees-Scheme(p) iterates $|E|$ times. In each iteration: (i) an optimal solution, if any, with at most $p-1$ colors for $G[E_{1,j}]$ is found using Theorem 2(i), in $O(|E|^{p-1} \cdot |E| \cdot \Delta^{3.5})$ time, and (ii) Algorithm Trees of complexity $O(|V| \cdot \Delta \cdot \log \Delta)$ is called for $G[E_{j+1,m}]$. Choosing p such that $\epsilon = \frac{1}{H_p}$, we get $p = O(2^{\frac{1}{\epsilon}})$. Consequently, we have a PTAS for the EC(w) problem on trees, that is an approximation ratio of $1 + \frac{1}{H_p} = 1 + \epsilon$ within time $O\left(|E| \left(|V| \cdot \Delta \cdot \log \Delta + |E|^p \cdot \Delta^{3.5}\right)\right)$ which is exponential to $\frac{1}{\epsilon}$. □

Chapter 6

Bounded Max-Edge-Coloring

In this chapter we deal with the complexity and approximability of the EC(w, b) problem. First, we present an approximation algorithm for general and bipartite graphs. Then, we prove that the problem is NP-complete for trees and we give a 2-approximation algorithm for this case. Note that this is the first complexity result for any max-coloring problem on trees.

6.1 General and bipartite graphs

Our approximation algorithm for general and bipartite graphs is based on tight bounds on the number of colors in a solution to the EC(w, b) problem. In fact, our bounds apply to any *nice* solution $\langle \mathcal{C} \rangle = \langle C_1, C_2, \ldots, C_k \rangle$ to the EC(w, b) problem. We call such a solution nice if each color C_i, $1 \leq i \leq k$, is of cardinality $|C_i| = b$ or C_i is maximal in the subgraph induced by the edges $\bigcup_{j=i}^{k} C_j$. It is easy to see that any solution to the EC(w, b) problem can be transformed into a nice one of the same total weight.

Proposition 8. *For the number, k, of colors in any nice solution to the* EC(w, b) *problem it holds that:*

$$\max\{\Delta, \left\lceil \frac{|E|}{b} \right\rceil\} \leq k \leq \begin{cases} \left\lceil \frac{|E|}{b} \right\rceil - \left\lceil \frac{\Delta^2}{2b} \right\rceil + (2\Delta - 1), & \textit{for general graphs} \\ \left\lceil \frac{|E|}{b} \right\rceil - \left\lceil \frac{\Delta^2}{b} \right\rceil + (2\Delta - 1), & \textit{for bipartite graphs} \end{cases}$$

Proof. The lower bounds follow trivially. For the upper bounds, let $\langle \mathcal{C} \rangle = \langle C_1, C_2, \ldots, C_k \rangle$ be a nice solution, $e = (u, v)$ be an edge in the last color C_k, and E_u and E_v be the sets of edges adjacent to vertices u and v, respectively. By the niceness of the solution $\mathcal{C}$ it follows that edge e does not appear in any color C_i, $1 \leq i \leq k-1$, because $|C_i| = b$ or C_i contains at least one edge in E_u or E_v. Let $W, X, Y \subseteq \{C_1, C_2, \ldots, C_{k-1}\}$ such that $W = \{C_i : |C_i| = b\}$, $X = \{C_i : |C_i| < b$ and C_i contains an edge $e \in E_u\}$ and $Y = \{C_i : |C_i| < b$ and C_i contains an edge $e \in E_v\}$. Let E_1 be the set of edges in the colors in W and $E_2 = E \setminus E_1$ be the set of edges in the colors in $X \cup Y \cup \{C_k\}$. Then, $k = \frac{|E_1|}{b} + x + y + 1$, where $x = |X|$ and $y = |Y|$.

Assume, w.l.o.g., that $X_1Y_1X_2Y_2\ldots X_lY_l$ is the order of colors in the nice solution $\langle\mathcal{C}\rangle$, where $X_i \subseteq X$, $Y_i \subseteq Y$, $1 \leq i \leq l$, and X_1 is possibly empty. Let $x_i = |X_i|$ and $y_i = |Y_i|$, $1 \leq i \leq l$.

For general graphs, consider a color $C \in X_i$ and let $S_i = \bigcup_{j=i}^{l} Y_j$ and $s_i = \sum_{j=i}^{l} y_j$. The edge $(u,z) \in C \cap E_u$ prevents at most one edge $(v,z) \in S_i \cap E_v$ from being into C. Moreover, each other edge $(p,q) \in C$ prevents at most two edges $(v,p),(v,q) \in S_i \cap E_v$ from being into C. As the colors in S_i contain exactly s_i edges from E_v and all of them are prevented from being into color C, it follows that $|C| \geq \lceil \frac{s_i-1}{2} \rceil + 1$. Therefore, there exist at least $x_i \cdot \lceil \frac{s_i-1}{2} \rceil + x_i$ edges in X_i. In a similar way, by considering a color $C \in Y_i$ there exist at least $y_i \cdot \lceil \frac{t_i-1}{2} \rceil + y_i$ edges in Y_i, where $t_i = \sum_{j=i+1}^{l} x_j$. Summing up these bounds, and taking into account that $y_l - 1 \geq 0$ (since Y_l is not empty), it follows that

$$\begin{aligned}
|E_2| &\geq \sum_{i=1}^{l}(x_i \cdot \left\lceil \frac{s_i-1}{2} \right\rceil + x_i) + \sum_{i=1}^{l}(y_i \cdot \left\lceil \frac{t_i-1}{2} \right\rceil + y_i) \\
&= \frac{x(y-1) - (y_1 + y_2 + \ldots + y_{l-1})}{2} + x + y + 1 \\
&\geq \frac{x(y-1) - (y_1 + y_2 + \ldots + y_{l-1} + y_l - 1)}{2} + x + y + 1 \\
&= \frac{(x-1)(y-1)}{2} + x + y + 1 \geq \frac{(x+1)(y+1)}{2} + 1 \geq \left\lceil \frac{(x+1)(y+1)}{2} \right\rceil.
\end{aligned}$$

Therefore, $k = \frac{|E \setminus E_2|}{b} + x + y + 1 \leq \left\lceil \frac{|E|}{b} \right\rceil - \left\lceil \frac{(x+1)(y+1)}{2b} \right\rceil + x + y + 1$ If $\Delta \leq 2b$ then this quantity is maximized when $x = y = \Delta - 1$ and hence $k \leq \left\lceil \frac{|E|}{b} \right\rceil - \left\lceil \frac{\Delta^2}{2b} \right\rceil + (2\Delta - 1)$. If $\Delta > 2b$ then the above quantity is maximized when $x = \Delta - 1$ and $y = 0$ and hence $k \leq \left\lceil \frac{|E|}{b} \right\rceil - \lceil \frac{\Delta}{2b} \rceil + \Delta \leq \left\lceil \frac{|E|}{b} \right\rceil - \left\lceil \frac{\Delta^2}{2b} \right\rceil + (2\Delta - 1)$.

For bipartite graphs, the proof is similar. The structure of a bipartite graph allows a tighter bound on the number of edges in the colors in X_i and Y_i. Consider, again, a color $C \in X_i$. For the edge $(u,z) \in C \cap E_u$, there is no edge $(v,z) \in S_i \cap E_v$, while each other edge $(p,q) \in C$ prevents at most one edge (v,p) or (v,q) in $S_i \cap E_v$ from being into C. Thus, $|C| \geq s_i + 1$ and, hence, there exist at least $x_i(s_i+1)$ edges in X_i. Similarly there exist at least $y_i(t_i+1)$ edges in Y_i. The rest of the proof is along the same lines, but using these bounds. □

Our algorithm for general and bipartite graphs adapts the greedy 2-approximation algorithm presented in [46] for the EC(w) problem to the EC(w,b) problem.

Algorithm Greedy

1: Let $\langle E\rangle = \langle e_1, e_2, \ldots, e_m\rangle$;
2: **for** $j = 1$ to m **do**
3: Insert edge e_j in the first color of cardinality less than b which does not contain other edges adjacent to e_j;
4: **end for**

The solution derived by ALGORITHM GREEDY is a nice one, since it is constructed in a first-fit manner. The analysis of this algorithm given in the next lemma is based on Proposition 8.

Lemma 5. ALGORITHM GREEDY *achieves approximation ratios of* $3-\frac{2}{\sqrt{2b}}$ *on general graphs, and* $3-\frac{2}{\sqrt{b}}$ *on bipartite graphs, for the* EC(w,b) *problem.*

Proof. Let $\langle\mathcal{C}\rangle = \langle C_1, C_2, \ldots, C_k\rangle$, be a solution derived by ALGORITHM GREEDY. Consider the color C_i and let e_j be the first edge inserted in C_i, i.e. $w_i = w(e_j)$. Let $E_i = \{e_1, e_2, \ldots, e_j\}$, G_i be the subgraph of G induced by the edges in E_i, and Δ_i be the maximum degree of G_i.

As the solution $\langle\mathcal{C}\rangle$ is a nice one and an optimal solution can be also considered to be nice, by Proposition 8, for general graphs, it follows that *(i)* $i \leq \left\lceil \frac{|E_i|}{b} \right\rceil - \left\lceil \frac{\Delta_i^2}{2b} \right\rceil + (2\Delta_i - 1)$, and *(ii)* in an optimal solution the edges of G_i appear in at least $i^* \geq \max\{\Delta_i, \left\lceil \frac{|E_i|}{b} \right\rceil\}$ colors, each one of weight at least w_i. Therefore, $\frac{i}{i^*} \leq \frac{\left\lceil \frac{|E_i|}{b} \right\rceil - \left\lceil \frac{\Delta_i^2}{2b} \right\rceil + (2\Delta_i - 1)}{\max\left\{\Delta_i, \left\lceil \frac{|E_i|}{b} \right\rceil\right\}}$. By distinguish between $\Delta_i \geq \left\lceil \frac{|E_i|}{b} \right\rceil$ and $\Delta_i < \left\lceil \frac{|E_i|}{b} \right\rceil$ it follows that in either case $\frac{i}{i^*} \leq 3 - \frac{\Delta_i^2 + 2b}{2b\Delta_i}$. This bound is maximized when $\Delta_i = \sqrt{2b}$, that is $\frac{i}{i^*} \leq 3 - \frac{2}{\sqrt{2b}}$. Thus, $w_i \leq w_{i^*}^* \leq w^*_{\left\lceil i/\left(3-\frac{2}{\sqrt{2b}}\right)\right\rceil}$. Summing up these inequalities for all i's, $1 \leq i \leq k$, we obtain the $\left(3 - \frac{2}{\sqrt{2b}}\right)$ ratio for general graphs.

A similar analysis yields the $\left(3 - \frac{2}{\sqrt{b}}\right)$ ratio for bipartite graphs. □

We present here an example for which the algorithm performs a ratio of $3 - \frac{2}{\sqrt{2b}}$ for general graphs. Consider the general graph shown in Figure 6.1(a), where $1 >> \epsilon$, and $b = 7$. The weight of the optimal solution shown in Figure 6.1(b) is $3 + 3\epsilon$. The weight of the solution obtained by ALGORITHM GREEDY, shown in Figure 6.1(c), is $7 - \epsilon$. Thus, the ratio for this instance is $\frac{7-\epsilon}{3+3\epsilon} \simeq \frac{7}{3} \simeq 3 - \frac{2}{\sqrt{14}}$.

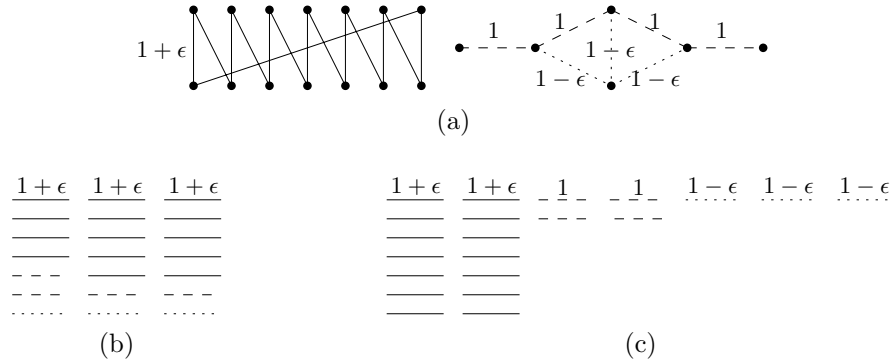

Figure 6.1: (a) An instance of the EC(w,b) problem on general graphs, $1 >> \epsilon$, $b = 7$. (b) An optimal solution. (c) The solution obtained by ALGORITHM GREEDY.

Consider, now, the bipartite graph shown in Figure 6.2(a), where $1 >> \epsilon$, and $b = 9$. The optimal solution is of weight $3 + 3\epsilon$ (Figure 6.2(a)), while the weight of the solution obtained by ALGORITHM GREEDY is $7 - \epsilon$ (Figure 6.2(c)). Hence, the ratio for this instance is $\frac{7-\epsilon}{3+3\epsilon} \simeq \frac{7}{3} = 3 - \frac{2}{\sqrt{9}} = 3 - \frac{2}{\sqrt{b}}$.

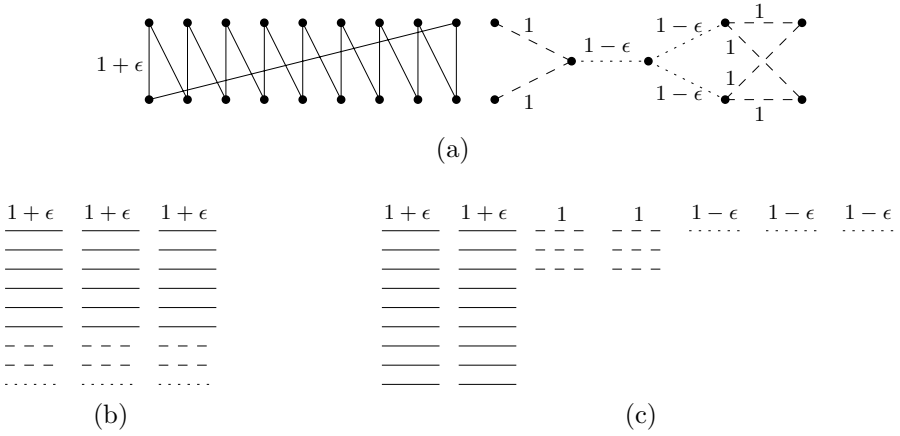

Figure 6.2: (a) An instance of the EC(w, b) problem on bipartite graphs, $1 >> \epsilon$, $b = 9$. (b) An optimal solution. (c) The solution obtained by ALGORITHM GREEDY.

Combining Lemma 5 and Theorems 3, 4 and 5, it follows that

Theorem 19. *The* EC(w, b) *problem can be approximated with a ratio of* $\min\{3 - 2/\sqrt{2b}, H_b, \lceil b/2 \rceil\}$ *for general graphs and* $\min\{e, 3 - 2/\sqrt{b}, H_b, \lceil b/2 \rceil\}$ *for bipartite graphs.*

Note that, the H_b ratio outperforms the other only for $b = 3$ or 5, for general graphs, and $b = 3$, for bipartite graphs, and, hence, b can be considered as fixed. These ratios are shown in Table 6.1, for several values of b.

b	General graphs		Bipartite graphs	
3	1.833	H_b	1.833	H_b
4	2.000	$\lceil b/2 \rceil$	2.000	$3 - 2/\sqrt{b}$ or $\lceil b/2 \rceil$
5	2.283	H_b	2.106	$3 - 2/\sqrt{b}$
6	2.423	$3 - 2/\sqrt{2b}$	2.184	$3 - 2/\sqrt{b}$
...	...	$3 - 2/\sqrt{2b}$	...	$3 - 2/\sqrt{b}$
50	2.800	$3 - 2/\sqrt{2b}$	2.717	$3 - 2/\sqrt{b}$
51	2.802	$3 - 2/\sqrt{2b}$	2.718	e
...	...	$3 - 2/\sqrt{2b}$	...	e

Table 6.1: Approximation ratios for the EC(w, b) problem.

6.2 NP-completeness for trees

To prove that the EC(w, b) problem is NP-complete on trees, we will present a reduction from the bounded list vertex-coloring problem, VC(ϕ, b_i). By Theorem 1(iii), VC(ϕ, b_i) problem is NP-complete even for chains, $|\phi(u)| \leq 2$, for all $u \in V$, and $b_i \leq 5$, $1 \leq i \leq k$.

We prove first that the bounded list edge-coloring, EC(ϕ, b), problem is NP-complete even if the graph $G = (V, E)$ is a set of chains, $|\phi(e)| = 2$, for all $e \in E$, and $b = 5$. We denote this problem as EC(chains, $|\phi(e)| = 2,\ b = 5$).

Proposition 9. *The EC(chains, $|\phi(e)| = 2,\ b = 5$) problem is NP-complete.*

Proof. By Theorem 1(iii), the VC(chains, $|\phi(v)| \leq 2,\ b_i \leq 5$) problem is NP-complete. Given that the line-graph of a chain is also a chain, it follows that the EC(chains, $|\phi(e)| \leq 2,\ b_i \leq 5$) problem is also NP-complete. The latter problem can be easily reduced to the EC(chains, $|\phi(e)| \leq 2,\ b = 5$) problem, where $b_i = b = 5$ for all colors: for every color C_i with $b_i < 5$, add $5 - b_i$ independent edges with just C_i in their lists. This last problem reduces to the EC(chains, $|\phi(e)| = 2,\ b = 5$) problem, where $|\phi(e)| = 2$ for all edges. This can be done by transforming an instance of EC(chains, $|\phi(v)| \leq 2, b = 5$) as following: *(i)* add two new colors C_{k+1} and C_{k+2}, both with cardinality bound $b = 5$, *(ii)* add color C_{k+1} to the list of every edge e with $|\phi(e)| = 1$, *(iii)* add ten independent edges and put in their lists both colors C_{k+1} and C_{k+2}. □

Theorem 20. *The* EC(w, b) *problem on trees is NP-complete.*

Proof. Our reduction is from EC(chains, $|\phi(e)| = 2,\ b = 5$) problem. We construct an instance of the EC(w, b) problem on a forest $G' = (V', E')$ as follows.

We replace every edge $e = (u, v) \in E$ with a chain of three edges: $e_1 = (u, u')$, $e_2 = (u', v')$ and $e_3 = (v', v)$, where $w(e_1) = w(e_2) = w(e_3) = 1$. Moreover, we create $k - |\phi(e)| = k - 2$ stars of $k - 1$ edges each. We add edges (u', s_t), $1 \leq t \leq k - 2$, between u' and the central vertex s_t of each of these $k - 2$ stars; thus every star has now exactly k edges. Let $\phi(e) = \{C_i, C_j\}$. The $k - 2$ edges (u', s_t) take different

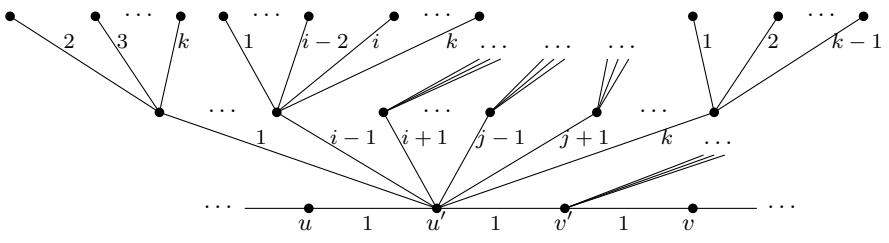

Figure 6.3: The gadget for an edge $e = (u, v)$ with $\phi(e) = \{C_i, C_j\}$.

weights in $\{1, 2, \ldots, k\} \setminus \{i, j\}$. Let q be the weight taken by an edge (u', s_t). The remaining $k - 1$ edges of the star t take different weights in $\{1, 2, \ldots, k\} \setminus \{q\}$. In the same way, we add $k - 2$ stars connected to v'. In Figure 6.3, is shown the u''s

part of this edge-gadget for $e = (u, v)$. For every edge e of G, we add $2(k-2)$ stars and $2(k-2)k+2$ edges.

To complete our construction we define f_i to be the frequency of color C_i in the lists of all edges and $F = \max\{f_i | 1 \leq i \leq k\}$. For every color C_i we add $F - f_i$ disconnected copies of the color-gadget shown in Figure 6.4. Such a gadget consists of an edge $e = (x, y)$ and $k-1$ stars with $k-1$ edges each. There are also edges between one of the endpoints of e, say y, and the central vertices of all stars; thus every star has now exactly k edges. The edge e takes weight i and the edges in the stars of such a color-gadget take weights similarly with those in the stars of an edge-gadget. For a color C_i we add $(F - f_i)(k-1)$ stars and $(F - f_i)(k-1)k + (F - f_i)$ edges.

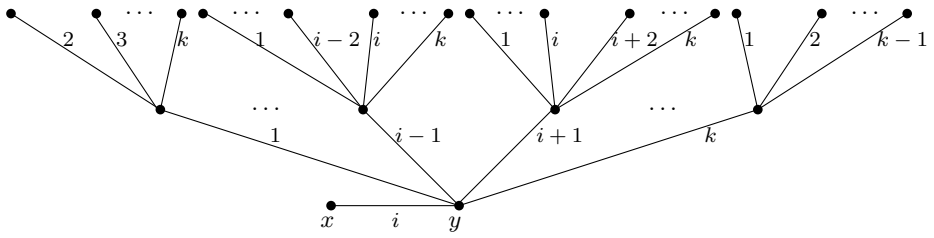

Figure 6.4: A gadget for the color C_i.

The number of stars in the forest G' we have constructed is $2|E|(k-2)+\sum_{i=1}^{k}(F-f_i)(k-1) = k(k-1)F-2|E|$, since $\sum_{i=1}^{k} f_i = 2|E|$. By setting $b' = k(k-1)F-2|E|+5+F$, we prove that: "*There is a k-coloring for* $\mathrm{EC}(\phi, b)$ *(chains,* $|\phi(e)| = 2,\ b = 5$*), if and only if, G' has a bounded max-edge-coloring of total weight* $\sum_{i=1}^{k} i$ *such that every color is used at most b' times*".

Consider, first, a solution $\mathcal{C}$ to the $\mathrm{EC}(\phi, b)$ problem. We construct a solution $\mathcal{C}'$ for the $\mathrm{EC}(w, b)$ problem as following. Let $e = (u, v) \in E$ be an edge with $\phi(e) = \{C_i, C_j\}$, which, w.l.o.g., appears in the color C_i of $\mathcal{C}$. Put the edges e_1 and e_3 of the edge-gadget for e in color C'_i, while the edge e_2 in color C'_j. After doing this for all edges in E, each color C'_i contains at most $2 \cdot 5 + 1 \cdot (f_i - 5) = f_i + 5$ edges. Next, put the edges with weight i, $1 \leq i \leq k$, from the $k(k-1)F - 2|E|$ stars into C'_i. Each color C'_i in $\mathcal{C}'$ constructed so far contains at most $k(k-1)F-2|E|+f_i+5 = b'-(F-f_i)$ edges and, by the construction of G', $\mathcal{C}'$ is a proper coloring. In the $F - f_i$ color-gadgets for C_i there are $F - f_i$ remaining (x, y) edges of weight i, which can still be inserted into color C'_i. Thus, we get a solution for the $\mathrm{EC}(w, b)$ problem of k colors, each one of at most b' edges, and total weight $\sum_{i=1}^{k} i$.

Conversely, consider a solution $\mathcal{C}'$ to the $\mathrm{EC}(w, b)$ problem. $\mathcal{C}'$ consists of exactly k colors of weights $1, 2, \ldots, k$, since each star in G' has k edges and each edge has a different weight in the range $\{1, 2, \ldots, k\}$. Thus, all edges of the same weight, say i, should belong in the same color C'_i of $\mathcal{C}'$. Therefore, C'_i contains one edge from each one of the $k(k-1)F - 2|E|$ stars as well as the $F - f_i$ remaining (x, y) edges of the color-gadgets having weight i. Consider, now, the edges of G' corresponding

to the edges e_1, e_2 and e_3 of the edge-gadget for an edge e with $\phi(e) = \{C_i, C_j\}$. By the construction of G' and the choice of edge weights, the edges e_1, e_2 and e_3 should appear into colors C'_i and C'_j. Thus, edges e_1 and e_3 should appear, w.l.o.g., into color C'_i, while e_2 into color C'_j. Therefore, the edge $e \in E$ can be colored by color $C_i \in \phi(e)$. Finally, a color C'_i contains at most 5 edges of type e_1 (or e_3), corresponding to at most 5 edges of E; otherwise $|C'_i| \geq k(k-1)F - 2|E| + (F - f_i) + (2 \cdot 6 + 1 \cdot (f_i - 6)) > b'$, a contradiction.

To complete our proof for the EC(w, b) problem on trees, let p be the number of trees in G'. We add a set of $p-1$ edges of weight $\epsilon < 1$ to transform the forest G' into a single tree T. This can be done as every tree of G' has at least two vertices. By keeping the same bound b', it is easy to see that there is a solution for the EC(w, b) problem on G' of weight $\sum_{i=1}^{k} i$, if and only if, there is a solution for the EC(w, b) problem on T whose weight is equal to $\sum_{i=1}^{k} i + \left\lceil \frac{p-1}{b'} \right\rceil \epsilon$. □

6.3 A 2-approximation algorithm for trees

In Section 5.2 a 2-approximation algorithm for the EC(w) problem on trees has been presented, which is also exploited to derive a ratio of 3/2 for that problem. This algorithm yields a solution of Δ colors, $\mathcal{M} = \{M_1, M_2, \ldots, M_\Delta\}$. Starting from this solution we obtain a solution to the EC(w, b) problem by finding the ordered b-partition of each color in $\mathcal{M}$. For the sake of completeness we give below the whole algorithm.

Algorithm Convert

1: Let T_r be the tree rooted in an arbitrary vertex r;
2: **for** each vertex v in pre-order traversal of T_r **do**
3: Let $\langle E_v \rangle = \langle e_1, e_2, \ldots, e_{d(v)} \rangle$ be the edges adjacent to v, and (v, p) be the edge from v, $v \not\equiv r$, to its parent;
4: Using ordering $\langle E_v \rangle$, insert each edge in E_v, but (v, p), into the first color which does not contain an edge in E_v;
5: **end for**
6: Let $\mathcal{M} = \{M_1, M_2, \ldots, M_\Delta\}$ be the colors constructed;
7: **for** $i = 1$ to Δ **do**
8: Let $\mathcal{P}_{M_i} = \{M_1^i, M_2^i, \ldots, M_{k_i}^i\}$ be the ordered b-partition of $\langle M_i \rangle$;
9: **end for**
10: Return a solution $\langle \mathcal{C} \rangle = \langle C_1, C_2, \ldots, C_k \rangle$, $\mathcal{C} = \bigcup_{i=1}^{\Delta} \mathcal{P}_{M_i}$;

Theorem 21. Algorithm Convert *is a* 2*-approximation one for the* EC(w, b) *problem on trees.*

Proof. Consider the color C_j in the solution $\langle \mathcal{C} \rangle$ and let e be the heaviest edge in C_j, i.e., $w(e) = w_j$. Let $X \subseteq \mathcal{C}_j = \{C_1, C_2, \ldots, C_{j-1}\}$ such that each color $C_p \in X$ has *(i)* $|C_p| = b$, and *(ii)* all edges of weight at least $w(e)$. Let also $Y = \mathcal{C}_j \setminus X$,

$|X| = x$ and $|Y| = y$. Clearly, $x + y = j - 1$. Let j^* be the number of colors in an optimal solution of weight at least $w(e)$, that is $w_j \leq w^*_{j^*}$.

There are at least $x \cdot b + y + 1$ edges of weight at least $w(e)$. These edges in an optimal solution appear in at least $\left\lceil \frac{x \cdot b + y + 1}{b} \right\rceil \geq x + 1$ colors, that is, $j^* \geq x + 1$.

We show, next, that all colors in $Y \cup \{C_j\}$ come from $y+1$ different colors in $\mathcal{M}$. Assume that two of these colors, C_q and C_r, come from the ordered b-partition of the same color $M_t \in \mathcal{M}$. Assume, w.l.o.g., that $w_q \geq w_r$, and let f be the heaviest edge in C_r. Note that C_r may coincide with C_j, while C_q cannot. As $C_q \in Y$, it follows that $|C_q| = b$ and there is an edge $f' \in C_q$ with $w(f') < w(e) \leq w(f)$, a contradiction to the definition of the ordered b-partition of M_t. Therefore, C_j comes from a color $M_i \in \mathcal{M}$, $i \geq y + 1$, that is $e \in M_i$. By the construction of the coloring $\mathcal{M}$, there are at least $i - 1$ edges, adjacent to each other, of weight at least $w(e)$ (i.e., $i - 2$ of them adjacent to e and e itself). These $i - 1$ edges appear in different colors in an optimal solution, that is, $j^* \geq y$.

Combining the two lower bounds for j^* and taking into account that $x+y = j-1$ we get $j^* \geq \left\lceil \frac{j}{2} \right\rceil$. Therefore, $w_j = w^*_{j^*} \leq w^*_{\lceil \frac{j}{2} \rceil}$ and summing up the weights w_j of all colors in $\mathcal{C}$ we get $W = \sum_{j=1}^{k} w_j \leq 2 \sum_{j=1}^{\lceil k/2 \rceil} w^*_j \leq 2 \sum_{j=1}^{k^*} w^*_j \leq 2OPT$, since $k^* \geq \lceil \frac{k}{2} \rceil$. A tight example for this algorithm, is given in Section 5.2, as for large values of b the $\mathrm{EC}(w, b)$ coincides with the $\mathrm{EC}(w)$ problem. By a careful analysis, the complexity of both Lines 2–5 and 7–9 of the algorithm is $O(|V| \log |V|)$. □

Chapter 7

Bounded Max-Vertex-Coloring

In this chapter we first present a simple 2-approximation algorithm for the VC(w, b) problem on bipartite graphs. The unweighted variant of this algorithm gives a $\frac{4}{3}$ approximation ratio for the VC(b) problem on bipartite graphs, which closes the approximability question for this case. Then, we give a generic scheme which becomes a $\frac{17}{11}$-approximation algorithm for bipartite graphs, a PTAS for bipartite graphs and fixed b, as well as a PTAS for trees. Recall also that by Theorem 3 there is an H_b approximation ratio for general graphs, if b is fixed.

7.1 A simple split algorithm

Let $G = (U \cup V, E)$, $|U \cup V| = n$, be a vertex weighted bipartite graph. Our first algorithm colors the vertices of each class of G separately, by finding the ordered b-partitions of classes U and V. For the minimum number of colors k^* it holds that $k^* \geq \left\lceil \frac{|U|+|V|}{b} \right\rceil$ and, therefore, $k = \left\lceil \frac{|U|}{b} \right\rceil + \left\lceil \frac{|V|}{b} \right\rceil \leq \left\lceil \frac{|U|+|V|}{b} \right\rceil + 1 \leq k^* + 1$.

Algorithm Split

1: Let $\mathcal{P}_U = \{U_1, U_2, \ldots, U_{k_U}\}$ be the ordered b-partition of U;
2: Let $\mathcal{P}_V = \{V_1, V_2, \ldots, V_{k_V}\}$ be the ordered b-partition of V;
3: Return the coloring $\mathcal{C} = \mathcal{P}_U \cup \mathcal{P}_V$;

Theorem 22. Algorithm Split *returns a solution of weight* $W \leq 2 \cdot w_1^* + w_2^* + \ldots + w_{k^*}^* \leq 2 \cdot OPT$ *for the* VC(w, b) *problem in bipartite graphs.*

Proof. Let $\langle \mathcal{C} \rangle = \langle C_1, C_2, \ldots, C_k \rangle$ be the colors constructed by Algorithm Split, that is $w_1 \geq w_2 \geq \ldots \geq w_k$. Assume, w.l.o.g., that U_x, $1 \leq x \leq k_U$, is the $i - th$ color in $\langle \mathcal{C} \rangle$. Let also u be the heaviest vertex of U_x, that is $w(u) = w_i$.

The ordered b-partition of U and V implies that the colors that appear before U_x in $\langle \mathcal{C} \rangle$ are the colors $U_1, U_2, \ldots, U_{x-1}$ and $V_1, V_2, \ldots, V_y$, $y = i - x$. The colors $U_1, U_2, \ldots, U_{x-1}$ are all of cardinality b and their $(x-1) \cdot b$ vertices are all of weight at least $w(v)$. The colors $V_1, V_2, \ldots, V_{y-1}$, are also all of cardinality b and their $(y-1) \cdot b$ vertices are all of weight at least the weight of the heaviest vertex of color V_y which

is at least $w(v)$. Taking into account the vertex v itself it follows that there are in G at least $(x-1)\cdot b+[(y-1)\cdot b+1]+1=(x+y-2)\cdot b+2=(i-2)\cdot b+2$ vertices of weight at least $w(v)=w_i$. In an optimal solution, these vertices belong into at least $\left\lceil\frac{(i-2)\cdot b+2}{b}\right\rceil=(i-1)$ colors, each one of weight at least w_i. Hence, $w^*_{i-1}\geq w_i$, $2\leq i\leq k$. Clearly, $w_1=w^*_1$, since both are equal to the weight of the heaviest vertex of the graph, and as $k\leq k^*+1$, we obtain

$$\begin{aligned} W &= \sum_{i=1}^{k} w_i = w^*_1+\sum_{i=2}^{k} w_i \leq w^*_1+\sum_{i=1}^{k-1} w^*_i \\ &\leq w^*_1+\sum_{i=1}^{k^*} w^*_i = 2\cdot w^*_1+w^*_2+\ldots+w^*_{k^*}\leq 2\cdot OPT. \end{aligned}$$

□

The complexity of Algorithm Split is dominated by the sorting needed to obtain the ordered b-partitions of U and V in Lines 1 and 2, that is $O(|V|\cdot\log|V|)$.

Algorithm Split applies also to the VC(b) problem on bipartite graphs. Moreover, the absence of weights in the VC(b) problem allows a tight analysis with respect to the $\frac{4}{3}$ inapproximability bound.

Theorem 23. *There is a $\frac{4}{3}$-approximation algorithm for the* VC(b) *problem on bipartite graphs.*

Proof. Assume, first, that $|U|+|V|\geq 2b+1$. Then, $k^*\geq\lceil\frac{2b+1}{b}\rceil=3$ and, since $k\leq k^*+1$, we get $\frac{k}{k^*}\leq\frac{4}{3}$.

Assume, next, that $b<|U|+|V|\leq 2b$. In this case the optimal solution consists of two or three colors and it is polynomial to decide between them. In fact, by Theorem 1(ii), it is polynomial to decide if a bipartite graph can be colored with two colors even for the general VC(ϕ,b_i) problem.

Assume, finally, that $|U|+|V|\leq b$. Then, an optimal solution consists of either two colors (if $E\neq\emptyset$) or one color (if $E=\emptyset$). □

7.2 A generic scheme

In this section we combine the ideas used in Section 5.4 for the EC(w) problem with Algorithm Split to design a generic scheme for the VC(w,b) problem.

To obtain our scheme we split a bipartite graph $G=(U\cup V,E)$, $|U\cup V|=n$, into two subgraphs $G_{1,j}$ and $G_{j+1,n}$ induced by the j heaviest and the $n-j$ lightest vertices of G, respectively (by convention, we consider $G_{1,0}$ as an empty subgraph). Our scheme depends on a parameter p such that all the vertices of G of weights $w^*_1,w^*_2,\ldots,w^*_{p-1}$ are in a subgraph $G_{1,j}$. This is always possible for some $j\leq b(p-1)$, since each color of an optimal solution for G contains at most b vertices. In fact, for every j, $1\leq j\leq b(p-1)$, we obtain a solution for the whole graph by concatenating

an optimal solution of at most $p-1$ colors for $G_{1,j}$, if there is one, and the solution obtained by ALGORITHM SPLIT for $G_{j+1,n}$.

ALGORITHM SCHEME(p)

1: Let $\langle U \cup V \rangle = \langle u_1, u_2, \ldots u_n \rangle$;
2: **for** $j = 0$ to $b \cdot (p-1)$ **do**
3: Split the graph into two vertex induced subgraphs:
- $G_{1,j}$ induced by vertices $u_1, u_2, \ldots, u_j$
- $G_{j+1,n}$ induced by vertices $u_{j+1}, u_{j+2}, \ldots, u_n$
4: **if** there is a solution for $G_{1,j}$ with at most $p-1$ colors **then**
5: Find an optimal solution for $G_{1,j}$ with at most $p-1$ colors;
6: Run ALGORITHM SPLIT for $G_{j+1,n}$;
7: Concatenate the two solutions found in Lines 5 and 6;
8: **end if**
9: **end for**
10: Return the best solution found;

Lemma 6. ALGORITHM SCHEME(p) *achieves a* $\left(1 + \frac{1}{H_p}\right)$ *approximation ratio for the* VC(w, b) *problem.*

Proof. Consider the iteration j, $j \leq b \cdot (p-1)$, of the algorithm where the weight of the heaviest vertex in $G_{j+1,n}$ equals to the weight of the i-th color of an optimal solution, i.e. $w(u_{j+1}) = w_i^*$, $1 \leq i \leq p$.

The vertices of $G_{1,j}$ are a subset of those appeared in the $i-1$ heaviest colors of the optimal solution. Thus, an optimal solution for $G_{1,j}$ is of weight

$$OPT_{1,j} \leq w_1^* + w_2^* + \ldots + w_{i-1}^*.$$

The vertices of $G_{j+1,n}$ are a superset of those appeared in the $k^* - (i-1)$ lightest colors of the optimal solution. The extra vertices of $G_{j+1,n}$ are of weight at most w_i^* and appear in an optimal solution into at most $i-1$ colors. Thus, an optimal solution for $G_{j+1,n}$ is of weight $OPT_{j+1,n} \leq w_i^* + w_{i+1}^* + \ldots + w_{k^*}^* + (i-1) \cdot w_i^* = i \cdot w_i^* + w_{i+1}^* + \ldots + w_{k^*}^*$. By Theorem 22, ALGORITHM SPLIT returns a solution for $G_{j+1,n}$ of weight

$$W_{j+1,n} \leq (i+1) \cdot w_i^* + w_{i+1}^* + \ldots + w_{k^*}^*.$$

Therefore, the solution found in this iteration j for the whole graph G is of weight

$$W_i = OPT_{1,j} + W_{j+1,n} \leq w_1^* + w_2^* + \ldots + w_{i-1}^* + (i+1) \cdot w_i^* + w_{i+1}^* + \ldots + w_{k^*}^*.$$

In all the iterations of the algorithm we obtain p such inequalities for W. By multiplying the i-th, $1 \leq i \leq p$, inequality by $\dfrac{1}{i \cdot (H_p + 1)}$ and adding up all of them, we have $\left(\sum_{i=1}^{p} \dfrac{1}{i \cdot (H_p+1)}\right) \cdot W \leq OPT$, that is $\dfrac{W}{OPT} \leq \dfrac{H_p+1}{H_p} = 1 + \dfrac{1}{H_p}$. □

The complexity of the ALGORITHM SCHEME(p) is $O(bp(f(p)+|V|\log|V|))$, where $O(f(p))$ is the complexity of checking for the existence of solutions with at most $p-1$ colors for $G_{1,j}$ and finding an optimal one among them, while $O(|V|\log|V|)$ is the complexity of ALGORITHM SPLIT. ALGORITHM SCHEME(1) coincides with ALGORITHM SPLIT. ALGORITHM SCHEME(2) has simply to check if the $j \leq b$ vertices of $G_{1,j}$ are independent from each other and, therefore, it derives a $\frac{5}{3}$ approximate solution in polynomial time. ALGORITHM SCHEME(3) has to check and find, a two color solution for $G_{1,j}$, if any. This can be done in polynomial time by Theorem 2(ii). Thus, ALGORITHM SCHEME(3) is a polynomial time $\frac{17}{11}$-approximation algorithm for the VC(w,b) problem on bipartite graphs.

However, when $p \geq 4$ and b is a part of the instance, finding an optimal solution in $G_{1,j}$ is an NP-hard problem (even for the VC(b) problem [6]). Hence, we consider that b is a fixed constant. In this case, we run an exhaustive algorithm for finding, if any, an optimal solution in $G_{1,j}$ of at most $p-1$ colors. The complexity of such an exhaustive algorithm is $O\left((p-1)^{b\cdot(p-1)}\right)$ and thus, the complexity of ALGORITHM SCHEME(p), $p \geq 4$, becomes $O\left(bp^{bp}+n^2\log n\right)$, since bp is $O(|V|)$. Choosing p such that $\epsilon = \frac{1}{H_p}$, we get $p = O(2^{\frac{1}{\epsilon}})$. Consequently, for fixed b, we have a PTAS for the VC(w,b) problem on bipartite graphs, that is an approximation ratio of $1+\frac{1}{H_p} = 1+\epsilon$ within $O\left(b\left(2^{\frac{1}{\epsilon}}\right)^{b2^{\frac{1}{\epsilon}}}+|V|^2\log|V|\right)$ time.

Furthermore, in the particular case of trees, checking the existence of solutions with at most $p-1$ colors for $G_{1,j}$, and finding an optimal one among them, can be done, by Theorem 2(i), in polynomial time for fixed p. The complexity of our scheme in this case becomes $O\left(b2^{\frac{1}{\epsilon}}\left(|V|^{2^{\frac{1}{\epsilon}}}+|V|^2\log|V|\right)\right)$. Therefore, the following theorem holds.

Theorem 24. *For the* VC(w,b) *problem,* ALGORITHM SCHEME(p) *is a*
(i) polynomial time $\frac{17}{11}$*-approximation algorithm for bipartite graphs (for* $p=3$*),*
(ii) PTAS for bipartite graphs if b *is fixed,*
(iii) PTAS for trees.

Chapter 8

Conclusions and open questions

We presented complexity results and approximation algorithms for (bounded) max-coloring problems that arise in computer and communication systems, with respect to the class of the underlying graph. Our results concern general and bipartite graphs, trees and bi-valued graphs.

For the EC(w) problem, recall that it is known to be approximable within a factor of 2 (for any class of graphs) and inapproximable within a factor less than 7/6 (even for bipartite graphs), while its complexity for trees remains open. We narrow these gaps in the approximability of the problem by presenting an 1.74-approximation algorithm for bipartite graphs, and a PTAS for trees. Moreover, we prove that the problem is NP-complete for complete bi-valued graphs and we present an asymptotic 4/3-approximation algorithm for general bi-valued graphs.

The EC(w, b) problem is NP-complete for bipartite graphs as a generalization of the EC(w) problem. We prove that it is NP-complete even on trees, which is the first complexity result for all the (bounded) max-coloring problems on this class of graphs. On the other hand, we present approximation algorithms of ratio at most 3 for general and bipartite graphs and 2 for trees.

The VC(w, b) problem is non approximable by a constant factor approximation ratio on general graphs, as a generalization of the classical vertex coloring problem, as well as by a factor within 4/3 on bipartite graphs, a result that comes from the VC(b) problem. We present approximation algorithms of ratio $\lceil \frac{b}{2} \rceil$ for general graphs and 17/11 for bipartite graphs, and a PTAS for trees. Finally, for the VC(b) problem, we show that the known 4/3 lower bound for bipartite graphs is tight by providing a 4/3-approximation algorithm.

The recent research activity on the (bounded) max-coloring problems has led to a significant progress on understanding their complexity and approximability, though there is enough room for further research. A first category of open questions can be picked up by looking the entries of Table 1.1.

i. Further identification of the frontier between polynomial and NP-hard variants of the problems with respect to the underlying graph. For example, the complexity of all the VC(w), VC(w, b) and EC(w) problems on trees remains the most interesting open question.

ii. Decrease (or close) the gaps between the best known approximation ratios and inapproximability bounds, either by obtaining new improved approximation algorithms, or by strengthening existing inapproximability results. For example, a 2-approximation algorithm and a 4/3-inapproximability result are known for the EC(w) problem on general graphs of any maximum degree. The gap is narrower for bipartite graphs, as we presented an 1.74-approximation algorithm and a 7/6-inapproximability result is known. Moreover, the NP-completeness proof for the EC(w, b) problem on trees does not prevent the existence of a better ratio than the 2 approximation one we presented, or even a PTAS, such as for the VC(w), VC(w, b) and EC(w) problems. Hence, it would be interesting to decrease these approximability gaps by improving either the lower or the upper bounds.

A second category of interesting questions is related to the nature of the applications of the (bounded) max-coloring problems:

i. As we have already mentioned in Chapter 1 all the (bounded) max-coloring problems are equivalent to the parallel batch scheduling problem with incompatibilities between jobs. Within this context, a natural idea to decrease the weight of a solution to such a weighted coloring problem is to allow preemption, that is interrupt the execution of a job and complete it later. It is known that the preemptive EC(w) problem for bipartite graphs is equivalent to the preemptive open shop scheduling problem which can be solved optimally in polynomial time [48]. An interesting question here is the complexity of the preemptive EC(w) problem on general graphs. Similar questions are also of interest for the VC(w) problem.

ii. In the same scheduling context, there a significant setup delay, say d, to establish each batch. For the (bounded) max-coloring problems the presence of such a delay can be easily handled by increasing the weight of the vertices or edges of G by d. This way the weight of each batch will be also increased by d, incorporating its set-up delay. However, for the preemptive variants of the problems the existence of the setup delay plays a crucial role, since the preemption increases the number of batches as well as the total setup overhead.

iii. The applications of all problems are on-line in nature. In practice, buffers are allocated as memory requests are created, and messages are scheduled as they arrive, with out knowledge of future. The online version of both problems is therefore of great practical and theoretical interest.

Finally, a third category of open questions includes some problems of technical nature.

i. The 1.74 approximation ratio of the algorithm we presented for the EC(w) problem on bipartite graphs, is obtained using computational tools (`Mathematica`). It would be interesting to have a close formula for this ratio.

ii. Our analysis of the approximation algorithm for the EC(w, b) problem presented, yields ratios of $3 - \frac{2}{\sqrt{2b}}$ for general graphs and $3 - \frac{2}{\sqrt{b}}$ for bipartite graphs. However, this algorithm reduces to the greedy 2-approximation one for the EC(w) problem when the cardinality bound b is large enough. Hence, the ratios for the EC(w, b) problem should tend to 2 instead of increasing with the bound b. Thus, a question here is a better analysis of our algorithm to match this fact.

Bibliography

[1] F. N. Afrati, T. Aslanidis, E. Bampis, and I. Milis. Scheduling in switching networks with set-up delays. *Journal of Combinatorial Optimization*, 9:49–57, 2005.

[2] N. Alon. A note on the decomposition of graphs into isomorphic matchings. *Acta Mathematica Hungarica*, 42:221–223, 1983.

[3] R. P. Anstee. An algorithmic proof of Tutte's f-factor theorem. *Journal of Algorithms*, 6:112–131, 1985.

[4] B. S. Baker and E. G. Coffman Jr. Mutual exclusion scheduling. *Theoretical Computer Science*, 162:225–243, 1996.

[5] A. Björklund and Th. Husfeldt. Inclusion–exclusion algorithms for counting set partitions. In *47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pages 575–582. IEEE Computer Society, 2006.

[6] H. L. Bodlaender and K. Jansen. Restrictions of graph partition problems. Part I. *Theoretical Computer Science*, 148:93–109, 1995.

[7] V. A. Bojarshinov. Edge and total coloring of interval graphs. *Discrete Applied Mathematics*, 114:23–28, 2001.

[8] G. Bongiovanni, D. Coppersmith, and C. K. Wong. An optimum time slot assignment algorithm for an SS/TDMA system with variable number of transponders. *IEEE Trans. on Communications*, 29:721–726, 1981.

[9] M. Boudhar. Scheduling on a batch processing machine with split compatibility graphs. *Journal of Mathematical Modelling and Algorithms*, 4:391–407, 2005.

[10] M. Boudhar and G. Finke. Scheduling on a batch machine with job compatibilities. *Belgian Journal of Oper. Res., Statistics and Computer Science*, 40:69–80, 2000.

[11] L. Cai and J. A. Ellis. Np-completeness of edge-coloring some restricted graphs. *Discrete Applied Mathematics*, 30:15–27, 1991.

[12] B.-L. Chen, H.-L. Fu, and M. T. Ko. Total chromatic number and chromatic index of split graphs. *Journal of Combinatorial Mathematics and Combinatorial Computing*, 17:137•–146, 1995.

[13] A. G. Chetwynd and A. J. W. Hilton. Regular graphs of high degree are 1-factorizable. In *Proceedings of the London Mathematical Society*, volume 50, pages 193–206, 1985.

[14] V. Chvátal. A greedy heuristic for the set-covering problem. *Mathematics of Operations Research*, 4:233–235, 1979.

[15] J. Cohen, E. Jeannot, N. Padoy, and F. Wagner. Messages scheduling for parallel data redistribution between clusters. *IEEE Transactions on Parallel and Distributed Systems*, 17:1163–1175, 2006.

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

[17] P. Crescenzi, X. Deng, and Ch. H. Papadimitriou. On approximating a scheduling problem. *Journal of Combinatorial Optimization*, 5:287–297, 2001.

[18] D. de Werra. Decomposition of bipartite multigraphs into matchings. *Mathematical Methods of Operations Research*, 16:85–90, 1972.

[19] D. de Werra. Restricted coloring models for timetabling. *Discrete Mathematics*, 165/166:161–170, 1997.

[20] D. de Werra, M. Demange, B. Escoffier, J. Monnot, and V. Th. Paschos. Weighted coloring on planar, bipartite and split graphs: Complexity and approximation. *Discrete Applied Mathematics*, 157:819–832, 2009.

[21] D. de Werra, A. Hertz, D. Kobler, and N. V. R. Mahadev. Feasible edge coloring of trees with cardinality constraints. *Discrete Mathematics*, 222:61–72, 2000.

[22] M. Demange, D. de Werra, J. Monnot, and V. Th. Paschos. Time slot scheduling of compatible jobs. *Journal of Scheduling*, 10:111–127, 2007.

[23] M. Dror, G. Finke, S. Gravier, and W. Kubiak. On the complexity of a restricted list-coloring problem. *Discrete Mathematics*, 195:103–109, 1999.

[24] L. Epstein and A. Levin. On the max coloring problem. In *5th Workshop on Approximation and Online Algorithms (WAOA'07)*, volume 4927 of *LNCS*, pages 142–155. Springer, 2008.

[25] B. Escoffier, J. Monnot, and V. Th. Paschos. Weighted coloring: Further complexity and approximability results. *Information Processing Letters*, 97:98–103, 2006.

[26] G. Finke, V. Jost, M. Queyranne, and A. Sebő. Batch processing with interval graph compatibilities between tasks. *Discrete Applied Mathematics*, 156:556–568, 2008.

[27] S. Fiorini and R. J. Wilson. *Edge-colourings of graphs*. Pitman, London, 1977.

[28] S. Fiorini and R. J. Wilson. Edge colorings of graphs. In L. W. Beineke and R. J. Wilson, editors, *Selected Topics in Graph Theory*, pages 103–126. Academic Press, 1978.

[29] H. N. Gabow, T. Nishizeki, O. Kariv, D. Leven, and O. Terada. Algorithms for edge-coloring graphs. Technical Report TRECIS-8501, Tohoku University, 1985.

[30] F. Gardi. Mutual exclusion scheduling with interval graphs or related classes. Part II. *Discrete Applied Mathematics*, 156:794–812, 2008.

[31] M. R. Garey, D. S. Johnson, G. L. Miller, and C. H. Papadimitriou. The complexity of coloring circular arcs and chords. *SIAM Journal on Algebraic and Discrete Methods*, 1:216–227, 1980.

[32] M. R. Garey, D. S. Johnson, and L. J. Stockmeyer. Some simplified NP-complete graph problems. *Theoretical Computer Science*, 1:237–267, 1976.

[33] F. Gavril. Algorithms for minimum coloring, maximum clique, minimum covering by cliques, and maximum independent set of a chordal graph. *SIAM Journal on Computing*, 1:180–187, 1972.

[34] M. C. Golumbic. *Algorithmic graph theory and perfect graphs.* Academic Press, New York, 1980.

[35] I. S. Gopal and C. Wong. Minimizing the number of switchings in a SS/TDMA system. *IEEE Transactions On Communications*, 33:497–501, 1985.

[36] S. Gravier, D. Kobler, and W. Kubiak. Complexity of list coloring problems with a fixed total number of colors. *Discrete Applied Mathematics*, 117:65–79, 2002.

[37] M. Grötschel, L. Lovász, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1:169–197, 1981.

[38] M. M. Halldórsson. A still better performance guarantee for approximate graph coloring. *Information Processing Letters*, 45:19–23, 1993.

[39] M. M. Halldórsson and H. Shachnai. Batch coloring flat graphs and thin. In *11th Scandinavian Workshop on Algorithm Theory (SWAT'08)*, volume 5124 of *LNCS*, pages 198–209. Springer, 2008.

[40] P. Hansen, A. Hertz, and J. Kuplinsky. Bounded vertex colorings of graphs. *Discrete Mathematics*, 111:305–312, 1993.

[41] I. Holyer. The NP-completeness of edge-coloring. *SIAM Journal on Computing*, 10:718–720, 1981.

[42] M. Jarvis and B. Zhou. Bounded vertex coloring of trees. *Discrete Mathematics*, 232:145–151, 2001.

[43] D. S. Johnson. The np-completeness column: An ongoing guide. *Journal of Algorithms*, 6:434–451, 1985.

[44] I. A. Karapetian. On coloring of arc graphs. *Akademiia nauk Armianskoi SSR Doklady*, 70:306•–311, 1980.

[45] R. M. Karp. Reducibility among combinatorial problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*, pages 85•–103. Plenum Press, 1972.

[46] A. Kesselman and K. Kogan. Nonpreemptive scheduling of optical switches. *IEEE Trans. on Communications*, 55:1212–1219, 2007.

[47] D. König. Über graphen und ihre anwendung auf determinantentheorie und mengenlehre. *Mathematische Annalen*, 77:453–465, 1916.

[48] E. L. Lawler and J. Labetoulle. On preemptive scheduling of unrelated parallel processors by linear programming. *Journal of the Association for Computing Machinery*, 25:612–619, 1978.

[49] R. C. S. Machado, C. M. H. de Figueiredo, and K. Vušković. Chromatic index of graphs with no cycle with a unique chord. preprint, 2009.

[50] S. Micali and V. V. Vazirani. An $O(\sqrt{|V|}|E|)$ algorithm for finding maximum matching in general graphs. In *21st Annual IEEE Symposium on Foundations of Computer Science (FOCS'80)*, pages 17–27. IEEE Computer Society, 1980.

[51] S. V. Pemmaraju and R. Raman. Approximation algorithms for the max-coloring problem. In *32nd International Colloquium on Automata, Languages and Programming (ICALP'05)*, volume 3580 of *LNCS*, pages 1064–1075. Springer, 2005.

[52] S. V. Pemmaraju, R. Raman, and K. R. Varadarajan. Buffer minimization using max-coloring. In *15th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'04)*, pages 562–571, 2004.

[53] F. Rendl. On the complexity of decomposing matrices arising in satellite communication. *Operations Research Letters*, 4:5–8, 1985.

[54] N. Robertson, D. P. Sanders, P. D. Seymour, and R. Thomas. The four colour theorem. *Journal of Combinatorial Theory, Series B*, 70:2–44, 1997.

[55] D. P. Sanders and Y. Zhao. Planar graphs of maximum degree seven are Class I. *Journal of Combinatorial Theory, Series B*, 83:201–212, 2001.

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

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

[58] D. Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. In *38th Annual ACM Symposium on the Theory of Computing (STOC'06)*, pages 681–690, 2006.