Contributions on Secretary Problems, Independent Sets of Rectangles and Related Problems

by

José Antonio Soto

Submitted to the Department of Mathematics
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2011

© 2011 José Antonio Soto. All rights reserved.

The author hereby grants to MIT permission to reproduce and to distribute publicly paper and electronic copies of this thesis document in whole or in part in any medium now known or hereafter created.

Author .....
Department of Mathematics
May 18, 2011

Certified by .....
Michel X. Goemans
Leighton Family Professor of Applied Mathematics
Thesis Supervisor

Accepted by .....
Michel X. Goemans
Chairman, Department Committee on Graduate Theses

Contributions on Secretary Problems, Independent Sets of Rectangles and Related Problems

by

José Antonio Soto

Submitted to the Department of Mathematics
on May 18, 2011, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

We study three problems arising from different areas of combinatorial optimization.

We first study the *matroid secretary problem*, which is a generalization proposed by Babaioff, Immorlica and Kleinberg of the classical secretary problem. In this problem, the elements of a given matroid are revealed one by one. When an element is revealed, we learn information about its weight and decide to accept it or not, while keeping the accepted set independent in the matroid. The goal is to maximize the expected weight of our solution. We study different variants for this problem depending on how the elements are presented and on how the weights are assigned to the elements. Our main result is the first constant competitive algorithm for the *random-assignment random-order model*. In this model, a list of hidden nonnegative weights is randomly assigned to the elements of the matroid, which are later presented to us in uniform random order, independent of the assignment.

The second problem studied is the *jump number problem*. Consider a linear extension L of a poset P. A *jump* is a pair of consecutive elements in L that are not comparable in P. Finding a linear extension minimizing the number of jumps is NP-hard even for chordal bipartite posets. For the class of posets having *two directional orthogonal ray comparability graphs*, we show that this problem is equivalent to finding a maximum independent set of a well-behaved family of rectangles. Using this, we devise combinatorial and LP-based algorithms for the jump number problem, extending the class of bipartite posets for which this problem is polynomially solvable and improving on the running time of existing algorithms for certain subclasses.

The last problem studied is the one of finding nonempty minimizers of a symmetric submodular function over any family of sets closed under inclusion. We give an efficient $O(n^3)$-time algorithm for this task, based on Queyranne's pendant pair technique for minimizing unconstrained symmetric submodular functions. We extend this algorithm to report all inclusion-wise nonempty minimal minimizers under hereditary constraints of slightly more general functions.

Thesis Supervisor: Michel X. Goemans
Title: Leighton Family Professor of Applied Mathematics

Acknowledgments

This thesis is the result of many experiences that I have encountered over the last five years at MIT, and the support of many wonderful people I have met here. There are always more people to acknowledge than one ever has the space and memory for.

First and foremost, I want to thank my advisor Michel Goemans. His perpetual enthusiasm for research, as well as his patience, encouragement and guidance beyond academic activities have been invaluable for me. Even on a very busy schedule, Michel always managed to dedicate a nontrivial amount of time to work together on different problems.

I also want to acknowledge all the patience and feedback provided by my thesis committee, Jonathan Kelner and Andreas Schulz. Special thanks go to Andreas not only for attending my thesis defense remotely from Germany, but also for inviting me to many mittag seminare and for proposing many interesting problems.

I gratefully acknowledge the support of the National Science Foundation (NSF) under contract CCF-0829878 and the Office of Naval Research (ONR) under grant number N00014-11-1-0053.

I extend my gratitude to all the teachers and professors who have motivated me to study Mathematics, specially to my undergraduate advisor Marcos Kiwi for introducing me to the world of scientific research and for encouraging me to come to MIT.

I do not want to forget to mention all the people who contributed to make my time at MIT enjoyable: my officemates, colleagues and friends; students, faculty, staff, visitors and postdocs; not only inside the math department, but also in CSAIL, and ORC. Many thanks to the chilean community in Boston and to my neighbors at Tang and Westgate, many of whom I am lucky to call friends. Special thanks go to Claudio Telha who not only has been a very good friend and neighbor but also a great coauthor: without him a big part of this thesis would have not been written.

I thank my parents, my brother and sister for their support, care and love. I would have never made it here without them. I also thank all my friends back in Chile for somehow always being present in my life.

Finally, but not least, I give my deepest love to my wife Giannina for her company, infinite love and understanding. I thank her for tolerating all the time that this thesis has stolen away from us. My last year at MIT has been the happiest by far and that is all because of her.

Contents

Introduction

In this thesis we focus on three problems in combinatorial optimization arising from different areas: the matroid secretary problem, the jump number problem and hereditarily constrained minimization of symmetric submodular functions.

The first problem belongs to the realm of online selection algorithms. We wish to select a set of elements satisfying certain properties from a stream revealed in random order, where the decision of selecting an element or not must be made at the moment the element arrives. We study different variants of this problem, exhibiting simple constant competitive algorithms for some of them.

The second is a scheduling problem coming from order theory. We show a surprising connection between the jump number problem of certain partially ordered sets and the geometric problem of finding a maximum collection of disjoint rectangles from a family of axis-parallel rectangles in the plane. Using this geometric interpretation we devise efficient algorithms for this problem and explore nontrivial relations to other problems in the literature.

The third problem is a set function minimization problem. We contribute a simple and efficient algorithm that solves it exactly and give some extensions.

In this chapter we describe each of the aforementioned problems, some motivation and historical background on them, and finally our results and the organization of the corresponding chapters. The rest of the thesis is divided into three parts, each one dedicated to one of these problems.

Basic notation

The common notation used in this thesis is fairly standard. We use $\mathbb{Z}$ and $\mathbb{R}$ to denote the sets of integer and real numbers respectively. We also use $\mathbb{Z}_+$ and $\mathbb{R}_+$ to denote the sets of nonnegative integer and real numbers respectively. For a finite number n, we use $[n]$ to denote the set $\{1, \dots, n\}$. The collection of all subsets of a set V is denoted as 2^V.

For vectors $v \in \mathbb{R}^d$, we use subindices to denote its coordinates $v = (v_1, \dots, v_d)$. For the specific case of $d = 2$ or $d = 3$, we also use v_x, v_y and v_z to denote the first, second and third coordinates. The notation $\mathbb{R}^E$ denotes the Euclidean space of dimension $|E|$ over $\mathbb{R}$, where the coordinates are indexed by elements of E. For every set $F \subseteq E$, we use $\chi_F \in \{0,1\}^E$ to denote its characteristic vector, where $\chi_F(e) = 1$ if $e \in F$ and 0 otherwise.

We use $G = (V, E)$ to denote a graph G having vertex set V and edge set E. On occasions, we use $G = (V, E, w)$ to denote a graph (V, E) along with a real weight

function $w\colon E \to \mathbb{R}$ on its edges. To describe the asymptotic behavior of a function when its value goes to infinity, we use the standard $O(\cdot)$, $\Omega(\cdot)$, $o(\cdot)$, and $\omega(\cdot)$ notations.

Finally, we use $\Pr(A)$ to denote the probability of a given event A and $\mathbb{E}[X]$ to denote the expected value of a random variable X. We use subindices on $\Pr(\cdot)$ and $\mathbb{E}[\cdot]$ if we want to be specific about the probability space over which the probability or expectation is taken.

In each part of the thesis, we introduce specific notation for each of the studied problems.

Matroid Secretary Problem

The optimal stopping problem known as the *classical secretary problem* can be regarded as follows. An employer wishes to hire the best secretary among n candidates about which she has no prior information. She interviews them in random order. After each interview, she is able to compare the candidates she has already interviewed and must decide immediately whether to hire the current one or continue. Candidates that are declined can not be recalled, and the employer is satisfied with nothing but the best.

This problem has a very elegant solution. Lindley [101] and Dynkin [48] have independently shown that the best strategy consists in observing roughly n/e of the candidates and then selecting the first one that is better than all the observed ones. This strategy returns the best candidate with probability $1/e$. Since then, this simple problem has been generalized and extended in many different directions, rapidly becoming a field in its own. People have studied different objective functions, the case where the number of candidates is unknown, selecting multiple candidates, allowing full or partial recall of candidates, and adding extra cost for selecting later candidates, among other variations. To learn more about the early history of this problem and some of its variants we recommend the entertaining article of Ferguson [55] and Freeman's survey [65]. We also mention different variants in Section 1.6.

In this thesis, we follow the line of work initiated by Babaioff, Immorlica and Kleinberg [8]. Their work is motivated by online auctions: An online algorithm can be regarded as an auctioneer having one or many identical items, and the secretaries as agents arriving at random times, each one having a different valuation for the items. The goal of the algorithm is to assign the items to the agents as they arrive while maximizing the total social welfare, subject to some combinatorial restrictions. In many cases, these restrictions can be modeled by matroid constraints.

More precisely, in the *matroid secretary problem*, the elements of the ground set of a given matroid arrive one by one, each one having a hidden weight. Upon arrival, an algorithm gets to know the weight and is given the choice to take the element or reject it, where this decision is irreversible. The only restriction is that the set of accepted elements must be independent in the matroid. The task is to select an independent set having total sum of weights as large as possible. It is usually assumed that the elements arrive in random order independent of their weights, but this is not a requirement. Another model assumes that a set of adversarially selected

weights is randomly assigned to the elements of the matroid before the execution of the algorithm.

Matroids are very rich combinatorial structures. They generalize certain aspects of linear independence in vector spaces and some aspects of graph theory. As independently shown by Rado [142], Gale [67] and Edmonds [50], matroids characterize those hereditary set systems for which optimizing a linear objective function is attained via a natural greedy algorithm. This property makes them suitable for online scenarios. Because of this, Babaioff et al. [8] have conjectured that it is possible to devise a constant competitive algorithm for the previous problem, as long as the order or the weight assignment is randomly selected.

There has been a significant amount of work on this conjecture on the model where the weights are adversarially selected but the order is random. Constant competitive algorithms are known for partition matroids [101, 48, 95, 6], transversal matroids [44, 97, 162], graphic matroids [8, 5, 97] and laminar matroids [87]. For general matroids, the best algorithm known so far, due to Babaioff et al. [8], is $O(\log r)$-competitive, where r is the rank of the matroid.

Our results

Part of the work of this thesis has already been presented in a recent SODA paper [152]. In this thesis we partially answer Babaioff et al.'s conjecture by exhibiting constant competitive algorithms for the random-assignment random-order model. This is the model for which both the arrival order is selected at random and the weights are randomly assigned. In [152], the author has already presented a $2e/(1-1/e) \approx 8.6005$ algorithm for this model. In this work, we present a new algorithm achieving a competitive ratio of at most 5.7187.

On a very high level our algorithm is based on a simple divide and conquer approach: replace the matroid by a collection of matroids of a simpler class for which we can easily derive a constant-competitive algorithm, and then return the union of the answers. The simpler matroids we use are known as uniformly dense matroids.

Uniformly dense matroids are those for which the density of a set, that is, the ratio of its cardinality to its rank, is maximized on the entire ground set. The simplest examples are precisely the uniform matroids. We show that uniformly dense matroids and uniform matroids of the same rank behave similarly, in the sense that the distribution of the rank of a random set is similar for both matroids. We use this fact to devise constant competitive algorithms for uniformly dense matroids in this model.

In order to extend the above algorithms to general matroids we exploit some notions coming from the *theory of principal partitions* of a matroid, particularly its *principal sequence*. Roughly speaking, the principal sequence of a matroid $\mathcal{M}$ is a decomposition of its ground set into a sequence of parts, each of which is the underlying set of a uniformly dense minor of $\mathcal{M}$. Furthermore, if we select one independent set in each of these minors, their union is guaranteed to be independent in $\mathcal{M}$. By employing separately the previous algorithms in each of these minors, we obtain an algorithm that returns an independent set of $\mathcal{M}$, while only increasing an

extra factor of $e/(e-1)$ on its competitive ratio. By comparing the weight of our solution to the optimum of certain randomly defined partition matroids, we give a tighter analysis for the competitive ratio of our algorithms.

As first noticed by Oveis Gharan and Vondrák [136], it is possible to also apply the methods in [152] to obtain constant competitive algorithms for the stronger model in which the weights are randomly assigned but the order in which the elements are presented is prescribed by an adversary (the random-assignment adversarial-order model). In this thesis, we present alternative algorithms for this second model achieving a competitive ratio of $16/(1-1/e) \approx 25.311$.

Babaioff et al.'s conjecture is still open for the "standard" adversarial-assignment random-order model. For this model, we present simple algorithms for various matroid classes. We show a ke-competitive algorithm for the case in which the matroid is representable by a matrix where each column has at most k non-zero entries. This result generalizes the $2e$-competitive algorithm for graphic matroids of Korula and Pál [97]. We also give algorithms for general matroids having competitive ratio proportional to the density of the matroid. Using this, we obtain a $3e$-competitive algorithm for cographic matroids, and a k-competitive algorithm for matroids where each element is in a cocircuit of size at most k.

For general matroids, we give a new $O(\log r)$-competitive algorithm. Unlike the previous algorithm of Babaioff et al. [8], our algorithm does not use the numerical value of the weights. It only needs the ability to make comparisons among seen elements. This is a desirable property since the features revealed by the elements may be of qualitative type (for example the *qualifications* of a person applying for a job), but the actual value or profit may be an unknown increasing function of the features revealed. In fact, all the algorithms proposed in this thesis have the mentioned desirable property.

Organization

This part of the thesis is presented in Chapters 1 and 2.

In Chapter 1 we formally define secretary problems and describe two of the simplest cases: the classical and the multiple choice secretary problems, presenting previous results for both of them. Later, we describe the generalized secretary problem, introduced by Babaioff et al. [8], in which the sets of elements that can be simultaneously selected obey arbitrary hereditary constraints. We pay special attention to the different models that arise depending on how the candidates are presented and how the weights are assigned. Afterwards, we turn our attention to the matroid secretary problem. Before tackling this problem, we present some background on matroids and their relation to the greedy algorithm. We conclude the chapter by giving a brief survey of related results.

Chapter 2 contains our new results. We start by introducing notation and the divide and conquer idea on which our algorithms for random-assignment models are based. Then, we study properties of uniformly dense matroids and give different algorithms for them on both the random-assignment random-order model and the random-assignment adversarial-order model. Afterwards, we describe the sequence of

uniformly dense minors arising from the principal partition of a loopless matroid and give some background on this construction. We show how to use these matroids to give constant competitive algorithms for general matroids in both random-assignment models.

Later in that chapter, we focus on algorithms for the adversarial-assignment random-order model. We present a new $O(\log r)$-competitive algorithm for matroids of rank r which only uses the relative order of the weights seen and not their actual values. We also present an algorithm for linear matroids represented by a matrix A having competitive ratio proportional to the maximum number of nonzero entries in a column of A, and an algorithm for general matroids having competitive ratio proportional to the density of the matroid. We conclude that chapter with a summary of results and a description of open problems in the area.

Jump Number of Two Directional Orthogonal Ray Graphs and Independent Sets of Rectangles

Although the *jump number problem* is a purely combinatorial problem arising from order theory, it admits a natural scheduling interpretation: schedule a collection of jobs on a single machine obeying the precedences induced by a partially ordered set (poset), in such a way that the total *setup cost* is minimized. Here, the setup cost of a job is 0 if the job immediately before it in the schedule is constrained to precede it, or 1 otherwise; we say in the latter case that there is a *jump* in the schedule.

It is known that this problem is NP-hard even for bipartite posets [139], and that the jump number itself, this is the minimum number of jumps that a given poset admits over all its linear extensions, depends only on the poset's comparability graph [82]. The literature of this problem is vast, specially in what respects determining natural classes of comparability graphs for which the problem is polynomial time solvable. For instance, there are polynomial time algorithms to compute this number on bipartite permutation graphs [156, 53, 16], biconvex graphs [16] and convex graphs [40].

An important open question in this area is to determine the complexity of the jump number problem as a function of the poset dimension. The dimension of a partial order P is the minimum number k for which P can be expressed as the intersection of k linear orders. Alternatively, this can be defined as the minimum k for which the poset P can be embedded in the k-dimensional Euclidean space $\mathbb{R}^k$ endowed with the natural coordinate-wise partial order relation $\leq_{\mathbb{R}^k}$. Many problems that are hard for general posets become polynomially solvable for two-dimensional posets. This has led Bouchitté and Habib [15] to conjecture that the jump number problem is also polynomial time solvable in this class. This conjecture remains open today.

From the description above, two-dimensional posets can be defined geometrically as follows. The elements are a collection of points in the plane and the precedences are given by the natural partial order on the plane: a point precedes another if the first is below and to the left of the second. In this thesis we consider a natural variant of

these posets in which the points have an associated color and relations between points of the same colors are ignored. More precisely *bicolored 2D-poset* can be defined as follows. Given a sets of red points R and blue points B in the plane, a red element r precedes a blue element b if r is located below and to the left of b. These are the only precedences included in the poset. The comparability graphs of these posets correspond exactly to the *two directional orthogonal ray graphs* considered recently by many authors [135, 151].

Our results

A big part of the material presented in this part of the thesis is based on joint work with Claudio Telha [153].

In this thesis we show that solving the jump number problem on bicolored 2D-posets defined as above, is equivalent to finding a maximum cardinality family of disjoint axis-parallel rectangles having a red point as bottom-left corner and a blue point as top-right corner. Furthermore, we show that this problem can be solved in polynomial time using either a linear programming based algorithm or a combinatorial algorithm. The combinatorial algorithm presented runs in time $\widetilde{O}(n^\omega)$, where $2 \leq \omega \leq 2.376$ is the exponent for the matrix multiplication problem.

As a biproduct of our work we show a min-max relation between two problems on rectangles: for the families of rectangles previously described, the maximum size of a disjoint subfamily of rectangles equals the minimum number of points needed to hit every rectangle in the family. This relation between the maximum independent set and the minimum hitting set of rectangles is not true for general rectangle families, although it is very simple to argue that the first quantity is always at most the second. A big conjecture in the area is whether or not the ratio between the sizes of the minimum hitting set and the maximum independent set of any rectangle family is bounded by a constant. Our result can be seen as a nontrivial step to answer this conjecture. While studying the aforementioned conjecture, we observe that a simple application of recent approximation algorithms for the maximum independent set and the minimum hitting set problems gives a bound of $O(\log^2 \log \alpha)$ for this ratio, where α is the size of the maximum independent set, improving the existing bounds of $O(\log \alpha)$.

The presented min-max relation translates immediately to min-max relations between other problems related to the jump number problem of two directional orthogonal ray graphs: the maximum cross-free matching problem and the minimum biclique cover problem. The minimum biclique cover problem of a bipartite graph is also equivalent to the problem of computing the *boolean rank* of its biadjacency matrix A: this is finding the minimum value k for which the $s \times t$ matrix A can be written as the boolean product of an $s \times k$ matrix P and a $k \times t$ matrix Q. The boolean rank is used, for example, to find lower bounds for *communication complexity* [99]. As a corollary, we also expand the class of matrices for which the boolean rank can be computed exactly in polynomial time.

Furthermore, we relate the previous min-max relations to other relations arising from apparently unrelated problems in combinatorial optimization: the *minimum*

rectangle cover and the *maximum antirectangle* of an orthogonal biconvex board, studied by Chaiken et al. [25], the *minimum base* and the *maximum irredundant subfamily* of an interval family, studied by Györi [80]; and the *minimum edge-cover* and the *maximum half-disjoint family of set-pairs*, studied by Frank and Jordán [63]. Our min-max relations can be seen in a certain way as both a generalization of Györi's result and as a non-trivial application of Frank and Jordán's result. By relating our problem to Györi's result we also give an efficient $O(n^2)$-time algorithm to compute the jump number of convex graphs, significantly improving on the previous $O(n^9)$-algorithm of Dahlhaus [40].

To conclude, we also study a weighted version of the maximum cross-free matching problem. We show that this problem is NP-hard on two directional orthogonal ray graphs, and give combinatorial algorithms for certain subclasses where the problem is polynomial time solvable.

Organization

This part of the thesis is presented in Chapters 3 through 6.

In Chapter 3 we introduce partially ordered sets and state some properties. Later, we give a small survey on different comparability graph classes, focusing on subclasses of two directional orthogonal ray graphs and discuss some useful geometrical characterizations. Afterwards, we briefly discuss perfect graphs and recall some of their properties, since we need them for some of our results.

Chapter 4 focuses on the jump number problem of a comparability graph and surveys some previous results. In this chapter we also define two related problems: the maximum cross-free matching and the minimum biclique cover of a graph. The former problem is known to be equivalent to the jump number problem for chordal bipartite graphs. Later, we review apparently unrelated problems on 0-1 matrices, planar geometry and interval combinatorics. The reviewed problems come in pairs of a minimization and a maximization problem; for some cases a min-max relation between them is known. We show that all these problems are either equivalent or special cases of the maximum cross-free matching and the minimum biclique cover problem.

In Chapter 5 we turn our attention to computing the jump number of two directional orthogonal ray graphs. For these graphs, we show an equivalence between the maximum cross-free matchings and minimum biclique covers and the maximum independent set and minimum hitting sets of certain families of rectangles. Because of this equivalence, we dedicate a section to review the two mentioned rectangle problems, paying special attention to existing linear programming relaxations. Later in the chapter, we use these relaxations to obtain a polynomial time algorithm for the maximum cross-free matching problem on two directional orthogonal ray graphs. Afterwards, we describe a combinatorial algorithm to compute both a cross-free matching and a biclique cover of a given two directional orthogonal ray graph having the same cardinality. This shows in particular that a min-max relation holds. Later, we explore the relation between our results with previous works. In particular, we show how to obtain our min-max relation as a particular case of a strong result by Frank

and Jordán [63]. Finally, we summarize our work and present open problems.

In Chapter 6 we describe the weighted version of the maximum cross-free matching problem. We show NP-hardness for the case of two directional orthogonal ray graphs and give some subclasses for which we can solve this problem in polynomial time.

Constrained Set Function Minimization

A real valued function $f\colon 2^V \to \mathbb{R}$ is called *submodular* if it satisfies the submodular inequality $f(A \cup B) + f(A \cap B) \leq f(A) + f(B)$ for every pair of subsets A and B of V. Submodularity is one of the most useful properties in combinatorial optimization (see, e.g. Lóvasz's article [104]).

Many important problems reduce to minimizing submodular functions. This is solved in polynomial time by Grötschel, Lovász and Schrijver [75, 76] using the ellipsoid method. Currently it is possible to minimize a general submodular function given by a value oracle in $O(n^6)$-time and using $O(n^5)$ oracle calls, where n is the size of the ground set, by a combinatorial algorithm due to Orlin [134].

When the submodular function has more structure, it is possible to solve the minimization problem faster. This is true, for instance for cut functions of weighted graphs. These functions are not only submodular, but also symmetric. In this case, the problem is equivalent to the minimum cut problem, which can be solved efficiently using network flow based algorithms.

Nagamochi and Ibaraki [125, 126] have proposed a combinatorial algorithm to solve the minimum cut problem on weighted graphs without relying on network flows. After some improvements and simplifications, Queyranne [141] obtains a purely combinatorial algorithm that finds a nontrivial minimizer of a symmetric submodular function running in $O(n^3)$-time and using only $O(n^3)$ function oracle calls. Here nontrivial means a set that is neither empty or the entire ground set.

We are interested in the problem of minimizing symmetric submodular functions over *hereditary subfamilies of* 2^V, this is, over families that are closed under inclusion. Many simple constraints can be expressed in this way. For example, we can add a maximum cardinality constraint of k to our problem by considering the family $\mathcal{F} = \{X\colon |X| \leq k\}$. It is important to notice that the problem of minimizing a *general* submodular function over the family $\mathcal{F}$ just defined is already NP-hard to approximate within an $o\left(\sqrt{n/\log n}\right)$ factor, as shown by Svitkina and Fleischer [159].

Our results

In this thesis, we show how to extend Queyranne's algorithm to return a nontrivial minimizer of a symmetric submodular function over any hereditary family. This hereditary minimization problem includes, for example, the problem of finding a planar induced subgraph in an undirected graph minimizing the number of edges having precisely one endpoint in the vertex set of the induce subgraph. Our algorithm runs in $O(n^3)$-time and uses $O(n^3)$ oracle calls.

For the unrestricted problem, Nagamochi and Ibaraki [127] have presented a modification of Queyranne's algorithm that finds all inclusion-wise minimal minimizers of a symmetric submodular function still using a cubic number of oracle calls. Using similar ideas, we can also list all minimal solutions of an hereditary minimization problem using only $O(n^3)$ oracle calls. As these minimal solutions can be shown to be disjoint, there are at most n of them.

We also give some general conditions for other function classes for which our methods can still be applied. For instance, we can find all the minimal minimizers of a function f under hereditary constraints when f is a *restriction* of a symmetric submodular function (also known as *submodular-posimodular functions*) or when $f(S)$ is defined as $d(S, V \setminus S)$ for a monotone and consistent symmetric by-set map d in the sense of Rizzi [145].

Organization

This part of the thesis is presented in Chapter 7 which is organized as follows.

We first give a brief introduction to the problem and some background on previous work. Later, we explore Queyranne's [141] technique to solve the unconstrained minimization of symmetric submodular functions. This technique is based on the concept of *pendant pairs* which we also describe in detail. We explain what are the conditions that a general set function must satisfy in order for this algorithm to work.

Afterwards, we move to the problem of minimizing set functions under hereditary constraints. We show how to modify the pendant pair technique in order to find a minimizer of the constrained problem. We further modify our algorithm so that it outputs all inclusion-wise minimal minimizers.

We later focus on the function class for which the previous techniques work. This class includes symmetric submodular functions, their restrictions and more general functions that we denote as weak Rizzi functions.

After the completion of this work, we were informed of an independently discovered algorithm of Nagamochi [124] which is able to perform a slightly stronger feat for certain function class that includes symmetric submodular functions. In the last part of this chapter we compare our results.

Part I

Matroid Secretary Problem

Chapter 1

Secretary Problems and Matroids

In this chapter we define secretary problems and describe two of the simplest cases: the classical secretary problem and the multiple choice secretary problem. Later, we describe a common generalization known as the matroid secretary problem introduced by Babaioff et al. [8]. We conclude by a brief survey of known results in this area.

1.1 Introduction to Secretary Problems

We call **secretary problem** any process in which we select one or more elements from a stream, under the condition that every element must be selected or rejected at the moment it is revealed.

Secretary problems arise as an idealization of real world sequential decision-making tasks. Examples of tasks modeled by secretary problems include selling an item to the highest bidder and hiring the most qualified person applying for a job – for example a secretary position (this is the reason why this problem got its, in my opinion, unfortunate name). Even though online selection problems like the ones mentioned seem natural, it was not until the second half of the twentieth century that serious attempts to formalize and study these problems were undertaken.

Many variations of the secretary problem have been introduced under different names or in the form of recreational games. Usually, these variants are obtained by changing some parameters of the problem, for example:

- Do we know the number of elements a priori?
- In what order are the elements presented?
- Can we select more than one element? If so, what sets of elements can we select simultaneously?
- What information we receive at the moment we examine an element?
- What information we know a priori about the elements?
- What is our objective function?

The rather large number of variants makes it very tedious to try to study each of them in detail. Since this is not our objective, we only focus on a few ones. In the rest of this section, we formulate the exact variants we attempt to study and explore some previous work on them.

1.1.1 Comparison and Value Based Algorithms

Consider the two following toy problems.

In the **beauty contest**, a collection of n contestants is presented one by one and in random order to a judge. After evaluating each contestant, the judge must make a choice before seeing the next one. He can either declare that contestant to be the winner, at which point the contest ends; or keep looking, disqualifying the current contestant. If the judge reaches the last contestant, she is declared the winner by default. What is the judge's best strategy to select the best contestant?

In the **game of googol**, Alice is given n slips of paper. In each paper she must write a positive real number. After that, the slips are put face down on a table and shuffled. Bob starts turning the slips face up and his aim is to stop turning when he sees a number which he guesses is the largest of the series. What is Bob's best strategy?

Both problems are very similar: the objective is to select the best of a sequence of randomly sorted elements. However, they differ in one small detail: in the second problem, Bob can actually *see* the values that Alice has written, and can use that information for his advantage. For example, if he knew that Alice is prone to select her numbers uniformly from a fixed range, he can probably devise a better strategy. In the first problem, this information is of no use, since the judge can not see values: he can only compare previously presented candidates.

We say that a decision maker or algorithm is **comparison-based** if it is able only to 'rank' or 'compare' previously seen elements, but it is not able to 'see' their actual values. Otherwise, we say it is a **value-based** algorithm. In this work we focus mostly on comparison-based algorithms (as in the case of the beauty contest problem). However, for reasons that will become apparent later, we won't restrict ourselves only to cases where the elements are revealed in a uniform random order.

We also assume that the decision maker is able to *break ties* in a *random but consistent way* using a uniformly random permutation $\tau\colon [n] \to [n]$ as follows: if two of the revealed elements have the same value, the one having larger τ-value is considered larger.

1.2 Classical Secretary Problem

The setting for the **classical secretary problem** is as follows.

Consider a set of n nonnegative hidden weights $w_1 \geq w_2 \geq \cdots \geq w_n \geq 0$. These weights are presented to an online algorithm in some order. When a weight is presented, the algorithm is able to compare it to previously seen ones, but can not see its numerical value (i.e. it is a comparison-based algorithm). Using this information,

the algorithm must decide whether to select the element or to ignore it and continue. Its objective is to select an element as heavy as possible.

We focus on two different **models**:

1. **Full-adversarial model**: An adversary selects both the hidden weights and the order in which the elements are presented to the algorithm.

2. **Random model**: An adversary selects the weights, but they are presented to the algorithm in uniform random order.

Every adversarial selection is taken before the execution of the algorithm. In other words, the adversary is oblivious to the decisions and random coins tosses of the algorithm.

Most of the literature regarding the classical secretary problem focus on designing algorithms that *maximize the probability of accepting the largest element in the sequence*. As we are interested in generalizations of this problem we instead take the route of competitive analysis.

Let ALG be the set returned by an algorithm $\mathcal{A}$ for the classical secretary problem. The set ALG can be either a singleton or the empty set. The **payoff** of $\mathcal{A}$ is the weight of the set ALG: $w(\text{ALG}) = \sum_{x \in \text{ALG}} w(x)$. Usually, the competitive ratio of $\mathcal{A}$ is defined as the ratio between the maximum possible payoff (in this case, the maximum weight) and the algorithm's payoff. As there is randomness involved, expected values are used. However, since the payoff could be zero, the previous ratio is not always well defined. We fix this by using the following definition.

We say that a randomized algorithm is **α-competitive** if for any adversarial selection that the model allows,

$$\mathbb{E}[\alpha w(\text{ALG}) - w^*] \geq 0, \tag{1.1}$$

where the expectation is over all the random choices given by the model and the random choices performed by the algorithm, and w^* is the maximum payoff of a possible answer under the current realization of the weights (in this case, the maximum weight w_1 of the stream). The **competitive ratio** of an algorithm is the minimum value of α for which this algorithm is α-competitive.

1.2.1 Previous Results for the Classical Secretary Problem

Consider the naive algorithm for any model of the classical secretary problem that chooses a random element of the stream. Since this algorithm recovers at least $1/n$-fraction of the weight of the maximum element, it is n-competitive.

We show that the full-adversarial model is, as expected, really hard and that for this model the naive algorithm above is optimal.

Lemma 1.2.1 (Folklore)**.** *No algorithm for the full-adversarial classic secretary problem admits a competitive ratio smaller than n.*

Proof. Let $\varepsilon > 0$ and fix any (possibly randomized) algorithm $\mathcal{A}$ for the problem. Consider the adversarial input given by the increasing sequence of weights $\varepsilon, 2\varepsilon, 3\varepsilon, \ldots, n\varepsilon$. Let p_i be the probability that $\mathcal{A}$ selects the i-th element on this sequence. Since the expected number of elements that $\mathcal{A}$ selects is at most one:

$$\sum_{i=1}^{n} p_i \leq 1. \tag{1.2}$$

In particular, there is an index j for which $p_j \leq 1/n$. Consider the sequence of weights given by $\varepsilon, 2\varepsilon, \ldots, (j-1)\varepsilon, n^2, 0, \ldots, 0$. Since the information available for the algorithm on this sequence up and including the revelation of the j-th element is equal to the information available for the totally increasing sequence (as the algorithm can only make comparisons), the probability that $\mathcal{A}$ selects the j-th element on this second sequence is still p_j. Hence, for the set ALG returned by the algorithm on the second sequence:

$$\mathbb{E}[w(\text{ALG})] \leq p_j n^2 + (j-1)\varepsilon \leq n(1+\varepsilon). \tag{1.3}$$

Since n^2 is the maximum payoff of the second sequence, the competitive ratio of algorithm $\mathcal{A}$ is at least $n/(1+\varepsilon) \geq n(1-\varepsilon)$. Because ε is arbitrary, we conclude the proof. □

Unlike the full-adversarial model, it is very easy to get a constant competitive algorithm for the random model. Before doing that, we give the following observation.

Lemma 1.2.2. *Any algorithm guaranteeing a probability p of selecting the largest element in the random model of the classical secretary problem is $1/p$-competitive and vice versa.*

Proof. For any algorithm selecting the largest element with probability at least p in the random model,

$$\mathbb{E}[1/p \cdot w(\text{ALG}) - w^*] \geq 1/p \cdot pw^* - w^* \geq 0. \tag{1.4}$$

Conversely, let $\mathcal{A}$ be a $1/p$-competitive algorithm for the random model. Let $0 < \varepsilon < 1$, $L = 1/\varepsilon$ and consider running algorithm $\mathcal{A}$ on the instances induced by the $n!$ permutations of the set of weights $W = \{L, L^2, \ldots, L^n\}$. Let q be the probability that $\mathcal{A}$ selects the largest element of the set. This probability is inherent to the algorithm and does not depend on the set W. Since the algorithm is $1/p$-competitive for the random model,

$$0 \leq \mathbb{E}[1/p \cdot w(\text{ALG}) - w^*] \leq 1/p \cdot (qL^n + L^{n-1}) - L^n = L^n/p \cdot (q + \varepsilon - p).$$

This implies that $q \geq p - \varepsilon$. Using that ε is arbitrary, we conclude the proof. □

Consider the following simple algorithm for the random model. Observe and reject the first $\lfloor n/2 \rfloor$ elements and then accept the first element (if any) that is better than all element seen so far. We claim that this algorithm selects the best element in the sequence with probability at least $1/4$ and thus, it is a 4-competitive algorithm. We

only show this for even values of n (as we will show a much more general bound later). Indeed, it is easy to see that if the second-best weight appears in the first $n/2$ positions and the top-weight appears in the last $n/2$-positions, then this algorithm returns the top-weight. The claim follows since the event just described has probability $1/4$.

In 1961, Lindley [101] gave a dynamic programming argument to show that the optimal algorithm for this problem is very similar to the one above: for every value of n, there is a number $r(n)$ such that the algorithm maximizing the probability of selecting the best element has the following form. Observe and reject the first $r(n)-1$ elements and then accept the first element (if any) that is better than all element seen so far. In 1963, Dynkin [48] gave an alternative proof of the previous fact using Markov chain arguments.

Lemma 1.2.3 (Lindley [101], Dynkin [48])**.** *Consider the algorithm for the random classical secretary problem obtained by observing and rejecting the first $r-1$ elements and then accepting the first element (if any) that is better than all elements seen so far. The probability $p(r)$ of selecting the best element under this algorithm is*

$$p(1) = \frac{1}{n} \text{ and } p(r) = \frac{r-1}{n} \sum_{i=r-1}^{n-1} \frac{1}{i}, \text{ for } 1 < r \leq n.$$

The optimum value $r(n)$, for $n \geq 2$ is obtained by picking the only r such that

$$\sum_{i=r}^{n-1} \frac{1}{i} \leq 1 < \sum_{i=r-1}^{n-1} \frac{1}{i}. \tag{1.5}$$

For a nice exposition and proof of the result above, we refer to the survey article of Gilbert and Mosteller [70]. Two properties of the above lemma are the following.

1. The optimum value $r(n)$ is such that $\lim_{n\to\infty} r(n)/n = 1/e$.

2. The optimal probability $p(r(n))$ is decreasing with n and $\lim_{n\to\infty} p(r(n)) = 1/e$.

In particular, in order to obtain an almost optimal e-competitive algorithm it is enough to set $r(n)$ to be roughly n/e and proceed as above.

Lindley–Dynkin algorithm is deterministic. Interestingly, we can obtain a *randomized* e-competitive algorithm with a somewhat simpler analysis by choosing the number of elements to "observe and reject" as a random variable with expectation n/e. This algorithm, denoted as Algorithm 1 below, is also easier to generalize to other versions of the secretary problem.

Algorithm 1 For the random-order classical secretary problem.

1: Choose m from the binomial distribution $\text{Bin}(n, p)$.
2: Observe and reject the first m elements – call this set the *sample*.
3: Accept the first element (if any) that is better than all the sampled ones.

Lemma 1.2.4.
Let w_i be the i-th top weight of the stream. Algorithm 1 returns an empty set with probability p, and returns the singleton $\{w_i\}$ with probability at least

$$p \int_p^1 \frac{(1-t)^{i-1}}{t}\, dt.$$

In particular, it returns w_1 with probability at least $(-p \ln p)$. Therefore, by setting $p = 1/e$, we obtain an e-competitive algorithm.

Proof. Consider the following offline simulation. Let $w_1 \geq w_2 \geq \cdots \geq w_n \geq 0$ be the weights in non-increasing order. Each weight w_i selects an arrival time t_i in the interval $(0, 1)$ uniformly and independently. The offline algorithm processes the weights in order of arrival, considering the ones arriving before time p as the sample. When a weight arriving after time p is processed, the algorithm checks if this weight is larger than every sampled weight. In that case, the algorithm selects the currently processed weight. Otherwise, it continues.

Since the cardinality of the sampled set has binomial distribution with parameters n and p, the set returned by this simulation has the same distribution as the one returned by Algorithm 1. Hence, we analyze the simulation.

Consider the i-th top weight w_i of the stream. Condition on the time t_i of its arrival being at least p (as otherwise the algorithm will not select it). Condition also on the event $\mathcal{E}_i$ that all $(i-1)$ weights higher than w_i arrive after time t_i.

Let $A(t_i) = \{w_j \colon t_j < t_i\}$ be the set of weights seen before w_i arrives. If this set is empty then w_i is selected. If not, let w_A be the top weight of $A(t_i)$. It is easy to see that under the previous conditioning, w_i is selected if and only if $t_A < p$.

Conditioned on t_i, $\mathcal{E}_i$ and on the set $A(t_i)$ being nonempty, the variable t_A is a uniform random variable in $(0, t_i)$. Hence,

$$\begin{aligned}
&\Pr(w_i \text{ is selected} \mid t_i \geq p, \mathcal{E}_i) \\
&\qquad = \Pr(A(t_i) = \emptyset \mid t_i \geq p, \mathcal{E}_i) + \Pr(A(t_i) \neq \emptyset, t_A < p \mid t_i \geq p, \mathcal{E}_i) \\
&\qquad \geq 1 \cdot \Pr(A(t_i) = \emptyset \mid t_i \geq p, \mathcal{E}_i) + \Pr(A(t_i) \neq \emptyset \mid t_i \geq p, \mathcal{E}_i) p/t_i \\
&\qquad \geq p/t_i.
\end{aligned}$$

As event $\mathcal{E}_i$ has probability $(1-t_i)^{i-1}$, we have

$$\Pr(w_i \text{ is selected}) \geq \int_p^1 \frac{p}{t_i} (1-t_i)^{i-1}\, dt_i.$$

In particular the probability that w_1 is selected is $\int_p^1 \frac{p}{t}\, dt = -p \ln p$. This is maximized by setting $p = 1/e$. For that value, the probability above also equals $1/e$.

To conclude the proof, note that the only way that this algorithm returns an empty set is when the top weight w_1 arrives before p. This happens with probability p. □

1.3 Multiple Choice Secretary Problem

A natural generalization of the classical secretary problem is the **multiple choice secretary problem**, also known as the ***r*-choice secretary problem**. In this problem, the algorithm is allowed to select a set ALG consisting on up to r elements of the stream, where r is a fixed parameter. The algorithm must follow the same rules as in the classical secretary problem: it has to select or reject elements as they appear. The **full-adversarial** model and the **random** model are defined analogously to the classical case. The **payoff** of the algorithm is, as before, the total weight of the set selected: $w(\text{ALG}) = \sum_{x \in \text{ALG}} w(x)$.

The r-choice secretary problem has a nice interpretation as an online auction. We can regard the algorithm as an auctioneer having r identical items, and the secretaries (the weights) as agents arriving at random times, each one having a different valuation for the item. The goal of the algorithm is to assign the items to the agents as they arrive while maximizing the total social welfare.

1.3.1 Previous Results for the Multiple Choice Secretary Problem

Lemma 1.3.1 (Folklore)**.** *No algorithm for the full-adversarial r-choice secretary problem admits a competitive ratio smaller than n/r.*

Proof. We offer a proof very similar to the proof of Lemma 1.2.1. Let $0 < \varepsilon < 1$ and fix any (possibly randomized) algorithm $\mathcal{A}$ for the problem with $1 \leq r \leq n$. Consider the adversarial input given by the increasing sequence of weights $\varepsilon, 2\varepsilon, 3\varepsilon, \ldots, n\varepsilon$. Let p_i be the probability that $\mathcal{A}$ selects the i-th element on this sequence. Since the expected number of elements that $\mathcal{A}$ selects is at most r:

$$\sum_{i=1}^{n} p_i \leq r. \tag{1.6}$$

In particular, there is an index j for which $p_j \leq r/n$. Consider the sequence of weights given by $\varepsilon, 2\varepsilon, \ldots, (j-1)\varepsilon, n^3, 0, \ldots, 0$. Since the information available for the algorithm on this sequence up and including the revelation of the j-th element is equal than the information available for the totally increasing sequence, the probability that $\mathcal{A}$ selects the j-th element on this second sequence is still p_j. This means that, for the second sequence:

$$\mathbb{E}[w(\text{ALG})] \leq p_j n^3 + r(j-1)\varepsilon \leq n^2(r+\varepsilon). \tag{1.7}$$

As the maximum payoff of the second sequence is bounded below by n^3, the competitive ratio of algorithm $\mathcal{A}$ is at least $w(\text{OPT})/\mathbb{E}[w(\text{ALG})] \geq n/(r+\varepsilon) \geq (1-\varepsilon)n/r$. Since ε is arbitrary, we conclude the proof. □

A natural optimum algorithm for the full-adversarial model would be simply to select a random set of r elements from the stream. Since each one of the top r weights

(say $w_1 \geq w_2 \geq \cdots \geq w_r$) is selected with probability at least r/n we obtain that $\mathbb{E}[w(\mathrm{ALG})] \geq (\sum_{i=1}^{r} w_i)r/n$, implying this algorithm is n/r-competitive.

We have seen that for the full-adversarial model, the multiple-choice secretary problem admits algorithms having better competitive ratios than the ones for the classical secretary problem. A natural question to ask is whether the same happens for the random model. Kleinberg [95] has shown that this holds asymptotically: the random multiple-choice secretary problem admits algorithms with competitive ratio tending to 1 as r goes to infinity.

Lemma 1.3.2 (Kleinberg [95])**.** *Kleinberg's algorithm for the random multiple-choice secretary problem returns a set having total expected weight at least* $1 - 5/\sqrt{r}$ *times the sum of the weights of the* r *largest elements. In our notation, his algorithm has a competitive ratio of* $1 + O\Big(\sqrt{1/r}\Big)$.

Kleinberg's algorithm is recursively defined. If $r = 1$, then simply apply Lindley–Dynkin algorithm. If $r > 1$, then let m be a random variable from the binomial distribution $\mathrm{Bin}(n, 1/2)$. On the first m weights of the stream apply recursively Kleinberg's algorithm to select $\lfloor r/2 \rfloor$ weights. Once they have been selected, let x be the top $\lfloor r/2 \rfloor$-th weight of the first m arriving ones. Use x as a threshold to select the remaining weights: accept any weight that is better than x until r weights have been selected or until we have run out of weights. Kleinberg offers a simple analysis for his algorithm in [95] which we do not reproduce. Also, he has claimed (but the proof, as far as the author knows, has not been published anywhere in the literature) that this algorithm is asymptotically optimal, in the sense that any algorithm for this problem returns a set having weight at least has competitive ratio at least $1 + \Omega\Big(\sqrt{1/r}\Big)$.

A much simpler suboptimal constant-competitive algorithm for this problem was proposed later by Babaioff et al. [6]: maintain the set T consisting on the top r elements seen so far (initialize this set with r dummy elements that are considered to be smaller than everyone else). Observe and reject the first $m = \lfloor n/e \rfloor$ elements without adding them to the output; refer to this set as the *sample*. An element arriving later will be added to the output only if at the moment it is seen, it enters T and the element that leaves T is either in the sample or a dummy element. Babaioff et al. have shown that this algorithm returns a set of at most r elements and that every element of the optimum is in the output of the algorithm with probability at least $1/e$, making this algorithm e-competitive.

The previous algorithm is deterministic. In the same way we modified Lindley–Dynkin algorithm for the classical secretary problem we can modify Babaioff et al.'s algorithm to obtain a *randomized* e-competitive algorithm having simpler analysis. The modification is depicted as Algorithm 2. The only difference with respect to the algorithm above is that the number of sampled elements is given by a binomial distribution $\mathrm{Bin}(n, p)$.

Lemma 1.3.3. *Algorithm 2 is* $(-p \ln p)^{-1}$*-competitive. In particular, by setting* $p = 1/e$*, we obtain an* e*-competitive algorithm for the random model of the multiple choice-secretary problem.*

Algorithm 2 For the random r-choice secretary problem.

1: Maintain a set T containing the top r elements seen so far at every moment. Initialize T with r dummy elements.
2: Choose m from the binomial distribution $\mathrm{Bin}(n, p)$.
3: Observe and reject the first m elements – call this set the *sample*.
4: **for** each element x arriving after the first m elements **do**
5: **if** x enters T and the element leaving T is in the sample or is a dummy element. **then** Add x to the output ALG.
6: **end if**
7: **end for**

Proof. The proof of this lemma is very similar to the proof of Lemma 1.2.4.

Consider the following offline simulation algorithm. Let $w_1 \geq w_2 \geq \cdots \geq w_n \geq 0$ be the weights in non-increasing order. Each weight w_i selects an arrival time t_i in the interval $(0, 1)$ uniformly and independently. The offline algorithm maintains a set T containing at every moment the top r weights seen so far, initializing T with a set of r dummy weights. The algorithm then processes the incoming weights in order of arrival, sampling the ones arriving before time p. When a weight arriving after time p is processed, the algorithm checks if this weight enters T and if the one leaving T is in the sample or a dummy weight. If this is the case, the algorithm selects it. Otherwise, it continues.

Since the cardinality of the sampled set has binomial distribution with parameters n and p, the set of elements returned by this simulation has the same distribution as the one returned by Algorithm 2. Hence, we analyze the simulation.

As every weight selected by the algorithm pushes out of T either a sampled weight or a dummy weight, the algorithm can select at most r elements. We estimate the probability that each one of the top r weights appears in the output set. Focus on one such weight w_i, $i \leq r$ and condition on the event that it is not sampled, i.e. that $t_i \geq p$.

Let $A(t_i) = \{w_j \colon t_j < t_i\}$ be the set of weights seen before w_i arrives. If this set has less than r elements then the element leaving T at time t_i will be a dummy weight. Consider the case where $A(t_i)$ has cardinality at least r and let w_A be the r-th top element of this set. Note that this element w_A and the time of its arrival t_A are independent random variables. In particular, conditioned on t_i, t_A is a uniform random variable in $(0, t_i)$. Hence,

$$\begin{aligned}
\Pr(w_i \text{ is selected} \mid t_i \geq p) &= \Pr(|A(t_i)| < r \mid t_i \geq p) + \Pr(|A(t_i)| \geq r, t_A < p \mid t_i \geq p) \\
&\geq 1 \cdot \Pr(|A(t_i)| < r \mid t_i \geq p) + \Pr(|A(t_i)| \geq r \mid t_i \geq p) p/t_i \\
&\geq p/t_i.
\end{aligned}$$

Then,

$$\Pr(w_i \text{ is selected}) \geq \int_p^1 \frac{p}{t_i}\, dt_i = -p \ln p.$$

Therefore,

$$\mathbb{E}[w(\text{ALG})] = \sum_{i=1}^{r} w_i \Pr(w_i \text{ is selected}) \geq (-p \ln p) \sum_{i=1}^{r} w_i.$$

By setting $p = 1/e$, we obtain a competitive ratio of e. □

In Section 2.3.1 we propose yet another simple algorithm, Algorithm 4, having a slightly worse competitive ratio, but having better behavior for some extensions of the r-choice secretary problem.

1.4 Generalized Secretary Problems

Babaioff et al. [8, 7] have proposed a framework that generalizes both the classical and multiple-choice secretary problems. In what follows, we describe the setting, models, standard assumptions and the tools we use to measure the performance of an algorithm on this framework.

1.4.1 Setting

In the **generalized secretary problem** there is a **ground set** $\mathcal{S}$ of so-called **elements** and a fixed collection of subsets $\mathcal{I} \subseteq 2^{\mathcal{S}}$ closed under inclusion that describes the sets of elements that can be simultaneously accepted. These sets are called the **feasible sets**, and $\mathcal{I}$ is known as the **domain** of the problem. Each element $x \in \mathcal{S}$ has a hidden nonnegative **weight** $w(x)$. The elements are presented to an online algorithm in a certain **order** $x_1, \ldots, x_n$.

As each element is presented the algorithm must decide whether to select it or reject it subject to the following constraints.

1. The decision to accept or reject an element is irreversible.
2. This decision must be made before the next element is presented.
3. The set of selected elements, denoted as ALG, must belong to $\mathcal{I}$ at all times.

The **payoff** of the algorithm is the total sum of the weights of the selected elements:

$$w(\text{ALG}) = \sum_{x \in \text{ALG}} w(x).$$

1.4.2 Models

We classify generalized secretary problems based on how the weights and the order of the elements are chosen.

Assignment of weights. Consider the following two variants.

1. **Adversarial-Assignment**: The weights are assigned to the elements of $\mathcal{S}$ by an adversary. This information is hidden to the algorithm.
2. **Random-Assignment**: An adversary selects a list $W = \{w_1, \ldots, w_n\}$ of weights, hidden from the algorithm. This list is assigned to the elements of $\mathcal{S}$ via a uniform random bijection.

Order of the elements. Consider the following two variants.

1. **Adversarial-Order**: An adversary selects the ordering of the elements, this order is unknown to the algorithm.
2. **Random-Order**: The elements are presented in uniform random order.

There are four models for this problem arising from combining the variants above. We assume that the random choices are performed after the adversarial choices. For example, in the adversarial-assignment random-order model, the adversary selecting the weights does not know the order in which the elements are going to be presented. If randomness is involved for both parameters, we assume the choices to be independent random variables.

1.4.3 Performance Tool: Competitive Analysis

We use the same performance tool as in the classical and multiple-choice secretary problems: the competitive ratio.

For any given model, we say that a (randomized) algorithm is **α-competitive** if for every adversarial selection that the model allows,

$$\mathbb{E}[\alpha w(\mathrm{ALG}) - w^*] \geq 0, \tag{1.8}$$

where the expectation is over all the random choices given by the model and the random choices performed by the algorithm and w^* is the maximum payoff of a feasible set under the current assignment of weights. The **competitive ratio** of an algorithm is the minimum value of α for which this algorithm is α-competitive.

1.4.4 Standard Assumptions

Unless explicitly stated otherwise, we assume that the algorithm has full access to the ground set $\mathcal{S}$ and the family $\mathcal{I}$ at all moments, even if $\mathcal{I}$ is exponentially larger than $\mathcal{S}$ (It is often the case though, that a compact representation of $\mathcal{I}$ is available.) When an element x is revealed, the algorithm learns its **role** in $\mathcal{S}$ (that is, it is able to distinguish x from any other element in $\mathcal{S}$).

We make this caveat here since one could consider a scenario in which we replace the above ability by the following one: the algorithm can only test if a given collection of elements already revealed is feasible or not. For example, consider the case where

the feasible sets are all the sets of size at most 2, except for a single pair $X = \{x, y\}$. In this setting we can not know if the element we are accepting is x or y until both elements have already been presented.

We also assume that at every moment, the algorithm is able to compare the values of two previously presented elements, but it is not able to see their numerical values. In other words, we look for comparison-based algorithms.

Regarding computational power, we assume that between the arrival of two consecutive elements, the algorithm is able to compute any function on the data it already knows. The polynomiality of the algorithm is not the real issue: unbounded computational time and space, or even the ability to solve the halting problem is not enough, in some cases, to achieve small competitive ratios. Having a polynomial-time algorithm is seen as a desirable feature, but it is not a requirement.

1.4.5 Special Cases

The two simplest problems fitting the settings for the generalized secretary problem are the classical and multiple-choice secretary problems. In these problems, the domains are the collection of all sets having cardinality at most one (classical case) or at most a fixed number r (multiple-choice case). We denote these domains as *uniform domains*.

Since uniform domains are totally symmetric, some of the models we have defined in Section 1.4.2 coincide. The adversarial-assignment adversarial-order model on uniform domains is equivalent with what we called the *full-adversarial model* for the classical and multiple-choice secretary problems. The other three models (random-assignment random-order model; random-assignment adversarial-order model; and adversarial-assignment random-order model) are equivalent to the *random model* for the classical and multiple-choice secretary problem.

Generalized secretary problems on different domains have many applications, specially for online auctions. Consider the following examples.

1. A popular website has space for advertising. This space is used to display a (probably) different banner ad to every visitor. The owner wishes to sell the ad slot to different advertisers. Each one arrives with a request for some number of impression to be displayed on a given day, and the price he is willing to pay. The owner accepts or rejects each request provided that the total number of impressions sold does not exceed the estimated number of different impressions the site gets that day. In the formalism described before, each element x of the set $\mathcal{S}$ of advertisers is assigned a *length* f_x, defined as the fraction of the total number of impressions he wants, and the domain $\mathcal{I}$ consists on those sets whose total length is at most 1. The domain for this problem is called a *knapsack domain*.

2. An auctioneer has a set M of non-identical items. Each agent $x \in \mathcal{S}$ wants to buy a specific item but the auctioneer is not allowed to sell more than $k \leq |M|$ items. The domain $\mathcal{I}$ of feasible sets that the auctioneer can choose in

the associated generalized secretary problem is known as a *truncated partition matroid*.

3. Certain internet provider has a collection of new high speed servers in a town, interconnected via routers forming a network. In order to have access to these new servers, clients must buy private routes. The sets of clients that this provider can serve are exactly those sets which can be connected to the servers using vertex disjoint paths. The domain $\mathcal{I}$ of feasible sets of clients is known as a *gammoid*.

Besides the examples above, there are other domains of interests: For instance *matching domains* (set of matchings of a graph or hypergraph), *matroid domains* (independent sets of a given matroid) and *matroid intersection domains*. Matroid domains appear often in many applications, for that reason there has been a long line of work dedicated to them. In Section 1.5 we focus on the matroid secretary problem and give some background on previous work.

1.4.6 Lower Bound for Generalized Secretary Problems

There is not much to say about the adversarial-assignment adversarial-order model for general domains. Since this problem includes uniform domains of unit size we know, using Lemma 1.2.1, that no algorithm achieves a competitive ratio smaller than n for general domains. For this reason, we only focus on the other three *random* models: adversarial-assignment random-order, random-assignment adversarial-order and random-assignment random-order.

Uniform domains admit constant-competitive algorithms on these three models (as they are equivalent on these domains). However, the following example, given by Babaioff et al. [8] shows that general domains do not have this feature.

Let $\mathcal{S} = [n]$ be the ground set and let $r = \lfloor \ln n \rfloor$. Partition $\mathcal{S}$ into $m = \lceil n/r \rceil$ subsets $S_1, \ldots, S_m$ each having size r or $r-1$. Consider the domain

$$\mathcal{I} = \{I \subseteq [n] \colon I \subseteq S_i \text{ for some } i\}. \tag{1.9}$$

Lemma 1.4.1 (Babaioff et al. [8])**.** *For any model of the generalized secretary problem, no algorithm has competitive ratio smaller than* $o(\log n / \log\log n)$ *for the domain* $\mathcal{I}$ *defined above.*

Proof. Babaioff et al.'s argument is to show that under certain independent assignment of weights, the expected weight of the set selected by any online algorithm is less than 2 while the expected value of the maximum weight in $\mathcal{I}$ is $\Omega(\log n / \log\log n)$. We reproduce their argument here for completeness.

For every element $x \in \mathcal{S}$, assign a weight $w(x)$ at random with the following probabilities:

$$\Pr(w(x) = 1) = \frac{1}{r} \text{ and } \Pr(w(x) = 0) = 1 - \frac{1}{r}. \tag{1.10}$$

Consider any randomized algorithm $\mathcal{A}$ for this problem. Suppose that the first element that $\mathcal{A}$ selects in one execution is $x \in S_i$. Then all the elements selected later must

also be in S_i. The weights of those elements are independent from the information observed so far. Since the weight of x is at most 1 and there are at most r additional elements that can be selected, each with expected value $1/r$, the expected value of the set selected by the algorithm is at most 2. Note that this is true *regardless* of the order in which the elements are presented: Even if the algorithm could *choose* the order in which the elements are presented, the expected value returned would still be at most 2.

For the rest of the proof, let $j = \lfloor \frac{r}{2 \ln r} \rfloor \leq \frac{\ln n}{2 \ln \ln n}$ and let $\mathcal{E}_i$ be the event that at least j elements of S_i have value 1. The probability of this event is

$$\Pr\left(\sum_{x \in S_i} w(x) \geq j\right) \geq \left(\frac{1}{r}\right)^j \geq \left(\frac{1}{\ln n}\right)^{\frac{\ln n}{2 \ln \ln n}} = \frac{1}{\sqrt{n}}. \tag{1.11}$$

Since the events $\mathcal{E}_1, \ldots, \mathcal{E}_m$ are independent, the probability that none of them occurs is at most $\left(1 - \frac{1}{\sqrt{n}}\right)^m = o(1)$. If one of the events $\mathcal{E}_i$ occurs, then the corresponding set $S_i \in \mathcal{I}$ has weight at least j. The previous implies that the expected value of the maximum weight set in $\mathcal{I}$ is at least $j = \Omega\left(\frac{\log n}{\log \log n}\right)$. □

The previous lower bound for the optimal competitive ratio is probably not tight. Currently, there are no known upper-bounds for generalized secretary problems apart from the trivial bound of n.

1.5 Matroids

In view of Lemma 1.4.1 in the previous section, a natural question is to determine other domains (apart from uniform) admitting constant-competitive algorithms for some of the three random models. Matroid domains are interesting candidates to consider. In this section, we define these domains and give some important properties and examples.

A **matroid** is a pair $\mathcal{M} = (\mathcal{S}, \mathcal{I})$, where $\mathcal{S}$ is denoted as the **ground set** and $\mathcal{I} \subseteq 2^{\mathcal{S}}$ is the family of **independent sets**, satisfying the following properties.

1. (*Nonempty*) $\emptyset \in \mathcal{I}$.
2. (*Hereditary*) If $A \in \mathcal{I}$ and $B \subseteq A$, then $B \in \mathcal{I}$.
3. (*Augmentation*) If $A, B \in \mathcal{I}$ and $|A| < |B|$, then there is $x \in B \setminus A$ such that $A \cup \{x\} \in \mathcal{I}$.

Matroids satisfy many properties. For a nice exposition of matroids we refer the reader to Oxley's [137] and Schrijver's [147] books.

In this thesis, we use the following matroid related definitions. A **basis** of a matroid $\mathcal{M}$ is an inclusion-wise maximal independent set. The augmentation property of matroids guarantees that all the bases of $\mathcal{M}$ have the same cardinality. The size of a base is known as the **total rank** of $\mathcal{M}$. The **rank** of a set A in $\mathcal{M}$, denoted as

$\mathrm{rk}_{\mathcal{M}}(A)$ is the size of any maximal independent set contained in A. A **circuit** is a minimal non-independent set, and a **loop** is a singleton circuit. The **span** of a set A, denoted $\mathrm{span}(A)$ is the set of all elements x in $\mathcal{S}$, such that $\mathrm{rk}_{\mathcal{M}}(A) = \mathrm{rk}_{\mathcal{M}}(A \cup \{x\})$.

An important property, that we often use, is that the rank function $\mathrm{rk}_{\mathcal{M}} \colon \mathcal{S} \to \mathbb{Z}_+$ is submodular, this is,

$$\mathrm{rk}_{\mathcal{M}}(A \cup B) + \mathrm{rk}_{\mathcal{M}}(A \cap B) \leq \mathrm{rk}_{\mathcal{M}}(A) + \mathrm{rk}_{\mathcal{M}}(B), \quad \text{for all } A, B \subseteq \mathcal{S}. \tag{1.12}$$

In the following sections we define some matroids and explore some properties.

1.5.1 Operations on Matroids

There are many ways to construct matroids, starting from existing ones. Here we describe some of the operations that allow us to do that.

1. **Truncation:** Given $\mathcal{M} = (\mathcal{S}, \mathcal{I})$ and an integer $k \geq 0$, the truncation of $\mathcal{M}$ to size k is the matroid $\mathcal{M}^{(k)} = (\mathcal{S}, \mathcal{I}')$ having the same ground set, where a set is independent in the new matroid if it is also independent in the original one and has cardinality at most k.

2. **Restriction:** Given $\mathcal{M} = (\mathcal{S}, \mathcal{I})$ and a subset $E \subseteq \mathcal{S}$ of elements, the restriction of $\mathcal{M}$ to E is the matroid $\mathcal{M}|_E = (E, \mathcal{I}')$ having E as ground set and where a subset of E is independent if it is independent in the original matroid.

3. **Contraction:** Given $\mathcal{M} = (\mathcal{S}, \mathcal{I})$ and a subset $E \subseteq \mathcal{S}$, the matroid obtained by contracting E is $\mathcal{M}/E = (\mathcal{S} \setminus E, \mathcal{I}')$ having $\mathcal{S} \setminus E$ as ground set, where the independent sets are defined as follows. Consider any maximal independent set X inside E, a set I is independent in $\mathcal{M}/E$ if and only if $I \cup X$ is independent in $\mathcal{M}$.

4. **Duality:** Given $\mathcal{M} = (\mathcal{S}, \mathcal{I})$, the dual matroid $\mathcal{M}^* = (\mathcal{S}, \mathcal{I}^*)$ is the matroid having the same ground set as $\mathcal{M}$ where the bases are the complement of the bases of $\mathcal{M}$. Equivalently, a set I is independent in $\mathcal{M}^*$ if $\mathcal{S} = \mathrm{span}_{\mathcal{M}}(\mathcal{S} \setminus I)$.

5. **Direct Sum:** Given two matroids $\mathcal{M}_1 = (\mathcal{S}_1, \mathcal{I}_1)$ and $\mathcal{M}_2 = (\mathcal{S}_2, \mathcal{I}_2)$, the direct sum $\mathcal{M}_1 \oplus \mathcal{M}_2$ is the matroid having as ground set two disjoint copies of $\mathcal{S}_1$ and $\mathcal{S}_2$ (i.e. even if $\mathcal{S}_1$ and $\mathcal{S}_2$ intersect in an element x, there are two copies of this element in the new ground set), where a set I is independent if and only if the set of elements of I belonging to the copy of $\mathcal{S}_i$ is independent in $\mathcal{M}_i$ for both $i = 1$ and $i = 2$.

6. **Union:** Given two matroids $\mathcal{M}_1 = (\mathcal{S}_1, \mathcal{I}_1)$ and $\mathcal{M}_2 = (\mathcal{S}_2, \mathcal{I}_2)$ where the ground sets possibly intersect, the union $\mathcal{M}_1 \cup \mathcal{M}_2$ is the matroid having $\mathcal{S}_1 \cup \mathcal{S}_2$ as ground set and where a set I is independent if it can be written as $I_1 \cup I_2$, where $I_1 \in \mathcal{I}_1$ and $I_2 \in \mathcal{I}_2$.

1.5.2 Matroid Examples

1. **Free matroids.**
   The free matroid on $\mathcal{S}$ is the matroid having ground set $\mathcal{S}$ where every set is independent.

2. **Uniform matroids.**
   For $r \geq 0$, the uniform matroid $\mathcal{U}(\mathcal{S}, r)$ is the matroid having $\mathcal{S}$ as ground set and where a set I is independent if it has cardinality at most r. In other words, it is the truncation of the free matroid on $\mathcal{S}$ to size r.

3. **Partition matroids.**
   A partition matroid is the direct sum of uniform matroids.

4. **Transversal matroids.** A matroid $\mathcal{M} = (\mathcal{S}, \mathcal{I})$ is transversal if there is a family of subsets $C_1, \ldots, C_r \subseteq \mathcal{S}$ such that a set I is independent if and only if we can assign each element of I to a set C_i in the family containing it, in an injective way. In other words, a transversal matroid is the union of rank 1 uniform matroids. Equivalently, one can define transversal matroids via a bipartite graph: the ground set are the vertices on one part of the bipartition, say A, and the independent sets are those subsets of A that can be covered by a matching.

5. **Laminar matroids.** A matroid $\mathcal{M} = (\mathcal{S}, \mathcal{I})$ is a laminar matroid if there is a laminar family[1] $\mathcal{L}$ of subsets of $\mathcal{S}$ and a collection of nonnegative integers $\{k_L\}_{L \in \mathcal{L}}$ such that I is independent if and only if $|I \cap L| \leq k_L$ for all $L \in \mathcal{L}$. Partition matroids and their truncations are examples of laminar matroids.

6. **Gammoids.** Consider a graph $G = (V, E)$ and two special sets of vertices $S, T \subseteq V$. The gammoid $\mathcal{M} = (S, \mathcal{I})$ induced by G, S and T is the matroid having S as ground set and where a set $I \subseteq S$ is independent if there is a collection of vertex-disjoint paths connecting each element of I to an element of T.

7. **Graphic matroids.** Consider a graph $G = (V, E)$. The graphic matroid $\mathcal{M}(G) = (E, \mathcal{I})$ is the matroid having the edges of G as ground set and where a set of edges is independent if it does not contain a cycle.

8. **Linear matroids.** Given a collection of vectors $V \subseteq \mathbb{F}^k$ in a finite-dimensional vector space over a field $\mathbb{F}$, the associated linear matroid $\mathcal{M} = (V, \mathcal{I})$ is the one having V as ground set and where a set of vectors is independent if and only if they are linearly independent in $\mathbb{F}^k$.

[1] A family is laminar if every time two sets A and B in the family have nonempty intersection, they are disjoint or one is contained in the other.

1.5.3 Greedy Algorithm

Consider a matroid $\mathcal{M} = (\mathcal{S}, \mathcal{I})$ and an ordering

$$\sigma(1), \sigma(2), \ldots, \sigma(n)$$

of its elements, where $\sigma\colon [n] \to \mathcal{S}$ is a bijective function. The **greedy procedure** on the ordered sequence above is the following. Initialize I as the empty set and then for all i from 1 to n, check if $I \cup \{\sigma(i)\}$ is independent. If so, add $\sigma(i)$ to I. Otherwise, continue.

The set $I = \{x_1, \ldots, x_r\}$ returned by this procedure is not only independent, but also a base of the matroid. This base is called the **lexicographic first base** of $\mathcal{M}$ according to the ordering σ. The name comes from the following lemma.

Lemma 1.5.1. *Let $I = \{x_1, \ldots, x_r\}$ be the base of $\mathcal{M}$ obtained by the greedy procedure on the sequence $\sigma(1), \ldots, \sigma(n)$, sorted in increasing order of σ^{-1}. Let $J = \{y_1, \ldots, y_r\}$ be a different base of the matroid, sorted again according to σ^{-1} and let j be the maximum index such that*

$$\{x_1, \ldots, x_j\} = \{y_1, \ldots, y_j\}.$$

Then

$$\sigma^{-1}(x_{j+1}) < \sigma^{-1}(y_{j+1}).$$

Furthermore, if $v\colon \mathcal{S} \to \mathbb{R}_+$ is a nonnegative weight function on the ground set such that

$$v(\sigma(1)) \geq v(\sigma(2)) \geq \cdots \geq v(\sigma(n)), \tag{1.13}$$

then the lexicographic first base I according to σ is the base having maximum total weight.

Proof. Suppose by contradiction that $\sigma^{-1}(x_{j+1}) > \sigma^{-1}(y_{j+1})$. Then, the greedy procedure processes y_{j+1} before x_{j+1} and, since $\{x_1, \ldots, x_j, y_{j+1}\} \subseteq J$ is independent, this algorithm would have added y_{j+1} to I, contradicting the hypothesis.

To prove the second claim, suppose that $J = \{y_1, \ldots, y_r\}$ is a base having strictly bigger weight than the lexicographic first base $I = \{x_1, \ldots, x_r\}$, where the elements are sorted according to their σ^{-1} value. Let k be the first index for which $v(y_k) > v(x_k)$. Consider the sets $A = \{x_1, \ldots, x_{k-1}\} \subseteq I$ and $B = \{y_1, \ldots, y_{k-1}, y_k\} \subseteq J$. Since $|A| < |B|$ and both sets are independent, there is an element $y_i \in B \setminus A$ so that $A \cup \{y_i\}$ is independent. But then, as $\sigma^{-1}(y_i) \leq \sigma^{-1}(y_k) < \sigma^{-1}(x_k)$, the greedy algorithm would have included y_i into I before processing x_k, which is a contradiction. This means that $v(y_k) \leq v(x_k)$ for all k, in particular, $v(J) \leq v(I)$. □

The previous lemma states that in order to find a maximum weight independent set of a nonnegatively weighted matroid, it is enough to sort the elements in decreasing order of weight and apply the greedy procedure to the sequence obtained. In fact, as Gale [67] and Edmonds [50] have shown, the previous property characterizes matroids as the only hereditary families for which the previous algorithm outputs a maximum weight set in the family.

Interestingly enough, the lexicographic first base according to σ can be obtained also by a different greedy-like algorithm.

Consider an arbitrary ordering of the elements of the matroid (not necessarily equal to σ): $\pi(1), \ldots, \pi(n)$. Perform the following procedure. Start with an empty set I. For all i from 1 to n, check if $I \cup \{\pi(i)\}$ is independent. If so, add $\pi(i)$ to I. Otherwise, find the element $x \in I$ with $\sigma^{-1}(x)$ as large as possible such that the set obtained from I by swapping x and $\pi(i)$ is independent. If $\sigma^{-1}(x) > \sigma^{-1}(\pi(i))$ perform the swap and continue. Otherwise, ignore element $\pi(i)$ and continue.

Lemma 1.5.2. *The greedy-like algorithm described above returns the lexicographic first base of $\mathcal{M}$ according to σ.*

Proof. Let B be the lexicographic first base of $\mathcal{M}$. Consider an element b of this base and let I be the independent set considered by the algorithm above just before b is considered. If b does not enter I at that moment, there must be a circuit $C \subseteq (I \cup \{b\})$ containing b. Let x be the element of $C \setminus \{b\}$ having largest $\sigma^{-1}(x)$. We must have $\sigma^{-1}(x) > \sigma^{-1}(b)$ as otherwise b would be in the span of elements having strictly smaller value of σ^{-1}, contradicting the fact that b is in the lexicographic first base. Therefore, x and b are swapped and b does enter the independent set.

Suppose now that at some moment b is swapped out by another element y arriving later. This means that there is a circuit C' containing y and b where b is the element having largest $\sigma^{-1}(\cdot)$-value. Again, this is a contradiction as b would be spanned by elements of smaller value of σ^{-1}. □

During the rest of this work we use many matroid properties that we do not describe in detail. Whenever we do so, we give a pointer to a reference where the property can be found.

1.5.4 Matroid Secretary Problems

Suppose that we relax the first condition in the setting of the generalized secretary problem, by allowing to revoke elements from the selected set. Consider the following greedy-like algorithm for a given domain $\mathcal{I}$: Whenever a new element can be added to the current set, do it. If this is not the case, check if it is possible to improve the total weight of the current selected set by discarding a subset of previous elements and selecting the new one while keeping feasibility[2]. If this is possible, perform the most beneficial swap. As noted in the previous section, this greedy algorithm is guaranteed to select the maximum weight feasible set when $\mathcal{I}$ is the family of independent sets of a matroid. In fact, this property characterizes matroids: Gale's [67] and Edmonds's [50] proofs can be modified to show that the previous greedy-like algorithm is guaranteed to return a maximum weight feasible set for every assignment of weights if and only if $\mathcal{I}$ defines a matroid.

[2]Checking the weight improvement of swapping the new element for an arbitrary subsets of previous elements requires the ability to see the values of the elements. However, if we only allow to discard singletons we can do it with comparisons.

The intuition in the last paragraph motivated Babaioff et al. [8, 7] to propose the following conjecture.

Conjecture 1.5.3 (Babaioff et al. [8, 7])**.** *The matroid secretary problem admits a constant competitive algorithm (where the constant is perhaps even e), for every one of the random models proposed.*

The conjecture above states that the obligation to honor past commitments is not too costly for matroids, as it only reduces the expected selected value by a constant. In Chapter 2 we partially answer the above conjecture by showing an affirmative answer for two of the three random models proposed.

For the random-assignment random-order model, the author has presented a constant-competitive algorithm in a recent SODA paper [152], which we improve in this work. As first noticed by Oveis Gharan and Vondrák [136], it is possible to apply the methods in [152] to also obtain constant-competitive algorithms for the random-assignment adversarial-order model. In this thesis, we present alternative algorithms achieving the same task.

The adversarial-assignment random-order model is still open, even for the case in which we allow to use value-based algorithms. For this last particular case, Babaioff et al. [8] have given an $O(\log r)$-competitive algorithm, where r is the total rank of the matroid.

Lemma 1.5.4 (Babaioff et al.[8])**.** *There is an $O(\log r)$-competitive algorithm for the adversarial-assignment random-order model of the matroid secretary problem, under the assumption that we can see the numerical values of the elements as they arrive.*

In Section 2.6.1 we prove that it is indeed possible to achieve a competitive ratio of $O(\log r)$ using a comparison-based algorithm.

There is a long line of work dedicated to devise constant competitive algorithms for specific matroid classes on the adversarial-assignment random-order model. Before describing these works we make the following important observations.

Lemma 1.5.5 (Folklore)**.**

1. *If $\mathcal{M}$ admits an α-competitive algorithm for the adversarial-assignment random-order matroid secretary problem, then so do all the restrictions of $\mathcal{M}$.*

2. *If $\mathcal{M}_1$ and $\mathcal{M}_2$ admit an α-competitive algorithm for the adversarial-assignment random-order matroid secretary problem then so does the direct sum $\mathcal{M}_1 \oplus \mathcal{M}_2$.*

Proof. For the first item, let $\mathcal{M}'$ be a restriction of a matroid $\mathcal{M}$ and $\mathcal{A}$ be an α-competitive algorithm for $\mathcal{M}$. Consider the modified algorithm $\mathcal{A}'$ that virtually extends $\mathcal{M}'$ to $\mathcal{M}$ by adding a set of zero weight dummy elements. Then, this algorithm simulates the augmented input in such a way that the dummy elements arrive uniformly at random similarly to the real ones. The resulting algorithm will also have a competitive ratio of α.

For the second point, run the corresponding α-competitive algorithms in parallel both in $\mathcal{M}_1$ and $\mathcal{M}_2$. In other words, when an element is processed, use the

corresponding algorithm and accept or reject it accordingly. This algorithm is α-competitive in the direct sum matroid. □

Using Karger's matroid sampling theorem [93], Babaioff et al. [8] have shown the following result: If an α-competitive algorithm is known for a given matroid $\mathcal{M}$ then they can devise a $\max(13\alpha, 400)$-competitive algorithm for any truncation of $\mathcal{M}$.

The previous paragraphs imply that the class of matroids admitting a constant-competitive algorithm for the adversarial-assignment random-order matroid secretary problem is closed under truncation, restriction and direct sum. It is still open if this class is also closed for duality, contraction and union.

In what follows we mention some of the previous results for the adversarial-assignment random-order matroid secretary problem.

Uniform and Partition Matroids. The uniform case corresponds exactly to the classical and multiple choice secretary problems described in Sections 1.2 and 1.3, for which e-constant competitive algorithms exist (e.g. Algorithms 1 and 2). Since partition matroids are direct sum of uniform matroids, they also admit e-competitive algorithms.

Transversal Matroids. Recall that transversal matroids can be defined using a bipartite graph, where the ground sets are the vertices on the left part, and the independent sets are those subsets of the left part that can be covered by a matching. Babaioff et al. [8] have given a $4d$-competitive algorithm for these matroids, where d is the maximum left-degree of the bipartite graph defining the matroid. In a later article, Dimitrov and Plaxton [44] give a 16-competitive algorithm for arbitrary transversal matroids. Korula and Pál [97] improve their analysis to show that their algorithm is in fact 8-competitive. Later, Thain [162], using a result by Goel and Mehta [72], shows that Dimitrov and Plaxton's algorithm is $4/(1 - 1/e) \approx 6.33$-competitive.

Graphic Matroids. Babaioff et al. [8] have shown a 16-competitive algorithm for graphic matroids, by modifying slightly their $4d$-competitive algorithm for transversal matroids. Later, Korula and Pál [97] give an improved $2e$-competitive algorithm for this class. Around the same time, Babaioff et al. [5] give an alternative $3e$-competitive algorithm.

Laminar Matroids. Very recently, Im and Wang [87] have shown that laminar matroids also admit constant competitive algorithms.

1.6 Related Work

As discussed in the introductory section of this chapter, the secretary problem admits many variants and extensions. In this section we present a very limited discussion about other studied variants.

An old line of work involve variants of the classical secretary problem where the algorithm is allowed to see values, and there is some information available a priori on the distribution of the incoming weights.

Gilbert and Mosteller [70] study the **full-information game** where we want to maximize the probability of selecting the top value of a sequence of weights drawn from a known (continuous) distribution. By applying a monotonic transformation, they argue that it is enough to focus on the uniform distribution on the interval $(0, 1)$ and give a strategy that selects the top element with probability at least $0.58016\ldots$, which is much larger than what we can get using comparison-based algorithms.

An intermediate problem between the full-information game and the classical secretary problem using only comparisons is called the **partial information game** in which weights are drawn from a distribution of known form but containing one or more unknown parameters; for example, a uniform distribution over an interval of unknown extremes. Many authors have studied this problem (see, e.g. Ferguson's survey [55]). Interestingly enough there are families of distributions for which the best strategy, in the limit when the number of elements tends to infinity, achieves a probability of at most $1/e$ [157]; for these distributions, value-based algorithms have no advantage over comparison-based ones.

Other variants of the classical secretary problem involve considering objective functions different from the total weight recovered or the probability of selecting the top weight. For example, Lindley [101] himself give a recurrence to solve the problem of minimizing the expected rank of the selected element, i.e. its ordinal position on the sorted list. Chow et al. [31] solve this problem completely, by showing that as n goes to infinity, the optimal strategy recovers an element of expected rank $\prod_{j=1}^{\infty}\left(\frac{j+2}{j}\right)^{1/(j+1)} \approx 3.8695\cdots$. A related problem, still open today, is Robbins' secretary problem in which we want to minimize the expected rank using a value-based algorithm and knowing that the weights are selected uniformly and independently in $(0, 1)$. It is know that the optimal achievable rank, as n goes to infinity tends to some value between 1.908 and 2.329, but the exact value is not yet known (see [20]).

Gusein-Zade [78] consider the problem of accepting any of the best r candidates. This corresponds to find the optimal stopping polity for an utility function that is equal to 1 if the selected element is one of the top r candidates and 0 otherwise. Mucci [119] was the first person to study general utility functions depending only on the ordinal position of each element in the sorted list.

Another possibility is to consider an objective function where the gain of selecting an element equals the product of its weight and a discount factor depending on the *time* we select it. Discounted versions of the secretary problem have been studied by Rasmussen and Pliska [143] and Babaioff et al. [5].

The r-choice secretary problem also admits different variants by changing the objective function. Ajtai et al. [1] consider the problem of minimizing the expected sum of the ranks of the selected objects. Babaioff et al. [5] consider the following weighted secretary problem in which up to r elements can be selected: at the moment an element is selected, it must be assigned to one of r possible positions, each one having a known weight. The benefit of assigning an element to a position is the product

of the element weight and the position weight, and the objective is to maximize the total benefit. Buchbinder et al. [21] study the J-choice K-best secretary problem, in which one can select at most J elements receiving profit for every element select from the top K of the stream. A different approach was taken by Bateni et al. [10], who consider the case where the gain of selecting a set is a nonnegative submodular objective function, which can be accessed only as the element arrive.

There has also been some work for the adversarial-assignment random-order generalized secretary problem on non-matroidal domains. Babaioff et al. [6] show a $10e$-competitive algorithms for knapsack domains even in the case where both the weights and lengths are revealed online. Korula and Pál [97] give constant competitive algorithms for some cases of intersection of partition matroids, specifically for matchings in graphs and hypergraphs where the edges have constant size. Im and Wang [87] have also studied an interval scheduling secretary problem that generalizes knapsack domains.

Chapter 2

New Results for Matroid Secretary Problems

2.1 Preliminaries

In this chapter we give new algorithms for matroid secretary problems. Here we briefly review the notation used.

Consider a matroid $\mathcal{M} = (\mathcal{S}, \mathcal{I})$ with ground set $\mathcal{S}$ of size n. An adversary selects a set W of n nonnegative weights $w_1 \geq \cdots \geq w_n \geq 0$, which are assigned to the elements of the matroid using an ordering of $\mathcal{S}$,

$$\sigma(1), \sigma(2), \sigma(3), \ldots, \sigma(n);$$

defined by a bijective map $\sigma\colon [n] \to \mathcal{S}$. In other words, the weight assignment is given by

$$w(\sigma(i)) = w_i.$$

The elements are then presented to an online algorithm in the order

$$\pi(1), \pi(2), \pi(3), \ldots, \pi(n);$$

defined by a certain bijective function $\pi\colon [n] \to \mathcal{S}$.

Depending on how the assignment σ and the ordering π are selected we recover the four models discussed in the previous chapter. Each one can be selected in an adversarial way or uniformly at random.

When an element is presented, the algorithm must decide whether to add it or not to the current solution set, denoted as ALG, under the condition that this set is independent ($\text{ALG} \in \mathcal{I}$) at all times. The objective is to output a solution set whose payoff $w(\text{ALG}) = \sum_{e \in \text{ALG}} w(e)$ is as high as possible.

We focus on *comparison-based algorithms*: We assume that when the i-th element of the stream, $\pi(i)$, is presented, *the algorithm only learns the relative order of the weight with respect to the previously seen ones.* This is, it can compare $w(\pi(j))$ with $w(\pi(k))$ for all $j, k \leq i$, but it can not use the actual numerical values of the weights. Without loss of generality, we assume that there are no ties in W, because otherwise

we break them using a new random permutation τ (independent of W, σ and π); if the comparison between two elements seen gives a tie, then we consider heavier the one having larger τ-value.

As usual, we say that an algorithm $\mathcal{A}$ returning an independent set ALG is α-competitive, if for any adversarial selection the model permits $\mathbb{E}[\alpha w(\text{ALG}) - w^*] \geq 0$, where the expectation is taken over all the random choices given by the model and the random choices performed by the algorithm, and w^* is the maximum possible payoff of an independent set in $\mathcal{I}$.

It is important to remark that for the matroid secretary problem, w^* depends on both W and σ, however the *set* achieving this maximum payoff depends only on σ.

Indeed, let $\text{OPT}_{\mathcal{M}}(\sigma)$ be the the *lexicographic first base* of $\mathcal{M}$ under ordering σ. In other words, $\text{OPT}_{\mathcal{M}}(\sigma)$ is the set obtained by applying the *greedy procedure* that selects an element if it can be added to the previously selected ones while preserving independence in $\mathcal{M}$, on the sequence $\sigma(1), \sigma(2), \ldots, \sigma(n)$.

Lemma 1.5.1 states that $\text{OPT}_{\mathcal{M}}(\sigma)$ is a maximum weight independent set with respect to any weight function v for which $v(\sigma(1)) \geq \cdots \geq v(\sigma(n)) \geq 0$. In particular, this is true for the weight function w defined before and $\mathbb{E}[w^*] = \mathbb{E}_{\sigma}[w(\text{OPT}_{\mathcal{M}}(\sigma))]$. We drop the subindex $\mathcal{M}$ in $\text{OPT}_{\mathcal{M}}(\sigma)$ whenever there is no possible confusion.

2.2 Divide and Conquer

A natural *divide and conquer* approach for any model of the matroid secretary problems is the following.

Given a matroid $\mathcal{M} = (\mathcal{S}, \mathcal{I})$, find matroids $\{\mathcal{M}_i = (E_i, \mathcal{I}_i)\}_i$, with $\mathcal{S} = \cup_i E_i$ satisfying three properties:

(i) Any independent set of $\bigoplus_i \mathcal{M}_i$ is independent in $\mathcal{M}$.

(ii) For each matroid $\mathcal{M}_i$, we have access to an α-competitive algorithm $\mathcal{A}_i$ for the same model.

(iii) The maximum (expected) weight in $\bigoplus_i \mathcal{M}_i$ is at least β times the maximum (expected) weight in $\mathcal{M}$.

If we can find such family of matroids, then we claim that the following algorithm is (at most) α/β-competitive: Run in parallel the α-competitive algorithm in every $\mathcal{M}_i$ and return the union of the answers. Call this algorithm $\mathcal{A}$.

To prove this claim, note first that the sub-algorithm $\mathcal{A}_i$ controlling the elements of $\mathcal{M}_i$ effectively receives an instance of the same model as the one we are trying to solve in the full matroid: If the ordering of the entire ground set was adversarial or uniformly at random selected, then so is the ordering of the elements presented to $\mathcal{A}_i$, and the same hold for the weight assignment.

Also, by conditions (i) and (ii), the set that algorithm $\mathcal{A}$ returns is independent in $\mathcal{M}$. By the second condition, the expected weight of the returned set is at least $1/\alpha$ times the expected maximum weight of $\bigoplus_i \mathcal{M}_i$. The claim follows immediately now from the third condition.

We use the divide and conquer approach above to devise constant-competitive algorithms for the random-assignment models of the matroid secretary problem.

In Section 2.3 we show a natural class of matroids admitting simple constant-competitive algorithms for random-assignment models: the class of *uniformly dense matroids*. In Section 2.4 we show how to use the powerful notion of *principal partition* to obtain a collection of uniformly dense matroids satisfying condition (i) above. We also show that a condition slightly weaker than (iii) is satisfied for the considered matroids. In Section 2.5 we study the obtained algorithms in detail.

2.3 Uniformly Dense Matroids

In order to give constant competitive algorithms for random-assignment models we start by defining a class of matroids for which this task is relatively simple to attain.

Define the **density** $\gamma(\mathcal{M})$ of a *loopless matroid*[1] $\mathcal{M} = (\mathcal{S}, \mathcal{I})$ with rank function $\mathrm{rk}\colon \mathcal{S} \to \mathbb{Z}_+$ to be the maximum over all nonempty sets X of the quantity $|X|/\mathrm{rk}(X)$.

The matroid $\mathcal{M}$ is **uniformly dense** if $\gamma(\mathcal{M})$ is attained by the entire ground set; that is, if

$$\frac{|X|}{\mathrm{rk}(X)} \leq \frac{|\mathcal{S}|}{\mathrm{rk}(\mathcal{S})}, \tag{2.1}$$

for every nonempty $X \subseteq \mathcal{S}$.

Examples of uniformly dense matroids include uniform matroids, the graphic matroid of a complete graph, and all the linear matroids arising from projective geometries $PG(r-1, \mathbb{F})$.

The following properties are important for our analysis.

Lemma 2.3.1. *Let $(x_1, \ldots, x_j)$ be a sequence of different elements of a uniformly dense matroid chosen uniformly at random. The probability that element x_j is selected by the greedy procedure on that sequence is at least $1 - (j-1)/r$, where $r = \mathrm{rk}(\mathcal{S})$.*

Proof. An element is selected by the greedy procedure only if it is outside the span of the previous elements. Denote by $A_i = \{x_1, \ldots, x_i\}$ the set of the first i elements of the sequence, and let n be the number of elements of the matroid, then:

$$\begin{aligned}\Pr[x_j \text{ is selected}] &= \frac{n - |\mathrm{span}(A_{j-1})|}{n - |A_{j-1}|} \geq \frac{n - \mathrm{rk}(A_{j-1})n/r}{n - (j-1)} \\ &\geq \frac{n}{n-(j-1)}\left(1 - \frac{j-1}{r}\right) \geq 1 - \frac{j-1}{r},\end{aligned}$$

where the first inequality holds since the matroid is uniformly dense and the second holds because the rank of a set is always at most its cardinality. □

A simple application of the previous lemma is the following. Consider a uniform random set X of j elements. The rank of $X = \{x_1, \ldots, x_j\}$ is equal to the cardinality

[1] A **loop** is an element x such that $\{x\}$ is not independent. A **loopless matroid** is a matroid having no loops.

of the set returned by the greedy procedure on any ordering of its elements. By Lemma 2.3.1,

$$\mathbb{E}[\mathrm{rk}(X)] \geq \sum_{i=1}^{j} \max\left\{\left(1 - \frac{i-1}{r}\right), 0\right\} = \sum_{i=1}^{\min\{j,r\}} \left(1 - \frac{i-1}{r}\right). \tag{2.2}$$

Note that if $j \leq r$, the right hand side is $j - \frac{j(j-1)}{2r} \geq j/2$, and if $j \geq r$, the right hand side is $r - \frac{r(r-1)}{2r} \geq r/2$. In any case the expected rank of X is at least $\min\{j, r\}/2$. This shows that the rank of a random set in a uniformly dense matroid is close to what it would have been if the matroid was uniform. The following lemma tightens this bound.

Lemma 2.3.2. *Let X be a set of a fixed cardinality j whose elements are chosen uniformly at random in a uniformly dense matroid. Then,*

$$\mathbb{E}[\mathrm{rk}(X)] \geq r\left(1 - \left(1 - \frac{1}{r}\right)^{j}\right) \geq r\left(1 - e^{-j/r}\right). \tag{2.3}$$

In particular,

$$\mathbb{E}[\mathrm{rk}(X)] \geq \min\{j, r\}\left(1 - \frac{1}{e}\right). \tag{2.4}$$

Proof. Let $(x_1, x_2, \ldots, x_n)$ be a random ordering of the elements of $\mathcal{S}$, and let X_j be the set $\{x_1, \ldots, x_j\}$. As j increases, the rank of X_j increases by one unit every time an element is outside the span of the previous elements. Then, for all $1 \leq j \leq n$,

$$\begin{aligned}\mathbb{E}[\mathrm{rk}(X_j)] - \mathbb{E}[\mathrm{rk}(X_{j-1})] &= \Pr(x_j \notin \mathrm{span}(X_{j-1})) = \mathbb{E}\left[\frac{n - |\mathrm{span}(X_{j-1})|}{n - |X_{j-1}|}\right] \\ &\geq \mathbb{E}\left[\frac{n - \mathrm{rk}(X_{j-1})n/r}{n - (j-1)}\right] \geq 1 - \frac{\mathbb{E}[\mathrm{rk}(X_{j-1})]}{r}.\end{aligned}$$

Therefore, the sequence $Z_j = \mathbb{E}[\mathrm{rk}(X_j)]$, for $j = 0, \ldots, n$ satisfies:

$$\begin{aligned}&Z_0 = 0, \\ &Z_j \geq 1 + Z_{j-1}\left(1 - \frac{1}{r}\right), \quad \text{for } j \geq 1.\end{aligned}$$

Solving the previous recurrence yields

$$Z_j \geq \sum_{i=0}^{j-1} \left(1 - \frac{1}{r}\right)^{i} = r\left(1 - \left(1 - \frac{1}{r}\right)^{j}\right) \geq r\left(1 - e^{-j/r}\right). \tag{2.5}$$

This completes the first part of the proof.

The function $(1 - e^{-x})$ is increasing, thus if $j \geq r$,

$$Z_j \geq r\left(1 - e^{-j/r}\right) \geq r\left(1 - e^{-1}\right).$$

In addition, the function $(1 - e^{-x})/x$ is decreasing, thus if $j \leq r$,

$$Z_j \geq j \frac{\left(1 - e^{-j/r}\right)}{j/r} \geq j \left(1 - e^{-1}\right). \quad \square$$

2.3.1 Random-Assignment Random-Order Model

We have seen that uniformly dense matroids behave similarly to uniform matroids on the same ground set and with the same total rank. A natural approach to obtain a constant-competitive algorithm for uniformly dense matroids is to adapt constant-competitive algorithms for uniform matroids.

Recall the e-competitive algorithm (Algorithm 2) for uniform matroids of rank r we presented in Section 1.3. We show that a slight modification of this algorithm is at most $2e$-competitive for uniformly dense matroids in the random-assignment random-order model. The only difference with respect to Algorithm 2 is that before adding an element to the output, we have to test if its addition maintains independence in the matroid. We depict the entire procedure as Algorithm 3 below.

Algorithm 3 for uniformly dense matroids of n elements and rank r in the random-assignment random-order model.

1: Maintain a set T containing the heaviest r elements seen so far at every moment (initialize T with r dummy elements).
2: ALG $\leftarrow \emptyset$.
3: Choose m from the binomial distribution $\text{Bin}(n, p)$.
4: Observe the first m elements and denote this set as the sample.
5: **for** each element x arriving after the first m elements **do**
6: **if** x enters T and the element leaving T is in the sample or is a dummy element **then** check if ALG $\cup \{x\}$ is independent. If so, add x to ALG.
7: **end if**
8: **end for**
9: Return the set ALG.

Theorem 2.3.3. *Let* ALG *be the set returned by Algorithm 3 when applied to a uniformly dense matroid $\mathcal{M}$ of rank r. Then*

$$\mathbb{E}_{\sigma,\pi}[w(\text{ALG}(\sigma))] \geq (-p^2 \ln p) \sum_{i=1}^{r} w_i \geq (-p^2 \ln p) \mathbb{E}_{\sigma}[w(\text{OPT}_{\mathcal{M}}(\sigma))].$$

In particular, by setting $p = e^{-1/2}$, we obtain a $2e$-competitive algorithm for uniformly dense matroids in the random-assignment random-order model of the matroid secretary problem.

Proof. Similar to the analysis of Algorithm 2, we use an offline simulation, but now we make a clear distinction between the elements of the matroid and their weights.

In the first step of the simulation, each weight w_i in the adversarial list W selects an arrival time t_i in the interval $(0, 1)$ uniformly and independently. The simulation keeps a set T containing the top r *weights* seen at every moment (initially containing r dummy weights of negative value) and processes the weights as they arrive, sampling the ones arriving before time p. When a weight arriving after time p is processed, the algorithm *marks it as a candidate* if, first, that weight enters T, and second, the one leaving T is either in the sample or a dummy weight.

In the second step of the simulation, the algorithm assigns to each weight marked as a candidate a different element of the matroid's ground set uniformly at random. Then, it runs the greedy procedure on the sequence of candidates in the order they arrived and returns its answer.

Using that the cardinality of the sampled set has binomial distribution with parameters n and p, it is not hard to check that the sets of elements and weights returned by this simulation have the same distribution as the ones returned by Algorithm 3. For this reason, we focus on the simulation.

We estimate the probability that each one of the top r weights appears in the output set. Focus on one such weight w_i with $i \leq r$. Condition on the event that w_i arrives after time p, and let ℓ be the random variable counting the number of weights in $\{w_1, \ldots, w_r\} \setminus \{w_i\}$ arriving in the time interval $(0, p)$. Each of these high weights enters T during the sample period and never leaves T. Since every weight marked as a candidate pushes out either a dummy or a sampled weight of T at the moment it is marked, the previous statement implies that the number of weights marked as a candidate by the simulation algorithm is at most $r - \ell$.

Since w_i is one of the top r weights, it enters the set T at the time it is considered. Thus, it will be marked as a candidate if and only if the weight leaving T at that time is either a dummy or a sampled weight.

Let $A(t_i) = \{w_j \colon t_j < t_i\}$ be the set of weights seen before w_i arrives. If this set has less than r elements then the element leaving T at t_i will be a dummy weight. Consider the case where $A(t_i)$ has cardinality at least r and let w_A be the r-th top element of this set. Since w_A is not one of the top r elements in the full adversarial list, its arrival time t_A is independent of ℓ. In particular, conditioned on $\{t_i, \ell\}$, t_A is a uniform random variable in $(0, t_i)$. Hence,

$$\begin{aligned}
&\Pr(w_i \text{ is as a candidate} \mid \ell, t_i) \\
&\qquad = 1 \cdot \Pr(|A(t_i)| < r \mid \ell, t_i) + \Pr(|A(t_i)| \geq r, t_A < p \mid \ell, t_i) \\
&\qquad = \Pr(|A(t_i)| < r \mid \ell, t_i) + \Pr(|A(t_i)| \geq r \mid \ell, t_i) p/t_i \\
&\qquad \geq p/t_i.
\end{aligned}$$

The elements of the matroid assigned to the weights marked as candidates form a random set. Conditioned on the value ℓ and on w_i being a candidate, Lemma 2.3.1 implies that no matter what position w_i takes in the list of at most $r - \ell$ candidates, the probability that the corresponding element gets added to the output is at least $1 - (r - \ell - 1)/r = (\ell + 1)/r$; therefore, the probability that w_i appears in the output

is at least

$$\mathbb{E}\left[\frac{\ell+1}{r}\int_p^1 \frac{p}{t_i}dt_i\right] = -p\ln p\frac{\mathbb{E}[\ell]+1}{r} = \frac{(-p\ln p)((r-1)p+1)}{r} \geq -p^2\ln p.$$

Theorem 2.3.3 follows easily from here. □

We can obtain an alternative constant-competitive algorithm for uniformly dense matroids based on Algorithm 1 for the classical secretary problem. The idea is to perform a random partition of the matroid's ground set into r consecutive groups. For each of them, apply Algorithm 1 to find its top-valued element and try to add it to the output set if this is possible. The procedure is depicted as Algorithm 4 below.

Algorithm 4 for uniformly dense matroids of n elements and rank r in the random-assignment random-order model.

1: ALG $\leftarrow \emptyset$.
2: Select n values $v_1, \ldots, v_n$ uniformly at random from $\{1, \ldots, r\}$ and let N_i be the number of times value i was selected.
3: Denote as $\mathcal{S}_1$ the set of the first N_1 elements arriving, $\mathcal{S}_2$ the set of the next N_2 elements, and so on.
4: For each i in $[r]$, run Algorithm 1 with parameter p (not necessarily $p = 1/e$) on the sequence $\mathcal{S}_i$. Mark the elements that are selected.
5: Whenever an element x is marked, check if ALG $\cup \{x\}$ is independent. If so, add x to ALG.
6: Return the set ALG.

Before stating the analysis of Algorithm 4, it is convenient to define the following auxiliary construction.

For any matroid $\mathcal{M} = (\mathcal{S}, \mathcal{I})$ of total rank r, the **associated random partition matroid** $\mathcal{P}(\mathcal{M}) = (\mathcal{S}, \mathcal{I}')$ is obtained as follows. Partition the set $\mathcal{S}$ into r classes, where each element selects its own class in a uniform random way. A set of elements in $\mathcal{S}$ is independent in $\mathcal{P}(\mathcal{M})$ if it contains at most one element of each class. The following lemma states that the weight of the optimum base of $\mathcal{P}(\mathcal{M})$ is at least a constant fraction of the weight of the optimum in $\mathcal{M}$.

Lemma 2.3.4. *Let $w_1 \geq w_2 \geq \cdots \geq w_n \geq 0$ be the set of weights determined by the adversary, then:*

$$\mathbb{E}_\sigma[w(\mathrm{OPT}_{\mathcal{P}(\mathcal{M})}(\sigma))] \geq \left(1 - \left(1 - \frac{1}{r}\right)^r\right)\sum_{i=1}^{r} w_i \geq (1-1/e)\mathbb{E}_\sigma[w(\mathrm{OPT}_{\mathcal{M}}(\sigma))]. \quad (2.6)$$

To prove this lemma, we need Chebyshev's sum inequality (see, e.g. [112]) which states that if

$$a_1 \geq a_2 \geq \cdots \geq a_r \text{ and } b_1 \geq b_2 \geq \cdots \geq b_r$$

then

$$\sum_{i=1}^{r} a_i b_i \geq \frac{1}{r}\left(\sum_{i=1}^{r} a_i\right)\left(\sum_{i=1}^{r} b_i\right).$$

Proof of Lemma 2.3.4. For $i \in [r]$, w_i is in $\mathrm{OPT}_{\mathcal{P}(\mathcal{M})}(\sigma)$ if and only if the weights $w_1, \ldots, w_{i-1}$ are assigned to elements in a different class of $\mathcal{P}(\mathcal{M})$ than the one the element of w_i is assigned. Then $\Pr(w_i \in \mathrm{OPT}_{\mathcal{P}(\mathcal{M})}(\sigma)) = (1-1/r)^{i-1}$. Note that both $((1-1/r)^{i-1})_{i=1,\ldots,r}$ and $(w_i)_{i=1,\ldots,r}$ are non-increasing sequences. Using Chebyshev's sum inequality we have

$$\begin{aligned}\mathbb{E}[w(\mathrm{OPT}_{\mathcal{P}(\mathcal{M})}(\sigma))] &\geq \sum_{i=1}^{r}\left(1-\frac{1}{r}\right)^{i-1} w_i \geq \left(\frac{1}{r}\sum_{i=1}^{r}\left(1-\frac{1}{r}\right)^{i-1}\right)\left(\sum_{i=1}^{r} w_i\right)\\ &= \left(1-\left(1-\frac{1}{r}\right)^{r}\right)\sum_{i=1}^{r} w_i. \qquad \square\end{aligned}$$

We give three different analysis for Algorithm 4.

Theorem 2.3.5. *Let* ALG *be the set returned by Algorithm 4 when applied to a uniformly dense matroid $\mathcal{M}$ of total rank r.*

(i) *By setting $p = 1/e$, we have:*

$$\mathbb{E}_{\sigma,\pi}[w(\mathrm{ALG}(\sigma))] \geq (1-1/e)(1/e)\mathbb{E}_{\mathcal{P}(\mathcal{M}),\sigma}[w(\mathrm{OPT}_{\mathcal{P}(\mathcal{M})}(\sigma))].$$

By Lemma 2.3.4, this algorithm is $(e/(1-1/e)^2) \approx 6.80291$-competitive.

(ii) *For $p \in (0,1)$, we have:*

$$\begin{aligned}\mathbb{E}_{\sigma,\pi}[w(\mathrm{ALG}(\sigma))] &\geq \gamma_1(p)\cdot\mathbb{E}_{\mathcal{P}(\mathcal{M}),\sigma}[w(\mathrm{OPT}_{\mathcal{P}(\mathcal{M})}(\sigma))],\\ \text{where } \gamma_1(p) &= \frac{1}{(1-p)}\,(1-e^{-(1-p)})(-p\ln p).\end{aligned}$$

This is optimized by setting $p = p_1 \approx 0.433509$. In that case, we get $\gamma_1(p_1) \approx 0.276632$. By Lemma 2.3.4, this algorithm is $((1-1/e)\gamma_1(p_1))^{-1} \approx 5.7187$-competitive.

(iii) *For $p \in (0,1)$, we have:*

$$\begin{aligned}\mathbb{E}_{\sigma,\pi}[w(\mathrm{ALG}(\sigma))] &\geq \gamma_2(p)\cdot\sum_{i=1}^{r} w_i,\\ \text{where } \gamma_2(p) &= \frac{p(1-e^{-(1-p)})}{1-p}\int_p^1 \frac{1-e^{-t}}{t^2}\,dt.\end{aligned}$$

This is optimized by setting $p = p_2 \approx 0.384374$. In that case, we get $\gamma_2(p_2) \approx 0.20322$. Thus, the competitive ratio of this algorithm is at most $1/(\gamma_2(p_2)) \approx 4.92078$.

Proof. As usual, we analyze the algorithm via a simulation. On the first step of the simulation every *weight* w_i of the adversarial list selects a random color $c(w_i)$ in $[r]$. Let W_j be the collection of weights that are assigned color j. For every j, we present the weights of W_j in uniform random order to Algorithm 1 with parameter p. This algorithm returns either a singleton weight or an empty set. Let T be the set of weights selected by this algorithm.

On the second step of the simulation, we randomly assign to each weight in T an element in $\mathcal{S}$ to obtain a set $X(T)$. Finally, we apply the greedy procedure on the elements $X(T)$ with weights in T, in *uniform random order*. Let ALG be the set obtained and $W(\text{ALG})$ the corresponding set of weights.

It is straightforward to check that the distribution of ALG and $W(\text{ALG})$ is the same both if we apply the greedy procedure on $X(T)$ in random order or if we apply it on increasing ordering of their colors (as the offline simulation does). We use random order because it makes our analysis simpler. From here, it is easy to see that the sets of elements and weights returned by the simulation have the same distribution as the ones returned by Algorithm 4.

Because of the way it was constructed, $W(\text{ALG})$ is a uniform random subset of T of size $\text{rk}(X(T))$. Then, for any weight w of the list of the adversary,

$$\Pr[w \in W(\text{ALG}) \,|\, w \in T,\ t = |T|] = \frac{\mathbb{E}[\text{rk}(X(T))]}{t} \geq (1 - 1/e), \tag{2.7}$$

where the last inequality comes from (2.4) in Lemma 2.3.2 and the fact that $X(T)$ is a random set of size $t = |T|$. Thus, we can remove the conditioning on t.

Recall that Algorithm 1 for the classical secretary problem is e-competitive when we set $p = 1/e$. Hence, for each color class $j \in [r]$, the expected value in $W_j \cap T$ is at least $1/e$-fraction of the maximum weight w_j^* in W_j (set $w_j^* = 0$ for the pathological case where W_j is empty). Therefore:

$$\mathbb{E}[\Sigma W(\text{ALG})] \geq \sum_{j=1}^{r} (1 - 1/e)\mathbb{E}[\Sigma(T \cap W_j)] \geq (1 - 1/e)(1/e) \sum_{j=1}^{r} \mathbb{E}[w_j^*], \tag{2.8}$$

where ΣV denotes the total sum of the weights inside V. We conclude the proof of part (i) by noting that $\sum_{j=1}^{r} \mathbb{E}[w_j^*]$ is the expected weight of the optimum in the random partition matroid $\mathcal{P}(\mathcal{M})$ induced by the coloring.

For part (ii), we refine the previous analysis. By using (2.3) of Lemma 2.3.2, we get an improved version of (2.7):

$$\Pr[w \in W(\text{ALG}) \,|\, w \in T,\ t = |T|] = \frac{\mathbb{E}[\text{rk}(X(T))]}{t} \geq \frac{r}{t}\left(1 - \left(1 - \frac{1}{r}\right)^t\right). \tag{2.9}$$

In order to remove the conditioning on t, we need to compute the expected value of the right hand side. First note that t equals 1 (because of w) plus the number of colors different to the color of w, for which the classical secretary algorithm did not return an empty set. For each color, the algorithm returns a nonempty set with probability at

most[2] $(1-p)$ (see Lemma 1.2.4). Therefore, the variable t is stochastically dominated by one plus the sum of $r-1$ Bernoulli random variables with parameter $(1-p)$. As the right hand side of (2.9) is decreasing in t, we can effectively replace t to obtain:

$$\begin{aligned}
\Pr(w \in W(\mathrm{ALG}) \mid w \in T) &\geq \sum_{k=0}^{r-1} \frac{r}{1+k}\left(1-\left(1-\frac{1}{r}\right)^{1+k}\right)\binom{r-1}{k}(1-p)^k p^{r-1-k} \\
&= \frac{1}{1-p}\left(\sum_{j=1}^{r}\binom{r}{j}(1-p)^j p^{r-j} - \sum_{j=1}^{r}\binom{r}{j}\left(1-\frac{1}{r}\right)^j (1-p)^j p^{r-j}\right) \\
&= \frac{1}{1-p}\Big(\left(1-p^r\right) - \left(((1-1/r)(1-p)+p)^r - p^r\right)\Big) \\
&= \frac{1-(1-(1-p)/r)^r}{1-p} \geq \frac{1-e^{-(1-p)}}{1-p}. \qquad (2.10)
\end{aligned}$$

Finally, using that Algorithm 1 returns the best weight of each class with probability $-p\ln p$ (see Lemma 1.2.4), we obtain that

$$\mathbb{E}[\Sigma W(\mathrm{ALG})] \geq \sum_{j=1}^{r} \frac{1-e^{-(1-p)}}{1-p}\mathbb{E}[\Sigma(T \cap W_j)] \geq \gamma_1(p)\sum_{j=1}^{r}\mathbb{E}[w_j^*],$$

where $\gamma_1(p) = \frac{1-e^{-(1-p)}}{1-p}(-p\ln p)$. This concludes the proof of part (ii).

The analysis for part (iii) is, again, a refinement of the previous one. For this one, we compute directly the probability that the simulation selects each one of the top r weights in the adversarial list. Let w_i, for $i \leq r$, be the i-th top weight. Let $\mathcal{E}_{ij}$ be the event that w_i is the j-th top weight of its own color class. We have

$$\Pr(\mathcal{E}_{ij}) = \begin{cases} \binom{i-1}{j-1}\left(\frac{1}{r}\right)^{j-1}\left(1-\frac{1}{r}\right)^{i-j}, & \text{if } j \leq i; \\ 0, & \text{otherwise.} \end{cases} \qquad (2.11)$$

Using Lemma 1.2.4, we conclude that the probability that w_i is marked is

$$\begin{aligned}
\Pr(w_i \in T) &= \sum_{j=1}^{i} \Pr(w_i \in T \,|\, \mathcal{E}_{ij}) \Pr(\mathcal{E}_{ij}) \\
&= \sum_{j=1}^{i} \int_p^1 \frac{(1-t)^{j-1}p}{t}\,dt \binom{i-1}{j-1}\left(\frac{1}{r}\right)^{j-1}\left(1-\frac{1}{r}\right)^{i-j} \\
&= \int_p^1 \frac{p}{t}\sum_{j=1}^{i}\binom{i-1}{j-1}\left(\frac{1-t}{r}\right)^{j-1}\left(1-\frac{1}{r}\right)^{i-j} dt \\
&= p\int_p^1 \frac{((1-t)/r + (1-1/r))^{i-1}}{t}\,dt = p\int_p^1 \frac{(1-t/r)^{i-1}}{t}\,dt.
\end{aligned}$$

The sequence $(\Pr(w_i \in T))_{i=1,\ldots,r}$ is non-increasing. Using (2.10) and Chebyshev's

[2]The reason why this is not exactly $(1-p)$ is the pathological case where the color class is empty. In this case the algorithm will always return an empty set.

sum inequality we conclude that

$$\begin{aligned}
\mathbb{E}[\Sigma W(\mathrm{ALG})] &\geq \frac{1-e^{-(1-p)}}{1-p} \sum_{i=1}^{r} w_i \Pr(w_i \in T) \\
&\geq \frac{1-e^{-(1-p)}}{1-p} \left(\sum_{i=1}^{r} w_i\right) p \int_p^1 \frac{1}{r} \sum_{i=1}^{r} \frac{(1-t/r)^{i-1}}{t}\, dt \\
&= \frac{1-e^{-(1-p)}}{1-p} \left(\sum_{i=1}^{r} w_i\right) p \int_p^1 \frac{1-(1-t/r)^{r}}{t^2}\, dt \\
&\geq \frac{1-e^{-(1-p)}}{1-p} \left(\sum_{i=1}^{r} w_i\right) p \int_p^1 \frac{1-e^{-t}}{t^2}\, dt.
\end{aligned}$$

This concludes the proof. □

The third part of the above analysis for Algorithm 4 shows that this algorithm has a better competitive ratio (≈ 4.92078) than Algorithm 3 ($2e \approx 5.43656$). The second part, as we will see in Section 2.5, is also useful for the case of general matroids.

2.3.2 Random-Assignment Adversarial-Order Model

After the first publication of some of the results of this thesis in [152], Oveis Gharan and Vondrák [136] have devised a 40-competitive algorithm for uniformly dense matroids on the random-assignment adversarial-order model.

In what follows we give an alternative algorithm for this task that is very similar to Algorithm 4. In this new algorithm we also partition the ground set of the matroid into r groups, in a way that is independent of the order. Afterwards, we use the threshold algorithm for the classical secretary in each group to find its top-valued element, but in a coupled way: the sample consists of all the elements of that group seen in the first half of the stream. After that, unlike Algorithm 4, we only try to add the elements found to the output set with certain fixed probability (otherwise, we discard it). The procedure is depicted as Algorithm 5 below.

As in the analysis of Algorithm 4, it is convenient to compare to the random partition matroid $\mathcal{P}(\mathcal{M}) = (\mathcal{S}, \mathcal{I}')$, associated to matroid $\mathcal{M}$.

Theorem 2.3.6. *Let* ALG *be the set returned by Algorithm 5 when applied to a uniformly dense matroid $\mathcal{M}$ of total rank r, in the random-assignment adversarial-order model. Then*

$$\mathbb{E}_\sigma[w(\mathrm{ALG}(\sigma))] \geq 1/16 \cdot \mathbb{E}_{\mathcal{P}(\mathcal{M}),\sigma}[w(\mathrm{OPT}_{\mathcal{P}(\mathcal{M})}(\sigma))].$$

By Lemma 2.3.4, Algorithm 5 is $16/(1-1/e) \approx 25.3116$*-competitive.*

Proof. As usual, we use a two-step offline simulation algorithm. On the first step, the simulation assigns to each weight w_i in the adversarial list W a color $c(w_i)$ in $[r]$ and an arrival time $t_i \in (0,1)$. After that, the simulation samples all the weights arriving before time $p = 1/2$ and, for each color, marks the first weight (if any) arriving after $p = 1/2$ that is larger than all the weights of that color seen so far.

Algorithm 5 for uniformly dense matroids of n elements and rank r in the random-assignment adversarial-order model.

ALG $\leftarrow \emptyset$.
Assign to each element of the matroid a color $i \in [r]$ uniformly at random.
Choose m from the binomial distribution $\text{Bin}(n, 1/2)$.
Observe and reject the first m elements and denote this set as the sample.
for each element x arriving after that **do**
    if the color i of x has not been marked and x is the heaviest element seen so far with color i **then**
        Mark color i.
        With probability $1/2$ ignore x and continue.
        Otherwise, check if ALG $\cup \{x\}$ is independent. If so, add x to ALG.
    end if
end for
Return the set ALG.

On the second step, the simulation assigns to the weight arriving in the k-th position the corresponding k-th element of the adversarially sorted list. Then, it *unmarks* each marked weight with probability $q = 1/2$, and applies the greedy procedure on the remaining marked elements in their order of arrival, returning its answer.

It is easy to see that the sets of elements and weights this simulation returns have the same distribution as the ones returned by Algorithm 5.

We say that color $j \in [r]$ is *successful* if either the top weight having color j is part of the set returned by the algorithm or if no weight receives color j.

Claim 1. Every color j is successful with probability at least $p(1-p)q(1-q)$.

If this claim holds then the total weight returned by the algorithm is at least $p(1-p)q(1-q) = 1/16$ times the weight of the optimum of the partition matroid induced by the coloring, where the independent sets are those containing at most one element of each color. As this matroid has the same distribution as $\mathcal{P}(\mathcal{M})$, the claim implies the lemma.

Let us prove the claim. Fix a nonempty color class j and let w' and w'' be the two top weights of color class j. In the case this class has only one element, let w'' be an arbitrary different weight of the list. Color j is successful if the weight w' is selected in the output of the algorithm. In what follows, we estimate the probability that this event occurs.

Let A be the set of weights arriving before time p and B be the set of weights arriving after time p. Note that if $w' \in B$ and $w'' \in A$ then no matter what relative position they take inside A and B, weight w' will be marked in the first step of the simulation. In order for w' to be part of the output of the greedy procedure, it is enough that this weight is not unmarked in the second step, and that the element x' in B that receives weight w' is outside the span of the set X of other marked elements.

In order to estimate $|X|$, we need another definition. We say that a color is *feasible* if one of the two following events occur. Either exactly one weight of that color is

marked in the first step of the simulation and it is not unmarked later, or no weight at all receives that color and two independent biased coins fall head, one with probability $1-p$ and another one with probability $1-q$.

Let s be the random variable counting the number of feasible colors different than j. Note that just before applying the greedy algorithm on the second part of the simulation, the number $|X|$ of remaining marked elements of color different than j, is always at most s. Then, we have

$$\begin{aligned}
&\Pr(w' \text{ is selected by the algorithm} \,|\, w' \in B, w'' \in A, s) \\
&\quad = (1-q)\,\Pr(x' \notin \operatorname{span}(X) \,|\, w' \in B, w'' \in A, s) \\
&\quad = (1-q)\,\mathbb{E}\left[\frac{|B \setminus \operatorname{span}(X)|}{|B|} \,\middle|\, w' \in B, w'' \in A, s\right]. \qquad (2.12)
\end{aligned}$$

As the matroid is uniformly dense,

$$\frac{|B \setminus \operatorname{span}(X)|}{|B|} \geq 1 - \frac{|\operatorname{span}(X)|}{|B|} \geq 1 - \frac{n}{r}\frac{|X|}{|B|} \geq 1 - \frac{ns}{r}\frac{1}{|B|}. \qquad (2.13)$$

We conclude that (2.12) is at least

$$(1-q)\left(1 - \frac{ns}{r}\,\mathbb{E}\left[\frac{1}{|B|} \,\middle|\, w' \in B, w'' \in A\right]\right). \qquad (2.14)$$

Conditioned on the fact that weight w' is already assigned to A and weight w'' is already assigned to B, the variable $|B|$ behaves like a binomial random variable with parameters $n-2$ and $1-p$ shifted up by one unit. Therefore,

$$\begin{aligned}
\mathbb{E}\left[\frac{1}{|B|} \,\middle|\, w' \in B, w'' \in A\right] &= \sum_{j=0}^{n-2} \frac{1}{1+j}\binom{n-2}{j}(1-p)^j p^{n-2-j} \\
&= \frac{1}{(1-p)(n-1)} \sum_{j=0}^{n-2} \binom{n-1}{j+1}(1-p)^{j+1} p^{n-1-(j+1)} \\
&= \frac{1-p^{n-1}}{(1-p)(n-1)}. \qquad (2.15)
\end{aligned}$$

By removing the conditioning on $w' \in B$ and $w'' \in A$ in (2.12), we obtain

$$\Pr(\text{Color } j \text{ is successful} \,|\, s) \geq p(1-p)(1-q)\left(1 - \frac{ns}{r}\frac{1-p^{n-1}}{(1-p)(n-1)}\right). \qquad (2.16)$$

By Lemma 1.2.4 the probability that a given color is feasible is at most $(1-p)(1-q)$, and since all these events are independent, the variable s is stochastically dominated by a binomial random variable with parameters $r-1$ and $(1-p)(1-q)$. As the right

hand side of (2.16) is decreasing in s, we can effectively replace s to obtain:

$$\begin{aligned}
&\Pr(\text{Color } j \text{ is successful}) \\
&\quad \geq p(1-p)(1-q)\left(1 - \frac{n\mathbb{E}[\mathrm{Bin}(r-1,(1-p)(1-q))]}{r}\frac{1-p^{n-1}}{(1-p)(n-1)}\right) \\
&\quad \geq p(1-p)(1-q)\left(1 - \frac{n(r-1)(1-q)(1-p^{n-1})}{r(n-1)}\right) \\
&\quad \geq p(1-p)(1-q)(1-(1-q)(1-p^{n-1})) \geq p(1-p)(1-q)q,
\end{aligned}$$

where in the third inequality we have used that $r \leq n$. This concludes the proof of the claim and the lemma. □

2.4 Principal Partition

Recall the divide and conquer strategy of Section 2.2. In this section we describe a technique which lets us find a collection of uniformly dense matroid whose direct sum approximates any given matroid.

Consider a loopless matroid $\mathcal{M} = (\mathcal{S}, \mathcal{I})$ that is not uniformly dense, and let E_1 be a *maximum cardinality set* achieving the density of $\mathcal{M}$. This is

$$\gamma(\mathcal{M}) = \max_{\emptyset \neq X \subseteq \mathcal{S}} \frac{|X|}{\mathrm{rk}(X)} = \frac{|E_1|}{\mathrm{rk}(E_1)}, \tag{2.17}$$

and $|E_1| \geq |X|$ for any set X achieving the same density.

We claim that the matroid $\mathcal{M}_1 = \mathcal{M}|_{E_1}$ restricted to the set E_1 is uniformly dense with density $\lambda_1 = \gamma(\mathcal{M})$. Indeed, since the rank function of $\mathcal{M}_1$ is equal to the one of $\mathcal{M}$, every subset of E_1 has the same density in both $\mathcal{M}$ and $\mathcal{M}_1$, making E_1 the densest set in $\mathcal{M}_1$, with density $|E_1|/\mathrm{rk}(E_1) = \gamma(\mathcal{M}) = \lambda_1$.

On the other hand, consider the matroid $\mathcal{M}_1' = \mathcal{M}/E_1$ obtained by contracting E_1. We show that this matroid is loopless and has strictly smaller density than $\mathcal{M}_1$. Indeed, recall that the rank function of the contracted matroid (see, e.g. [137]) is

$$\mathrm{rk}_{\mathcal{M}_1'}(X) = \mathrm{rk}_{\mathcal{M}}(E_1 \cup X) - \mathrm{rk}_{\mathcal{M}}(E_1), \text{ for all } X \subseteq \mathcal{S} \setminus E_1.$$

Hence, if $x \in \mathcal{S} \setminus E_1$ is a loop of $\mathcal{M}_1'$, then $\mathrm{rk}_{\mathcal{M}}(E_1 \cup \{x\}) = \mathrm{rk}_{\mathcal{M}}(E_1)$, and so

$$\frac{|E_1 \cup \{x\}|}{\mathrm{rk}_{\mathcal{M}}(E_1 \cup \{x\})} = \frac{|E_1|+1}{\mathrm{rk}_{\mathcal{M}}(E_1)} > \frac{|E_1|}{\mathrm{rk}_{\mathcal{M}}(E_1)},$$

contradicting the definition of E_1. Therefore, $\mathcal{M}_1'$ is loopless. By maximality of E_1, every set X with $\emptyset \neq X \subseteq \mathcal{S} \setminus E_1$ satisfies

$$\frac{|E_1 \cup X|}{\mathrm{rk}_{\mathcal{M}}(E_1 \cup X)} < \frac{|E_1|}{\mathrm{rk}_{\mathcal{M}}(E_1)}.$$

Hence,

$$\frac{|X|}{\mathrm{rk}_{\mathcal{M}_1'}(X)} = \frac{|E_1 \cup X| - |E_1|}{\mathrm{rk}_{\mathcal{M}}(E_1 \cup X) - \mathrm{rk}_{\mathcal{M}}(E_1)} < \frac{\frac{|E_1|}{\mathrm{rk}_{\mathcal{M}}(E_1)}\left(\mathrm{rk}_{\mathcal{M}}(E_1 \cup X) - \mathrm{rk}_{\mathcal{M}}(E_1)\right)}{\mathrm{rk}_{\mathcal{M}}(E_1 \cup X) - \mathrm{rk}_{\mathcal{M}}(E_1)} = \frac{|E_1|}{\mathrm{rk}_{\mathcal{M}}(E_1)},$$

implying that $\gamma(\mathcal{M}_1') < \gamma(\mathcal{M})$. Then, we have the following lemma.

Lemma 2.4.1. *Let $\mathcal{M} = (\mathcal{S}, \mathcal{I})$ be a non-uniformly dense loopless matroid and E_1 be the* unique *maximum cardinality set with $\gamma(\mathcal{M}) = |E_1|/\mathrm{rk}_{\mathcal{M}}(E_1)$, then the matroid $\mathcal{M}_1 = \mathcal{M}|_{E_1}$ is uniformly dense with density $\lambda_1 = \gamma(\mathcal{M})$ and the matroid $\mathcal{M}_1' = \mathcal{M}/E_1$ is loopless with density strictly smaller than the one of $\gamma(\mathcal{M})$.*

Proof. The only missing step to prove is that E_1 is unique. Indeed suppose that there are two sets E_1 and E_1' of the same cardinality achieving the density λ_1 of $\mathcal{M}$, in particular they also have the same rank, $\mathrm{rk}_{\mathcal{M}}(E_1) = \mathrm{rk}_{\mathcal{M}}(E_1')$. By submodularity of the rank function:

$$\frac{|E_1 \cup E_1'|}{\mathrm{rk}_{\mathcal{M}}(E_1 \cup E_1')} \geq \frac{|E_1| + |E_1'| - |E_1 \cap E_1'|}{\mathrm{rk}_{\mathcal{M}}(E_1) + \mathrm{rk}_{\mathcal{M}}(E_1') - \mathrm{rk}_{\mathcal{M}}(E_1 \cap E_1')},$$

Using that $\lambda_1 = |E_1|/\mathrm{rk}_{\mathcal{M}}(E_1) = |E_1'|/\mathrm{rk}_{\mathcal{M}}(E_1') \geq |E_1 \cap E_1'|/\mathrm{rk}_{\mathcal{M}}(E_1 \cap E_1')$, we have

$$\frac{|E_1 \cup E_1'|}{\mathrm{rk}_{\mathcal{M}}(E_1 \cup E_1')} \geq \frac{\lambda_1\left(\mathrm{rk}_{\mathcal{M}}(E_1) + \mathrm{rk}_{\mathcal{M}}(E_1') - \mathrm{rk}_{\mathcal{M}}(E_1 \cap E_1')\right)}{\mathrm{rk}_{\mathcal{M}}(E_1) + \mathrm{rk}_{\mathcal{M}}(E_1') - \mathrm{rk}_{\mathcal{M}}(E_1 \cap E_1')} = \lambda_1.$$

Thus, $E_1 \cup E_1'$ is a set strictly larger than E_1 with the same density. This contradicts the choice of E_1. □

If the loopless matroid $\mathcal{M}_1'$ is not uniformly dense, we can use the above lemma in this matroid to find a second uniformly dense matroid $\mathcal{M}_2 = \mathcal{M}_1'|_{E_2}$ with density

$$\lambda_2 = \gamma(\mathcal{M}_1') = \frac{|E_2|}{\mathrm{rk}_{\mathcal{M}_1'}(E_2)} = \frac{|E_2|}{\mathrm{rk}_{\mathcal{M}}(E_1 \cup E_2) - \mathrm{rk}_{\mathcal{M}}(E_1)} < \lambda_1,$$

and a loopless matroid $\mathcal{M}_2' = \mathcal{M}_1'/E_2$ of strictly smaller density. Here, E_2 is the maximum cardinality set achieving $\mathcal{M}_1'$'s the density. By repeating this process we obtain a sequence of sets $(E_1, \ldots, E_k)$ partitioning $\mathcal{S}$ and a sequence of values $\lambda_1 > \lambda_2 > \cdots > \lambda_k \geq 0$ for which the following holds.

Theorem 2.4.2 (Principal Partition)**.** *Let $\mathcal{M} = (\mathcal{S}, \mathcal{I})$ be a loopless matroid and let $(E_i)_{i=1}^k$ and $(\lambda_i)_{i=1}^k$ be the partition and sequence of values just described. Let also $F_0 = \emptyset$ and $F_i = \bigcup_{j=1}^{i} E_j$. Then, for every $1 \leq i \leq k$, the matroid*

$$\mathcal{M}_i = (\mathcal{M}/F_{i-1})|_{E_i}$$

is uniformly dense with density

$$\lambda_i = \frac{|E_i|}{\mathrm{rk}_{\mathcal{M}}(F_i) - \mathrm{rk}_{\mathcal{M}}(F_{i-1})}.$$

Furthermore, if for every i, I_i is an independent set of $\mathcal{M}_i$, then the set $\bigcup_{i=1}^{k} I_i$ is independent in $\mathcal{M}$.

Proof. We know from properties of matroids that contractions and restrictions commute (see [137]). Therefore, the matroids $(\mathcal{M}_i)_{i=1}^{k}$ coincide with the uniformly dense matroids constructed iteratively using Lemma 2.4.1. The density condition follows since

$$\gamma(\mathcal{M}_i) = \frac{|E_i|}{\mathrm{rk}_{\mathcal{M}_i}(E_i)} = \frac{|E_i|}{\mathrm{rk}_{\mathcal{M}}(E_i \cup F_{i-1}) - \mathrm{rk}_{\mathcal{M}}(F_{i-1})}.$$

To finish, let I_i be an independent set of matroid $\mathcal{M}_i$ for every $i \in [k]$. We prove by induction that $\bigcup_{i=1}^{j} I_i$ is independent in $\mathcal{M}|_{F_j}$. The claim holds for $j = 1$ trivially. For $2 \leq j \leq k$, the fact that I_j is independent in $\mathcal{M}_j = (\mathcal{M}/F_{j-1})\,|_{E_j} = \left(\mathcal{M}|_{F_j}\right)/F_{j-1}$ implies that $I_j \cup J$ is independent in $\mathcal{M}|_{F_j}$ for any J independent in $\mathcal{M}|_{F_{j-1}}$. By setting $J = \bigcup_{i=1}^{j-1} I_i$ we conclude the proof. □

The unique sequence of sets $\emptyset = F_0 \subset F_1 \subset \cdots \subset F_k = E$ is called the **principal sequence** of the matroid $\mathcal{M}$; $\lambda_1 > \cdots > \lambda_k \geq 1$ is the associated sequence of **critical values** and $\mathcal{M}_1, \ldots, \mathcal{M}_k$ are the **principal minors** of $\mathcal{M}$. These sequences have been extensively studied in the past under different names, becoming part of what today is known as the *theory of principal partitions* of matroids and submodular systems.

2.4.1 Background

In this section we briefly introduce some background on principal partitions. For more detailed explanations, we refer the reader to the Fujishige's survey [66] and Narayanan's monograph [131].

The ideas of this area started with the following *principal partition of a graph*, studied by Kishi and Kajitani [94]. Suppose we are given a connected graph $G = (V, E)$. Define the distance $d(T_1, T_2)$ between two spanning trees T_1 and T_2 as the value $|T_1 \setminus T_2| = |T_2 \setminus T_1|$. A pair of trees is a maximally distant pair if this distance is maximized. Kishi and Kajitani have shown the following.

Theorem 2.4.3 (Kishi and Kajitani [94])**.** *The edge set of any connected graph $G = (V, E)$ can be partitioned into three sets (F^-, F^0, F^+) satisfying the following conditions.*

1. *For any $e \in F^-$, there is a maximally distant pair of spanning trees T_1 and T_2 such that $e \notin T_1 \cup T_2$.*
2. *For any pair of maximally distant spanning trees T_1 and T_2, every element $e \in F^0$ belongs to exactly one of T_1 or T_2.*

3. *For any $e \in F^+$, there is a maximally distant pair of spanning trees T_1 and T_2 such that $e \in T_1 \cap T_2$.*

The above partition has many applications. For example, it can be used to find the *topological degree of freedom* of an electrical network. This is defined as the minimum number of current and voltage variables whose value uniquely determine all the currents and voltages of the network using Kirchhoff's law (see, e.g. [94] and [131, Chapter 14]). It can also be used to solve *Shannon's switching game* (see, e.g. [19]).

Before continuing it is useful to observe the following (see, e.g. [129, Chapter 9]). Let $\mathcal{S}$ be a finite set and $f\colon 2^{\mathcal{S}} \to \mathbb{R}$ be a submodular function, this is

$$f(A \cup B) + f(A \cap B) \leq f(A) + f(B), \quad \text{for all } A, B \in \mathcal{S}. \tag{2.18}$$

Then the family of minimizers of f form a **distributive lattice**, that is, a collection closed for set union and set intersection. In particular this family has a unique minimal and a unique maximal set.

Kishi and Kajitani's principal partition can be characterized by the next theorem.

Theorem 2.4.4. *Consider the set $\mathcal{D}_G$ of all the minimizers of the submodular function*

$$f(X) = 2\,\mathrm{rk}(X) + |E \setminus X|,$$

for $X \in 2^E$, where $\mathrm{rk}(\cdot)$ is the rank function of the graphic matroid of G. The unique minimal element of $\mathcal{D}_G$ is given by F^- and the unique maximal element of $\mathcal{D}_G$ by $F^- \cup F^0 = E \setminus F^+$, where (F^-, F^0, F^+) is the Kishi–Kajitani partition of E for $G = (V, E)$.

Recall also the following min-max theorem for the union of matroids (see, e.g. [147, Chapter 42]).

Theorem 2.4.5. *For a matroid $\mathcal{M} = (\mathcal{S}, \mathcal{I})$ with rank function $\mathrm{rk}(\cdot)$, and any $p \geq 1$,*

$$\max\left\{\left|\bigcup_{i=1}^{p} I_i\right| \colon I_i \in \mathcal{I}\right\} = \min\left\{p\,\mathrm{rk}(X) + |\mathcal{S} \setminus X| \colon X \subseteq \mathcal{S}\right\}.$$

In particular, the value $\min_{X \subseteq E} f(X)$, with $f(X)$ as in Theorem 2.4.4, corresponds exactly to the maximum size of the union of 2 trees in G. Bruno and Weinberg [19] have noticed the matroidal structure of Kishi and Kajitani's result and have extended their partition to unions of p copies of a given matroid. Their result can be stated as follows.

Theorem 2.4.6 (Bruno and Weiberg [19])**.** *For any matroid $\mathcal{M} = (\mathcal{S}, \mathcal{I})$, and $p \geq 1$, let $\mathcal{D}_p$ be the distributive lattice of the minimizers of the submodular function*

$$f_p(X) = p\,\mathrm{rk}(X) + |\mathcal{S} \setminus X|.$$

Let F_p^- and F_p^+ be the unique minimum and maximum elements of $\mathcal{D}_p$ respectively. Suppose that the collection of distinct $\mathcal{D}_p$ is given by $\mathcal{D}_{p_1}, \ldots, \mathcal{D}_{p_k}$ with $p_1 > \cdots > p_k$,

then we have

$$F_{p_1}^- \subseteq F_{p_1}^+ \subseteq F_{p_2}^- \subseteq F_{p_2}^+ \subseteq \cdots \subseteq F_{p_k}^- \subseteq F_{p_k}^+.$$

In particular, if the difference set $F_{p_i}^+ \setminus F_{p_i}^-$ is nonempty, the minor $(\mathcal{M}|_{F_{p_i}^+})/F_{p_i}^-$ is a matroid having p_i disjoint bases partitioning $F_{p_i}^+ \setminus F_{p_i}^-$.

Tomizawa [163] and Narayanan [128] independently generalized the decomposition above by considering rational numbers instead of integers p. For a rational number p/q with p and q positive integers, they find a minor of $\mathcal{M}$ that has p bases uniformly covering each element of the underlying set q times. More precisely, consider the following extended min-max theorem for the union of matroids.

Theorem 2.4.7. *For a matroid $\mathcal{M} = (\mathcal{S}, \mathcal{I})$ with rank function $\operatorname{rk}(\cdot)$, and $p, q \geq 1$,*

$$\max\left\{\sum_{i=1}^{p} |I_i| \colon I_i \in \mathcal{I},\ |\{i \colon i \in [p], s \in I_i\}| \leq q, \forall s \in \mathcal{S}\right\} = \min\left\{p \operatorname{rk}(X) + q\,|\mathcal{S} \setminus X| \colon X \subseteq \mathcal{S}\right\}.$$

The above result follows from applying Theorem 2.4.5 to the q-parallel extension of $\mathcal{M}$, which is the matroid obtained by replacing replacing each element of $\mathcal{M}$ with q parallel elements[3]. Tomizawa and Narayanan show the following.

Theorem 2.4.8 (Tomizawa [163], Narayanan [128])**.** *For any matroid $\mathcal{M} = (\mathcal{S}, \mathcal{I})$, and any $\lambda = p/q$ for positive integers $p, q \geq 1$, let $\mathcal{D}_\lambda$ be the distributive lattice of the minimizers of the submodular function*

$$f_\lambda(X) = p \operatorname{rk}(X) + q\,|\mathcal{S} \setminus X|.$$

We call the value λ critical *if $\mathcal{D}_\lambda$ contains more than one element. Then, there is a finite sequence of critical values $\lambda_1 > \cdots > \lambda_k \geq 0$. For each $i \in [k]$, let $F_{\lambda_i}^-$ and $F_{\lambda_i}^+$ be the minimum and maximum elements of $\mathcal{D}_{\lambda_i}$, respectively. Then we have:*

$$\emptyset = F_{\lambda_1}^- \subset F_{\lambda_1}^+ = F_{\lambda_2}^- \subset F_{\lambda_2}^+ = \cdots \subset F_{\lambda_{k-1}}^+ = F_{\lambda_k}^- \subset F_{\lambda_k}^+ = \mathcal{S}.$$

And the minor $\mathcal{M}_{\lambda_i} = (\mathcal{M}|_{F_{\lambda_i}^+})/F_{\lambda_i}^-$, for $\lambda_i = p/q$ is a matroid having p bases covering each element of $F_{\lambda_i}^+ \setminus F_{\lambda_i}^-$ exactly q times.

Note that we could have replaced the function $f_\lambda(X)$ in the previous theorem by $f_\lambda(X) = \lambda \operatorname{rk}(X) + |\mathcal{S} \setminus X|$ or by $f_\lambda(X) = \lambda \operatorname{rk}(X) - |X|$ and we would have obtained the same result.

The sequence of distinct sets $\emptyset = F_{\lambda_1}^- \subset F_{\lambda_1}^+ \subset F_{\lambda_2}^+ \subset \cdots \subset F_{\lambda_k}^+ = \mathcal{S}$ is denoted as the principal sequence of $\mathcal{M}$, the values $(\lambda_i)_{i=1}^k$ are the associated critical values and

[3]To be precise, let $\mathcal{S}'$ be a set having q copies of each element of $\mathcal{S}$. The q-parallel extension of $\mathcal{M}$ is the matroid having $\mathcal{S}'$ as ground set and where a set I' is independent if it has at most one copy of each element of $\mathcal{S}$ and the respective set I of original elements is independent in $\mathcal{M}$.

the minors $(\mathcal{M}_{\lambda_i})_{i=1}^k$ are the principal minors of $\mathcal{M}$. Furthermore, the union of all the distributed lattices $(\mathcal{D}_{\lambda_i})_{i=1}^k$ is usually known as the **principal partition** of $\mathcal{M}$.

It is important to remark that when $\mathcal{M}$ is loopless, the construction above coincides with the one given in Theorem 2.4.2. (See also [131] and [22].)

To conclude this small survey, we remark that the constructions above for the rank function of a matroid can be extended also to polymatroid ranks and to even more general functions (see Fujishige's survey [66]). An interesting extension is the following (see [131, Chapter 10]).

Theorem 2.4.9. *Let $f\colon 2^{\mathcal{S}} \to \mathbb{R}$ be a submodular function and $g\colon 2^{\mathcal{S}} \to \mathbb{R}$ be a strictly increasing polymatroid rank function (this is, $g(\emptyset) = 0$ and for all $X \subset Y$, $g(X) < g(Y)$). For every $\lambda \geq 0$, let $\mathcal{D}_\lambda$ be the distributive lattice of the minimizers of the submodular function*

$$h_\lambda(X) = \lambda f(X) + g(\mathcal{S} \setminus X).$$

We call the value λ critical *if $\mathcal{D}_\lambda$ contains more than one element. Then, there is a finite sequence of critical values $\lambda_1 > \cdots > \lambda_k \geq 0$. For each $i \in [k]$, let $F_{\lambda_i}^-$ and $F_{\lambda_i}^+$ be the minimum and maximum elements of $\mathcal{D}_{\lambda_i}$, respectively. Then we have*

$$\emptyset = F_{\lambda_1}^- \subset F_{\lambda_1}^+ = F_{\lambda_2}^- \subset F_{\lambda_2}^+ = \cdots \subset F_{\lambda_{k-1}}^+ = F_{\lambda_k}^- \subset F_{\lambda_k}^+ = \mathcal{S}.$$

The sequence of distinct sets $\emptyset = F_{\lambda_1}^- \subset F_{\lambda_1}^+ \subset F_{\lambda_2}^+ \subset \cdots \subset F_{\lambda_k}^+ = \mathcal{S}$ is called the principal sequence *of $(f(\cdot), g(\cdot))$.*

As shown, for example in [131, Chapter 10], for the particular case where f is submodular and g is a weight function (such as in the case of the principal partition of matroids) there are polynomial time algorithms to find the principal sequence of $(f(\cdot), g(\cdot))$ and the associated critical values.

2.4.2 Matroids Related to the Principal Partition

Consider a general (not necessarily loopless) matroid $\mathcal{M} = (\mathcal{S}, \mathcal{I})$ and let L be its set of loops. Let $(F_i)_{i=0}^k$, $(\lambda_i)_{i=1}^k$ and $(\mathcal{M}_i)_{i=1}^k$ be the principal sequence, critical values and principal minors of the loopless matroid $\mathcal{M} \setminus L$. Also, for every $i \in [k]$, let $E_i = F_i \setminus F_{i-1}$ and r_i denote the ground set and the total rank of matroid $\mathcal{M}_i$. Recall also that the *density* $\gamma(\mathcal{M}_i)$ of the matroid $\mathcal{M}_i$ is equal to $\lambda_i = |E_i|/r_i$.

The family $\{L, E_1, \ldots, E_k\}$ is a partition of the ground set $\mathcal{S}$. Define two new matroids $\mathcal{M}'$ and $\mathcal{P}'$ with ground set $\mathcal{S}$ and independent sets as follows:

$$\mathcal{I}(\mathcal{M}') = \left\{ \bigcup_{i=1}^k I_i \colon I_i \in \mathcal{I}(\mathcal{M}_i) \right\}, \text{ and}$$

$$\mathcal{I}(\mathcal{P}') = \left\{ \bigcup_{i=1}^k I_i \colon I_i \subseteq E_i, |I_i| \leq r_i \right\}.$$

In other words, if $\mathcal{U}(X, r)$ is the uniform matroid on set X having rank r then

$$\mathcal{M}' = \mathcal{U}(L, 0) \oplus \bigoplus_{i=1}^{k} \mathcal{M}_i. \tag{2.19}$$

$$\mathcal{P}' = \mathcal{U}(L, 0) \oplus \bigoplus_{i=1}^{k} \mathcal{U}(E_i, r_i). \tag{2.20}$$

Our main task now is to show that $\mathcal{M}'$ and $\mathcal{P}'$ are, in a well-defined manner, good approximations of $\mathcal{M}$. Before doing that, let us define a third *random* matroid related to the previous ones.

Recall the definition of the random partition matroid $\mathcal{P}(\mathcal{M}_i)$ associated to $\mathcal{M}_i$. In $\mathcal{P}(\mathcal{M}_i)$, each element of E_i receives a color in $[r_i]$ uniformly at random. Let B_{ij} be the set of elements in E_i that are assigned color j. The independent sets of $\mathcal{P}(\mathcal{M}_i)$ are those subsets of E_i having at most one element in each part B_{ij}.

Consider the random matroid $\mathcal{Q}'$ obtained by replacing each summand $\mathcal{M}_i$ of $\mathcal{M}'$ by the matroid $\mathcal{P}(\mathcal{M}_i)$. This is,

$$\mathcal{Q}' = \mathcal{U}(L, 0) \oplus \bigoplus_{i=1}^{k} \mathcal{P}(\mathcal{M}_i) = \mathcal{U}(L, 0) \oplus \bigoplus_{i=1}^{k} \bigoplus_{j=1}^{r_i} \mathcal{U}(B_{ij}, 1). \tag{2.21}$$

Now we show some properties of the previous matroids.

Lemma 2.4.10. *Any independent set of $\mathcal{M}'$ is independent in $\mathcal{M}$.*

Proof. It follows directly from the definition of each $\mathcal{M}_i = (\mathcal{M}/F_{i-1})\big|_{E_i}$. □

Lemma 2.4.11. *Any independent set of $\mathcal{Q}'$ is independent in $\mathcal{P}'$.*

Proof. As any independent set of $\mathcal{Q}'$ contains at most one element in each B_{ij}, it also contains at most r_i element in each E_i. □

The following theorem is the main result of this section. It states that random independent sets in $\mathcal{Q}'$ are likely to have large rank in the original matroid $\mathcal{M}$.

Theorem 2.4.12. *Let $1 \leq \ell \leq n$, and X_ℓ be a random subset of ℓ elements of $\mathcal{S}$. Then*

$$\mathbb{E}_{X_\ell, \mathcal{Q}'}[\mathrm{rk}_{\mathcal{Q}'}(X_\ell)] \geq \left(1 - \frac{1}{e}\right) \mathbb{E}_{X_\ell}[\mathrm{rk}_{\mathcal{M}}(X_\ell)]. \tag{2.22}$$

In order to prove Theorem 2.4.12, we need two technical lemmas.

Lemma 2.4.13. *For every $1 \leq \ell \leq n$, and every $1 \leq i \leq k$,*

$$\mathbb{E}_{X_\ell, \mathcal{Q}'}[\mathrm{rk}_{\mathcal{Q}'}(X_\ell \cap E_i)] \geq r_i\Big(1 - \exp(-\lambda_i \ell / n)\Big).$$

Proof. The value on the left hand side depends only on X_ℓ and on the subpartition $\{B_{ij}\}_{j \in [r_i]}$ of E_i. Since these two objects are chosen independently at random, we can assume they are constructed as follows.

First select $X_\ell \subseteq \mathcal{S}$ uniformly at random. Assign then to each element of X_ℓ a color in $\{1, \ldots, r_i\}$. Let $X_{\ell,j}$ be the set of elements in X_ℓ having color j. Use those colors to assign the partition of E_i: Every element in $E_i \cap X_{\ell,j}$ is assigned to the set B_{ij}. Finally, each of the elements in $E_i \setminus X_\ell$ selects a value $j \in \{1, \ldots, r_i\}$ uniformly, and is assigned then to the corresponding B_{ij}.

Using the definition of $\mathcal{Q}'$,

$$\mathbb{E}_{X_\ell, \mathcal{Q}'}\left[\mathrm{rk}_{\mathcal{Q}'}(X_\ell \cap E_i)\right] = \sum_{j=1}^{r_i} \Pr(X_{\ell,j} \cap E_i \neq \emptyset) = \sum_{j=1}^{r_i} \left(1 - \Pr(X_{\ell,j} \cap E_i = \emptyset)\right). \quad (2.23)$$

Focus on the j-th term of the sum above and condition on the size t of $X_{\ell,j}$. Under this assumption, $X_{\ell,j}$ is a random subset of $\mathcal{S}$ of size t. From here,

$$\Pr\left(X_{\ell,j} \cap E_i = \emptyset \;\middle|\; |X_{\ell,j}| = t\right) = \frac{\binom{n-|E_i|}{t}}{\binom{n}{t}} = \prod_{\ell=0}^{t-1} \left(1 - \frac{|E_i|}{n-\ell}\right) \leq \left(1 - \frac{|E_i|}{n}\right)^t.$$

By removing the conditioning and using that t is a binomial random variable with parameters ℓ and $1/r_i$,

$$\begin{aligned}
\Pr(X_{\ell,j} \cap E_i = \emptyset) &\leq \sum_{t=0}^{\ell} \left(1 - \frac{|E_i|}{n}\right)^t \cdot \binom{\ell}{t} \left(\frac{1}{r_i}\right)^t \left(1 - \frac{1}{r_i}\right)^{\ell - t} \\
&= \left(\frac{1}{r_i}\left(1 - \frac{|E_i|}{n}\right) + \left(1 - \frac{1}{r_i}\right)\right)^\ell \\
&= \left(1 - \frac{|E_i|}{n r_i}\right)^\ell.
\end{aligned}$$

Replacing this in (2.23), and using that $\lambda_i = |E_i|/r_i$ we have

$$\mathbb{E}_{X_\ell, \mathcal{Q}'}\left[\mathrm{rk}_{\mathcal{Q}'}(X_\ell \cap E_i)\right] \geq r_i \left(1 - \left(1 - \frac{|\lambda_i|}{n}\right)^\ell\right) \geq r_i\Big(1 - \exp(-\lambda_i \ell / n)\Big). \quad \square$$

Consider a random set X_ℓ of size ℓ in $\mathcal{S}$. The rank of X_ℓ in $\mathcal{Q}'|_{E_i} = \mathcal{P}(\mathcal{M}_i)$ is simply the number of subparts in $\{B_{i1}, \ldots, B_{ir_i}\}$ this set intersect. If E_i has high density (say $\lambda_i \geq n/\ell$), then we expect E_i to contain $|E_i|(\ell/n) \geq r_i$ elements of X_ℓ. As they are roughly equally distributed among the subparts of E_i, we expect the rank of $X_\ell \cap E_i$ to be close to r_i. On the other hand, if the set E_i has low density then we expect it to contain less than r_i elements of X_ℓ, and so we expect the rank of $X_\ell \cap E_i$ to be closer to its expected cardinality. The following lemma formalizes this intuition.

Lemma 2.4.14. *For every $1 \le \ell \le n$, and every $1 \le i \le k$,*

$$\begin{aligned}\mathbb{E}_{X_\ell,\mathcal{Q}'}[\mathrm{rk}_{\mathcal{Q}'}(X_\ell \cap E_i)] &\ge \left(1-\frac{1}{e}\right)\min\Big\{\mathbb{E}_{X_\ell}[|X_\ell \cap E_i|], r_i\Big\} \\ &= \begin{cases}(1-1/e)\ r_i, & \text{if } \lambda_i \ge n/\ell;\\ (1-1/e)|E_i|\ell/n, & \text{if } \lambda_i \le n/\ell.\end{cases}\end{aligned}$$

Proof. First note that $\mathbb{E}_{X_\ell}[|X_\ell \cap E_i|] = |E_i|(\ell/n)$. This quantity is larger than or equal to r_i if and only if $\lambda_i \ge n/\ell$. Suppose that this is the case. Using Lemma 2.4.13 and that the function $(1-e^{-x})$ is increasing, we obtain

$$\begin{aligned}\mathbb{E}_{X_\ell,\mathcal{Q}'}[\mathrm{rk}_{\mathcal{Q}'}(X_\ell \cap E_i)] &\ge \Big(1-\exp(-\lambda_i\ell/n)\Big)r_i \\ &\ge \left(1-\frac{1}{e}\right)r_i = \left(1-\frac{1}{e}\right)\min\Big\{\mathbb{E}_{X_\ell}[|X_\ell \cap E_i|], r_i\Big\}.\end{aligned}$$

Suppose now that $\lambda_i \le n/\ell$. Since the function $(1-e^{-x})/x$ is decreasing, we obtain

$$\begin{aligned}\mathbb{E}_{X_\ell,\mathcal{Q}'}[\mathrm{rk}_{\mathcal{Q}'}(X_\ell \cap E_i)] &\ge \frac{\Big(1-\exp(-\lambda_i\ell/n)\Big)}{\lambda_i\ell/n} r_i\lambda_i\ell/n \\ &\ge \left(1-\frac{1}{e}\right)|E_i|\ell/n = \left(1-\frac{1}{e}\right)\min\Big\{\mathbb{E}_{X_\ell}[|X_\ell \cap E_i|], r_i\Big\}. \quad \square\end{aligned}$$

Now we are ready to prove Theorem 2.4.12

Proof of Theorem 2.4.12. Since the densities $(\lambda_i)_{i=1}^k$ form a decreasing sequence, there is an index i^* such that $\lambda_i \ge n/\ell$ if and only if $1 \le i \le i^*$. The set $\bigcup_{i=1}^{i^*} E_i$ is equal to the set F_{i^*} in the principal sequence of the matroid $\mathcal{M} \setminus L$. Let $F = L \cup F_{i^*}$.

Every set in the principal sequence has the same rank in both $\mathcal{M}$ and $\mathcal{Q}'$. Using this fact and properties of the rank function we get:

$$\begin{aligned}\mathbb{E}_{X_\ell}[\mathrm{rk}_{\mathcal{M}}(X_\ell)] &\le \mathbb{E}_{X_\ell}[\mathrm{rk}_{\mathcal{M}}(X_\ell \cap F) + \mathrm{rk}_{\mathcal{M}}(X_\ell \cap (\mathcal{S} \setminus F))] \\ &\le \mathrm{rk}_{\mathcal{M}}(F_{i^*}) + \mathbb{E}_{X_\ell}[|X_\ell \cap (\mathcal{S} \setminus F)|] \\ &= \sum_{i=1}^{i^*} r_i + \sum_{i=i^*+1}^{k} |E_i|(j/n). \\ &\le \frac{\sum_{i=1}^k \mathbb{E}_{X_\ell,\mathcal{Q}'}[\mathrm{rk}_{\mathcal{Q}'}(X_\ell \cap E_i)]}{(1-1/e)} \\ &= \frac{\mathbb{E}_{X_\ell,\mathcal{Q}'}[\mathrm{rk}_{\mathcal{Q}'}(X_\ell)]}{(1-1/e)},\end{aligned}$$

where the last inequality follows from Lemma 2.4.14. This concludes the proof. $\square$

To end this section, we prove a lemma that translates the result of Theorem 2.4.12 to a more useful setting for the matroid secretary problem.

Lemma 2.4.15. *For two matroids $\mathcal{M}_1$ and $\mathcal{M}_2$ on the same ground set $\mathcal{S}$ of size n (but possibly having randomly defined independent set families), and a constant $\alpha \geq 0$, the following two statements are equivalent:*

(i) *For all $1 \leq \ell \leq n$,*

$$\mathbb{E}_{X_\ell,\mathcal{M}_1}[\mathrm{rk}_{\mathcal{M}_1}(X_\ell)] \geq \alpha\, \mathbb{E}_{X_\ell,\mathcal{M}_2}[\mathrm{rk}_{\mathcal{M}_2}(X_\ell)], \tag{2.24}$$

where X_ℓ is a cardinality ℓ subset of $\mathcal{S}$ selected uniformly at random, independently from any random choice defining the matroids.

(ii) *For every adversarial list of weights $w_1 \geq w_2 \geq \cdots \geq w_n \geq 0$,*

$$\mathbb{E}_{\sigma,\mathcal{M}_1}[w(\mathrm{OPT}_{\mathcal{M}_1}(\sigma))] \geq \alpha \mathbb{E}_{\sigma,\mathcal{M}_2}[w(\mathrm{OPT}_{\mathcal{M}_2}(\sigma))], \tag{2.25}$$

where $\sigma\colon [n] \to \mathcal{S}$ is a bijective map selected uniformly at random, independently from any random choice defining the matroids.

Proof. We start by rewriting $\mathbb{E}_{\sigma,\mathcal{M}}[w(\mathrm{OPT}_{\mathcal{M}}(\sigma))]$ in a more useful way.

Let $X_\ell^\sigma = \{\sigma(1), \sigma(2), \ldots, \sigma(\ell)\}$ be the (random) set of elements in $\mathcal{S}$ receiving the top ℓ weights of the adversarial list of weights. Note that

$$\begin{aligned}
\Pr(\sigma(\ell) \in \mathrm{OPT}_{\mathcal{M}}(\sigma)) &= \Pr(\mathrm{rk}_{\mathcal{M}}(X_\ell^\sigma) - \mathrm{rk}_{\mathcal{M}}(X_{\ell-1}^\sigma) = 1) \\
&= \mathbb{E}_{\sigma,\mathcal{M}}[\mathrm{rk}_{\mathcal{M}}(X_\ell^\sigma)] - \mathbb{E}_{\sigma,\mathcal{M}}[\mathrm{rk}_{\mathcal{M}}(X_{\ell-1}^\sigma)] \\
&= \mathbb{E}_{X_\ell,\mathcal{M}}[\mathrm{rk}_{\mathcal{M}}(X_\ell)] - \mathbb{E}_{X_{\ell-1},\mathcal{M}}[\mathrm{rk}_{\mathcal{M}}(X_{\ell-1})],
\end{aligned}$$

where for the last line we used that X_ℓ^σ is a uniform random set of ℓ elements. Then,

$$\begin{aligned}
\mathbb{E}_\sigma[w(\mathrm{OPT}_{\mathcal{M}}(\sigma))] &= \sum_{\ell=1}^{n} w_\ell \Pr(\sigma(\ell) \in \mathrm{OPT}_{\mathcal{M}}(\sigma)) \\
&= \sum_{\ell=1}^{n} w_\ell \left(\mathbb{E}_{X_\ell,\mathcal{M}}[\mathrm{rk}_{\mathcal{M}}(X_\ell)] - \mathbb{E}_{X_{\ell-1},\mathcal{M}}[\mathrm{rk}_{\mathcal{M}}(X_{\ell-1})]\right) \\
&= w_n\, \mathbb{E}_{X_n,\mathcal{M}}[\mathrm{rk}_{\mathcal{M}}(X_n)] + \sum_{\ell=1}^{n-1} (w_\ell - w_{\ell+1})\, \mathbb{E}_{X_\ell,\mathcal{M}}[\mathrm{rk}_{\mathcal{M}}(X_\ell)]. \qquad (2.26)
\end{aligned}$$

Assume that condition (i) holds, then each term for $\mathcal{M}_1$ in the above sum is at least α times the corresponding term for $\mathcal{M}_2$, implying that condition (ii) holds.

On the other hand, if condition (i) does not hold, then then there is an index j, for which

$$\mathbb{E}_{X_\ell,\mathcal{M}_1}[\mathrm{rk}_{\mathcal{M}_1}(X_\ell)] < \alpha \mathbb{E}_{X_\ell,\mathcal{M}_2}[\mathrm{rk}_{\mathcal{M}_2}(X_\ell)].$$

Consider the sequence of weights given by $w_1 = w_2 = \cdots = w_j = 1$ and $w_{j+1} = \cdots = w_n = 0$. For this sequence

$$\begin{aligned}
\mathbb{E}_\sigma[w(\mathrm{OPT}_{\mathcal{M}_1}(\sigma))] &= \mathbb{E}_{X_\ell,\mathcal{M}_1}[\mathrm{rk}_{\mathcal{M}_1}(X_\ell)] \\
&< \alpha \mathbb{E}_{X_\ell,\mathcal{M}_2}[\mathrm{rk}_{\mathcal{M}_2}(X_\ell)] = \alpha \mathbb{E}_\sigma[w(\mathrm{OPT}_{\mathcal{M}_2}(\sigma))].
\end{aligned}$$

□

As a corollary, we obtain the following result for the partition matroids $\mathcal{P}', \mathcal{Q}'$ associated to $\mathcal{M}$.

Lemma 2.4.16. *For every adversarial list of weights $w_1 \geq w_2 \geq \cdots \geq w_n \geq 0$,*

$$\mathbb{E}_\sigma[w(\mathrm{OPT}_{\mathcal{P}'}(\sigma))] \geq \mathbb{E}_{\sigma,\mathcal{Q}'}[w(\mathrm{OPT}_{\mathcal{Q}'}(\sigma))] \geq \left(1 - \frac{1}{e}\right) \mathbb{E}_\sigma[w(\mathrm{OPT}_{\mathcal{M}}(\sigma))].$$

Proof. The first inequality follows by Lemma 2.4.11, and the second from Theorem 2.4.12 and Lemma 2.4.15. □

2.5 General Matroids

In this section we apply the divide and conquer idea of Section 2.2 to obtain constant competitive algorithms for general matroids.

2.5.1 Random-Assignment Random-Order Model

We use the same notation as in the previous section. Consider Algorithm 6 depicted below.

Algorithm 6 for general matroids in the random-assignment random-order model.

1: Compute the principal minors $(\mathcal{M}_i)_{i=1}^k$ of the matroid obtained by removing the loops of $\mathcal{M}$.
2: Run Algorithm 4 (with parameter p) for uniformly dense matroids in parallel on each $\mathcal{M}_i$ and return the union of the answers.

As the set returned is independent in the matroid $\mathcal{M}' = \mathcal{U}(L, 0) \oplus \bigoplus_{i\in[k]} \mathcal{M}_i$, it is also independent in $\mathcal{M}$. Therefore, the algorithm is correct. In order to estimate its competitive ratio, we use the properties of Algorithm 4 for uniformly dense matroids.

Theorem 2.3.5 states that for every uniformly dense matroid $\mathcal{N}$ of total rank r, Algorithm 4 (with parameter p) returns a set of expected weight at least $\gamma_2(p)$ times the weight of the top r weights in the matroid (this is stronger than just being $(\gamma_2(p))^{-1}$-competitive). In other words, Algorithm 4 returns a set of expected weight at least $\gamma_2(p)$ times the one of the optimum independent set in the uniform matroid of total rank r having the same ground set as $\mathcal{N}$. As every matroid $\mathcal{M}_i = (E_i, \mathcal{I}_i)$ is uniformly dense, the previous argument suggests that Algorithm 4 recovers, in expectation $\gamma_2(p)$-fraction of the optimum weight of the partition matroid $\mathcal{P}' = \mathcal{U}(L, 0) \oplus \bigoplus_{i\in[k]} \mathcal{U}(E_i, r_i)$ defined in the previous section. We now formalize this result.

Lemma 2.5.1. *Let* ALG *be the set returned by Algorithm 6. Then,*

$$\mathbb{E}_{\sigma,\pi}[w(\mathrm{ALG})] \geq \gamma_2(p)\, \mathbb{E}_\sigma[w(\mathrm{OPT}_{\mathcal{P}'}(\sigma))]. \tag{2.27}$$

Proof. The only ingredient left is to argue that Algorithm 4 is effectively being applied over an instance of the random-assignment random-order model. Observe that

the random bijection $\sigma\colon [n] \to W$ that is used to assign the weights in W to the elements of the matroid can be viewed as the composition of a random partition of $[n]$ and W into blocks of sizes $(|L|, |E_1|, |E_2|, \ldots, |E_k|)$, and a collection of random bijections between the corresponding blocks. Conditioned on the random partition, each block receives a hidden list of weights which are assigned uniformly at random to the elements of the block. To complete the proof we only need to observe that the random order in which the elements of each block are presented to the algorithm processing that block is also uniform. □

Now we can a give a first bound for the competitive ratio of Algorithm 6.

Theorem 2.5.2. *For $p = p_2 \approx 0.384374$, Algorithm 6 is $1/(\gamma_2(p_2)(1-1/e)) \approx 7.78455$-competitive for the random-assignment random-order model.*

Proof. Direct from Lemmas 2.5.1 and 2.4.16. □

We note here that this competitive ratio is already an improvement over the $2e/(1-1/e) \approx 8.60052$ algorithm presented by the author in the SODA paper [152], which used Algorithm 3 instead of Algorithm 4 as a subroutine.

We can further improve the estimation of this competitive ratio by using another property of Algorithm 4. Theorem 2.3.5 states that, when applied on a uniformly dense matroid $\mathcal{N}$, Algorithm 4 (with parameter p) returns a set of expected weight at least $\gamma_1(p)$ times the expected weight of the optimum in the random partition matroid $\mathcal{P}(\mathcal{N})$. Recall now that the random partition matroid $\mathcal{Q}'$ defined from a general matroid $\mathcal{M}$ contains, for every uniformly dense matroid $\mathcal{M}_i$, a summand $\mathcal{P}(\mathcal{M}_i)$. Since every summand is treated independently in Algorithm 6, we conclude that this algorithm recovers, in expectation, $\gamma_1(p)$-fraction of the optimum weight of the partition matroid $\mathcal{Q}'$.

Lemma 2.5.3. *Let* ALG *be the set returned by Algorithm 6. Then,*

$$\mathbb{E}_{\sigma,\pi}[w(\text{ALG})] \geq \gamma_1(p)\mathbb{E}_{\sigma,\mathcal{Q}'}[w(\text{OPT}_{\mathcal{Q}'}(\sigma))]. \tag{2.28}$$

Proof. The proof is analogous to the one of Lemma 2.5.1. □

Now we can give the current tightest bound for the competitive ratio of Algorithm 6.

Theorem 2.5.4. *For $p = p_1 \approx 0.433509$, Algorithm 6 is $1/(\gamma_1(p_1)(1-1/e)) \approx 5.7187$-competitive for the random-assignment random-order model of the matroid secretary problem.*

Proof. Direct from Lemmas 2.5.3 and 2.4.16. □

2.5.2 Random-Assignment Adversarial-Order Model

Oveis Gharan and Vondrák [136] have noticed that by combining their 40-competitive algorithm for uniformly dense matroids on the random-assignment adversarial-order model with our Lemma 2.4.16 one can get a $40/(1-1/e)$-competitive algorithm for general matroids. We improve this result by using Algorithm 5 instead.

Consider the procedure depicted as Algorithm 7 below.

Algorithm 7 for general matroids in the random-assignment adversarial-order model.

1: Compute the principal minors $(\mathcal{M}_i)_{i=1}^k$ of the matroid obtained by removing the loops of $\mathcal{M}$.
2: Run Algorithm 5 for uniformly dense matroids in parallel on each $\mathcal{M}_i$ and return the union of the answers.

Theorem 2.3.6 states that for every uniformly dense matroid $\mathcal{N}$ of total rank r, Algorithm 5 returns a set of expected weight at least $1/16$ times the expected weight of the optimum in the random partition matroid $\mathcal{P}(\mathcal{N})$. In particular, since each matroid $\mathcal{M}_i$ is uniformly dense, and the elements of each matroid receive a random permutation of certain hidden list of weights (see the proof of Lemma 2.5.1), Algorithm 7 recovers in expectation $1/16$ times the optimum weight of the partition matroid $\mathcal{Q}'$ defined in Section 2.4.2. Therefore, we have the following theorem.

Theorem 2.5.5. *Let* ALG *be the set returned by Algorithm 7 when applied on a uniformly dense matroid. Then,*

$$\mathbb{E}_\sigma[w(\mathrm{ALG})] \geq \frac{1}{16}\mathbb{E}_{\sigma,\mathcal{Q}'}[w(\mathrm{OPT}_{\mathcal{Q}'}(\sigma))]. \tag{2.29}$$

By Lemma 2.4.16, Algorithm 7 is $16/(1-1/e) \approx 25.3116$*-competitive for the random-assignment adversarial-order model of the matroid secretary problem.*

2.6 New Results for the Adversarial-Assignment Random-Order Model

Constant competitive algorithms for the adversarial-assignment random-order model of the matroid secretary problem remain elusive.

Unfortunately, the strategy used for random-assignment models is not very useful in this setting. Lai [100] has shown that every matroid $\mathcal{M}$ is a restriction of a uniformly dense matroid $\mathcal{M}'$. This means that any algorithm for uniformly dense matroids can be transformed immediately into one for general matroids having the same competitive ratio (see Lemma 1.5.5).

The best algorithm for the adversarial-assignment random-order model so far is an $O(\log r)$-competitive due to Babaioff et al. [8] (see Lemma 1.5.4). This algorithm has many features, including the fact that it does not need to know the matroid beforehand; it only needs to know the number of elements and have access to an oracle

that test independence only on subsets of elements it has already seen. Nevertheless, this algorithm makes use of the actual values of the weights being revealed (in other words, it is not a comparison-based algorithm, as our definition for the matroid secretary problem requires). In this section we present a new algorithm having the same features but that only uses the relative order of weights seen and not their numerical values.

Later in this section, we show new constant-competitive algorithms in this model for certain matroid classes.

We preserve the same notation used in Section 2.1. The only difference with the previous settings is that the assignment $\sigma\colon [n] \to \mathcal{S}$ is adversarial. In particular, $w_1 \geq w_2 \geq \cdots \geq w_n \geq 0$ denotes the collection of element weights in decreasing order (without loss of generality we assume there are no ties as otherwise the algorithm can break them using a random permutation τ). Also, we use $\sigma(1), \sigma(2), \ldots, \sigma(n)$ to denote the elements of $\mathcal{S}$ sorted from largest to smallest weight.

We also use indistinctly $\mathrm{OPT}_{\mathcal{M}}$ or $\mathrm{OPT}_{\mathcal{M}}(\sigma)$ to denote the optimum independent set $\{x_1, \ldots, x_r\}$ of $\mathcal{M}$ under this ordering, with $\sigma^{-1}(x_1) < \sigma^{-1}(x_2) < \cdots < \sigma^{-1}(x_r)$ (or equivalently $w(x_1) > w(x_2) > \cdots > w(x_r)$). We drop the subindex $\mathcal{M}$ whenever there is no possibility for confusion.

2.6.1 General $O(\log r)$-Competitive Algorithm

The following algorithm returns an independent set of the matroid $\mathcal{M}$. With probability $1/2$, run the classical secretary algorithm (say, Algorithm 1) on the set of non-loops of the matroid. This returns the heaviest non-loop of the stream with probability at least $1/e$. Otherwise, observe the first m elements of the stream, where m is chosen from the binomial distribution $\mathrm{Bin}(n, 1/2)$ (as usual, denote this set of elements as the *sample*) and compute the optimum base $A = \{a_1, \ldots, a_k\}$ (with $w(a_1) \geq \cdots \geq w(a_k)$) of the sampled elements. Afterwards, select a number $\ell \in \{1, 3, 9, \ldots, 3^t\}$ with $t = \lfloor \log_3 r \rfloor$ uniformly at random, run the greedy procedure on the set of non-sampled elements having weight at least the one of a_ℓ as they arrive and return its answer (if $\ell > k$, run the greedy procedure over the entire set of non-sampled elements).

It is possible to implement this algorithm without knowing the matroid beforehand; we only need to know the number of elements n and have access to an oracle to test independence on subsets of already seen elements. For that we need to make two changes to the algorithm above.

First we require to make a slight modification to the algorithm for the classical secretary problem we use (Algorithm 1) so that it only consider the non-loops of the stream *without knowing a priori the number of non-loops*. The modification samples the first N elements of the stream, where N is distributed as $\mathrm{Bin}(n, 1/e)$, and then returns the first non-loop having weight larger than any sampled non-loop (if any). Observe that if n' is the number of non-loops of the matroid, then the number of non-loops sampled has distribution $\mathrm{Bin}(n', 1/e)$. This observation implies this algorithms does exactly what Algorithm 1 would do if it knew the number of non-loops beforehand.

The second change deals with the number $t = \lfloor \log_3 r \rfloor$ in the algorithm above. As we do not know the rank of the matroid a priori, we can not use this value. Instead, we use the rank k of the sampled set (which we can compute) to estimate it: We select $t \in \{\lfloor \log_3 k \rfloor, \lfloor \log_3 k \rfloor + 1\}$ uniformly at random and use this value in the previous algorithm.

The full description of this algorithm is depicted as Algorithm 8.

Algorithm 8 for general matroids.

1: With probability 1/2 run an e-competitive algorithm for the classical secretary problem *on the set of non-loops* and return its answer.
2: Otherwise, set ALG $\leftarrow \emptyset$.
3: Choose m from the binomial distribution $\mathrm{Bin}(n, 1/2)$.
4: Observe the first m elements and denote this set as the sample.
5: Compute the optimum base A for the sample. Let $a_1, \ldots, a_k$ be the elements of A in decreasing order of weight.
6: If the total rank r of the matroid is known, set $t = \lfloor \log_3 r \rfloor$, otherwise, select $t \in \{\lfloor \log_3 k \rfloor, \lfloor \log_3 k \rfloor + 1\}$ uniformly at random.
7: Select ℓ uniformly at random from the set $\{1, 3, 9, \ldots, 3^t\}$.
8: **for** each element x arriving after the first m elements **do**
9: If ALG $\cup \{x\}$ is independent and $w(x) \geq w(a_\ell)$ (where $w(a_\ell) = 0$ if $\ell > k$), add x to ALG.
10: **end for**
11: Return the set ALG.

To analyze this algorithm, note the following. For every ℓ the algorithm can choose (provided $\ell \leq k$), the sample contains an independent set of size ℓ containing only elements of weight at least the one of a_ℓ (namely the set $\{a_1, \ldots, a_\ell\}$ itself). Since the sampled set behaves similarly to the non-sampled one, we expect that the same happens outside the sample. In particular, the greedy procedure should recover a weight of roughly $\ell w(a_\ell)$. By taking the expectation over the choices of ℓ it is not hard to check that the expected weight returned by the algorithm is at least $\Omega(\mathbb{E}[w(A)/\log_3(r)]) = \Omega(\mathbb{E}[w(\mathrm{OPT})/\log_3(r)])$. We give the formal proof below.

Theorem 2.6.1. *Algorithm 8 is $O(\log r)$-competitive for any matroid of rank r.*

Proof. Assume first that the rank r of the matroid is known. Let $\mathrm{OPT} = \{x_1, \ldots, x_r\}$ with $w(x_1) > \cdots > w(x_r)$ be the maximum independent set of the matroid, T the set of sampled elements and T' the set of non-sampled ones.

Let $A = \{a_1, \ldots, a_k\}$ be the optimum set in T, with $w(a_1) > \cdots > w(a_k)$ (independent of whether the algorithm computes A or not). This set can be obtained by sorting T in decreasing order of weights and applying the greedy procedure. This fact implies that if an element x_i of the optimum is sampled, then x_i appears in the set A.

Every element of the matroid is sampled independently with probability 1/2, including the elements of the optimum. Therefore, by the previous paragraph,

$$\mathbb{E}[w(A)] \geq \frac{w(\mathrm{OPT})}{2}. \tag{2.30}$$

To simplify our analysis, in the following we assume that for $i > k$, a_i is a dummy element with $w(a_i) = 0$. Given the number ℓ chosen by the algorithm (if the algorithm reaches that state), the weight of the set returned will be at least $w(a_\ell)$ times the number of elements the greedy procedure selects; therefore, $\mathbb{E}[w(\mathrm{ALG})]$ is at least

$$\frac{w(x_1)}{2e} + \frac{1}{2(1+\lfloor \log_3 r \rfloor)} \sum_{j=0}^{\lfloor \log_3 r \rfloor} \mathbb{E}\left[w(a_\ell) \cdot |\mathrm{ALG}| \,\middle|\, \ell = 3^j \text{ was selected} \right]. \tag{2.31}$$

Let $H(a_\ell)$ be the collection of non-sampled elements that are at least as heavy as a_ℓ. If the algorithm chooses the number ℓ, it will then execute the greedy procedure on $H(a_\ell)$ and return a set of cardinality equal to the rank of $H(a_\ell)$. Note that for every ℓ, $w(x_\ell) \geq w(a_\ell)$; therefore, the rank of $H(a_\ell)$ is at least the number of nonsampled elements in $\{x_1, \ldots, x_\ell\}$.

Using a Chernoff bound (see, e.g. [118]),

$$\Pr\Big(|\{x_1, \ldots, x_\ell\} \cap T'| \leq \ell/4 \Big) \leq \exp(-\ell/8).$$

In particular, if $\ell \geq 9$,

$$\mathbb{E}[w(a_\ell) \cdot |\mathrm{ALG}| \,\big|\, \ell\,] \geq \mathbb{E}[w(a_\ell)](1 - \exp(-\ell/8))\ell/4 \geq \mathbb{E}[w(a_\ell)]\ell/6.$$

Replacing this in (2.31), and dropping the values of j for which $\ell < 9$, we get

$$\begin{aligned}
\mathbb{E}[w(\mathrm{ALG})] &\geq \frac{w(x_1)}{2e} + \frac{1}{12(1+\lfloor \log_3 r \rfloor)} \sum_{j=2}^{\lfloor \log_3 r \rfloor} \mathbb{E}[w(a_{3^j})] 3^j \\
&\geq \mathbb{E}\left[\frac{w(\{a_1, \ldots, a_8\})}{16e} + \frac{1}{24(1+\lfloor \log_3 r \rfloor)} \sum_{j=2}^{\lfloor \log_3 r \rfloor} w(\{a_{3^j}, \ldots, a_{3^{j+1}-1}\}) \right] \\
&\geq \frac{\mathbb{E}[w(A)]}{16e(1+\lfloor \log_3 r \rfloor)}.
\end{aligned}$$

Using Inequality (2.30), we obtain

$$\mathbb{E}[w(\mathrm{ALG})] \geq \frac{w(\mathrm{OPT})}{32e(1+\lfloor \log_3 r \rfloor)},$$

which implies the algorithm is $O(\log r)$-competitive.

Suppose now that the rank r is unknown. If r is small, say $r \leq 12$, then with probability $1/(2e)$ the algorithm will run the standard secretary algorithm and return element x_1. This element has weight at least $1/12$ fraction of the optimum; therefore the algorithm is $24e$-competitive for this case.

For the case where $r > 12$ we use a different analysis. The random variable k

denoting the rank of the sampled set could be strictly smaller than r. However, the probability that $k \leq r/3$ is small. Indeed, for that event to happen we require that at most $1/3$ of the elements of OPT are in the sample. By Chernoff bound, this happens with probability

$$\Pr\Big(|\{x_1, \ldots, x_r\} \cap T| \leq r/3\Big) \leq \exp(-r/18) \leq \exp(-13/18) \leq 1/2.$$

Noting that $r/3 \leq k \leq r$ implies that $\lfloor \log_3 r \rfloor \in \{\lfloor \log_3 k \rfloor, \lfloor \log_3 k \rfloor + 1\}$, we deduce that with probability at least $1/4$ our algorithm guesses $t = \lfloor \log_3 r \rfloor$ right; therefore, the competitive ratio of this algorithm is at most 4 times worse than the one that knows the rank beforehand. □

2.6.2 Column-Sparse Linear Matroids

Let $\mathcal{M} = (V, \mathcal{I})$ be a linear matroid represented by a matrix A. Consider the following algorithm for $\mathcal{M}$ (depicted as Algorithm 9): Randomly permute the rows of A and define for every row i, the sets $C_i = \{v \in V : v_i \neq 0\}$ and $B_i = C_i \setminus \bigcup_{j<i} C_j$, where v_i denotes the i-th coordinate of column v in the permuted matrix. Next, run any e-competitive secretary algorithm for the partition matroid that accepts at most one element from each B_i.

Algorithm 9 for a matroid $\mathcal{M} = (V, \mathcal{I})$ represented by a matrix A.

1: Permute the rows of A at random to obtain a matrix A'. Index the rows of A' as $1, 2, \ldots, \ell$.
2: Let $C_i = \{v \in V : v_i \neq 0\}$ and $B_i = C_i \setminus \bigcup_{j<i} C_j$ where v_i is the i-th coordinate of the column associated to v in matrix A'
3: Let $\mathcal{P}$ be the partition matroid on V whose independent sets contain at most one element from each B_i.
4: Run any e-competitive secretary algorithm for $\mathcal{P}$ on the stream of elements and return its answer.

Theorem 2.6.2. *Algorithm 9 returns an independent set of $\mathcal{M}$. Furthermore, if the matrix A representing $\mathcal{M}$ is such that every column contains at most k non-zero entries, then this algorithm is ke-competitive (assuming a random-order model).*

Proof. We show first that the returned set is independent. If this was not the case there would be a circuit C inside the output. Let $v \in C$ be the element belonging to the set B_i of smallest index i. By definition of v, the elements of $C \setminus v$ are not in C_i; therefore, C and C_i intersects only in v. This is a contradiction since C_i is in the cocircuit space of the matroid. (Use, e.g. [137, Proposition 2.1.11].)

To show that the algorithm is ke-competitive, construct the bipartite graph G having parts the rows and columns of A, where there is an edge (i, v) from row i to column v if the corresponding entry of A is non-zero. Assign to every edge incident to column v a weight equal to the weight of the matroid element v.

Consider the following simulation algorithm: Randomly permute the vertices in the row part of the graph. Delete all the edges, except the ones going from a column vertex to its lowest neighbor (this is, to the row having smallest index in the random permutation). Finally, run any e-competitive secretary algorithm for the partition matroid that accepts for each row vertex, at most one edge incident to it. This returns a matching in G with the same weight as the set of elements the original algorithm returns.

If X is a set of columns independent in $\mathcal{M}$ then the row-rank of the submatrix of A induced by X is equal to its cardinality. In particular, the number of row vertices that X dominates in G is at least $|X|$. Using Hall's Theorem we conclude that there is a matching covering each independent set of columns. In particular, the weight of the maximum weight matching M^* of G is at least the one of the optimum independent set of $\mathcal{M}$. On the other hand, M^* has weight at most the one of the edge set $\{(i, v^*(i))\colon i \in \text{rows}(A)\}$, where $v^*(i)$ is the maximum weight neighbor of i in G. Since each edge $(i, v^*(i))$ is not deleted with probability $1/k$ and, given that it is not deleted, the simulation selects it with probability $1/e$, we conclude that Algorithm 9 is ke-competitive. □

By applying this algorithm to graphic matroids, which are representable by matrices having only 2 ones per column, we recover the $2e$-competitive algorithm of Korula and Pál [97].

2.6.3 Low Density Matroids

The **matroid polytope** $\mathrm{P}_{\mathcal{M}} \subseteq \mathbb{R}^{\mathcal{S}}$ of a matroid $\mathcal{M} = (\mathcal{S}, \mathcal{I})$ is the convex hull of the indicator vectors of its independent sets. This polytope can be characterized (see, e.g. [147, Chapter 40]) as

$$\mathrm{P}_{\mathcal{M}} = \{y \in \mathbb{R}^{\mathcal{S}}\colon y \geq 0 \text{ and } y(U) \leq \mathrm{rk}(U), \text{ for all } U \subseteq \mathcal{S}\}.$$

Let $\mathcal{M} = (\mathcal{S}, \mathcal{I})$ be a loopless matroid of density

$$\gamma(\mathcal{M}) = \max_{\emptyset \neq U \subseteq \mathcal{S}} \frac{|U|}{\mathrm{rk}(U)},$$

and $\tau^{\mathcal{M}} \in \mathbb{R}^{\mathcal{S}}$ be the vector having all its coordinates equal to $1/\gamma(\mathcal{M})$, then we have the following property.

Lemma 2.6.3. *For every loopless matroid $\mathcal{M}$, the corresponding vector $\tau^{\mathcal{M}}$ is in the matroid polytope $\mathrm{P}_{\mathcal{M}}$.*

Proof. This follows since for every $U \subseteq \mathcal{S}$,

$$\tau^{\mathcal{M}}(U) = \sum_{u \in U} \tau^{\mathcal{M}}(u) = \frac{|U|}{\gamma(\mathcal{M})} \leq \mathrm{rk}(U). \qquad \square$$

The previous lemma implies that $\tau^{\mathcal{M}}$ admits a decomposition as convex combination of independent sets of $\mathcal{M}$:

$$\tau^{\mathcal{M}} = \sum_{I \in \mathcal{I}} \lambda_I \chi_I, \text{ with } \sum_{I \in \mathcal{I}} \lambda_I = 1.$$

This decomposition can be found in polynomial time given access to an independent oracle of $\mathcal{M}$ (see [147, Chapter 40]). Consider the following algorithm (depicted as Algorithm 10) for a loopless matroid $\mathcal{M} = (\mathcal{S}, \mathcal{I})$.

Algorithm 10 for loopless matroid $\mathcal{M} = (\mathcal{S}, \mathcal{I})$.

1: Compute the decomposition $\tau^{\mathcal{M}} = \sum_{I \in \mathcal{I}} \lambda_I \chi_I$.
2: Select and return a set $I \in \mathcal{I}$ according to the probability distribution $(\lambda_I)_{I \in \mathcal{I}}$.

Lemma 2.6.4. *For every loopless matroid $\mathcal{M}$, Algorithm 10 is $\gamma(\mathcal{M})$-competitive in* any *model of the matroid secretary problem (even adversarial-assignment adversarial-order)*

Proof. Every element $u \in \mathcal{S}$ is in the output with probability

$$\sum_{I \in \mathcal{I}:\, u \in I} \lambda_I = \tau^{\mathcal{M}}(u) = 1/\gamma(\mathcal{M}).$$

Therefore, the expected weight returned is at least $1/\gamma(\mathcal{M})$ times the collective total weight of all the elements in the matroid. □

If a loopless matroid $\mathcal{M}$ contains parallel elements[4] we can potentially get a better competitive ratio by using its **simple version** $\mathcal{M}'$. This matroid $\mathcal{M}' = (\mathcal{S}', \mathcal{I}')$ is obtained by deleting all but one element in each parallel class of $\mathcal{M} = (\mathcal{S}, \mathcal{I})$. In particular, for every element u in $\mathcal{M}$ there is a unique element in $\mathcal{M}'$ representing u's parallel class.

For every independent set $I' \in \mathcal{M}'$, let $\mathcal{Q}(I')$ be the partition matroid in $\mathcal{S}$ induced by I'. In other words, the independent sets of $\mathcal{Q}(I')$ are those subsets of $\mathcal{S}$ containing at most one element from each parallel class represented in I'. In particular, every independent set in $\mathcal{Q}(I')$ is independent in the original matroid $\mathcal{M}$.

Consider Algorithm 11 depicted below.

Lemma 2.6.5. *For every loopless matroid $\mathcal{M}$, Algorithm 11 is $e\gamma(\mathcal{M})$-competitive in the adversarial-assigment random-order model of the matroid secretary problem.*

Proof. The set returned by this algorithm is independent in the original matroid $\mathcal{M}$, hence the algorithm is correct. Note that every element u of the optimum base

[4] Two elements u and v are parallel in $\mathcal{M}$ if $\{u, v\}$ is a minimal dependent set. Being parallel is an equivalence relation (considering that every element is parallel to itself). The parallel classes of $\mathcal{M}$ are the equivalence classes of this relation. A matroid is called simple if it has no loops and no pair of parallel elements.

Algorithm 11 for loopless matroid $\mathcal{M} = (\mathcal{S}, \mathcal{I})$.

1: Construct the simple matroid $\mathcal{M}' = (\mathcal{S}', \mathcal{I}')$ and compute the decomposition $\tau^{\mathcal{M}'} = \sum_{I' \in \mathcal{I}'} \lambda_{I'} \chi_{I'}$.
2: Select a set $I' \in \mathcal{I}'$ according to the probability distribution $(\lambda'_I)_{I' \in \mathcal{I}'}$.
3: Run any e-competitive algorithm for partition matroids on the matroid $\mathcal{Q}(I')$ and return its answer.

of $\mathcal{M}$ is the heaviest of its own parallel class. Provided that the parallel class of u is represented in the set I' selected in line 2, Algorithm 11 returns u with probability at least $1/e$. We conclude the proof by noting that every parallel class (i.e. every element of $\mathcal{S}'$) is selected with probability $\gamma(\mathcal{M}')$. □

It is possible to modify the previous algorithms to work on matroids having loops: Simply run them on the matroid obtained by removing the loops. The competitive ratio of the resulting algorithm depends on the density of the resulting matroid.

The competitive ratios of the described algorithms are linear in the density of the matroid (or in the density of its simple version). In particular for matroids of constant density, these algorithms are constant-competitive.

In the next subsections we describe two interesting classes of matroids having this property: cographic matroids and small cocircuit matroids.

2.6.4 Cographic Matroids

The cographic matroid $\mathcal{M}^*(G)$ of a graph G is the dual of its graphic matroid $\mathcal{M}(G)$. The independent sets in $\mathcal{M}^*(G)$ are those sets of edges that, when removed do not increase the number of connected components of G. The bases in $\mathcal{M}^*(G)$ are the complements of the maximum forests of G. The circuits (minimal dependent sets) of $\mathcal{M}^*(G)$ are exactly the minimal edge-cuts of the connected components of G. This means that the loops of $\mathcal{M}^*(G)$ are the bridges of G.

A well-known result of graph-theory states that when G is 3-edge-connected then we can find three spanning trees T_1, T_2 and T_3, such that the union of their complements covers $E(G)$ (This follows from e.g. Edmonds' Matroid Partitioning Theorem [49]). In particular, we have the following.

Lemma 2.6.6. *If every connected component of $G = (V, E)$ is 3-edge-connected, then $\gamma(\mathcal{M}^*(G)) \leq 3$.*

Proof. The above result implies that there are 3 forests F_1, F_2 and F_3 whose complements cover all the edges of G. Let $B_i = E \setminus F_i$. For every set of edges $X \subseteq E$,

$$|X| \leq \sum_{i=1}^{3} |X \cap B_i| = \sum_{i=1}^{3} \mathrm{rk}_{\mathcal{M}^*(G)}(X \cap B_i) \leq 3\,\mathrm{rk}_{\mathcal{M}^*(G)}(X),$$

where the middle equality follows since $X \cap B_i \subseteq B_i$ is independent in $\mathcal{M}^*(G)$. □

This result implies that Algorithm 10 is 3-competitive for cographic matroids of graphs having only 3-edge-connected components. An alternative algorithm in the same spirit is depicted as Algorithm 12.

Algorithm 12 for the cographic matroid of a graph $G = (V, E)$ having only 3-edge-connected components.

1: Find three forests F_1, F_2 and F_3 whose complement cover E.
2: Select $i \in \{1, 2, 3\}$ uniformly at random and return the set $B_i = E \setminus F_i$.

Lemma 2.6.7. *Algorithm 12 is 3-competitive for cographic matroids of graphs with 3-edge-connected components. This holds for any model of the matroid secretary problem.*

Proof. This follows since every edge of G is selected with probability at least $1/3$. □

We can not extend the previous results to arbitrary graphs: the cographic matroid of a bridgeless graph G can have arbitrarily high density. To see this consider the cycle C_n on n vertices. For this graph

$$\frac{|E(C_n)|}{\mathrm{rk}_{\mathcal{M}^*(C_n)}} = n. \tag{2.32}$$

Nevertheless we can still show the following result.

Lemma 2.6.8. *For any bridgeless graph G, the simple version of its cographic matroid has density at most 3.*

Proof. Let $\{C_1, \ldots, C_k\}$ be the collection of 2-edge-connected components[5] of the bridgeless graph G. Consider the graph H obtained by taking the disjoint union of the graphs C_i (using copies of the vertices that are in two or more of the C_i's). The graph H has the same graphic and cographic matroids as G, so we use H instead.

Let $P_1, \ldots, P_\ell$ be the cographic parallel classes of $\mathcal{M}^*(H)$. The simple version $\mathcal{M}'$ of $\mathcal{M}^*(H)$ is the matroid obtained from $\mathcal{M}^*(H)$ by deleting (in a matroid-sense) all but one element in each P_j. Since deleting an element of a matroid corresponds to contracting the same element in its dual, we conclude that the matroid $\mathcal{M}'$ is the cographic matroid of the graph H' obtained by contracting (in the graph-sense) all but one edge in each of the P_j.

Since $\mathcal{M}'$ has no pair of parallel elements, the components of H' have no edge-cut of size 2. Therefore, all the components of H' are 3-edge-connected. Using Lemma 2.6.6, we conclude that $\mathcal{M}'$ has density at most 3. □

The result above implies that the following algorithm is $3e$-competitive-algorithm for cographic matroids of an arbitrary graph G: Remove the bridges of G and then apply Algorithm 11. An alternative algorithm in the same spirit is depicted as Algorithm 13.

[5] A 2-edge-connected component is a maximal 2-edge-connected subgraph. The 2-edge-connected components of a bridgeless graph provide a partition of the edges of the graph.

Algorithm 13 for the cographic matroid of a graph $G = (V, E)$.

1: Remove the bridges of G.
2: Construct the associated graph H' described in the proof of Lemma 2.6.8.
3: Find three forests F_1, F_2 and F_3 whose complements cover H'.
4: Define the partition matroids $\mathcal{Q}_i = \mathcal{Q}(E(H') \setminus F_i)$ having $E(G)$ as ground set (see Section 2.6.3).
5: Select $i \in \{1, 2, 3\}$ uniformly and run any e-competitive algorithm for partition matroids on $\mathcal{Q}_i$, returning its answer.

Lemma 2.6.9. *Algorithm 13 is 3e-competitive for cographic matroids of general graphs in the adversarial-assignment random-order model.*

Proof. Analogous to the proof of Lemma 2.6.5. □

2.6.5 Matroids with Small Cocircuits

For each element u of a loopless matroid $\mathcal{M} = (\mathcal{S}, \mathcal{I})$, let $c^*(u)$ be the size of the smallest cocircuit (i.e. circuits of the dual matroid) containing u, and let

$$c^*(\mathcal{M}) = \max_{u \in \mathcal{S}} c^*(u). \tag{2.33}$$

Consider the algorithm that greedily constructs an independent set of $\mathcal{M}$ selecting elements as they appear without looking at their weights.

Theorem 2.6.10. *The algorithm described above is $c^*(\mathcal{M})$-competitive (in random-order models).*

Proof. To see this, fix an element $u \in \mathcal{S}$ and let C^* be a cocircuit of minimum size containing it. If u appears before all the other elements of C^* in the random order then it has to be selected by the algorithm. Otherwise, there would be a circuit C that intersects C^* only in element u, which is a contradiction. (See, e.g. [137, Proposition 2.1.11].) From here, we conclude that u is selected with probability at least $1/c^*(u) \geq 1/c^*(\mathcal{M})$. □

We also have the following property.

Lemma 2.6.11. *For every loopless matroid $\mathcal{M} = (\mathcal{S}, \mathcal{I})$, $\gamma(\mathcal{M}) \leq c^*(\mathcal{M})$.*

Proof. Let n be the size of $\mathcal{S}$. An element u is selected by the algorithm above if and only if u is in the lexicographic first base $\mathrm{OPT}(\pi)$ of the ordering $\pi\colon [n] \to \mathcal{S}$ in which the elements are presented. Consider the vector $\rho \in \mathbb{R}^{\mathcal{S}}$ having each coordinate equal to $1/c^*(\mathcal{M})$. Using the proof of Theorem 2.6.10, we conclude that

$$\rho(u) \leq \Pr_\pi(u \text{ is selected by the algorithm}) = \frac{1}{n!} \sum_\pi \chi_{\mathrm{OPT}(\pi)}(u).$$

In particular, for every set $U \subseteq \mathcal{S}$ we have:

$$\frac{|U|}{c^*(\mathcal{M})} = \rho(U) \leq \frac{1}{n!}\sum_{\pi} |U \cap \mathrm{OPT}(\pi)| \leq \frac{1}{n!}\sum_{\pi} \mathrm{rk}(U) = \mathrm{rk}(U),$$

where we used the fact that $U \cap \mathrm{OPT}(\pi)$ is an independent subset of U. □

In particular, the algorithm presented is no better than the Algorithm 10 for low-density matroids at least in terms of its competitive ratio. Nevertheless, the algorithm above is simpler and does need require to know the matroid beforehand.

2.7 Summary and Open Problems

In this chapter we have given new algorithms for the matroid secretary problem. Some of these results have already been presented by the author in a recent SODA paper [152].

Regarding the random-assignment random-order model, our main contribution is Algorithm 6. This algorithm is the best constant-competitive algorithm known so far for this model, achieving a competitive ratio of at most 5.7187. (See Theorem 2.5.4.) A slightly weaker algorithm achieving a competitive ratio of $2e/(1-1/e) \approx 8.60052$ was already presented by the author in [152]. The existence of constant-competitive algorithm solves an open question of Babaioff et al. [8] (see Conjecture 1.5.3).

On a very high level Algorithm 6 is an application of a simple divide and conquer idea: replace the matroid by a collection of disjoint simpler matroids for which we can easily derive a constant-competitive algorithm, and then return the union of the answers. The proposed simpler matroids are the uniformly dense matroids.

In Section 2.3 we have studied some properties of the class of uniformly dense matroids and shown how to modify existent algorithms for uniform matroids to work on this class. The best algorithm presented, Algorithm 4 has competitive ratio of at most 4.92078, improving on the $2e \approx 5.43653$ competitive algorithm previously presented by the author [152] (see Algorithm 3).

In Section 2.4 we have studied the notion of principal partition of a matroid and how to use it to construct a collection of uniformly dense minors. These matroids are used for the development of Algorithm 6 in Section 2.5. Roughly speaking, Lemma 2.4.16 states that every algorithm for uniformly dense matroids can be transformed into an algorithm for general matroids, increasing an extra factor of at most $(e/(e-1))$ on its competitive ratio.

As first noticed by Oveis Gharan and Vondrák [136] after the first publication of some of the results above, it is possible to apply our methods in the random-assignment adversarial-order model. Oveis Gharan and Vondrák have devised a 40-competitive algorithm for uniformly dense matroids on this stronger model, which by our results, can be transformed in a $40/(1-1/e) \approx 63.279$-competitive algorithm for general matroids.

In this thesis we have improved those algorithm showing a $16/(1-1/e) \approx 25.311$ for both uniformly dense matroids (Algorithm 5) and general matroids (Algorithm 7).

The last sections of this chapter deal with the "standard" matroid secretary problem: the arbitrary-assignment random-order model.

We have presented the first $O(\log r)$-competitive *comparison-based* algorithm for general matroids. The best previous result for this problem was an *valued-based* algorithm of Babaioff et al. [8] achieving the same asymptotic competitive ratio.

Afterwards, we have given a ke-competitive algorithm for linear matroids representable by k-column sparse matrices. This result contains as a special case the $2e$-competitive for graphic matroids of Korula and Pál [97].

We have also given algorithms for general matroids having competitive ratio proportional to the density of the matroid. Two special cases studied are the cographic matroids, for which we give $3e$-competitive algorithms, and matroids where each element is in a cocircuit of size at most k, for which we give a k-competitive algorithm.

2.7.1 Open Problems

Besides the obvious problem of finding a constant-competitive algorithm for general matroids in the adversarial-assignment random-order model there are several other problems to work on. Here is a list of some of them.

Consider the following setting. A weighted matroid with arbitrary hidden weights is presented to an algorithm. At every step, the algorithm can pick any element, ask for its weight, and depending on the answer add it to the solution set or reject it, provided that the solution set is always an independent set. Does this ability of choosing the order help to attain a constant-competitive algorithm for adversarial-value models of the matroid secretary problem?

Another interesting question is to determine if there is an approximate notion of principal partition for domains extending matroids, in particular for matroid intersection domains. An affirmative answer to this question could allow us to use the divide and conquer approach presented for matroids to give low competitive algorithms for the random-assignment random-order model of the corresponding generalized secretary problem.

Babaioff et al. [8] (see Lemma 1.4.1) have shown that for general domains, it is impossible to achieve a competitive ratio of $o(\log n/\log\log n)$. The proof of this lemma shows that this is true even for cases where (i) each element receives their weights from a known distribution, (ii) the algorithm is value-based and (iii) the algorithm can choose the order in which the elements are revealed. Currently, there are no known algorithms achieving an $o(n)$-competitive ratio for general domains even if the three conditions above hold. It would be interesting to close the gap between the upper and lower bounds on the competitive ratio for this problem.

Finally, it is open to determine if there are cases of the adversarial-assignment model of the generalized secretary problem in which value-based algorithms outperform comparison-based ones. This seems to be the case since comparison-based algorithms are not allowed to compare the weight of sets of elements, but currently no examples are available.

Part II

Jump Number of Two Directional Orthogonal Ray Graphs and Independent Sets of Rectangles.

Chapter 3

Posets and Perfect Graphs

In this chapter, we define partially ordered sets (posets), comparability graphs and related notions; and state some properties. We then give a survey on different classes of comparability graphs, focusing on subclasses of two directional orthogonal ray graphs and discuss some geometrical characterizations. Finally, we discuss perfect graphs and recall some of their properties.

3.1 Basic Notions of Posets

We mostly follow the notation of Trotter [164].

A **partial order relation** over a set V is a reflexive, antisymmetric and transitive binary relation. A pair $P = (V, \leq_P)$ where $\leq_P$ is a partial order over V is called a **partially ordered set** or **poset**.

Two elements u and v in V are **comparable** if $u \leq_P v$ or $v \leq_P u$. Otherwise they are **incomparable**. For u and v satisfying $u \leq_P v$, the **interval** $[u, v]_P$ is the set of elements x such that $u \leq_P x \leq_P v$. A **linear order**, also known as a **total order** is a poset in which every pair of elements is comparable. The poset $([n], \leq)$ where $\leq$ is the natural order relation is an example of total order.

An element $v \in V$ **covers** $u \in V$ if $u <_P v$ and there is no $w \in V$ such that $u <_P v <_P w$. Two important undirected graphs associated to the poset P are defined as follows.

The graph G_P having V as vertex set such that uv is an edge if u and v are comparable and $u \neq v$ is the **comparability graph** of the poset P. A graph G is a comparability graph if there is a poset P such that G is isomorphic to G_P.

The graph C_P having V as vertex set such that uv is an edge if u covers v or if v covers u is the **covering graph** of the poset P. A graph G is a covering graph if there is a poset P such that G is isomorphic to C_P.

Different posets can share the same comparability graph or the same covering graph. Covering graphs are usually represented in the plane with a structured drawing denoted as a **Hasse diagram** in such a way that if v covers u, then v is above u in the plane. Since the problems considered in this part of the thesis deal with bipartite graphs, the following remark is relevant.

Remark 3.1.1. Every bipartite graph is both a comparability and a covering graph.

Given two posets $P = (V, \leq_P)$ and $Q = (V, \leq_Q)$ on the same set V, the **intersection** $P \cap Q$ is defined as the poset $(V, \preceq)$ where $u \preceq v$ if and only if $u \leq_P v$ and $u \leq_Q v$. The intersection of a family of posets is defined analogously.

Given two posets $P = (U, \leq_P)$ and $Q = (V, \leq_Q)$ on possibly different sets, the **product** $P \times Q$ is defined as the poset $(U \times V, \preceq)$, where $(u, v) \preceq (u', v')$ if $u \leq_P u'$ and $v \leq_Q v'$. The product of a sequence of posets is defined analogously. For $d \geq 1$, we use P^d to denote the iterative product of d copies of P.

A poset $P = (V, \leq_P)$ can be **embedded** in a poset $Q = (W, \leq_P)$ if there is a map $\varphi\colon V \to W$ for which $u \leq_P v$ if and only if $\varphi(u) \leq_Q \varphi(v)$. Note that if P can be embedded in Q and Q can be embedded in R, then the first poset can be embedded in the third one by just composing the corresponding maps.

Unless specifically stated, we assume the set V to be finite and use n to denote its cardinality. The only infinite posets we consider are the usual orders of the integers $(\mathbb{Z}, \leq)$, the reals $(\mathbb{R}, \leq)$, and their finite powers $(\mathbb{Z}^d, \leq_{\mathbb{Z}^d})$ and $(\mathbb{R}^d, \leq_{\mathbb{R}^d})$. We reserve the symbol $\leq$, without subindex, to denote the total order between numbers.

3.2 Chains and Antichains

Let $P = (V, \leq_P)$ be a poset. A subset $\{u_1, u_2, \ldots, u_k\}$ of V satisfying $u_1 \leq_P u_2 \leq_P \cdots \leq_P u_k$ is called a **chain**. A set of elements that are mutually incomparable is called an **antichain**. The **height** and **width** of P are the maximum sizes of a chain and an antichain respectively. Two important results relating these concepts are the following.

Theorem 3.2.1 (Mirsky's antichain partitioning [116])**.** *The height of a poset, that is the size of a maximum chain, is equal to the minimum number of antichains into which the poset can be partitioned.*

Theorem 3.2.2 (Dilworth's chain partitioning [43])**.** *The width of a poset, that is the size of a maximum antichain, is equal to the minimum number of chains into which the poset can be partitioned.*

Mirsky's theorem is very simple to prove: for a given element $u \in V$, let $h(u)$ be its height, that is, the size of a maximum chain in P having u as its largest element. The set of elements having the same height forms an antichain, and these antichains partition P into a number of antichains equal to the poset height. This means that we can efficiently find a maximum chain and a minimum antichain partition by solving a longest path problem in the directed acyclic graph obtained from P by appending one element u_0 that is considered smaller than all the others. As finding longest paths in directed acyclic graphs can be done in linear time (see, e.g. [38]), we have the following result.

Lemma 3.2.3 (Algorithm for antichain partitioning)**.** *Let $P = (V, \leq_P)$ be a poset and let n and m be the number of vertices and edges of the comparability graph of P. We can find a maximum chain and a minimum antichain partition of P in $O(n+m)$-time.*

Dilworth's theorem is equivalent to König's theorem on bipartite graph matchings and many other related theorems including Hall's marriage theorem. An important algorithmic consequence of this is that we can find a minimum partition of a poset into chains by solving a maximum bipartite matching problem. The following reduction can be found in [59].

Given a poset $P = (V, \leq_P)$, construct a bipartite graph G having two copies of V as bipartition, where (u, v) is an edge of G if $u \leq_P v$. For any partition of P into chains, we can construct a matching by including all the edges between pairs of elements that are consecutive in some chain. The converse also hold: for any matching M we can construct a collection of disjoint chains by including u and v in the same chain whenever $(u, v) \in M$. Observe that the trivial partition of P into $n = |V|$ singleton chains corresponds to the empty matching, and that adding an edge to a matching has the effect of *merging* two of the chains in the collection. The previous observation implies that this construction maps chain partitions of size k into matchings of size $n - k$; therefore, finding a minimum chain partition of P corresponds to finding a maximum cardinality matching in G.

Computing maximum cardinality matchings in a bipartite graph G with n vertices and m edges is a well-studied problem. The current best algorithms for this task are the deterministic algorithm of Hopcroft and Karp [86] that runs in time $O(n+m\sqrt{n})$, and the randomized algorithm of Mucha and Sankowski [120] that runs in time $O(n^\omega)$, where ω is the exponent for matrix multiplication. That is, ω is the smallest exponent for which there is an algorithm running in time $O(n^\omega)$ to multiply two $n \times n$ matrices. Currently it is known that $2 \leq \omega \leq 2.376$ (the upper bound is due to the fast matrix multiplication algorithm of Coppersmith and Winograd [37]). Using this, we have the following result.

Lemma 3.2.4 (Algorithms for chain partitioning)**.** *Let $P = (V, \leq_P)$ be a poset and let n and m be the number of vertices and edges of the comparability graph of P. We can find a maximum antichain and a minimum chain partition of P in $O(n+m\sqrt{n})$-time using the deterministic algorithm of Hopcroft and Karp or in $O(n^\omega)$-time using the randomized algorithm of Mucha and Sankowski.*

3.3 Extensions and Poset Dimension

Chains and linear orders correspond to our intuition of "sorted lists" of elements. We can represent the linear order $L = \Big(\{u_1, \ldots, u_n\}, \{(u_i, u_j) \colon 1 \leq i \leq j \leq n\}\Big)$, compactly as $L = (u_1, \ldots, u_n)$. Using this notation, we say that u_i is the element in the i-th position on the linear order L.

Any poset can be sorted into a linear order maintaining all preexistent relations. We formalize this observation as follows. Given two partial orders $P = (V, \leq_P)$ and $Q = (V, \leq_Q)$ on the same set V, we say that Q is an **extension** of P if for any two elements u and v in V, $u \leq_P v$ implies that $u \leq_Q v$. A **linear extension** is an extension that is also a total order.

Theorem 3.3.1 (Szpilrajn [161]). *The collection $\mathcal{L}$ of linear extensions of a poset P is nonempty and $P = \bigcap_{L \in \mathcal{L}} L$.*

Theorem 3.3.1 implies that every poset is fully characterized by its family of linear extensions. An interesting parameter of a poset is the minimum number of linear extensions needed to characterize it.

A family $\{L_1, \ldots, L_d\}$ of linear orders in V is called a **realizer** of P if their intersection is P. The **dimension** of a poset P is the smallest cardinality of a realizer for P. A d-dimensional poset is a poset of dimension at most d.

Even though the above is the standard way to define it, the usual idea of dimension is something based on geometry, not combinatorics. The following alternative definition is implicit in a book by Ore [132]. The **product dimension** of a poset P is the smallest number of total orders $C_1, \ldots, C_d$, with $C_i = (V_i, \leq_i)$ such that P can be embedded in $(\mathbb{R}^d, \leq_{\mathbb{R}^d})$.

Theorem 3.3.2 (Ore [132]). *The dimension and the product dimension of a poset are the same.*

Proof. For completeness, we include a proof of this theorem. Let $\{L_1, \ldots, L_d\}$ be a realizer of a d-dimensional partial order $P = (V, \leq_P)$. Consider the map $\varphi \colon V \to \mathbb{R}^d$ where the i-th coordinate of $\varphi(u)$ corresponds to the position of u in L_i. By definition of a realizer, we have $u \leq_P v$ if and only if $\varphi(u) \leq_{\mathbb{R}^d} \varphi(v)$. This embedding shows that the product dimension of P is at most its dimension.

For the other direction, let $\varphi \colon V \to \mathbb{R}^d$ be an embedding of a poset P. Consider the linear order $L_i = (V, \leq_i)$ obtained by setting $u <_i v$ if $\varphi(u)_i < \varphi(v)_i$ or if $\varphi(u)_i = \varphi(v)_i$ and $\varphi(u)_j < \varphi(v)_j$, where j is the first index in $[d]$ such that $\varphi(u)_i \neq \varphi(v)_j$. It is easy to check that L_i is a total order on V.

We prove now that $\{L_1, \ldots, L_d\}$ is a realizer for P. Consider two points $u \neq v$ such that $u <_i v$ for all $i \in [d]$, then we have $\varphi(u) \leq_{\mathbb{R}^d} \varphi(v)$ for otherwise there would be a coordinate j such that $\varphi(u)_j > \varphi(v)_j$, contradicting that $u <_j v$. Therefore, by the definition of the embedding, $u \leq_P v$.

Conversely, consider two points $u \neq v$ such that $\varphi(u) \leq_{\mathbb{R}^d} \varphi(v)$ and suppose for contradiction that for some i, $u >_i v$. By definition, this means that either $\varphi(u)_i > \varphi(v)_i$ or for some $j \neq i$, the strict inequality $\varphi(u)_j > \varphi(v)_j$ holds. Both cases contradict the hypothesis. □

One-dimensional posets correspond simply to linear orders. Two-dimensional posets are well understood and have many different characterizations. The following one is worth mentioning.

Theorem 3.3.3 (Dushnik and Miller [47]). *A poset P has dimension at most two if and only if it admits a **nonseparating linear extension**. This is, a linear extension $L = (v_1, \ldots, v_n)$ of P such that if $i < j < k$ and $v_i \leq_P v_k$ then v_j is comparable in P to at least one of v_i and v_k.*

Two-dimensional posets can be recognized in polynomial time (see Theorem 3.4.6). However, no simple characterization of d-dimensional posets is known for $d \geq 3$.

Yannakakis [168] has shown that deciding whether a poset has dimension d is NP-complete, for $d \geq 3$. Hegde and Jain [83] have shown that the problem of computing the dimension of a poset is hard to approximate within a factor $n^{0.5-\varepsilon}$ unless NP = ZPP.

The following theorem, often attributed to Galai [68] and Hiragushi [84], states that the dimension of a poset only depends on its comparability graph.

Theorem 3.3.4 (Gallai [68]; Hiragushi [84])**.** *Posets having the same comparability graph have the same dimension.*

Properties of posets depending only on their comparability graphs are called **comparability invariants**. In Chapter 4 we study another comparability invariant: the jump number.

3.4 Survey on Comparability Graph Classes

To efficiently solve algorithmic problems on graphs, it is important to know the structural properties of the objects in consideration. Certain graph problems, such as computing a clique of maximum size, are NP-hard for general graphs, but they become polynomial time solvable when restricted to special classes, such as comparability graphs. For two graphs classes $\mathcal{G}_1 \subset \mathcal{G}_2$, if a problem is polynomially tractable for the larger class it will remain easy for the smaller one, and if the problem is NP-hard in the smaller class it will remain hard in the larger one. Assuming P $\neq$ NP, it is interesting to refine the *threshold* classes between P and NP for certain graph problems. Even for classes that are "easy" for a given problem, it is interesting to find subclasses where faster algorithms can be provided.

Special graph classes are an interesting topic. For additional results and references we refer the reader to the excellent survey of Brandstädt, Le and Spinrad [17].

In this section, we survey a particular collection of comparability graphs and their associated posets which are of relevance for the problems we study in the next chapter. For some of these classes we present different characterizations and give new proofs of equivalence between some of them. We start by defining some basic concepts.

Given a collection of objects X for which intersection makes sense, the **intersection graph** $\mathcal{I}(X)$ is the graph having X as vertex set, and where two objects are adjacent if their intersection is nonempty. Similarly, if inclusion makes sense, the **containment graph** of X is the graph having X as vertex set and where two objects are adjacent if one is included in the other.

Given a bipartite graph $G = (A \cup B, E)$, an ordering $(a_1, \ldots, a_s)$ of A, and an ordering $(b_1, \ldots, b_t)$ of B where $\{A, B\}$ is a fixed bipartition of the vertices of G, the associated **biadjacency matrix** is the $s \times t$ matrix M having $M_{i,j} = 1$ if $a_i b_j$ is an edge and $M_{i,j} = 0$ otherwise. For a matrix M and a collection of matrices $\mathcal{N}$, we say that M is $\mathcal{N}$-free if M does not contain any matrix of $\mathcal{N}$ as an (ordered) submatrix. If $\mathcal{N}$ consists of only one matrix N, we say that a matrix is N-free if it is $\mathcal{N}$-free.

3.4.1 Geometric Representation of Posets in the Plane

A **two-dimensional comparability graph** (or **2D-graph** for short) is the comparability graph of a two-dimensional poset. Theorem 3.3.2 states that any such poset P can be obtained from a subset of points V in $\mathbb{R}^2$, where for two points u and v in V, $u \leq_P v$ if and only if $u_x \leq v_x$ and $u_y \leq v_y$. For a given 2D-graph G, we denote any collection of points $V \subseteq \mathbb{R}^2$ realizing it as above a **2D-representation** of G.

In what follows we introduce bipartite versions of the previous concepts. The following definitions are not standard, but they are deeply related to the class of two directional orthogonal ray graphs we later introduce (see Section 3.4.4).

Given two multisets[1] A and B in $\mathbb{R}^2$, define the bipartite poset $P(A, B)$ on the disjoint union[2] $A \sqcup B$, where $a \leq_P b$ if and only if $a \in A$, $b \in B$, $a_x \leq b_x$ and $a_y \leq b_y$. The comparability graph of any such poset is denoted as a **bicolored 2D-graph**. Any pair of multisets $A, B \subseteq \mathbb{R}^2$ realizing a bicolored 2D-graph G as above is called a **bicolored 2D-representation** of G.

Let G be a (bicolored) 2D-graph with n vertices. A (bicolored) 2D-representation of G having every point in the grid $[n]^2 = \{1, \ldots, n\} \times \{1, \ldots, n\}$, and where no two points are in the same horizontal or vertical line is called a (bicolored) **rook representation** of G. In particular, every rook representation of a 2D-graph can be obtained from a collection of points $\{(i, \pi(i)) \colon i \in [n]\}$ for some permutation π of $[n]$. We say in this case that the rook representation is induced by π. We remark that if (A, B) is a bicolored rook representation of a graph then A and B are disjoint sets (we do not consider them as multisets anymore).

Lemma 3.4.1. *Any (bicolored) 2D-graph admits a (bicolored) rook representation.*

Proof. By Theorem 3.3.2, any 2D-representation $V \subseteq \mathbb{R}^2$ defines a poset admitting a realizer $\{L_1, L_2\}$ of size 2. We recover a rook representation by mapping each element v of this poset to the point having x-coordinate equal to the position of v in L_1 and y-coordinate equal to the position of v in L_2.

Consider now the bicolored case. The main difficulty is that A and B are not necessarily disjoint multisets: two points of $A \sqcup B$ can share the same position in the plane. For a bicolored 2D-representation (A, B), consider the map $\varphi \colon A \sqcup B \to \mathbb{R}^3$ obtained by appending to each point in A (resp. in B) a third z-coordinate equal to some negative number (resp. positive number), in such a way that no two elements of $A \sqcup B$ have the same z-coordinate. The poset Q defined by $u \leq_Q v$ if $\varphi(u) \leq_{\mathbb{R}^3} \varphi(v)$ is an extension of $P(A, B)$ satisfying $a \leq_P b$ if and only if $a \leq_Q b$ for all $(a, b) \in A \times B$.

Using the same construction as in the proof of Theorem 3.3.2, we obtain a realizer $\{L_1, L_2, L_3\}$, $L_i = (A \sqcup B, \leq_i)$ for Q. Consider the set A' obtained by moving each element $a \in A$ to the point a' in $[n]^2$ having x-coordinate equal to the position of v in L_1 and y-coordinate equal to the position of v in L_2. Define B' from B analogously. We claim that (A', B') is a bicolored rook representation of $P(A, B)$.

[1] For our purposes a multiset of points in the plane is a collection of objects where even though two of them may share the same position in the plane, they are considered as different elements.

[2] By $A \sqcup B$ we mean the multiset containing a copy of A and a copy of B where elements repeated in both are considered as different.

It is clear that in $A' \cup B'$ no two points are in the same horizontal or vertical line. We only need to check that $a \leq_{\mathbb{R}^2} b$ if and only if $a' \leq_{\mathbb{R}^2} b'$, for $a \in A$, and $b \in B$. One direction is easy: If $a \leq_{\mathbb{R}^2} b$ then $\varphi(a) \leq_{\mathbb{R}^3} \varphi(b)$ and so, $a <_i b$ for $i = 1, 2, 3$. This means that $a' \leq_{\mathbb{R}^2} b'$. For the converse, suppose that $a' \leq_{\mathbb{R}^2} b'$. By construction, a' is located strictly to the left and strictly below b'. This is, $a <_1 b$ and $a <_2 b$. Since we are using the same construction as in the proof of Theorem 3.3.2, the fact that $\varphi(a)_3 < 0 < \varphi(b)_3$ implies that $a <_3 b$ also holds. From here, we get $\varphi(a) \leq_{\mathbb{R}^3} \varphi(b)$, implying that $a \leq_{\mathbb{R}^2} b$. □

Using the previous lemma we can show the following.

Lemma 3.4.2. *Bicolored 2D-graphs are comparability graphs of dimension at most 3.*

Proof. Let (A, B) be a bicolored rook representation in $[n]^2$ of a bicolored 2D-graph G. Define $\varphi\colon A \cup B \to \mathbb{R}^3$ as

$$\varphi(v) = \begin{cases} (v_x, v_y, -v_x) & \text{if } v = (v_x, v_y) \in A; \\ (v_x, v_y, n + 1 - v_x) & \text{if } v = (v_x, v_y) \in B. \end{cases}$$

Note that if $v = (v_x, v_y)$ and $w = (w_x, w_y)$ are two different elements in $A \cup B$ then, by assumption, $v_x \neq w_x$. In particular, if both v and w are in the same color class (A or B), then the first and third coordinates of $(\varphi(v) - \varphi(w))$ have different sign. This means that $\varphi(v)$ and $\varphi(w)$ are incomparable in $(\mathbb{R}^3, \leq_{\mathbb{R}^3})$. On the other hand, if $a \in A$ and $b \in B$ then the third coordinate of $\varphi(a)$ is always negative, while the third coordinate of $\varphi(b)$ is always positive. Therefore, $a \leq_{\mathbb{R}^2} b$ if and only if $\varphi(a) \leq_{\mathbb{R}^3} \varphi(b)$.

We have proved that φ is an embedding in $\mathbb{R}^3$ of a poset having comparability graph G. This concludes the proof. □

Now we are ready to describe the classes of graphs and posets that we focus on this thesis. We start by defining permutation graphs and then we proceed to define a chain of bipartite graph classes, each one contained in the previous one.

3.4.2 Permutation Graphs

Permutation graphs have many equivalent characterization. Their standard definition is the following: A graph $G = (V, E)$ is a **permutation graph** if there is a labeling for $V = \{v_1, \ldots, v_n\}$ and a permutation σ on $[n]$ such that $v_i v_j$ is an edge of G if and only if $(i - j)(\sigma^{-1}(i) - \sigma^{-1}(j)) < 0$.

Two equivalent geometric definitions are the following.

Lemma 3.4.3 (Even et al. [52] and Baker er al. [9])**.** *The following are equivalent:*

(i) *G is a permutation graph.*

(ii) *G is the intersection graph of a family $\{S_1, \ldots, S_n\}$ of line segments connecting two parallel lines in the plane.*

(iii) *G is a 2D-graph.*

Proof. We give the proofs for completeness.
(i) $\Leftrightarrow$ (ii): Consider two parallel (say horizontal) lines in the plane. On the top line select n points and label them from left to right as $1, 2, \ldots, n$. On the bottom line select n points and label them from left to right as $\sigma(1), \ldots, \sigma(n)$, where σ is the permutation defining G. Let S_i be the line segment connecting the points labeled i in both the top and bottom line. Then S_i intersects S_j if and only if $i < j$ and $\sigma^{-1}(i) > \sigma^{-1}(j)$, or if $i > j$ and $\sigma^{-1}(i) < \sigma^{-1}(j)$. Therefore, G is the intersection graph of the segments $\{S_1, \ldots, S_n\}$. Since this construction is reversible we obtain the equivalence.
(i) $\Leftrightarrow$ (iii): Consider a permutation graph given by some permutation σ. Let π be the related permutation $\pi(i) = n + 1 - \sigma^{-1}(i)$ and define the mapping $\varphi\colon V \to \mathbb{R}^2$ given by $\varphi(v_i) = (i, \pi(i))$. From here, two elements v_i and v_j are adjacent in G if and only if $\varphi(v_i)$ and $\varphi(v_i)$ are comparable by $\leq_{\mathbb{R}^2}$ in the plane. Therefore, G is the $2D$-graph having rook representation induced by π. Since this construction is reversible we obtain the equivalence. □

Using Lemma 3.4.3 we can give an alternative geometric proof for yet another characterization of permutation graphs.

Lemma 3.4.4 (Baker et al. [9]; Dushkin and Miller [47])**.** *A graph G is a permutation graph if and only if it is the containment graph of a collection of (distinct) closed intervals.*

Proof. Consider a graph G having rook representation induced by a permutation π on $[n]$. Let v_i be the vertex represented by $(i, \pi(i))$. Associate v_i to the interval $I_i = [-\pi(i), i]$. Noting that $I_i \subseteq I_j$ if and only if $(i, \pi(i)) \leq_{\mathbb{R}^2} (j, \pi(j))$ we conclude that G is the containment graph of the intervals $\{I_i : i \in [n]\}$. Conversely, given the containment graph G of a family of intervals, we map each interval $I_i = [a_i, b_i]$ to the point $p_i = (-b_i, a_i)$ in the plane. As before, $I_i \subseteq I_j$ if and only if $p_i \leq_{\mathbb{R}^2} p_j$. Hence, the points p_i are a $2D$-representation of the graph G and therefore, G is a $2D$-graph. Intuitively, the previous assignment gives a bijection between intervals and points weakly above the diagonal line $y = -x$ in such a way that their horizontal and vertical projections onto this line define the corresponding intervals. We illustrate this construction in Figure 3-1. □

We conclude by mentioning a couple of properties of these graphs.

Theorem 3.4.5 (Dushnik and Miller [47])**.** *A graph G is a permutation graph if and only if both G and its complement $\overline{G}$ are comparability graphs.*

Theorem 3.4.6 (McConell and Spinrad [115])**.** *Permutation graphs can be recognized in time $O(n + m)$ where n and m are the number of vertices and edges of the graph.*

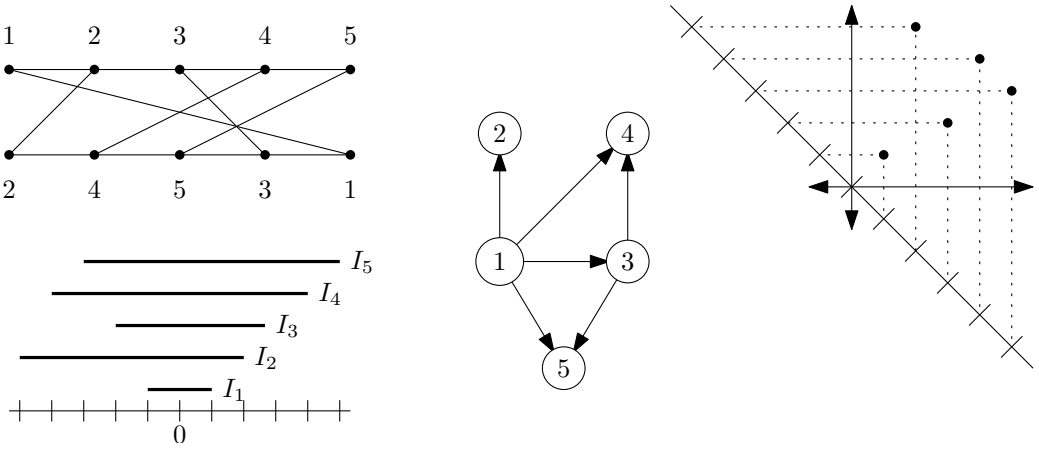


Figure 3-1: Different representations of the permutation graph given by $\sigma = (2, 4, 5, 3, 1)$ or equivalently, by the rook representation induced by $\pi = (1, 5, 2, 4, 3)$.

3.4.3 Chordal Bipartite Graphs

A **chord** of a cycle is an edge connecting two vertices that are not consecutive in the cycle. A graph is **chordal** (or triangulated) if every cycle of length at least 4 has at least one chord. We are interested in the following related class.

A **chordal bipartite graph** is a bipartite graph where each cycle of length at least 6 has a chord. We must be careful as this definition can be misleading: chordal bipartite graphs are not chordal in general, since the cycle on four vertices C_4 is chordal bipartite. In other words, chordal bipartite graphs are not the intersection of bipartite graphs and chordal graphs.

Chordal bipartite graphs have many important properties. However, we only mention some theorems that are relevant for our purposes.

Theorem 3.4.7 (Hoffman, Kolen and Sakarovitch [85])**.** *A graph G is chordal bipartite if and only if it admits a Γ-free biadjacency matrix, where* $\Gamma = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$.

Theorem 3.4.8 (e.g. Lubiw [108])**.** *Chordal bipartite graphs can be recognized in polynomial time.*

3.4.4 Two Directional Orthogonal Ray Graphs (2DORGs)

An **orthogonal ray graph** is the intersection graph of rays (closed half-lines) in the plane that are parallel to either the x or the y axis, provided we only consider intersections between horizontal and vertical rays (i.e. if two horizontal rays intersect, there is no associated edge in the graph).

A **two directional orthogonal ray graph** (or 2DORG) is an orthogonal ray graph where all the horizontal rays go in the same direction (without loss of generality, they go to the right) and all the vertical rays go in the same direction (without loss of generality, they go down). Formally, a 2DORG is a bipartite graph on $A \cup B$ where

each vertex v is associated to a point $(v_x, v_y) \in \mathbb{R}^2$, so that $a \in A$ and $b \in B$ are neighbors if and only the rays $[a_x, \infty) \times \{a_y\}$ and $\{b_x\} \times (-\infty, b_y]$ intersect each other.

Lemma 3.4.9. *Two directional orthogonal ray graphs are exactly the bicolored 2D-graphs.*

Proof. This follows directly from the fact that $[a_x, \infty) \times \{a_y\}$ intersects $\{b_x\} \times (-\infty, b_y]$ if and only if $a \leq_{\mathbb{R}^2} b$. □

This class of graphs can also be characterized by their biadjacency matrices.

Lemma 3.4.10 (Shrestha, Tayu and Ueno [151])**.** *The following are equivalent:*

(i) *G is a two directional orthogonal ray graph.*

(ii) *G admits a biadjacency matrix that is γ-free, where* $\gamma = \left\{ \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \right\}$.

(iii) *G admits a biadjacency matrix such that if M_{ij} is a 0-entry, then at least one of the following holds: every entry above it is a 0-entry or every entry to its left is a 0-entry.*

Proof. It is easy to check the equivalence between the second and third conditions. The proof of equivalence between the first and second ones we present is similar to the one given in [151]. Let G be a 2DORG with rook representation (A, B). Sort A from top to bottom as $a_1, \ldots, a_s$ and B from left to right as $b_1, \ldots, b_t$. We claim this ordering induces a γ-free biadjacency matrix M. Indeed, suppose that $i \leq i'$ and $j \leq j'$ are such that the submatrix $M_{\{i,i'\},\{j,j'\}}$ is in γ. As $M_{i',j} = M_{i,j'} = 1$, we have $(a_{i'})_x \leq (b_j)_x \leq (b_{j'})_x$ and $(a_{i'})_y \leq (a_i)_y \leq (b_{j'})_y$, contradicting the fact that $M_{i',j'} = 0$. Figure 3-2 illustrates this construction.

Conversely, let M be a γ-free $s \times t$ biadjacency matrix of G. For every $i \in [s]$, let $L(i)$ be the leftmost index j in the matrix such that $M_{ij} = 1$ (set $L(i) = 0$ if no such entry exists). Similarly, for every $j \in [t]$, let $T(j)$ be the topmost index i in the matrix such that $M_{ij} = 1$ (set $T(j) = 0$ if no such entry exists). Map each vertex $a_i \in A$ to the position $(L(i), s - i + 1)$ and vertex $b_j \in B$ to position $(j, s - T(j) + 1)$. The corresponding pair (A', B') of obtained multisets in $\mathbb{R}^2$ is a 2D-representation of G. Indeed, since M is γ-free, $M_{ij} = 1$ if and only if $L(i) \leq j$ and $T(j) \leq i$. This is equivalent to $(L(i), s - i + 1) \leq_{\mathbb{R}^2} (j, s - T(j) + 1)$. □

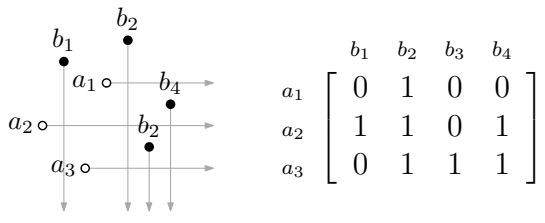


Figure 3-2: A 2DORG and its γ-free biadjacency matrix.

Using the previous lemma we can show the following.

Lemma 3.4.11. *The class of two directional orthogonal ray graphs is strictly contained in the class of chordal bipartite graphs.*

Proof. The inclusion follows since γ-free matrices are also Γ-free. To see that the inclusion is strict, consider the 3-claw G in Figure 3-3.

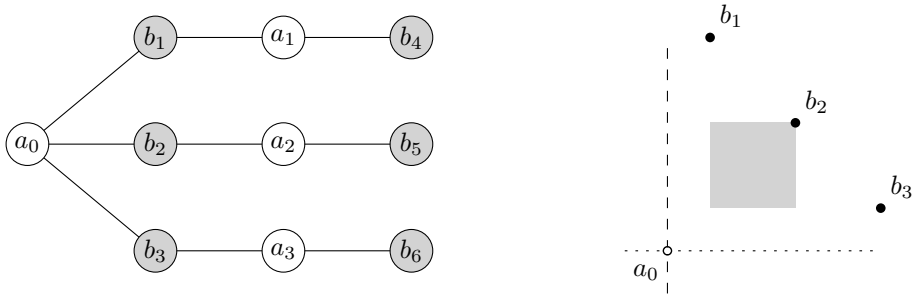


Figure 3-3: Chordal bipartite graph that is not a 2DORG and a sketch for the impossibility of a bicolored rook representation.

This graph is obtained by replacing each edge in a star with three edges ($K_{1,3}$) by a path of length 3. Since G has no cycles, it is chordal bipartite. Shrestha et al. [150] have shown that G is not a 2DORG by using more involved characterizations of these graphs. Here we give a direct proof. Let a_0 be the root, $\{b_1, b_2, b_3\}$, $\{a_1, a_2, a_3\}$ and $\{b_4, b_5, b_6\}$ be the vertices at distance 1, 2 and 3 from the root and assume that G admits a bicolored rook representation (A, B). In this representation, a_0 divides the plane in four quadrants. The points b_1, b_2, b_3 must be in the quadrant above and to the right of a_0. Furthermore, they must form an antichain for $\leq_{\mathbb{R}^2}$ since, if for two of them we have $b_i \leq_{\mathbb{R}^2} b_j$, the fact that $a_i \leq_{\mathbb{R}^2} b_i$ would contradict that $a_i b_j$ is not an edge of G.

Without loss of generality, assume that b_1, b_2, b_3 are the elements of this antichain from left to right as in Figure 3-3. Since a_2 is connected to b_2, but not to b_1 or b_3, the point a_2 must be in the gray rectangular area shown in the figure, located between the vertical line passing through b_1, the horizontal line passing through b_3 and the vertex b_2. But then, since a_2 is above and to the right of a_0, so must be b_5 contradicting the fact that a_0 and b_5 are incomparable. □

Finally, we mention the following result.

Theorem 3.4.12 (Shrestha, Tayu and Ueno [151])**.** *Two directional orthogonal ray graphs can be recognized in polynomial time.*

3.4.5 Interval Bigraphs

An **interval graph** is the intersection graph of a collection of intervals $\{I_1, \ldots, I_n\}$ of the real line. An interesting bipartite variant of these graphs is the following.

An **interval bigraph** is a bipartite graph $G = (A \cup B, E)$, where each vertex $v \in A \cup B$ is associated to a real interval I_v so that $a \in A$ and $b \in B$ are adjacent if and only if $I_a \cap I_b \neq \emptyset$.

Since we are only considering graphs with finite cardinality, it is very simple to prove that every interval graph (and every interval bigraph) can be represented using only closed intervals having extreme points in the set $\{1, 2, \ldots, 2n\}$, where n is the number of vertices of the graph.

Lemma 3.4.13. *The class of interval bigraphs is strictly contained in the class two directional orthogonal ray graphs.*

Proof. We give a simple proof of the inclusion using the bicolored 2D-representation of two directional orthogonal ray graphs. Let G be an interval bigraph with parts A and B. For $a \in A$ with interval $I_a = [s, t]$ and $b \in B$ with interval $I_b = [s', t']$, we identify a with the point $(s, -t) \in \mathbb{Z}^2$ and b with the point $(t', -s') \in \mathbb{Z}^2$. By definition, ab is an edge of G if and only if $[s, t] \cap [s', t'] \neq \emptyset$, or equivalently if $(s, -t) \leq_{\mathbb{Z}^2} (t', -s')$. Intuitively, the previous identification maps A (resp. B) to points weakly below (resp. weakly above) the diagonal line $y = -x$ in such a way that their horizontal and vertical projections onto this line define the corresponding intervals. We illustrate this construction in Figure 3-4. We call this the **natural bicolored 2*D*-representation** of an interval bigraph.

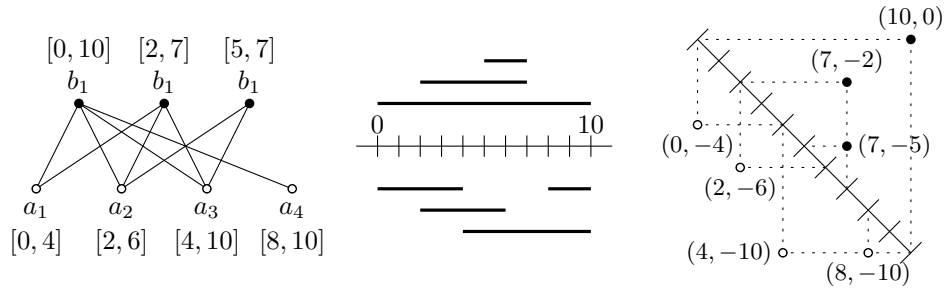


Figure 3-4: An interval bigraph, its interval representation and a geometric representation as 2DORG.

We have shown that every interval bigraph is a 2DORG. To see that the inclusion is strict, consider the graph G in Figure 3-5.

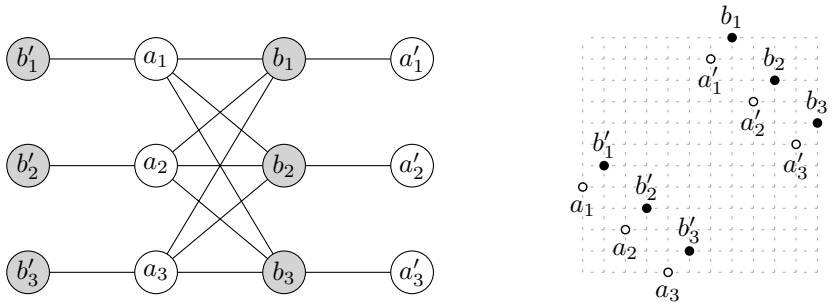


Figure 3-5: A two directional orthogonal ray graph that is not an interval bigraph and one bicolored rook representation.

Müller [123] calls G an "insect" and gives a proof that this graph is not an interval bigraph. For completeness, we include a slightly different proof here.

Suppose that G admits an interval representation as an interval bigraph. Consider its natural bicolored 2D-representation (A, B). The set $B_0 = \{b_1, b_2, b_3\}$ must be an antichain for $\leq_{\mathbb{R}^2}$ since if for two of them, $b_i \leq_{\mathbb{R}^2} b_j$ then the fact that $a'_i \leq_{\mathbb{R}^2} b_i$ would contradict that $a'_i b_j$ is not an edge. By symmetry, the set $A_0 = \{a_1, a_2, a_3\}$ is also an antichain. By possibly relabeling the vertices, we may assume that b_2 and a_2 are the middle points (from left to right) of their respective antichains. Let p be the bottommost and leftmost point in the plane that is larger than all elements of A_0 in the $\leq_{\mathbb{R}^2}$ order and q be the topmost and rightmost point in the plane that is smaller than all elements of B_0 in the $\leq_{\mathbb{R}^2}$ order. Since every $a_i b_j$ is an edge, we must have $p \leq_{\mathbb{R}^2} q$. On the other hand, since a'_2 is only comparable to the middle point of B_0, $q \leq_{\mathbb{R}^2} a'_2$. Symmetrically, $b'_2 \leq_{\mathbb{R}^2} p$. This implies that $b'_2 \leq_{\mathbb{R}^2} a'_2$. Since the points in A are weakly below the diagonal and the points in B are weakly above it, we must have $a'_2 = b'_2$, which is a contradiction since $a'_2 b'_2$ is not an edge. □

We conclude, by mentioning the following results (See Figure 3-6)

Theorem 3.4.14 (Das et al. [41])**.** *A graph G is an interval bigraph if it admits a biadjacency matrix M with the following* ***zero-partition property****: Every* 0*-entry in M can be labeled as R or C in such a way that every entry above a C is also a C and every entry to the left of an R is also an R.*

	b_1	b_2	b_3	b_4	b_5	b_6
a_1	1	1	1	0^C	1	0^C
a_2	0^R	1	1	1	1	0^C
a_3	1	1	1	1	1	0^C
a_4	0^R	0^R	1	1	1	0^C
a_5	0^R	0^R	1	1	1	1

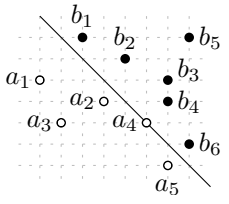


Figure 3-6: A biadjacency matrix with the zero partition property and a bicolored 2D-representation of the same interval graph.

Theorem 3.4.15 (Müller [123])**.** *Interval bigraphs can be recognized in polynomial time.*

3.4.6 Convex Graphs

A bipartite graph with bipartition $\{A, B\}$ is **convex on** A if there is a labeling for $A = \{a_1, \ldots, a_s\}$ so that the neighborhood of each $b \in B$ is a set of elements of A consecutive in this labeling. A graph is **convex** (also known as convex bipartite) if it admits a partition $\{A, B\}$ of its vertices such that G is bipartite with parts A and B and convex on one of its parts.

Another, perhaps more usual, characterization of convex graphs is the following. Given the graph of a finite path P, define A to be the set of edges of the path and B to be a collection of subpaths of P. It is easy to verify that the intersection graph of $A \cup B$ is a convex graph and that every convex graph can be obtained in this way.

Lemma 3.4.16. *The class of convex graphs is strictly contained in the class of interval bigraphs.*

Proof. Consider a convex graph $G = (A \cup B, E)$ such that G is convex on A under the labeling $A = \{a_1, \ldots, a_s\}$. Map each element $a_i \in A$ to the (singleton) interval $\{i\} \subseteq \mathbb{R}$, and each element $b \in B$ having neighborhood $\{a_{\ell(b)}, \ldots, a_{r(b)}\}$ to the interval $[\ell(b), r(b)]$. Then, G is the interval bigraph associated to the intervals defined in this way. This proves that every convex graph is an interval bigraph.

To prove that the inclusion is strict, consider the interval bigraph induced by two equal families of intervals $A = B = \{\{1\}; \{2\}; \{3\}; [1,2]; [2,3]; [1,3]\}$. This graph is depicted in Figure 3-7.

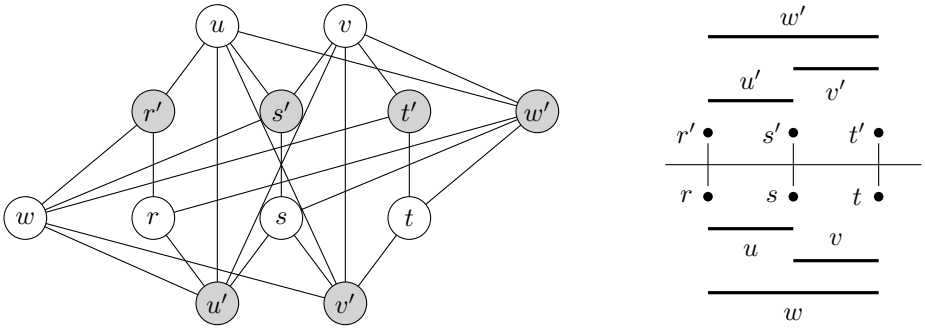


Figure 3-7: Interval bigraph that is not a convex graph.

We claim that this graph is not convex on A. By symmetry, this implies that G is not convex on B either. Thus, G is not a convex graph because $\{A, B\}$ is the only partition of the graph that makes G bipartite. To prove the claim, denote $r = \{1\}$, $s = \{2\}$, $t = \{3\}$, $u = [1,2]$, $v = [2,3]$ and $w = [1,3]$ to the vertices of A and r', s', t', u', v', w' the corresponding vertices of B. Assume, for sake of contradiction, that there is a labeling for A that witness convexity.

The vertex v' is adjacent to every vertex of A except for r, similarly the vertex u' is adjacent to every vertex except for t. This implies that the first and last vertices of the labeling must be r and t. Assume without loss of generality then that $a_1 = r$ and $a_6 = t$. Since the neighborhood of r' is $\{r, u, w\}$, these three vertices should appear consecutively in the labeling. Thus, w has label a_2 or a_3. On the other hand, the neighborhood of t' is $\{t, v, w\}$, implying that w must have label a_4 or a_5, which is a contradiction. □

By using the natural bicolored $2D$-representation for interval bigraphs, we can represent geometrically a convex graph $G = (A \cup B, E)$ in the triangular region $\{(i, -j) \in \mathbb{Z}^2 \colon 1 \leq j \leq i \leq |A|\}$ with the points of A lying on the line $y = -x$.

It is very simple to characterize convex graphs by their biadjacency matrices. A 0-1 matrix is **vertically** (resp. **horizontally**) convex if the ones in every column (resp. in every row) are consecutive.

Theorem 3.4.17. *A bipartite graph $G = (A \cup B, E)$ is convex on A if it admits a biadjacency matrix that is vertically convex.*

Proof. Direct from the definitions. □

We also have the following result.

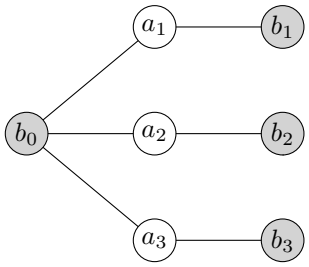
Theorem 3.4.18 (Booth and Lueker [14])**.** *Convex bigraphs can be recognized in polynomial time.*

3.4.7 Biconvex Graphs

A **biconvex** graph is a bipartite graph with bipartition $\{A, B\}$ that is convex on both A and B. We say that the graph is biconvex on A and B if this happens.

Lemma 3.4.19. *The class of biconvex graphs is strictly contained in the class of convex graphs.*

Proof. The inclusion holds trivially by definition. To prove that the inclusion is strict, we show that a 2-claw, that is, the tree obtained by replacing each edge in a star with three edges $K_{1,3}$ by a path of length 2 is convex but not biconvex. This graph is depicted in Figure 3-8.



$$M_G = \begin{array}{c} \\ a_1 \\ a_2 \\ a_3 \end{array} \begin{array}{c} \begin{array}{cccc} b_0 & b_1 & b_2 & b_3 \end{array} \\ \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} \end{array}$$

Figure 3-8: Convex graph that is not biconvex, and one of its convex biadjacency matrices.

Let b_0 be the root, $\{a_1, a_2, a_3\}$ and $\{b_1, b_2, b_3\}$ be the vertices at distance 1 and 2 from the root. The only bipartition of this graph is $A = \{a_1, a_2, a_3\}$ and $B = \{b_0, b_1, b_2, b_3\}$. Since the neighborhood of every vertex of B consists of either one vertex or all the vertices in A, any labeling of A witness convexity on this set. However, there is no way to order B so that the neighbors of all the vertices of A are consecutive: we would require b_0 to be consecutive to all b_1, b_2 and b_3 which is impossible. □

We can characterize biconvex graphs by their biadjacency matrices. A matrix is **biconvex** if it is both horizontally and vertically convex.

Theorem 3.4.20. *A bipartite graph G is biconvex if it admits a biconvex biadjacency matrix.*

Finally, we remark that we can recognize biconvex graphs in polynomial time using the algorithms for convex graph recognition.

3.4.8 Bipartite Permutation Graphs

A graph G is a bipartite permutation graph if it is simultaneously bipartite and a permutation graph. In particular, G admits a bicolored rook representation (A, B) where both A and B are antichains for $\leq_{\mathbb{R}^2}$.

We can characterize bipartite permutation graphs by their biadjacency matrices.

Theorem 3.4.21 (Spinrad, Brandstädt and Steward [154]; Chen and Yesha [30])**.** *A bipartite graph G without isolated vertices is a bipartite permutation graph if it admits a biadjacency matrix M in **2-staircase normal form**. This is, M satisfies the following properties.*

1. *The matrix M is biconvex.*
2. *Let $[\ell(i), r(i)]$ be the (horizontal) interval of indices j for which $M_{ij} = 1$. Then $\ell(1) \leq \ell(2) \leq \cdots \leq \ell(|A|)$ and $r(1) \leq r(2) \leq \cdots \leq r(|A|)$.*

Equivalently, M is β-free, where $\beta = \left\{ \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \right\}$.

Proof. Consider a rook representation of a bipartite permutation graphs and let $A = \{a_1, \ldots, a_s\}$ and $B = \{b_1, \ldots, b_t\}$ be the orderings obtained on A and B by sorting them in increasing order of x-coordinates. Since A and B are antichains with respect to $\leq_{\mathbb{R}^2}$, in the above orderings the elements are also sorted in decreasing order of their y-coordinates. We claim these orderings induce a biadjacency matrix M in 2-staircase normal form.

To prove the biconvexity of M, consider any element $a_i \in A$ with degree at least two, and let b_j and $b_{j'}$ with $j \leq j'$ be two of its neighbors. Suppose that $b_k \in B$ is an element with $j < k < j'$. This implies that $(a_i)_x < (b_j)_x < (b_k)_x < (b_{j'})_x$ and $(b_j)_y > (b_k)_y > (b_{j'})_y > (a_i)_y$, and so $a_i b_k$ must be an edge of the graph. Therefore, the matrix M is horizontally convex. The proof of vertical convexity is analogous.

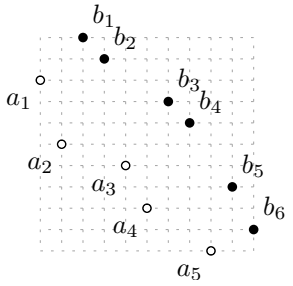
Let a_i and $a_{i'}$ be two elements of A with $i < i'$. We show that $\ell(i) \leq \ell(i')$. Indeed, suppose for contradiction that $j > j'$, where $j = \ell(i)$ and $j' = \ell(i')$, then we have $(a_i)_y < (b_j)_y < (b_{j'})_y$. Since $i < i'$, we also have $(a_i)_x < (a_{i'})_x < (b_{j'})_x$. Therefore $a_i \leq_{\mathbb{Z}^2} b_{j'}$, contradicting the fact that j was the first neighbor of a_i. The proof that $r(i) \leq r(i')$ is analogous. The fact that β-free matrices are exactly the ones in 2-staircase normal form follows easily from the definitions. □

For an example of a 2-staircase normal form matrix, see Figure 3-9.

Lemma 3.4.22. *The class of bipartite permutation graphs is strictly contained in the class of biconvex graphs.*

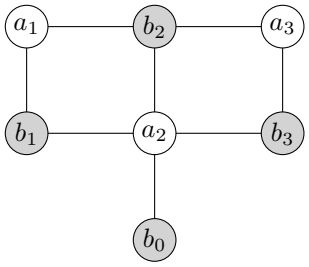
Proof. The inclusion follows from Theorem 3.4.21 and the fact that both bipartite permutation graphs and biconvex graphs are closed under adding isolated vertices.

To see that the inclusion is strict consider the graph G in Figure 3-10. The biconvex matrix M_G witness the biconvexity of G.



$$M_G = \begin{array}{c} \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{array} \begin{array}{c} \begin{array}{cccccc} b_1 & b_2 & b_3 & b_4 & b_5 & b_6 \end{array} \\ \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix} \end{array}$$

Figure 3-9: A rook representation and a 2-staircase normal form matrix associated to a bipartite permutation graph G.



$$M_G = \begin{array}{c} \\ a_1 \\ a_2 \\ a_3 \end{array} \begin{array}{c} \begin{array}{cccc} b_0 & b_1 & b_2 & b_3 \end{array} \\ \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \end{array}$$

Figure 3-10: Biconvex graph that is not a permutation graph and one of its biadjacency matrices.

Suppose for contradiction that G is a bipartite permutation graph. Then it would admit a biadjacency matrix in 2-staircase normal form. But it is easy to see that (up to relabeling of the vertices), the only biconvex biadjacency matrices of G are M_G and the ones obtained by taking the horizontal reflection, vertical reflection or both. None of them is in 2-staircase normal form. □

Finally, we remark that we can recognize bipartite permutation graphs in polynomial time using the same algorithms used for permutation graphs.

3.4.9 Summary

Figure 3-11 depicts all the strict containments among the presented comparability graph classes. In Table 3.1 we summarize some of the properties of their bicolored 2D-representations and admitted biadjacency matrices.

Table 3.1: Geometric and biadjacency properties of comparability graph classes.

Class	Bicolored 2D-representation	Biadjacency Matrix
Chordal Bipartite	None in general.	Γ-free.
2DORGs	General.	γ-free.
Interval bigraphs	A: weakly below $y = -x$, B: weakly above $y = -x$.	Zero-partition property.
Convex graphs	A: on $y = -x$, B: weakly above $y = -x$.	Vertically convex.
Biconvex graphs	–	Biconvex.
Bip. permutation graphs	A and B antichains.	β-free.

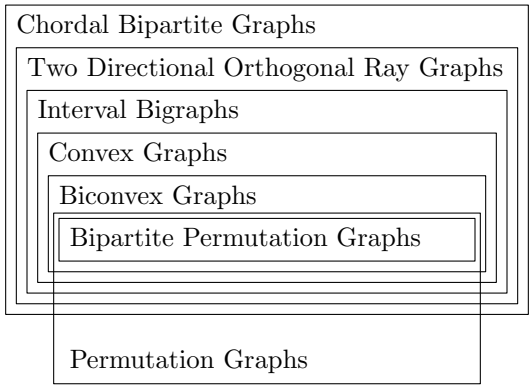


Figure 3-11: Summary of presented comparability graph classes.

3.5 Perfect Graphs

We now recall some important notions about graphs. A **clique** is a subset of the vertices of a graph that are mutually adjacent to one another. A **stable set** is the graph complement of a clique, this is, a subset of vertices that are mutually non-adjacent. A **clique cover** of a graph G is a collection of cliques containing every vertex of G. A **stable set cover** of G is a collection of stable sets containing every vertex of G. A **clique partition** is a clique cover where all the cliques are mutually disjoint. Similarly, a **stable set partition** is a stable set cover where all the stable sets are mutually disjoint. Since cliques are closed under inclusion, any clique cover can be transformed into a clique partition of the same cardinality by possibly dropping some vertices of some cliques in the cover. The same property holds for stable set covers and stable set partitions.

A **proper coloring** of a graph G is an assignment of labels, traditionally called **colors**, to the vertices of G in such a way that no two adjacent vertices share the same color. Note that **color classes**, this is maximal sets of vertices sharing the same color, are stable sets. Therefore, the collection of color classes of a proper coloring defines naturally a stable set partition.

The **clique number** of G, denoted $\omega(G)$, is the size of a largest clique in G. The **stability number** of G, denoted $\alpha(G)$, is the size of a largest stable set in G. The

clique partition number of G, denoted $\kappa(G)$, is the size of a smallest clique cover of G. The **coloring number** of G, denoted $\chi(G)$, is the minimum number of colors needed in a proper coloring of G.

Since cliques in a graph G correspond exactly to stable sets in the graph complement $\overline{G}$, it follows easily that

$$\omega(G) = \alpha(\overline{G}),\ \alpha(G) = \omega(\overline{G}),\ \kappa(G) = \chi(\overline{G}) \ \text{ and } \ \chi(G) = \kappa(\overline{G}). \tag{3.1}$$

Also, since cliques and stable sets intersects in at most one vertex, we easily conclude that

$$\omega(G) \leq \chi(G) \ \text{ and } \ \alpha(G) \leq \kappa(G). \tag{3.2}$$

An important question in graph theory is to identify classes of graphs for which equality holds above. Note that by taking the disjoint union of any graph with a very large clique and a very large stable set we obtain a graph where equality holds for both inequalities in (3.2). This (pathological) example shows that it might be difficult to find conditions to characterize the cases where equality holds. This example also partially explains the study of perfect graphs, which are defined below.

A **(vertex) induced subgraph** of a graph is a subgraph preserving all the adjacency relations of the subset of the vertices in which it is defined. Formally $H = (W, F)$ is an induced subgraph of $G = (V, E)$ if W is a subset of V and for any two vertices $u, v \in W$, $u, v \in F$ if and only if $u, v \in E$.

A graph G is **ω-perfect** if for every nonempty induced subgraph H of G, $\omega(H) = \chi(H)$. A graph G is **α-perfect** if for every nonempty induced subgraph H of G, $\alpha(H) = \kappa(H)$. A graph is **perfect** if it is both α-perfect and ω-perfect.

The **Weak Perfect Graph Theorem** [103] states that a graph is α-perfect if and only if its complement is α-perfect. By using the complementary relations in (3.1), we obtain that the three notions of perfectness (perfect, α-perfect and ω-perfect) are equivalent.

Cycles of odd length are not perfect since their largest clique has size 2 whereas their chromatic number is 3. By the perfect graph theorem, their complements are also imperfect graphs. However, any proper induced subgraph of an odd cycle or of its complement is perfect: they are minimally imperfect. In the nineteen sixties, Berge [13] conjectured that these two graph classes are the only families of minimally imperfect graphs. This conjecture attracted much attention during the next forty years. It was proved in 2002 by Chudnovsky, Robertson, Seymour and Thomas [32] and it is now known as the Strong Perfect Graph Theorem.

We can restate this theorem as follows. A **hole** of a graph is an induced chordless cycle of length at least 4. An **antihole** is the graph complement of a hole. A hole (or an antihole) is **odd** if it contains an odd number of vertices. A graph is a **Berge graph** if it contains no odd hole and no odd antihole.

Theorem 3.5.1 (Strong Perfect Graph Theorem [32])**.** *A graph is perfect if and only if it is a Berge graph.*

Some important classes of perfect graphs are the following.

- Bipartite graphs and their complements.
- Line graphs of bipartite graphs and their complements.
- Comparability graphs and their complements (co-comparability graphs).
- Interval graphs and their complements.
- Chordal graphs (graphs having no hole of length at least 4).
- Weak chordal graphs (graphs having no hole or antihole of length at least 5).

Perfect graphs can also be characterized polyhedrally. Given a graph $G = (V, E)$, the **stable set polytope** $\mathrm{STAB}(G)$ and the **clique-constrained stable set polytope** $\mathrm{QSTAB}(G)$ are defined as

$$\mathrm{STAB}(G) = \text{convex-hull}(\{\chi^S \in \mathbb{R}^V : S \text{ stable set of } G\}), \text{ and} \tag{3.3}$$

$$\mathrm{QSTAB}(G) = \left\{x \in \mathbb{R}^V \colon x \geq 0, \sum_{v \in K} x_v \leq 1, \text{ for all } K \text{ maximal clique of } G\right\}. \tag{3.4}$$

The vectors in $\mathrm{QSTAB}(G)$ are known as **fractional stable sets**. Since for every stable set S, $\chi^S \in \mathrm{QSTAB}(G)$, we conclude that $\mathrm{STAB}(G) \subseteq \mathrm{QSTAB}(G)$. However, the polytope $\mathrm{QSTAB}(G)$ might have non-integral vertices associated to fractional stable sets. Chvátal [33], using results from Lovász [103, 102] characterizes perfect graphs in terms of these two polytopes.

Theorem 3.5.2 (Chvátal [33])**.** $\mathrm{QSTAB}(G)$ *is integral if and only if* G *is a perfect graph and in this case* $\mathrm{QSTAB}(G) = \mathrm{STAB}(G)$*.*

We can find maximum cliques, maximum independent sets, minimum clique partition and minimum colorings of perfect graph in polynomial time (see e.g. [76, Chapter 9]). For special classes faster algorithms are available. For example, for comparability graphs, the four mentioned objects are in correspondence with maximum chains, maximum antichains, minimum chain partitions and minimum antichain partitions respectively; therefore, we can compute them using the algorithms described in Lemmas 3.2.4 and 3.2.3.

Chapter 4

A Primer on the Jump Number Problem

In this chapter, we define and motivate the jump number problem of a comparability graph and survey previous results. We also define two related problems, the maximum cross-free matching and the minimum biclique cover problems. The former problem is known to be equivalent to the jump number problem for chordal bipartite graphs.

Later, we define and review some apparently unrelated problems on 0-1 matrices, on planar geometry and on interval combinatorics. Most of the defined problems come in pairs: a minimization and a maximization problem. For some of these pairs, a min-max relation is known.

We show that all these problems are either equivalent or special cases of the maximum cross-free matching and the minimum biclique cover problems. Some of the connections presented in this chapter are very simple, yet they have not been explicitly noticed in the literature before our joint work with Telha [153] and this thesis.

As simple corollaries of the presented equivalences we obtain new min-max relations for the maximum cross-free matching and the minimum biclique cover on biconvex and convex graphs, and an $O(n^2)$-time algorithm for the jump number problem on convex graphs, considerably improving the previous $O(n^9)$-time algorithm of Dahlhaus [40].

4.1 Jump Number

Let us revisit for a minute both Dilworth's chain partitioning theorem (Theorem 3.2.2) and Szpilrajn's linear extension theorem (Theorem 3.3.1).

Consider a family $\mathcal{C}$ of disjoint chains partitioning P and having minimum cardinality. It is not always the case that we can sort the chains in $\mathcal{C} = \{C_1, \ldots, C_k\}$ in such a way that the obtained sequence induces a linear extension of P. On the other hand, even though every linear extension L of P can be naturally partitioned into chains by "breaking" the sequence induced by L whenever an element and the next one are incomparable in P, this partition is not necessarily of minimum size. As an

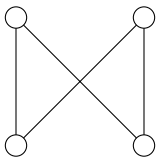


Figure 4-1: Hasse diagram of $K_{2,2}$

example, consider a poset having the complete bipartite graph $K_{2,2}$ as comparability graph (see Figure 4-1 for its Hasse diagram). This poset can be partitioned into 2 chains, but every linear extension induces a partition into 3 chains.

Suppose that we want to find a linear extension of P for which the natural partition described above yields the minimum number of chains possible. This problem, formally defined later in this section, conciliates in a way, both Dilworth's and Szpilrajn's problems.

Let us motivate this problem a little more. We can reinterpret it as a scheduling problem. This interpretation is often attributed to an unpublished manuscript of Pulleyblank [140]. We want to schedule a collection of jobs V on a single machine, respecting the precedences induced by a poset $P = (V, \leq_P)$. These precedence constraints model the fact that certain jobs can only be processed after a collection of jobs have finished. Moreover, every time a job is scheduled immediately after a task not constrained to precede it we incur in a unit "setup" cost. We want to find a schedule that minimizes the total setup cost. Let us define this problem formally using poset terminology.

Let $P = (V, \leq_P)$ be a poset and $L = (u_1, \ldots, u_n)$ be a linear extension of P. The pair (u_i, u_{i+1}), where $1 \leq i \leq n-1$ is a **jump** (or **setup**) of P in L if $u_i \not\leq_P u_{i+1}$. Otherwise, it is said to be a **bump** of P in L.

We use $J(P, L)$ and $B(P, L)$ to denote the number of jumps and bumps of P in L respectively. Obviously, $J(P, L) + B(P, L) = n - 1$ where n is the cardinality of V. The **jump number** $j(P)$ of a poset P is defined as

$$j(P) = \min\{J(P, L) \colon L \text{ linear extension of } P\}. \tag{4.1}$$

Following the notation of Chaty and Chein [27] we also define the **step number**[1] $s(P)$ as

$$s(P) = \max\{B(P, L) \colon L \text{ linear extension of } P\}. \tag{4.2}$$

From the relation above, we also have that for every poset of cardinality n,

$$j(P) + s(P) = n - 1. \tag{4.3}$$

The **jump number problem** is defined as follows.

Problem 1 (Jump Number Problem)**.** Given a finite poset P, find $j(P)$ and a linear

[1]While it would be more natural to use the notation "bump number", traditionally this term is reserved to denote the *minimum* number of bumps in a linear extension of the poset. We use step number to avoid confusion.

extension L of P such that $j(P) = J(P, L)$.

As commented at the beginning of this section, every linear extension L of P induces naturally a partition of V into $1 + J(P, L)$ chains separated by bumps. Moreover, each of these chains is **convex**, that is, they are also intervals: indeed, if an element v is such that $u <_P v <_P w$ for u and w in some chain C of the partition, then v can not appear before C or after C in the linear order, implying that v must be in C. A natural question is then: What are necessary and sufficient conditions for a partition of a poset into convex chains to induce a linear extension?

To answer this question, denote any partition $\mathcal{C}$ of a poset P into convex chains as a **chaining**. A chaining is **proper** if there is a linear extension L such that the elements of each chain are consecutive in L. Since each chain C has $|C| - 1$ bumps, it is immediate that

$$B(P, L) = \sum_{C \in \mathcal{C}} (|C| - 1) \tag{4.4}$$

and so

$$s(P) = \min_{\mathcal{C} \text{ proper chaining of } P} \sum_{C \in \mathcal{C}} (|C| - 1). \tag{4.5}$$

Theorem 4.1.1 below characterizes the proper chainings of a poset in terms of alternating cycles. A **2k-alternating cycle** $(a_1, b_1, a_2, b_2, \ldots, a_k, b_k)$ of a set of chains $\{C_1, C_2, \ldots, C_k\}$ is a collection of elements $a_i, b_i \in C_i$ such that the only comparisons under $\leq_P$ in $\{a_1, \ldots, a_k, b_1, \ldots, b_k\}$ are

$$a_i \leq_P b_i, \qquad i = 1, \ldots, k. \tag{4.6}$$

$$b_i \geq_P a_{(i+1) \pmod k}, \qquad i = 1, \ldots, k. \tag{4.7}$$

or compactly

$$a_1 \leq_P b_1 \geq_P a_2 \leq_P b_2 \geq_P \cdots \leq_P b_{k-1} \geq_P a_k \leq_P b_k \geq_P a_1. \tag{4.8}$$

In particular note that all the points $a_1, \ldots, a_k, b_1, \ldots, b_k$ are distinct.

A chaining $\mathcal{C}$ of a poset P is **cycle-free** if it does not contain a subset of chains $C_1, \ldots, C_k$ and points $a_i, b_i \in C_i$ such that $(a_1, b_1, a_2, b_2, \ldots, a_k, b_k)$ is a $2k$-alternating cycle. The following theorem is already part of the folklore of the field, and can be obtained as a consequence of the work of Duffus et al. [45].

Theorem 4.1.1 (Duffus et al. [45])**.** *A chaining $\mathcal{C}$ of a poset P is proper if and only if it is cycle-free.*

The previous theorem, together with (4.3) and (4.5), implies that the jump number problem is equivalent to the problem of finding a cycle-free chaining of maximum cardinality.

This observation has interesting corollaries for **bipartite posets**. A poset is bipartite if its comparability graph is bipartite or equivalently, when every chain contains at most two elements. In particular, for a bipartite poset P with comparability graph $G_P = (A \cup B, E)$:

1. Every nonempty chain in P is convex and corresponds to either a singleton or an edge in G_P.

2. Chainings in P are in bijection with matchings in G_P.

3. Proper chainings in P correspond to matchings M in G_P such that there are no edges $e_1, \ldots, e_k$ in M and $f_1, \ldots, f_k$ in $E \setminus M$ where $e_1 f_1 e_2 f_2 \cdots e_k f_k$ is a cycle in G_P. This is, there are no M-alternating cycles in G_P.

The last consequence implies the following theorem:

Theorem 4.1.2 (Chaty and Chein [27])**.** *Solving the jump number problem in a bipartite poset is equivalent to finding an alternating-cycle-free matching M of maximum size in the corresponding comparability graph.*

As a corollary, we obtain that the jump number of bipartite posets only depends on the comparability graph of the poset. Habib [82] has shown that, in fact, this holds for every poset: the jump number is a comparability invariant.

Theorem 4.1.3 (Habib [82])**.** *Posets having the same comparability graph have the same jump number.*

The previous theorem states that the jump number $j(G)$ of a comparability graph G is a well-defined parameter.

4.1.1 Complexity of the Jump Number Problem

The literature concerning the jump number is vast. In what follows we mention some of the most relevant results regarding the complexity of this parameter.

Pulleyblank [139] has shown that determining the jump number of a general poset is NP-hard. On the other hand, there are efficient algorithms for many interesting classes of posets. Chein and Martin [29] give a simple algorithm to determine the jump number in forests. Habib [81] presents new algorithms for forests, for graphs having a unique cycle, and for graphs having only disjoint cycles. Cogis and Habib [35] introduce the notion of greedy linear extension. They prove that every such extension is optimal for the jump number in *series-parallel posets*, obtaining an algorithm for this poset class. Rival [144] extends this result to N-free posets. Later, Gierz and Poguntke [69] show that the jump number is polynomial for *cycle-series-parallel posets*.

Duffus et al. [45] have shown that for alternating-cycle-free posets, the jump number of a poset is always one unit smaller than its width (see Theorem 4.1.1). Chein and Habib [28] give algorithms for posets of width two and later Colbourn and Pulleyblank [36] provide a dynamic program algorithm for posets of bounded width. Steiner [155] also gives polynomial time algorithms for posets admitting a particular *bounded decomposition width*.

Since the jump number is polynomial for posets of bounded width, it is natural to ask the same question for posets of bounded height. On the negative side, Pulleyblank's proof [139] of NP-hardness shows that the jump number is NP-hard even

when restricted to posets of height 2 or equivalently, to bipartite graphs. Müller [121] has extended this negative result to chordal bipartite graphs. On the positive side, there are efficient algorithms to compute the jump number of bipartite permutation graphs: an $O(n + m)$-time algorithm by Steiner and Stewart [156] and $O(n)$-time algorithms independently discovered by Fauck [53] and Brandstädt [16], where n and m represent the numbers of vertices and edges of the graph respectively. The last two algorithms require a succinct representation of the graphs (e.g. a permutation). There are also an $O(n^2)$-time algorithm for biconvex graphs by Brandstädt [16], an $O(m^2)$-time and an $O(nm)$-time algorithm for bipartite distance hereditary graphs developed by Müller [121] and Amilhastre et al. [2] respectively, and an $O(n^9)$-time dynamic program algorithm by Dahlhaus [40] for convex graphs.

All the results in the previous paragraph are based on the equivalence for bipartite graphs between the jump number and the maximum alternating-cycle free matching problem (Theorem 4.1.2). For chordal bipartite graphs, the only induced cycles have length 4. Therefore, in this class, the jump number problem is equivalent to finding a maximum matching having no 4-alternating cycle. We study the latter problem in detail in Section 4.2. The algorithms for bipartite permutation and biconvex graphs are further inspired by a problem in computational geometry: the minimum rectangle cover of a biconvex rectangular region. We explore this problem in Section 4.3.1.

An important class of posets for which the complexity of this problem is still not settled is the class of two-dimensional posets. The tractability of the jump number problem on permutation graphs (that is, comparability graph of two-dimensional posets, see Lemma 3.4.3) is still open and conjectured to be polynomially solvable by Bouchitté and Habib [15]. As mentioned above, the jump number is polynomial for bipartite permutation graphs. Ceroi [23] extended this result to two-dimensional posets having bounded height. Interestingly enough, Ceroi [24] also gives a proof of the NP-hardness of a weighted version of this problem and conjectures, in contrast to Bouchitté and Habib, that the unweighted case is also NP-hard. Both results of Ceroi use a clever reduction to the maximum weight independent set of a particular family of rectangles. A similar construction for other types of graphs is studied in Section 5.2.

We end this survey by mentioning other results regarding the jump number. Mitas [117] has shown that the jump number problem is NP-hard also for *interval posets*. However, 3/2-approximation algorithms have been independently discovered by Mitas [117], Felsner [54] and Syslo [160]. As far as we know, these are the only approximation results in this area, and it would be interesting to obtain approximation algorithms for other families of posets. It is also worth mentioning that, as shown by von Arnim and de la Higuera [3], this parameter is polynomial for the subclass of semiorders.

There has also been work on posets avoiding certain substructures (for example, N-free posets as stated before in this survey). Sharary and Zaguia [149] and Sharary [148] have shown that the jump number is polynomial for K-free posets and for Z-free posets respectively. Lozin and Gerber [106] propose some necessary conditions for the polynomial-solvability of the jump number on classes of bipartite graphs characterized by a finite family of forbidden induced graphs and give polynomial time

algorithms for some of these classes. In a different article, Lozin [105] shows that the jump number problem is also polynomial for E-free bipartite graphs.

Another interesting result is that the jump number is *fixed parameter tractable*: El-Zahar and Schmerl [51] have shown that the decision problem asking whether the jump number is at most a fixed number k is solvable in polynomial time and McCartin [114] has given an algorithm linear in the number of vertices (but factorial in k) to solve the same question.

4.2 Cross-Free Matchings and Biclique Covers

In this section we introduce new notation. Given a graph $G = (V, E)$, a **biclique** is the edge set of a (not necessarily induced) complete bipartite subgraph of G. A **biclique cover** is a family of bicliques whose union is E. Two distinct edges $e = ab$ and $f = a'b'$ **cross** if and only if ab' and $a'b$ are also edges of the graph. A **cross-free matching** is a collection of edges that are pairwise non-crossing. We denote the maximum size of a cross-free matching by $\alpha^*(G)$ and the minimum size of a biclique cover[2] by $\kappa^*(G)$. Two natural problems are the following.

Problem 2 (Maximum Cross-Free Matching Problem)**.** Given a graph $G = (V, E)$, find a cross-free matching M of maximum cardinality $\alpha^*(G)$.

Problem 3 (Minimum Biclique Cover Problem)**.** Given a graph $G = (V, E)$, find a biclique cover B of minimum cardinality $\kappa^*(G)$.

Even though the definitions above make sense for any type of graph, let us focus for the rest of this work on *bipartite graphs*. An important observation is that two edges cross if and only if they are incident to the same vertex or are part of a 4-cycle. Hence, the cross-free matchings of a graph are exactly the matchings having no 4-alternating cycle. This observation implies the following.

Lemma 4.2.1 (Müller [121])**.** *Let G be a chordal bipartite graph. Then cross-free matchings and alternating-cycle-free matchings of G coincide*[3]*.*

Using this lemma, together with Theorem 4.1.2, (4.3) and (4.5), we obtain the following theorem.

Theorem 4.2.2 (Implicit in Müller [121])**.** *Computing the jump number of a bipartite chordal graph G is equivalent to finding a maximum cross-free matching G, and*

$$j(G) + \alpha^*(G) = n - 1, \tag{4.9}$$

where n is the number of vertices of G.

[2]Different authors have introduced these concepts before under different names. Müller [121, 122] uses *edge-cliques*, *dependent edges*, and *alternating C_4-free matchings* to denote bicliques, crossing edges and cross-free matchings respectively. He also uses $\alpha_e(G)$ and $\kappa_e(G)$ instead of $\alpha^*(G)$ and $\kappa^*(G)$. Dawande [42] denotes crossing edges that are not incident as *cross-adjacent edges* and uses $m^*(G)$ and $\beta^*(G)$ instead of $\alpha^*(G)$ and $\kappa^*(G)$. Fishburn and Hammer [56] call $\kappa^*(G)$ the *bipartite dimension of G*.

[3]Müller shows this also holds for *distance-hereditary graphs* and *strongly chordal graphs*.

Observe that we could also have defined crossing edges of bipartite graphs as follows. Two distinct edges cross if and only if there is a biclique containing them. This observation implies that for any cross-free matching M and for every biclique cover B of a bipartite graph, $|M| \leq |B|$. Therefore, $\alpha^*(G) \leq \kappa^*(G)$. It is a very interesting problem to find classes of graphs for which equality holds.

Define the **crossing graph**[4] $X(G)$ as the graph on the edges of G where the adjacencies are given by the crossing relation between them, this is

$$X(G) = (E(G), \{ef : e \text{ crosses } f\}). \tag{4.10}$$

It is immediate from their definition that cross-free matchings of a bipartite graph G correspond exactly to stable sets of $X(G)$. On the other hand, while every biclique of G is a clique of $X(G)$, the opposite is not necessarily true. However, maximal cliques in $X(G)$ correspond to maximal bicliques of G. Noting that there is always a minimum cardinality biclique cover of G that uses only maximal bicliques we conclude that

$$\alpha^*(G) = \alpha(X(G)) \text{ and } \kappa^*(G) = \kappa(X(G)), \tag{4.11}$$

where $\alpha(H)$ and $\kappa(H)$ denote the size of a maximum stable set and the size of a minimum clique cover of H, justifying our choice of notation. An important case where both quantities can be computed in polynomial time is when the graph $X(G)$ is perfect. In this case we say that the graph G is **cross-perfect**. We refer the reader to Section 3.5 for notions related to perfect graphs. For a survey of known algorithmic results for the cross-free matchings and biclique covers we refer the reader to Section 4.3.4.

4.3 Related Problems

4.3.1 Matrices: Boolean rank, Antiblocks and Block Covers

Here we explore some problems involving 0-1 matrices. To define them, we need some definitions.

The **boolean algebra** consists of elements 0 and 1, where the *sum* '+' is replaced by the logical AND (or the minimum) '$\wedge$' and the *product* '$\cdot$' is replaced by the logical OR (or the maximum) '$\vee$'. A **boolean matrix** is a matrix whose entries belong to the boolean algebra. Addition and multiplication of boolean matrices are defined as usual, except that we use operations in the boolean algebra. The **boolean rank** of an $s \times t$ matrix M is the minimum r such that there exists an $s \times r$ boolean matrix P and an $r \times t$ boolean matrix Q such that $M = P \cdot Q$.

Problem 4 (Boolean Rank Problem)**.** Given a boolean matrix M, find its boolean rank and the corresponding matrices P and Q with $M = P \cdot Q$.

[4] Also denoted as the *dependence graph* by Müller [122] and as the modified line graph $L'(G)$ by Dawande [42].

A **(combinatorial) block** of an $s \times t$ matrix M is a submatrix which has only 1-entries. More formally, it is a set $S \times T$, where $S \subseteq [s]$ is a set of rows and $T \subseteq [t]$ is a set of columns, such that $M_{ij} = 1$ for all $i \in S$ and $j \in T$. If $(i, j) \in S \times T$, we say that the entry (i, j) is covered by the block $S \times T$.

Problem 5 (Block Cover Problem)**.** Given a boolean matrix M, find a minimum cardinality collection of *blocks* of M such that every 1-entry of M is covered by at least one block.

The following lemma shows that the two problems above are equivalent. This lemma can be found with a slightly different notation in an article by Gregory et al. [74]. We prove it for completeness.

Lemma 4.3.1 (Gregory et al. [74])**.** *The boolean rank of a matrix M is the minimum number r such that the set of 1-entries of M can be covered by an union of r combinatorial blocks of M.*

Proof. Let M be an $s \times t$ matrix whose 1-entries are covered by the union of r combinatorial blocks $S_k \times T_k, 1 \leq k \leq r$. Define P as the $s \times r$ matrix having as columns the characteristic vectors of all S_k and Q as the $r \times t$ matrix having as rows the characteristic vectors of all T_k. This is, $P_{ik} = 1$ if and only if S_k contains element i, and $Q_{kj} = 1$ if and only if T_k contains element j. Note that $M_{ij} = 1$ if and only if there is k such that $(i, j) \in S_k \times T_k$, or equivalently, if and only if there is k such that $P_{ik} \wedge Q_{kj} = 1$. Therefore $M = P \cdot Q$, and so its boolean rank is at most r.

Conversely, let M be equal to $P \cdot Q$, where P and Q are $s \times r$ and $r \times t$ boolean matrices respectively. Consider the sets $S_k = \{i \colon P_{ik} = 1\}$ and $T_k = \{j \colon Q_{jk} = 1\}$. Each $S_k \times T_k$ defines a combinatorial block of M. Indeed, for $i \in S_k, j \in T_k$, $M_{ij} = (P \cdot Q)_{ij} = \bigvee_{\ell=1}^{r} P_{i\ell} \wedge Q_{\ell j} \geq P_{ik} \wedge Q_{kj} = 1$, which means that $M_{ij} = 1$. By definition of boolean product, $M_{ij} = 1$ if and only if there exists k such that $P_{ik} = 1$ and $Q_{kj} = 1$, or equivalently if and only if (i, j) is a 1-entry of some combinatorial block $S_k \times T_k$. Therefore the 1-entries of M are covered by the union of at most r combinatorial blocks of M. □

Consider any biadjacency matrix M for a bipartite graph $G = (A \cup B, E)$. It is easy to see that the combinatorial blocks of M are in bijection with the bicliques of G. This observation implies the following result.

Lemma 4.3.2 (Orlin [133], Gregory et al. [74])**.** *The minimum biclique cover problem on bipartite graphs is equivalent to the boolean rank problem and the block cover problem.*

Proof. The proof follows directly from Lemma 4.3.1. □

The problems related by the previous lemma also have applications to communication complexity as the non-deterministic communication complexity of a boolean function $f \colon X \times Y \to \{0, 1\}$ is precisely the binary logarithm of the boolean rank of the input matrix of f (see, e.g. [99]).

We have seen how the minimum biclique cover problem of a bipartite graph is restated in terms of biadjacency matrices. The same can be done for the maximum cross-free matching problem. For that, define a **(combinatorial) antiblock** of a boolean matrix M as a collection of 1-entries of M, no two of them contained in the same block of M.

Problem 6 (Maximum Antiblock Problem)**.** Given a boolean matrix M, find a maximum cardinality *antiblock* of M.

The next lemma follows immediately by interpreting the cross-free matchings of a bipartite graph in its corresponding biadjacency matrix.

Lemma 4.3.3. *The maximum cross-free matching problem of a bipartite graph is equivalent to the maximum antiblock problem.*

Abusing notation, we use $\alpha^*(M)$ and $\kappa^*(M)$ to denote the size of a maximum antiblock and a minimum block cover of a boolean matrix M. It is direct from their definitions that for every such matrix $\alpha^*(M) \leq \kappa^*(M)$. As the example in Figure 4-2 shows these quantities do not need to be equal.

$$M_1 = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}. \qquad M_2 = \begin{bmatrix} 0 & 1 & \mathbf{1} & 0 \\ 1 & \mathbf{1} & 1 & 0 \\ \mathbf{1} & 1 & 1 & \mathbf{1} \\ 0 & 0 & \mathbf{1} & 1 \end{bmatrix}.$$

Figure 4-2: The matrix on the left (example by Lubiw [109]) is not firm: the maximum size of an antiblock is 5, but the minimum size of a block cover is 6. The matrix on the right is firm, but the corresponding graph G is not cross-perfect: the bold entries $(1,3),(2,2),(3,1),(3,4),(4,3)$ form a *hole* in $X(G)$.

Lubiw [109] has proposed the problem of characterizing **firm matrices**, that is, matrices for which the min-max relation $\kappa^*(M) = \alpha^*(M)$ holds for M and for all submatrices M' of M. Despite its similarity, this is not the same as characterizing cross-perfect bipartite graph (see Section 4.2) which are the graphs for which $X(G)$ is perfect: Given a graph G with biadjacency matrix M, the *vertices* of $X(G)$ correspond to 1-entries of M, and two entries are connected in $X(G)$ if there is a block containing both. In particular, $X(G)$ is perfect if for any matrix M' obtained from M by changing some 1-entries into 0-entries, the min-max relation $\kappa^*(M') = \alpha^*(M')$ holds. From here, we get that cross-perfection implies firmness, however the converse does not hold as the example in Figure 4-2 shows.

4.3.2 Geometry: Antirectangles and Rectangle Covers

We define two geometrical problems related to the problems presented in the previous section. An **orthogonal polygon** is a polygon in the plane having only horizontal and vertical segments.

Problem 7 ((Geometric) Minimum Rectangle Cover Problem)**.** Given an orthogonal polygon P, find a collection of orthogonal rectangles whose union is P having minimum cardinality.

Problem 8 ((Geometric) Maximum Antirectangle Problem)**.** Given an orthogonal polygon P, find a collection of points inside P, no two of them contained in an orthogonal rectangle contained in P.

It is a simple exercise to restate these problems in terms of matrices: A **rectangle** of a boolean matrix M is a block whose rows and columns are consecutive in M. An **antirectangle** is a collection of 1-entries of M no two of them contained in the same rectangle of M.

Problem 9 ((Matrix) Minimum Rectangle Cover Problem)**.** Given a boolean matrix M, find a minimum cardinality collection of *rectangles* of M such that every 1-entry of M is covered by at least one rectangle.

Problem 10 ((Matrix) Maximum Antirectangle Problem)**.** Given a boolean matrix M, find a maximum cardinality *antirectangle* of M.

Problems 9 and 10 are more general than their geometrical counterparts as the set of 1-entries of a boolean matrix does not necessarily define a polygon: the region defined could be disconnected. Just like for the case of block covers and antiblocks, the size of a maximum antirectangle is always at most the size of a minimum rectangle cover.

Chvátal once conjectured that both quantities are equal for any polygon. Small counterexamples were quickly proposed by Chung even for simply connected polygons (see Figure 4-3). Chaiken et al. [25] have shown that the min-max relation holds for biconvex polygons (respectively for biconvex matrices). Later, Györi [80] extends this result to vertically convex polygons (resp. for convex matrices). Györi obtains this result by studying a different problem, namely computing a minimum base of a family of intervals. We revisit that problem and related ones, in the next section.

It is important to remark that in the case of biconvex matrices, maximal blocks are rectangles. This follows directly from the following fact. If two 1-entries (i, j) and (i', j') are in the same block, then (i, j') and (i', j) are also 1-entries. Therefore, by biconvexity, all the entries in the rectangle with corners in (i, j) and (i', j') are also 1-entries. Because in any block cover (resp. rectangle cover) we can assume every block (resp. rectangle) to be maximal without increasing the cardinality of the cover, the minimum block cover and the minimum rectangle cover problem coincide for biconvex matrices. Similarly, since antiblocks and antirectangles can be defined as collection of 1-entries, no two of them in the same maximal blocks or maximal

$$M = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

Figure 4-3: Chung's counterexample (see [25]). The maximum size of an antirectangle is 7, but the minimum size of a rectangle cover is 8.

antirectangle respectively, the maximum antiblock and the maximum antirectangle problem coincide for biconvex matrices. In particular, using Lemmas 4.3.2, 4.3.3 and Chaiken et al.'s min-max result [25] for rectangle cover in biconvex polygons, we obtain the following theorem.

Theorem 4.3.4. *For a biconvex graph G, the size $\alpha^*(G)$ of a maximum cross-free matching and the size $\kappa^*(G)$ of a minimum biclique cover coincide.*

Even though the theorem above follows from simple observations, to the author knowledge it has not been noted in the literature for the minimum biclique cover problem before our joint work with Telha [153] and this thesis.

4.3.3 Interval Combinatorics: Bases and Irredundancy

To simplify our discussions later, in this section we denote as **interval** every set of consecutive elements of a finite linear order.

We say that a sequence $(I_1, \ldots, I_k)$ of intervals is **irredundant**[5] if every interval contains a point not contained in the union of the preceding intervals. We extend this definition to unordered families by saying that a family of intervals is irredundant if can be sorted into a irredundant sequence.

We say that a family $\mathcal{F}$ of intervals **generates** an interval I if the latter can be obtained as the union of intervals contained in $\mathcal{F}$. A family of intervals is a **basis** for another family if every interval of the latter family is generated by the former. Two natural problems are then the following.

Problem 11 (Maximum Irredundant Subfamily Problem)**.** Given a family $\mathcal{F}$ of intervals, find an irredundant subfamily of maximum cardinality.

Problem 12 (Minimum Basis Problem)**.** Given a family $\mathcal{F}$ of intervals, find an interval basis $\mathcal{B}$ for $\mathcal{F}$ of minimum size, where $\mathcal{B}$ is not restricted to be a subfamily of $\mathcal{F}$.

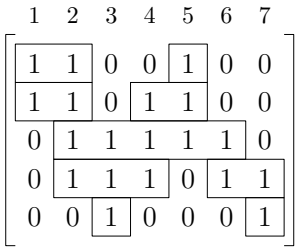
If $\mathcal{B}$ generates an irredundant family of intervals $\mathcal{I}$ then we clearly have that $|\mathcal{B}| \geq |\mathcal{I}|$ as every element of the sequence requires a new generator. Frank conjectured

[5] Also called $\bigcup$-increasing by Györi [80].

that the maximum size of an irredundant subfamily is always equal to the minimum size of a basis. This question has been answered affirmatively by Györi [80] in a very beautiful work. Györi reached this conclusion while studying the rectangle cover problem defined in the previous section. Indeed, as we describe next, his result implies as a corollary the min-max relation for convex matrices between the maximum antirectangle and the minimum rectangle cover.

Consider a vertically convex matrix M and let $B = \{1, 2, \ldots, t\}$ its set of columns. Since the matrix is vertically convex, every maximal rectangle of 1-entries of M is defined uniquely by an interval in B.

For each row of M, consider the intervals on B induced by maximal sets of consecutive 1-entries. Let $\mathcal{F}$ be the full collection of (different) intervals obtained in this way. Using the previous paragraph, it is easy to see that the collection of maximal rectangles of M are in bijection with the intervals of $\mathcal{F}$. See Figure 4-4 for an example.



$$\mathcal{I} = \{\, \{1,2\}; \{5\}; \{4,5\}; \{2,3,4,5,6\}; \\ \{2,3,4\}; \{6,7\}; \{3\}; \{7\} \,\}.$$

Figure 4-4: Horizontal intervals associated to a vertically convex matrix.

Consider a minimal base $\mathcal{B}$ and a maximum irredundant subfamily $\mathcal{I}$ for $\mathcal{F}$ and let k be their common cardinality. The rectangles associated to $\mathcal{B}$ form a rectangle cover of M. We can construct an antirectangle of M of the same cardinality as follows. Let $(I_1, \ldots, I_k)$ be the irredundant sequence of $\mathcal{I}$. There must be a point b_i in each interval I_i that is not contained in $\bigcup_{j<i} I_j$. Recall that each I_i was selected as a maximum interval of consecutive 1-entries for some row a_i. Let P_i be the 1-entry $M_{a_i b_i}$. The collection $\{P_1, \ldots, P_k\}$ must form an antirectangle, for suppose there was a rectangle covering two of these points, say P_i and P_j with $i < j$. This means that in row a_i, the entire interval between columns b_i and b_j is formed by 1-entries, contradicting the definition of point b_j. This proves the min-max relation between rectangle covers and antirectangles for convex matrices.

Interestingly enough, Györi's results also allow us to show a min-max relation between block covers and antiblocks for convex matrices (or equivalently, between biclique covers and cross-free matchings of convex graphs). We can not use the exact construction above though since, unlike the biconvex case, for convex matrices maximum blocks and maximum rectangles are not equivalent, and neither are antiblocks and antirectangles.

To do that, it is convenient to introduce some notation. A collection of point-interval pairs $\{(p_i, I_i) \colon i \in [k]\}$ is called **independent** if for every $i \neq j$, we have $p_i \notin I_j$ or $p_j \notin I_i$. Consider the following problem.

Problem 13. [Maximum Independent Set of Point-Interval Pairs] Given a family of intervals $\mathcal{F}$, find a maximum cardinality independent collection of point-interval pairs having the intervals in $\mathcal{F}$.

Györi's proof of his min-max relation for intervals uses the following fact which we do not prove.

Theorem 4.3.5 (Györi [80], see also Lubiw [110] and Knuth [96])**.** *Problems 11 and 13 are equivalent: From any irredundant family of intervals we can obtain an independent family of point-interval pairs and vice versa.*

Consider now a family of intervals $\mathcal{F}$ over a finite total ordered set X. It is straightforward to see that the containment bipartite graph G having parts X and $\mathcal{F}$ is convex on X, and that every (finite) convex graph arise in this way.

Lemma 4.3.6. *Let X, $\mathcal{F}$ and G be as above. The cross-free matchings of G correspond to independent families of point-interval pairs in $X \times \mathcal{F}$. Also, from every biclique cover of G we can obtain a base of $\mathcal{F}$ of the same cardinality and vice versa.*

Proof. The first part follows since, by definition, two point-interval pairs are independent if and only if the corresponding edges do not cross in G. For the second part consider a biclique cover. We can assume that every biclique of this cover is maximal without changing the cardinality of the cover. Since G is biconvex, the elements in X covered by a maximal biclique form an interval. It is easy to see that the family of all such intervals obtained from the maximal bicliques in this cover is a base for $\mathcal{F}$.

Conversely, consider a base for $\mathcal{F}$. For every interval I in this base, consider the largest biclique having I as one of its parts. The collection of bicliques constructed in this way covers all the edges. □

From here, we obtain as a corollary the following.

Theorem 4.3.7. *For a convex graph G, the size of maximum cross-free matching, $\alpha^*(G)$ and the size of a minimum biclique cover, $\kappa^*(G)$, coincide.*

Proof. Use Györi's min-max relation for intervals and Lemma 4.3.6. □

4.3.4 Survey on the Complexity of the Presented Problems

We have already established that for chordal bipartite graphs, the jump number problem and the maximum cross-free matching problem are equivalent (see Theorem 4.2.2). For this reason, most of the algorithmic results concerning the maximum cross-free matching problem were already mentioned in Section 4.1.1.

There are three types of results for the described problems we wish to survey: hardness results, min-max relations, and algorithmic results. We review each type in no particular order (the problems presented are so intertwined, that it is somehow hard to separate them by problem).

We review first the hardness results. In 1977, Orlin [133] shows that the minimum biclique cover problem is NP-hard even for bipartite graphs. This implies that

computing the boolean rank or finding a minimum block cover of a matrix is also NP-hard. Two years later, Masek [113], in an unpublished manuscript (in)famous for its elusiveness (see Johnson's article [92]) shows that the geometric problem of covering orthogonal polygons by a minimum number of rectangles is NP-hard. Masek's proof, however, requires the use of polygons that are not simply connected. Several years later, Culberson and Reckhow [39] show that the rectangle cover problem is still NP-hard when restricted to simply connected polygons. In 1990, Müller [121] extends Orlin's result by showing that the minimum biclique cover is NP-hard even for chordal bipartite graphs.

Regarding min-max relations, Chaiken et al. [25] show in 1981, that minimum rectangle covers and maximum antirectangles have the same size for biconvex polygons. Three years later, Györi [80] proves that for any collection of intervals, maximum irredundant subfamilies and minimum bases have the same cardinality. Using this, he also extends Chaiken et al.'s result to vertically convex polygons. There are also generalizations of Györi's min-max result for intervals given by Lubiw [110] and Frank and Jordán [63]. These results are described in Sections 6.2.2 and 5.5 respectively.

Now we focus on algorithms for the presented problems.

Chaiken et al.'s result is algorithmic but Gyori's proof of his min-max relation is not. Franzblau and Kleitman [64] have given an algorithmic proof of Györi's result which can find maximum irreducible families and minimum bases of intervals in $O(n^2)$-time. This translates immediately to an $O(n^2)$ algorithm for minimum rectangle covers and maximum antirectangles of biconvex and convex polygons. A nice implementation of this algorithm can be found in an article by Knuth [96]. Lubiw [107, 109] also gives polynomial time algorithms for a somewhat larger class of polygons, called plaid polygons. For a simpler class of biconvex polygons, denoted as monotone orthogonal polygons in [53], Branstädt [16] and Fauck [53] independently present $O(n)$-time algorithms for the minimum rectangle cover problem.

Some of the results for the minimum rectangle cover problem translate immediately to the minimum biclique cover problem. Franzblau and Kleitman's $O(n^2)$-time algorithm for rectangle covers, when applied to biconvex polygon, becomes an $O(n^2)$-time algorithm for minimum biclique covers of biconvex graphs. Similarly, since monotone orthogonal polygons corresponds exactly to matrices having 2-staircase normal form (see Theorem 3.4.21), and these matrices are the biadjacency matrices of bipartite permutation graphs, the minimum biclique cover of these graphs can be computed in $O(n)$-time using Branstädt's or Fauck's algorithm.

Regarding other algorithms for the biclique cover problem, Müller [121, 122] has shown that if G is a C_4-free bipartite graph, a strongly chordal graph, a distance hereditary graph or a bipartite permutation graph, then the associated crossing graph $X(G)$ is perfect, and so (in view of (4.11)) both the minimum biclique cover and maximum cross-free matchings are polynomially computable in those classes. Amilhastre et al. [2] also give polynomial algorithms for the minimum biclique cover of bipartite domino-free graphs.

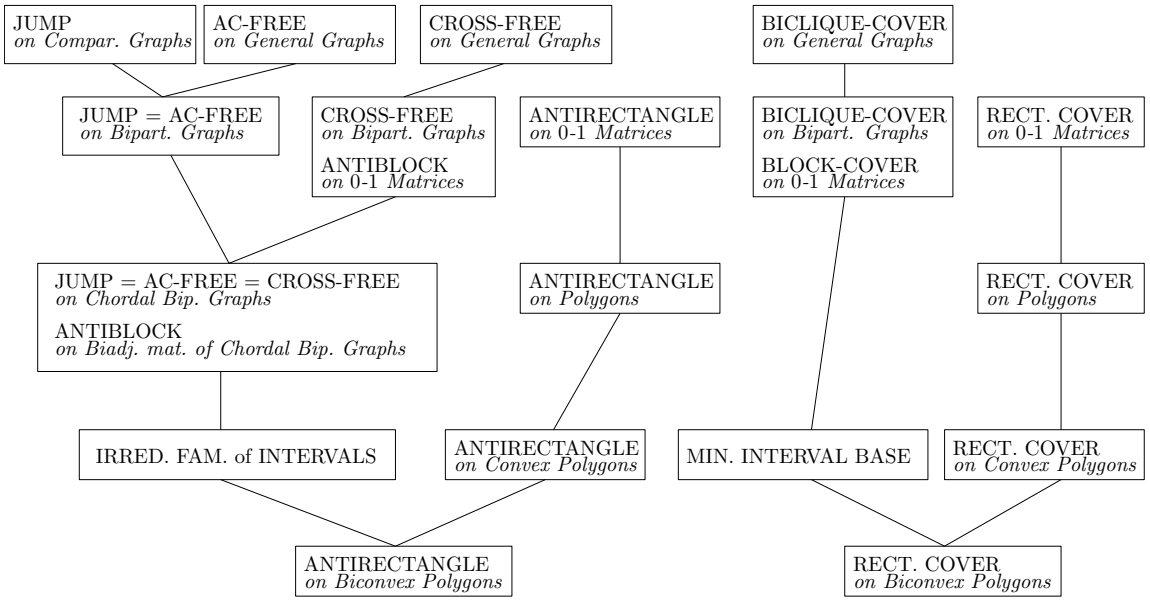


Figure 4-5: Relation between problems

4.4 Summary of Presented Problems and Results

In Figure 4-5, we present a visual summary of some of the problems presented in this section, where a line between one problem and another means that the problem located below is a particular case of the problem located higher.

In Table 4.1, we summarize some of the known algorithmic results for the minimum biclique cover and the maximum cross-free matching problems. In this table we include a result marked as *new* for the jump number of convex graphs. Even though the same result for biclique covers has already been hinted by Amilhastre et al. [2], the possibility of efficiently computing the maximum cross-free matching of convex graphs has often been overlooked: In the literature concerning the jump number problem before our work, the best algorithm reported for convex graphs is still the $O(n^9)$-time dynamic program algorithm by Dahlhaus [40].

Table 4.1: Table of known results.

	Bip. Perm.	Biconvex	Convex
Max. cross-free matching (Jump number)	$O(n)$ [16, 53]	$O(n^2)$ [16]	$O(n^9)$ [40]
(new)	-	-	$\boldsymbol{O(n^2)}$[a]
Min. biclique-cover	$O(n)$ [16, 53]	$O(n^2)$[a]	$O(n^2)$[a]

[a] These results follows from Theorem 4.3.7 and the algorithm of Franzblau and Kleitman [64] for geometric rectangle cover. The possibility of applying this algorithm to compute the biclique cover of convex graphs has been noticed by Amilhastre et al. [2].

Chapter 5

Jump Number of 2DORGs

In this chapter we show that the jump number problem is polynomially solvable for the class of two directional orthogonal ray graphs, using its equivalence to the maximum cross-free matching problem. A key tool for this result, in a way the most important contribution of this part of the thesis, is a new *geometrical reformulation* of the maximum cross-free matching and the minimum biclique cover problems as the problem of finding *maximum independent set* and the *minimum hitting set* of an associated collection of rectangles in the plane respectively.

Most of the results of this chapter are joint work with Claudio Telha. This chapter is organized as follows.

In Section 5.1, we review some results for independent and hitting sets of rectangles. A big conjecture in the area is whether or not the ratio between the size of the minimum hitting set and the size of the maximum independent set of any family of axis-aligned rectangles is bounded by a constant. In passing, we observe that a simple application of recent approximation algorithms for both problems gives a bound of $O(\log^2 \log \alpha)$ for this ratio, where α is the size of the maximum independent set, improving the existing bounds of $O(\log \alpha)$.

In Section 5.2, we describe our key tool. In Section 5.3 we give a linear programming based algorithm for the maximum cross-free matching on 2DORGs. The main result of Section 5.4 is an $\widetilde{O}(n^\omega)$-time combinatorial algorithm to compute, for a given 2DORG, both a cross-free matching and a biclique cover of the same size. In particular, this shows that the min-max relation holds.

We later explore the relation between our results with previous work. In particular, we show how our min-max relation can be obtained as a particular case of a strong result by Frank and Jordán [63]. Finally, we present some open problems.

5.1 Maximum Independent Sets and Minimum Hitting Sets of Rectangles

As preparation for our results, we review two geometric problems: the maximum independent set of rectangles and the minimum hitting set of rectangles.

We start by defining the geometric notion of a rectangle. For our purposes, a

rectangle is the cartesian product of two closed intervals, viewed as a subset of the plane $\mathbb{R}^2$. In other words, we only consider possibly degenerated rectangles in the plane having their sides parallel to the x and y axes. Alternatively, we can define a rectangle by describing its bottom-left corner and its top-right corner. Formally, given two points a and b in $\mathbb{R}^2$, the **(geometric) rectangle** $\Gamma(a, b)$ is defined as:

$$\Gamma(a, b) = \{p \in \mathbb{R}^2 \colon a_x \leq p_x \leq b_x, a_y \leq p_y \leq b_y\}. \tag{5.1}$$

Note that if $a \not\leq_{\mathbb{R}^2} b$, the rectangle $\Gamma(a, b)$ is empty. Given a nonempty rectangle R, we use $\mathrm{bl}(R)$ and $\mathrm{tr}(R)$ to denote its bottom-left corner and top-right corner respectively.

Two rectangles are **independent** if they do not intersect, otherwise they are **intersecting**. Given a family $\mathcal{C}$ of rectangles, an **independent set** of $\mathcal{C}$ is a family of pairwise independent rectangles. We say that a point $p \in \mathbb{R}^2$ **hits** a rectangle R if $p \in R$. A set of points $H \subseteq \mathbb{R}^2$ is a **hitting set** of $\mathcal{C}$ if every rectangle in $\mathcal{C}$ is hit by at least one point of this set. We denote by $\mathrm{mis}(\mathcal{C})$ and $\mathrm{mhs}(\mathcal{C})$ the sizes of a *maximum independent set of rectangles* in $\mathcal{C}$ and a *minimum hitting set* for $\mathcal{C}$ respectively. It is natural to study the following problems.

Problem 14 (Maximum Independent Set of Rectangles Problem)**.** Given a family $\mathcal{C}$ of rectangles, find an independent set $\mathcal{I} \subseteq \mathcal{C}$ of maximum cardinality, this is $|\mathcal{I}| = \mathrm{mis}(\mathcal{C})$.

Problem 15 (Minimum Hitting Set of Rectangles Problem)**.** Given a family $\mathcal{C}$ of rectangles, find a hitting set H of $\mathcal{C}$ of minimum cardinality, this is $|H| = \mathrm{mhs}(\mathcal{C})$.

Consider the intersection graph of a family $\mathcal{C}$ of rectangles,

$$\mathcal{I}(\mathcal{C}) = (\mathcal{C}, \{RS : R \cap S \neq \emptyset\}). \tag{5.2}$$

Naturally, independent sets of $\mathcal{C}$ correspond to stable sets in $\mathcal{I}(\mathcal{C})$. Therefore,

$$\mathrm{mis}(\mathcal{C}) = \alpha(\mathcal{I}(\mathcal{C})). \tag{5.3}$$

It is a well known fact that rectangles have the **Helly property**. This is, if a collection of rectangles pairwise intersect, then all of them share a point in the plane. In fact, their intersection is a nonempty rectangle. In particular, we can assign to every clique C in $\mathcal{I}(\mathcal{C})$ a unique **witness point**, defined as the bottom-left point of the rectangle formed by the intersection of all elements in C.

Consider now a hitting set H for $\mathcal{C}$. The set of rectangles hit by a point p in H is a (not necessarily maximal) clique $C(p)$ of $\mathcal{I}(\mathcal{C})$. By replacing p by the witness point of any maximal clique containing $C(p)$ we obtain a new hitting set of the same cardinality. From here we conclude that $\mathcal{C}$ admits a minimum hitting set consisting only of witness points of maximal cliques, or equivalently a clique cover of $\mathcal{I}(\mathcal{C})$ using only maximal cliques. From here,

$$\mathrm{mhs}(\mathcal{C}) = \kappa(\mathcal{I}(\mathcal{C})). \tag{5.4}$$

Since any point hits at most one rectangle of an independent set, we deduce the relation

$$\mathrm{mis}(\mathcal{C}) \leq \mathrm{mhs}(\mathcal{C}). \tag{5.5}$$

An elementary, but important observation is that for both problems defined above we can restrict ourselves to the family $\mathcal{C}_\downarrow$ of **inclusion-wise minimal** rectangles in $\mathcal{C}$.

Lemma 5.1.1.

$$\mathrm{mis}(\mathcal{C}) = \mathrm{mis}(\mathcal{C}_\downarrow) \quad \textit{and} \quad \mathrm{mhs}(\mathcal{C}) = \mathrm{mhs}(\mathcal{C}_\downarrow).$$

Proof. We can transform any independent set in $\mathcal{C}$ into an independent set in $\mathcal{C}_\downarrow$ of the same cardinality by replacing each non-minimal rectangle by a minimal one contained in it. Since every independent set in $\mathcal{C}_\downarrow$ is also independent in $\mathcal{C}$, we obtain the first equality. For the second one, use that every hitting set of $\mathcal{C}_\downarrow$ is also a hitting set of $\mathcal{C}$ and vice versa. □

Given any hitting set $H \subseteq \mathbb{R}^2$ containing the witness points of all maximal cliques in $\mathcal{C}$, define the following polytopes:

$$\mathrm{P}_H(\mathcal{C}) = \left\{ x \in \mathbb{R}^{\mathcal{C}} \colon \sum_{R\colon\ p \in R} x_R \leq 1, \text{ for all } p \in H, x \geq 0 \right\}, \tag{5.6}$$

$$\mathrm{D}_H(\mathcal{C}) = \left\{ y \in \mathbb{R}^{H} \colon \sum_{p\colon\ p \in R} y_p \geq 1, \text{ for all } R \in \mathcal{C}, y \geq 0 \right\}, \tag{5.7}$$

and the associated integer and linear programs:

$$\mathrm{IP}_H(\mathcal{C}) = \max \left\{ \sum_{R \in \mathcal{C}} x_R \colon x \in \mathrm{P}_H(\mathcal{C}) \cap \{0,1\}^{\mathcal{C}} \right\}, \tag{5.8}$$

$$\mathrm{LP}_H(\mathcal{C}) = \max \left\{ \sum_{R \in \mathcal{C}} x_R \colon x \in \mathrm{P}_H(\mathcal{C}) \right\}, \tag{5.9}$$

$$\mathrm{IP}'_H(\mathcal{C}) = \max \left\{ \sum_{p \in H} y_p \colon y \in \mathrm{D}_H(\mathcal{C}) \cap \{0,1\}^{H} \right\}, \tag{5.10}$$

$$\mathrm{LP}'_H(\mathcal{C}) = \max \left\{ \sum_{p \in H} y_p \colon y \in \mathrm{D}_H(\mathcal{C}) \right\}. \tag{5.11}$$

Here, abusing notation we use $\mathrm{IP}_H(\mathcal{C}), \mathrm{LP}_H(\mathcal{C}), \mathrm{IP}'_H(\mathcal{C}), \mathrm{LP}'_H(\mathcal{C})$ to denote both the program formulations and their optimal values.

The characteristic vector of any independent set of $\mathcal{C}$ is inside $\mathrm{P}_H(\mathcal{C})$. Conversely, every integer vector x of $\mathrm{P}_H(\mathcal{C})$ is the characteristic vector of an independent set of $\mathcal{C}$: If there is a pair of intersecting rectangles in the support of x, then the inequality associated to any witness point of a maximal clique containing both rectangles is not satisfied. In particular, (5.8) is an integer program formulation for the maximum independent set problem and (5.9) is its corresponding linear program relaxation.

Note also that every integer vector y of $\mathrm{D}_H(\mathcal{C})$ is the characteristic vector of a hitting set of $\mathcal{C}$. However, only the characteristic vectors of hitting sets that are subsets of H are contained in $\mathrm{D}_H(\mathcal{C})$. Nevertheless, since there is an optimum hitting set using only witness points, the set $\mathrm{D}_H(\mathcal{C})$ contains at least one such optimum. Therefore, (5.10) and (5.11) are integer and linear program relaxations for the minimum hitting set. It is important to note that the values $\mathrm{LP}_H(\mathcal{C})$ and $\mathrm{LP}'_H(\mathcal{C})$ coincide since they correspond to dual linear programs.

Another important observation is that $\mathrm{P}_H(\mathcal{C})$ does not depend on the choice of H. As the following lemma shows, it only depends on the intersection graph $\mathcal{I}(\mathcal{C})$.

Lemma 5.1.2. *For any hitting set $H \subseteq \mathbb{R}^2$ containing the witness points of all maximal cliques in $\mathcal{C}$,*

$$\mathrm{P}_H(\mathcal{C}) = \mathrm{QSTAB}(\mathcal{I}(\mathcal{C})).$$

Proof. It follows directly from the definition of $\mathrm{QSTAB}(\mathcal{I}(\mathcal{C}))$ (see (3.4) in Section 3.5) and the fact that inequalities associated to non-witness points are implied by the ones associated to witness points. □

Denote by $\mathrm{P}(\mathcal{C})$ and $\mathrm{LP}(\mathcal{C})$ the common polytope and associated linear program value. For every collection of rectangles $\mathcal{C}$ we have

$$\mathrm{mis}(\mathcal{C}) \leq \mathrm{LP}(\mathcal{C}) \leq \mathrm{mhs}(\mathcal{C}). \tag{5.12}$$

It is interesting to find cases where (5.12) holds with equality. This happens, for instance when the polytope $P(\mathcal{C})$ is integral. We can characterize those cases.

Lemma 5.1.3. *Given a family of rectangles $\mathcal{C}$ the polytope $\mathrm{P}(\mathcal{C})$ is integral if and only if the intersection graph $\mathcal{I}(\mathcal{C})$ is* perfect.

Proof. The lemma follows using that $\mathrm{P}(\mathcal{C}) = \mathrm{QSTAB}(\mathcal{I}(\mathcal{C}))$ (Lemma 5.1.2) and Theorem 3.5.2 for perfect graphs. □

We give two examples of families where $\mathcal{I}(\mathcal{C})$ is perfect.

Interval Families.

Lemma 5.1.4. *Let $\mathcal{C}$ be a family of rectangles such that there is a single horizontal (or vertical) line intersecting all of them. Then $\mathcal{I}(\mathcal{C})$ is an interval graph, and therefore a perfect graph.*

Proof. Assume without loss of generality, by rotating and translating the collection, that the line intersecting every rectangle is the x-axis. For each rectangle R, let $I(R)$ be the interval obtained by projecting R over the x-axis. For every pair of rectangles R and R' on the family, $R \cap R' \neq \emptyset$ implies that $I(R) \cap I(R') \neq \emptyset$. Conversely, if $I(R) \cap I(R') \neq \emptyset$, then both R and R' contain the line segment $I(R) \cap I(R')$, and therefore, they intersect. This means that the intersection graph of $\mathcal{C}$ is equal to the intersection graph of the corresponding intervals, completing the proof. □

Skew Intersecting Families. Every rectangle R is defined as the cartesian product of two closed intervals R_x and R_y (where one or both could possibly be a singleton interval). We say that R and R' **skew-intersect** if

$$R_x \subseteq R'_x \text{ and } R_y \supseteq R'_y \tag{5.13}$$

or vice versa. We give some examples of pairs of skew-intersecting rectangles in Figure 5-1.

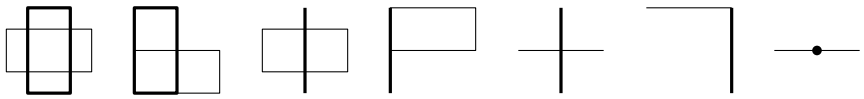


Figure 5-1: Skew-intersecting rectangles.

Lemma 5.1.5. *Let $\mathcal{C}$ be a family of rectangles such that every pair of intersecting rectangles are skew-intersecting. Then $\mathcal{I}(\mathcal{C})$ is a comparability graph, and therefore a perfect graph.*

Proof. Consider the following partial order in $\mathcal{C}$:

$$R \preceq R' \text{ if and only if } R_x \subseteq R'_x \text{ and } R_y \supseteq R'_y. \tag{5.14}$$

Note that two rectangles R and R' skew-intersect if and only if $R \preceq R'$ or $R' \preceq R$. Therefore, every pair of intersecting rectangles in $\mathcal{C}$ are comparable by $\preceq$. This means that $\mathcal{I}(\mathcal{C})$ is a comparability graph. In every pair of intersecting rectangles of Figure 5-1, the rectangle in bold precedes the other one in the partial order $\preceq$. □

For families $\mathcal{C}$ of rectangles having perfect intersection graph $\mathcal{I}(\mathcal{C})$, not only we have equality in (5.12), but we can also compute a maximum independent set by finding an optimal extreme solution of the linear program $\mathrm{LP}(\mathcal{C})$. This can be done in polynomial time since there is a formulation of the linear program using a polynomial number of inequalities. For that, use that $\mathrm{LP}(\mathcal{C}) = \mathrm{LP}_H(\mathcal{C})$, where

$$\begin{aligned} H = \{(x,y)\colon\ & x \text{ is the } x\text{-coordinate of a corner of a rectangle in } \mathcal{R} \text{ and} \\ & y \text{ is the } y\text{-coordinate of a corner of a rectangle in } \mathcal{R}\}. \end{aligned} \tag{5.15}$$

It is easy to see that H is a hitting set containing the witness points of maximal cliques of $\mathcal{C}$. Furthermore, $|H| \leq (2m)^2$, where $m = |\mathcal{C}|$.

Since there are polynomial time algorithms to find minimum clique-covers of perfect graphs (see Section 3.5), we can also find a minimum hitting set of these families in polynomial time. For the particular case of interval families and skew-intersecting families, there are more efficient algorithms that exploit the fact that the corresponding intersecting graphs are interval graphs, where a simple greedy algorithm works [77], or comparability graphs, as observed in Lemma 3.2.3.

Both the maximum independent and the minimum hitting set problems are NP-hard, as shown by, e.g. Fowler et al. [60]. Hence, the problem of approximating

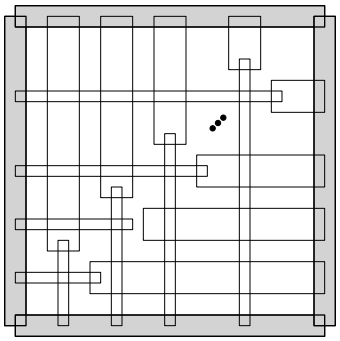


Figure 5-2: The family $\mathcal{C}_n$ contains $4n+4$ rectangles: n fat vertical, n thin vertical, n fat horizontal, n thin horizontal and 4 rectangles on the outside.

these quantities is also of relevance. An important related conjecture attributed to G. Wegner[1] is the following. Is it always the case that

$$\text{mis}(\mathcal{C}) \leq 2 \cdot \text{mhs}(\mathcal{C}) - 1 \ ? \tag{5.16}$$

Note that this conjecture is trivially tight by letting $\mathcal{C}$ be a single rectangle, where $\text{mhs}(\mathcal{C}) = \text{mis}(\mathcal{C}) = 1$. Gyárfás and Lehel [79] relaxed this conjecture to the following. Is there a constant c such that

$$\text{mis}(\mathcal{C}) \leq c \cdot \text{mhs}(\mathcal{C}) \ ? \tag{5.17}$$

Their conjecture is also still open. An important question is then: how large can the ratio $\text{mhs}(\mathcal{C})/\text{mis}(\mathcal{C})$ be?

Gyárfás and Lehel [79] give a simple example for which $\text{mhs}(\mathcal{C})/\text{mis}(\mathcal{C}) \geq 3/2$. Fon-Der-Flaass and Kostochka [58] improve this lower bound to $5/3$. In an unpublished manuscript, Cibulka et al. [34] give a family $\{\mathcal{C}_n\}_{n\geq 3}$ of examples, where $\text{mis}(\mathcal{C}_n) = n+2$ and $\text{mhs}(\mathcal{C}_n) = 2n+2$, from which

$$\lim_{n\to\infty} \frac{\text{mhs}(\mathcal{C}_n)}{\text{mis}(\mathcal{C}_n)} = 2. \tag{5.18}$$

The examples achieving this bound are depicted in Figure 5-2. In particular, this example shows that if Gyárfás and Lehel's conjecture holds, then the constant c in (5.17) is at least 2.

On the upper bound side, many authors (e.g. Károly and Tardos [98], Fon-Der-Flaas and Kostochka [58]) have shown that

$$\frac{\text{mhs}(\mathcal{C})}{\text{mis}(\mathcal{C})} \leq O(\log(\text{mis}(\mathcal{C}))). \tag{5.19}$$

[1] In the original text by Wegner [167], he actually asked whether for the case where $\text{mis}(\mathcal{C}) \geq 2$, $\text{mis}(\mathcal{C}) \leq 2\text{mhs}(\mathcal{C}) - 3$ or not.

We use this opportunity to give an improvement on this bound[2].

Theorem 5.1.6.

$$\frac{\mathrm{mhs}(\mathcal{C})}{\mathrm{mis}(\mathcal{C})} \leq O\Big(\log^2 \log(\mathrm{mis}(\mathcal{C}))\Big). \tag{5.20}$$

In order to prove this bound, we need two recent approximation algorithms for $\mathrm{mis}(\mathcal{C})$ and $\mathrm{mhs}(\mathcal{C})$.

Chalermsook and Chuzhoy [26] have developed an $O(\log\log m)$-approximation algorithm for the maximum independent set of a family of m rectangles. More precisely, they have shown that for any family of rectangles $\mathcal{C}$ having their corners in a grid $[t]^2$, it is possible to find an independent set $\mathcal{K}$ with

$$\mathrm{mis}(\mathcal{C}) \leq \mathrm{LP}_{[t]^2}(\mathcal{C}) \leq |\mathcal{K}| O(\log\log(t)) \tag{5.21}$$

On the other hand, Aronov et al. [4] have shown the existence of $O(\frac{1}{\varepsilon}\log\log\frac{1}{\varepsilon})$-nets for families of axis-parallel rectangles. This implies, using the approach of Brönnimann and Goodrich [18], that for every family $\mathcal{C}$ of rectangles, there exist a (polynomial time computable) hitting set H, with

$$|H| \leq O(\mathrm{LP}(\mathcal{C}) \log\log(\mathrm{LP}(\mathcal{C}))). \tag{5.22}$$

To prove Theorem 5.1.6, we require the following lemma.

Lemma 5.1.7. *For every family of rectangles $\mathcal{C}$, with $\alpha = \mathrm{mis}(\mathcal{C})$, there is another family $\mathcal{C}'$ of rectangles with corners in the grid $[\alpha]^2$ such that*

$$\mathrm{mis}(\mathcal{C}') \leq \mathrm{mis}(\mathcal{C}) \leq \mathrm{LP}(\mathcal{C}) \leq 9\mathrm{LP}(\mathcal{C}'). \tag{5.23}$$

Proof. Let $\mathcal{C}_x$ (resp. $\mathcal{C}_y$) be the family of intervals obtained by projecting $\mathcal{C}$ on the x-axis (resp. y-axis). Let H_x (resp. H_y) be a minimum hitting set for $\mathcal{C}_x$ (resp. $\mathcal{C}_y$). The intersection graph of $\mathcal{C}_x$ is an interval graph, and therefore, a perfect graph. Hence,

$$|H_x| = \mathrm{mhs}(\mathcal{C}_x) = \mathrm{mis}(\mathcal{C}_x) \leq \mathrm{mis}(\mathcal{C}) = \alpha, \tag{5.24}$$

and similarly $|H_y| \leq \alpha$.

Consider the grid $H_x \times H_y$ of size $\leq \alpha \times \alpha$. By translating and piece-wise scaling the plane, we can identify H_x with the set $\{(i,0) : 1 \leq i \leq |H_x|\}$ and H_y with the set $\{(0,j) : 1 \leq j \leq |H_y|\}$ without changing the intersection graph associated to $\mathcal{C}$. Thus, we can identify the grid $H_x \times H_y$ with a subgrid of $[\alpha] \times [\alpha]$. Note that this grid is itself, a hitting set of $\mathcal{C}$.

Furthermore, consider the family $\widetilde{\mathcal{C}} = \{R \cap [1,\alpha] \times [1,\alpha] \colon R \in \mathcal{C}\}$. This is, $\widetilde{\mathcal{C}}$ is obtained by trimming the rectangles to the rectangular region $[1,\alpha]\times[1,\alpha]$. It is easy to see that this operation does not change the intersection graph of the family. So, for our purposes, we can assume w.l.o.g. that $\mathcal{C} = \widetilde{\mathcal{C}}$.

[2]Even though this observation is an almost straightforward application of [26] and [4], as far as we know, it has not been yet published in any medium.

Let $\mathcal{C}'$ be the family of rectangles obtained by "growing" each rectangle of $\mathcal{C}$ in such a way that every corner of it is on a vertex of the grid. This is, we replace $R = \Gamma(a, b)$ by $R_+ = \Gamma\Big((\lfloor a_x \rfloor, \lfloor a_y \rfloor), (\lceil b_x \rceil, \lceil b_y \rceil)\Big)$.

The first inequality of (5.23) follows since any independent set of $\mathcal{C}'$ induces an independent set of $\mathcal{C}$ of the same size. The second inequality follows from (5.12). The only non-trivial inequality is the last one.

Since $[\alpha]^2$ is a hitting set for $\mathcal{C}$ and $\mathcal{C}'$, $\mathrm{LP}_{[\alpha]^2}(\mathcal{C}) = \mathrm{LP}(\mathcal{C})$ and $\mathrm{LP}_{[\alpha]^2}(\mathcal{C}') = \mathrm{LP}(\mathcal{C})$. Consider a fractional optimal solution y' for $\mathrm{LP}'_{[\alpha]^2}(\mathcal{C}')$ (see (5.11)) and recall that the support of y' is contained in $[\alpha]^2$. Observe that if p is a point in the support of y that fractionally hits some grown rectangle R_+, then either p, one of its 4 immediate neighbors in the grid or one of its 4 diagonal neighbors in the grid will hit the original rectangle R. Define y as

$$y_q = y'_q + \sum_{p \in [\alpha]^2 \colon p \text{ immediate or diagonal neighbor of } q} y'_p, \text{ for all } q \in [\alpha]^2. \tag{5.25}$$

By the previous observation, y is a fractional feasible solution for the dual of $\mathrm{LP}_{[\alpha]^2}(\mathcal{C})$, and by definition, its value is at most 9 times the value of y'. □

Now we are ready to prove Theorem 5.1.6.

Proof of Theorem 5.1.6. Let $\mathcal{C}$ be a family of rectangles with $\mathrm{mis}(\mathcal{C}) = \alpha$ and $\mathcal{C}'$ the family guaranteed by Lemma 5.1.7. Then, by combining (5.22) and (5.21), we have:

$$\begin{aligned}
\mathrm{mhs}(\mathcal{C}) &\leq O(\mathrm{LP}(\mathcal{C}) \log\log(\mathrm{LP}(\mathcal{C}))) \leq O(\mathrm{LP}(\mathcal{C}') \log\log(\mathrm{LP}(\mathcal{C}'))) \\
&= O(\mathrm{LP}_{[\alpha]^2}(\mathcal{C}') \log\log(\mathrm{LP}_{[\alpha]^2}(\mathcal{C}'))) \\
&\leq O(\alpha \log\log(\alpha) \log\log(\alpha \log\log(\alpha))) = O(\alpha(\log\log(\alpha))^2).
\end{aligned}$$ □

5.2 Geometric Interpretation for 2DORGs

In this section we establish a geometric interpretation of maximum cross-free matchings and minimum biclique covers of 2DORGs as maximum independent sets and minimum hitting sets of certain families of rectangles in the plane.

Consider a 2DORG G with bicolored $2D$-representation (A, B). For every edge ab in G, the rectangle $\Gamma(a, b)$ is nonempty. We denote by $\mathcal{R}(A, B)$ the set of all nonempty rectangles $\Gamma(a, b)$ with $a \in A$ and $b \in B$.

We remark that since A and B are maybe multisets, some of the rectangles in $\mathcal{R}(A, B)$ could be equal. Also, since A and B are not necessarily disjoint, some of the rectangles in $\mathcal{R}(A, B)$ may consist of a single point in the plane. These degeneracies do not happen if (A, B) is a bicolored rook representation: In this case, all the rectangles in $\mathcal{R}(A, B)$ are distinct, they have corners in a fixed grid and have nonempty interior.

In what follows, we use $G(A, B, \mathcal{R})$ to represent the 2DORG G having bicolored $2D$-representation (A, B) and such that $\mathcal{R} = \mathcal{R}(A, B)$. We do not impose (A, B) to be a bicolored rook representation.

The following theorem characterizes crossing edges of $G(A, B, \mathcal{R})$ in terms of $\mathcal{R}$ (see Figure 5-3.)

Theorem 5.2.1 (Soto and Telha [153]). *Two edges ab and $a'b'$ of $G(A, B, \mathcal{R})$ cross if and only if $\Gamma(a, b)$ and $\Gamma(a', b')$ intersect as rectangles.*

Proof. Let $R = \Gamma(a, b)$ and $R' = \Gamma(a', b')$ be distinct rectangles in $\mathcal{R}$. The edges ab and $a'b'$ cross if and only if $\Gamma(a, b')$ and $\Gamma(a', b)$ are also in $\mathcal{R}(A, B)$. This is equivalent to $\max(a_x, a'_x) \leq \min(b_x, b'_x)$ and $\max(a_y, a'_y) \leq \min(b_y, b'_y)$. From here, if ab and $a'b'$ cross, then the point $p = (\max(a_x, a'_x), \max(a_y, a'_y))$ is in the intersection of R and R' as rectangles. Conversely, if there is a point $p \in R \cap R'$, then $\max(a_x, a'_x) \leq p_x \leq \min(b_x, b'_x)$ and $\max(a_y, a'_y) \leq p_y \leq \min(b_y, b'_y)$, implying that ab and $a'b'$ cross. □

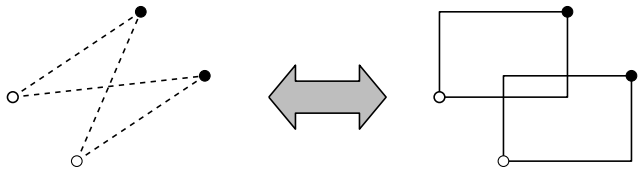


Figure 5-3: For 2DORGs, crossing edges are equivalent to intersecting rectangles

Recall the definition of the crossing graph of a graph G (see (4.10)). This is the graph $X(G)$ on the edges of G, where two elements are adjacent if they cross. Theorem 5.2.1 states that for a 2DORG $G = G(A, B, \mathcal{R})$ the intersection graph $\mathcal{I}(\mathcal{R})$ and the crossing graph $X(G)$ are isomorphic.

As a consequence of this, all the properties we explored in Section 5.1 about independent sets and hitting sets of rectangles translate to $X(G)$. For example, as shown in Figure 5-4, we can associate to every maximal biclique of G a unique witness point in the plane hitting all the corresponding rectangles, and to every point in the plane we can associate a (possibly empty) biclique of G.

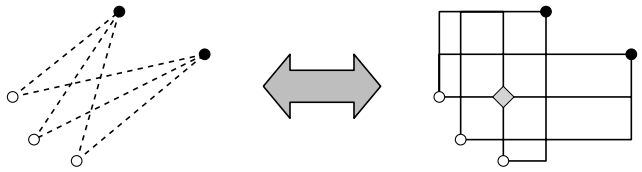


Figure 5-4: For 2DORGs, maximal bicliques corresponds to witness points.

More importantly, we obtain immediately the following theorem.

Theorem 5.2.2 (Soto and Telha [153]). *For a 2DORG $G = G(A, B, \mathcal{R})$,*

$$\alpha^*(G) = \text{mis}(\mathcal{R}) = \text{mis}(\mathcal{R}_\downarrow) \quad \textit{and} \quad \kappa^*(G) = \text{mhs}(\mathcal{R}) = \text{mhs}(\mathcal{R}_\downarrow).$$

5.3 Linear Programming Formulation

In this section, we use the linear programming relaxation for the maximum independent set of rectangles (see Section 5.1) to obtain a polynomial time algorithm for the maximum cross-free matching of 2DORGs.

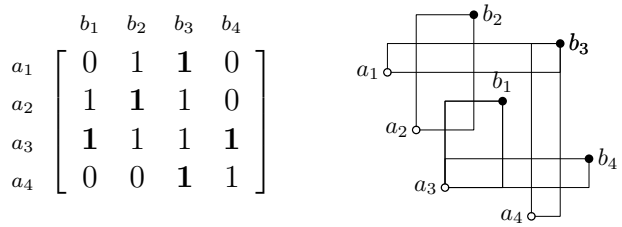
If $G = G(A, B, \mathcal{R})$ is a 2DORG such that $\mathcal{I}(\mathcal{R})$ (or $\mathcal{I}(\mathcal{R}_\downarrow)$) is perfect, then, by Theorem 5.2.2 and Lemma 5.1.3,

$$\alpha^*(G) = \kappa^*(G), \tag{5.26}$$

and solving the linear program $\mathrm{LP}(\mathcal{R})$ (or $\mathrm{LP}(\mathcal{R}_\downarrow)$) gives a polynomial time algorithm for finding a maximum cross-free matching. As observed in Section 5.1, we can find minimum clique-covers of perfect graphs in polynomial time. Therefore, we can also obtain a minimum biclique cover of G.

Not every 2DORG $G = G(A, B, \mathcal{R})$ is such that $\mathcal{I}(\mathcal{R})$ or $\mathcal{I}(\mathcal{R}_\downarrow)$ is a perfect graph. However, this holds for some subclasses. For a bipartite permutation graph G, the graph $X(G)$ is known to be perfect since it is both weakly chordal [122] and co-comparability [16]. Since $X(G)$ corresponds to $\mathcal{I}(\mathcal{R})$ for any representation of $G = G(A, B, \mathcal{R})$ as a 2DORG, we can find using the previous observation a maximum cross-free matching and minimum biclique cover for bipartite permutation graphs in polynomial time. As mentioned in Section 4.1.1, linear-time algorithms have been developed for both problems [156, 16, 53]. These algorithms use other properties of the structure of bipartite permutation graphs.

Let $G = G(A, B, \mathcal{R})$ be a biconvex graph. As Figure 5-5 shows, the graph $X(G) = \mathcal{I}(\mathcal{R})$ is not necessarily perfect.



(a) biadjacency matrix (b) 5-hole

Figure 5-5: A biadjacency matrix and a bicolored $2D$-representation of a biconvex graph. The bold entries form a 5-hole. This 5-hole is represented as a 5-cycle of rectangles in the right.

However, for biconvex graphs we can show the following result.

Theorem 5.3.1. *For every biconvex graph G, there is a representation as a* 2DORG *$G = G(A, B, \mathcal{R})$ for which $\mathcal{I}(\mathcal{R}_\downarrow)$ is perfect.*

Proof. Consider a graph G with biconvex adjacency matrix M. Let $\{a_1, \ldots, a_s\}$ and $\{b_1, \ldots, b_t\}$ be the vertices of both parts of G, in the order induced by M. Consider

each element $b_j \in B$ as an interval $I_j = [\ell(j), r(j)]$ of elements in A, this is $\ell(j)$ and $r(j)$ are the minimum and maximum indices i for which $M_{ij} = 1$.

It is easy to check that there are no four indices j_1, j_2, j_3, j_4 for which the left and right extremes of the first three intervals are strictly increasing,

$$\ell(j_1) < \ell(j_2) < \ell(j_3),\ r(j_1) < r(j_2) < r(j_3), \tag{5.27}$$

and simultaneously, the second and fourth interval satisfy

$$\ell(j_4) < \ell(j_2),\ r(j_2) < r(j_4). \tag{5.28}$$

This follows from the fact that no matter how j_1, j_2, j_3 and j_4 are sorted in the labeling of B, there is always an element $a \in A$ for which the corresponding ones in the associated row of M, restricted to the previous four columns, are not consecutive.

We use the previous property to prove the lemma. Recall that the graph G admits a bicolored representation (A, B), with A on the line $y = -x$ and B weakly above this line. This representation is obtained by the identification

$$a_i = (i, -i), \tag{5.29}$$
$$b_i = (r_i, -l_i). \tag{5.30}$$

We claim that the family of rectangles $\mathcal{R} = \mathcal{R}(A, B)$ obtained from the above representation is such that the intersection graph of $\mathcal{R}_\downarrow$ is perfect. We show this using the Strong Perfect Graph Theorem [32].

Suppose there is a hole $\mathcal{H} = \{R_1, R_2, \ldots, R_k\} \subseteq \mathcal{R}_\downarrow$ of $\mathcal{I}(\mathcal{R}_\downarrow)$ with $k \geq 5$, where R_ℓ intersects $R_{\ell-1}$ and $R_{\ell+1}$ (mod k), and $R_\ell = \Gamma(a_{i_\ell}, b_{j_\ell})$.

Assume that R_1 is the rectangle in $\mathcal{H}$ having i_1 as small as possible. It is easy to check that $i_1 \neq i_2$, since otherwise any rectangle in $\mathcal{H}$ intersecting the thinnest[3] of R_1 and R_2 must intersect the other one. By a similar argument, $i_1 \neq i_k$ and $i_k \neq i_{k-1}$. Also, since R_2 and R_k do not intersect, $i_2 \neq i_k$. Therefore, without loss of generality we can assume that $i_1 < i_2 < i_k$. For the rest of the argument, we use Figure 5-6. We use λ_1 to denote the vertical line $x = (a_{i_k})_x = i_k$, λ_2 to denote the horizontal line $y = (a_{i_1})_y = -i_1$, Z_1 to denote the triangular zone bounded by λ_1, λ_2 and the line $y = -x$, Z_2 to denote the rectangular area strictly above λ_2 and strictly to the left of λ_1, and Z_3 to denote the rectangular area strictly below λ_2 and strictly to the right of λ_1. Note that the point where λ_1 and λ_2 intersect is in $R_1 \cap R_k$.

First, we claim that $i_\ell < i_k$, for all $\ell \neq k$. Assume, for contradiction, that $i_k < i_\ell$ for some ℓ, meaning that a_{i_ℓ} is outside the triangular zone Z_1 in Figure 5-6. Since $\mathcal{H}$ is a hole, it is possible to draw a continuous curve in the plane going from the point $a_{i_2} \in R_2$ to the point $a_{i_\ell} \in R_\ell$ without ever touching rectangles R_1 and R_k. But this is impossible since to leave Z_1 we must touch either λ_1 or λ_2. Here, we have used that $Z_1 \cap \lambda_1 \subseteq R_k$ and $Z_1 \cap \lambda_2 \subseteq R_1$.

Recall that R_2 intersects R_1 and that R_{k-1} intersects R_k. From here it is easy to see that $i_2 < i_{k-1}$ (as in the picture) as otherwise R_2 and R_{k-1} would intersect,

[3]The one having smallest length in the x-direction.

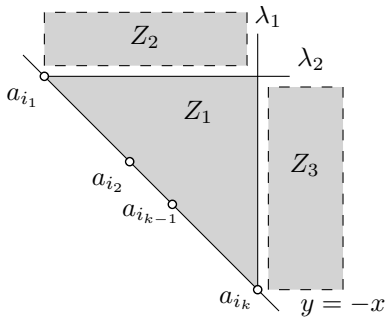


Figure 5-6: Proof for no holes.

contradicting the fact that $\mathcal{H}$ is a hole with $k \geq 5$.

Now we can deduce where the corners in B are located.

Both b_{i_1} and b_{i_k} are weakly above λ_2 and to the right of λ_1. Since R_2 intersects R_1 but not R_k, the corner b_{i_2} of R_2 must be located strictly to the left of the line λ_1, as otherwise R_2 would intersect R_k. The corner b_{i_2} must also be strictly above the line λ_2: It can not be strictly below λ_2 as otherwise R_1 and R_2 would not intersect, and it can not be on the line λ_2 since then $b_{i_2} \in R_1$, contradicting the inclusion-wise minimality of R_1. This means that b_{i_2} is in zone Z_2 of Figure 5-6. By an analogous argument, the corner $b_{i_{k-1}}$ of R_{k-1} lies in zone Z_3. Also, since all the points $\{a_{i_1}, \ldots, a_{i_k}\}$ are in the zone Z_1 and the only rectangles intersecting λ_1 or λ_2 are R_{k-1}, R_k, R_1 and R_2, we conclude that the points $\{b_{i_3}, \ldots, b_{i_{k-2}}\}$ are also in Z_1.

Note that at least one of the following holds:

1. $b_{i_3} \in Z_1$ is located above $b_{i_{k-1}}$.

2. $b_{i_{k-2}} \in Z_1$ is located to the right of $b_{i_{k-1}}$.

If neither does, it is not hard to show that R_2 and R_{k-1} intersect. Set j_2 equal to i_3 if the first condition holds and equal to i_{k_2} if the second condition holds. Also set $j_1 = i_2$, $j_3 = i_{k-1}$ and $j_4 = i_1$. Then the sequence (j_1, j_2, j_3, j_4) satisfies (5.27) and (5.28), contradicting that G is biconvex.

We have shown that $\mathcal{I}(\mathcal{R}_\downarrow)$ contains no odd-hole of size ≥ 5. Now we focus on antiholes. Suppose that $\mathcal{A} = \{R_1, R_2, \ldots, R_k\}$ is an antihole of $\mathcal{I}(\mathcal{R}_\downarrow)$ of length at least 7, where R_j intersects every rectangle except $R_{j-1 \pmod k}$, and $R_{j+1 \pmod k}$. Let R_1 and R_m be the rectangles in $\mathcal{A}$ with minimum and maximum value of its lower left corner (i_1 and i_m respectively). As in the case of holes, we can check that R_1 and R_j are the only rectangles with corners at a_{i_1} and a_{i_m} respectively.

We now consider two cases, depending on whether R_1 and R_m intersect.

If R_1 and R_m have empty intersection, then $m = 2$ or $m = k$. For both values of m, R_4 and R_5 intersect both R_1 and R_m (since the antihole has length $k \geq 7$). This implies that R_4 and R_5 must contain the point p where the line through the bottom side of R_1 and the line through the left side of R_m intersect (see Figure 5-7), which contradicts the antihole definition.

The second case is if R_1 and R_m intersect (see Figure 5-8). Since the length of the antihole is $k \geq 7$, there is $j \in [k]$ such that $\{1, m\} \cap \{j-1, j, j+1\}_{\bmod k} = \emptyset$. This

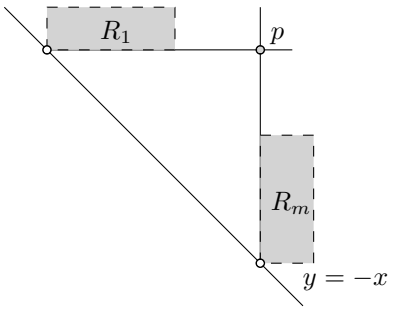


Figure 5-7: First case of the proof for no antiholes.

means that R_j intersects both R_1 and R_m. Consider the rectangles R_{j-1} and R_{j+1}, since they do not intersect R_j, each one must be completely contained in one of the semi-infinite zones Z_1 or Z_2. Furthermore, since R_{j-1} and R_{j+1} do intersect, they both lie in the same zone. But this implies that one of R_1 and R_m is not intersected by both R_{j-1} and R_{j+1}, which is a contradiction with the definition of the antihole.

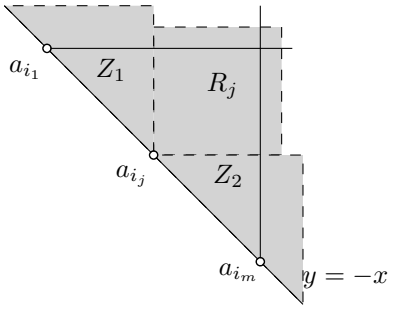


Figure 5-8: Second case of the proof for no antiholes.

We have shown that $\mathcal{I}(\mathcal{R}_\downarrow)$ contains no hole of size ≥ 5 and no odd antihole of size ≥ 7 (size 5 is covered since antiholes and holes of size 5 coincide). By the Strong Perfect Graph Theorem [32], we conclude that $\mathcal{I}(\mathcal{R}_\downarrow)$ is perfect. □

Let us go back to the general case of 2DORGs. Recall that any such graph admits a bicolored rook representation (by Lemmas 3.4.1 and 3.4.9). In what follows let $G = G(A, B, \mathcal{R})$ where (A, B) is a bicolored rook representation. In this case no two points of $A \cup B$ share the same position and every rectangle has full-dimensional interior.

An elementary, but important observation is the following.

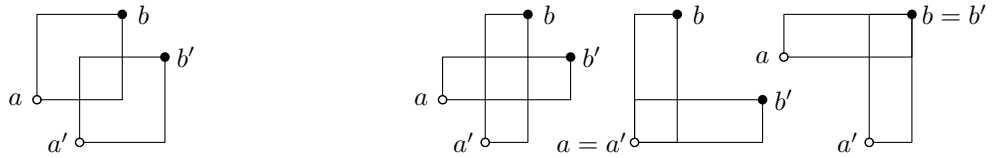
Remark 5.3.2. If $R = \Gamma(a, b)$ is an inclusion-wise minimal rectangle in $\mathcal{R}_\downarrow$, then R does not contain any point in $(A \cup B) \setminus \{a, b\}$.

This remark holds since the existence of a point $c \in R \cap ((A \cup B) \setminus \{a, b\})$ implies that either $\Gamma(a, c)$ or $\Gamma(c, b)$ is a rectangle in $\mathcal{R}$ completely contained in R, contradicting its minimality.

By Remark 5.3.2, there are only four ways in which a pair of rectangles of $\mathcal{R}_\downarrow$ can intersect. They are shown in Figure 5-9. We say that two intersecting rectangles have

corner-intersection if one rectangle contains a corner of the other in its topological interior. If both are in $\mathcal{R}_\downarrow$, the only way this can happen is that one rectangle contains the top-left corner of the other, while the other contains the bottom-right corner of the first.

If the previous case does not happen, we say that the intersection is **corner-free**. A **corner-free-intersection (c.f.i.)** family is a collection of inclusion-wise minimal rectangles having no corner-intersections.



(a) corner-intersection (b) corner-free intersections

Figure 5-9: The only ways two rectangles in $\mathcal{R}_\downarrow$ can intersect each other.

Note that c.f.i. families are *skew-intersecting families*. Therefore, by Lemma 3.4.9, if $\mathcal{R}_\downarrow$ has no pair of corner-intersecting rectangles, $\mathcal{I}(\mathcal{R}_\downarrow)$ is perfect.

Let z^* be the optimal value of $\mathrm{LP}(\mathcal{R}_\downarrow)$ and for every rectangle $R \in \mathcal{R}_\downarrow$ let $\mu(R) > 0$ be its area[4]. Let $\bar{x}$ be an *optimal extreme point* of the following linear program

$$\mathrm{LP}(z^*, \mathcal{R}_\downarrow) = \min \left\{ \sum_{R \in \mathcal{R}_\downarrow} \mu(R) x_R \colon \sum_{R \in \mathcal{R}_\downarrow} x_R = z^* \text{ and } x \in \mathrm{P}(\mathcal{R}_\downarrow) \right\}.$$

Note that $\bar{x}$ is a solution to $\mathrm{LP}(\mathcal{R}_\downarrow)$ minimizing the total **weighted area** covered.

It is interesting to note that $\bar{x}$ is also an optimal extreme point of the linear program

$$\max \left\{ \sum_{R \in \mathcal{R}_\downarrow} (1 - \varepsilon \mu(R)) x_r \colon x \in \mathrm{P}(\mathcal{R}_\downarrow) \right\},$$

for sufficiently small values of ε. In particular, this means we can find $\bar{x}$ by solving only one linear program.

Theorem 5.3.3. *The point $\bar{x}$ is an integral point.*

Proof. Suppose that $\bar{x}$ is not integral. Let $\overline{\mathcal{R}} = \{R \in \mathcal{R}_\downarrow \colon \bar{x}_R > 0\}$ be the set of rectangles in the support of $\bar{x}$. We claim that $\overline{\mathcal{R}}$ is a c.f.i. family. Assume for sake of contradiction that $R = \Gamma(a, b)$ and $R' = \Gamma(a', b')$ are two rectangles in $\overline{\mathcal{R}}$ having corner-intersection as in Figure 5-9a. We apply the following **uncrossing procedure**: let $\varepsilon = \min(\bar{x}_R, \bar{x}_{R'}) > 0$ and consider the rectangles $S = \Gamma(a, b')$ and $S' = \Gamma(a', b)$ in $\mathcal{R}_\downarrow$. Consider the vector $\hat{x}$ obtained from $\bar{x}$ by decreasing x_R and $x_{R'}$ by ε and increasing by the same amount x_S and $x_{S'}$. It is easy to see that $\hat{x}$ is a feasible

[4]For our discussion, the *area* or a rectangle $R = \Gamma(a, b)$ is defined as $(b_x - a_x)(b_y - a_y)$. However, our techniques also works if we define the *area* of a rectangle as the number of grid points it contains.

solution of $\mathrm{LP}(z^*, \mathcal{R}_\downarrow)$ with strictly smaller weighted area than $\bar{x}$, contradicting its optimality.

Since $\overline{\mathcal{R}}$ is a c.f.i. family, $\mathcal{I}(\overline{\mathcal{R}})$ is a perfect graph. Therefore, $\mathrm{P}(\overline{\mathcal{R}})$ is an integral polytope. Consider now the set

$$F = \mathrm{P}(\overline{\mathcal{R}}) \cap \left\{ x \in \mathbb{R}^{\mathcal{R}_\downarrow} : \sum_{R \in \overline{\mathcal{R}}} x_R = z^*, \sum_{R \in \overline{\mathcal{R}}} \mu(R) x_R = \sum_{R \in \overline{\mathcal{R}}} \mu(R) \bar{x}_R \right\}. \tag{5.31}$$

This set is a face of $\mathrm{P}(\overline{\mathcal{R}})$ containing only optimum solutions of $\mathrm{LP}'(z^*, \mathcal{R}_\downarrow)$. To conclude the proof we show that $\bar{x}$ is a vertex of F, and therefore, a vertex of $\mathrm{P}(\overline{\mathcal{R}})$. If $\bar{x}$ is not a vertex of F, we can find two different points in F such that $\bar{x}$ is a convex combination of those points. This means that $\bar{x}$ is a convex combination of different optimum solutions of $\mathrm{LP}'(z^*, \mathcal{R}_\downarrow)$, contradicting the choice of $\bar{x}$. □

We can find a maximum cross-free matching of $G = G(A, B, \mathcal{R})$ where (A, B) is a bicolored rook representation, using Theorem 5.3.3 as follows: Find the optimal value z^* of the linear program $\mathrm{LP}_{[n]^2}(\mathcal{R}_\downarrow) = \mathrm{LP}(\mathcal{R}_\downarrow)$, where $n = |A \cup B|$ and then find an optimal extreme point of $\mathrm{LP}(z^*, \mathcal{R}_\downarrow)$.

Using that 2DORGs are chordal-bipartite graphs, for which the maximum cross-free matching and the jump number problems are equivalent, we conclude the following.

Theorem 5.3.4. *The maximum cross-free matching and, equivalently, the jump number of a* 2DORG *can be computed in polynomial time.*

In this section we have shown not only that the value of the linear program $\mathrm{LP}(\mathcal{R}_\downarrow)$ equals $\mathrm{mis}(\mathcal{R}_\downarrow)$ but also that an integer optimal solution of $\mathrm{LP}(\mathcal{R}_\downarrow)$ can be found by optimizing an easy to describe linear function over the optimal face of this linear program. In the next section, we give an algorithmic proof that $\mathrm{mis}(\mathcal{R}_\downarrow) = \mathrm{mhs}(\mathcal{R}_\downarrow)$, implying that the dual linear program $\mathrm{LP}'_{[n]^2}(\mathcal{R}_\downarrow)$ (see (5.11)) also admits an integral vertex. We conjecture that there is also a simple direction such that optimizing on this direction over the optimal face of $\mathrm{LP}'_{[n]^2}(\mathcal{R}_\downarrow)$ yields the integral solution.

5.4 Combinatorial Algorithm

In this section we give a combinatorial algorithm that computes simultaneously a maximum cross-free matching and a minimum biclique cover of a 2DORG. Our procedure shares ideas with the algorithmic proof of Györi's min-max result of intervals [80] given by Frank [62]. In order to keep the discussion self-contained, we do not rely on that result and prove every step.

In Section 5.3 we have shown that a maximum cross-free matching of a 2DORG $G = (A, B, \mathcal{R})$ can be obtained from a maximum independent set of a c.f.i. family $\overline{\mathcal{R}}$ (as a matter of fact, we have shown that $\overline{\mathcal{R}}$ itself is an independent set). In what follows, we show that this result also holds if we replace $\overline{\mathcal{R}}$ by a certain *maximal greedy* c.f.i. subfamily of $\mathcal{R}_\downarrow$.

Consider a 2DORG $G = (A, B, \mathcal{R})$ where (A, B) is a bicolored rook representation. In particular, the rectangles are in the grid $[n]^2$, with $n = |A \cup B|$. We say that a rectangle $R \in \mathcal{R}_\downarrow$ appears before a rectangle $S \in \mathcal{R}_\downarrow$ in **right-top order** if either

1. $\mathrm{bl}(R)_x < \mathrm{bl}(S)_x$, or
2. $\mathrm{bl}(R)_x = \mathrm{bl}(S)_x$ and $\mathrm{tr}(R)_y < \mathrm{tr}(S)_y$,

where, as we recall, $\mathrm{bl}(\cdot)$ stands for the bottom left corner (the corner in A) and $\mathrm{tr}(\cdot)$ stands for the top right corner (the corner in B). The right-top order is a total order on $\mathcal{R}_\downarrow$.

Construct a family $\mathcal{K}$ by processing the rectangles in $\mathcal{R}_\downarrow$ in right-top order and adding only those that keep $\mathcal{K}$ corner-free.

Since $\mathcal{I}(\mathcal{K})$ is the comparability graph of $(\mathcal{K}, \preceq)$, where $\preceq$ is defined as in (5.14), the size of a maximum independent set $\mathcal{R}_0$ of $\mathcal{K}$ equals the size of a minimum hitting set H_0 for $\mathcal{K}$. We can find optimal solutions of these problems by computing a maximum antichain and a minimum chain-cover of the poset $(\mathcal{K}, \preceq)$, using any polynomial time algorithm for the Dilworth chain-partitioning problem (see, for example, Lemma 3.2.3). We modify H_0 to obtain a set of points H^* of the same size hitting all $\mathcal{R}$.

An **admissible flip** of a hitting set H for $\mathcal{K}$ is an ordered pair of points p and q in H with $p_x < q_x$ and $p_y < q_y$, such that the set $H \setminus \{p, q\} \cup \{(p_x, q_y), (q_x, p_y)\}$ obtained by **flipping p and q** is still a hitting set for $\mathcal{K}$.

Construct H^* from H_0 by flipping admissible flips while this is possible. Since we only use points of the grid $[n]^2$, flipping two points reduces the potential $\psi(H) = \sum_{p \in H} p_x p_y$ by at least one unit. Using that ψ is positive and $\psi(H_0) \leq |H_0| n^2 \leq n^3$, we do at most this many flips. Therefore, we can construct H^* in polynomial time.

To continue, we need some definitions and a technical lemma. We say that a rectangle $R \in \mathcal{R}_\downarrow \setminus \mathcal{K}$ **blames** a rectangle $S \in \mathcal{K}$ if S is the first rectangle (in right-top order) of $\mathcal{K}$ having corner-intersection with R. We denote this rectangle as $S = \mathrm{blame}(R)$. Also, given a set of points H, let $\mathrm{hit}(H, \mathcal{R}_\downarrow)$ be the set of rectangles in $\mathcal{R}_\downarrow$ hit by H.

Lemma 5.4.1. *Let H be a hitting set of $\mathcal{K}$ but not of $\mathcal{R}_\downarrow$ and let $R = \Gamma(a, b)$ be a rectangle of $\mathcal{R}_\downarrow \setminus \mathcal{K}$ that is not hit by H. Let also $S = \mathrm{blame}(R) = \Gamma(c, d) \in \mathcal{K}$. Then:*

C1. *The rectangle $U = \Gamma(c, b)$ is in $\mathcal{K}$.*

If furthermore, R is the last *rectangle of $\mathcal{R}_\downarrow \setminus \mathcal{K}$ in right-top order that is not hit by H then*

C2. *The rectangle $T = \Gamma(a, d) \in \mathcal{R}_\downarrow$ is hit by H.*

C3. *If p is any point of H hitting U and q is any point of H hitting T then (p, q) is an admissible flip for H.*

C4. *Let $H' = H \setminus \{p, q\} \cup \{(p_x, q_y), (p_y, q_x)\}$ be the set obtained by flipping (p, q) as above. Then $\mathrm{hit}(H, \mathcal{R}_\downarrow) \cup \{R\} \subseteq \mathrm{hit}(H', \mathcal{R}_\downarrow)$.*

Proof. Since $S = \Gamma(c,d)$ and $R = \Gamma(a,b)$ have corner intersection and S appears before R in top-right order, we have $c_x < a_x < d_x < b_x$ and $d_y > b_y > c_y > a_y$ (see Figure 5-10). In particular, the rectangles $T = \Gamma(a,d)$ and $U = \Gamma(c,b)$ are in $\mathcal{R}$. They are also inclusion-wise minimal, otherwise there would be a point of $A \cup B \setminus \{a,b,c,d\}$ in $T \cup U \subseteq R \cup S$, contradicting the minimality of R or S.

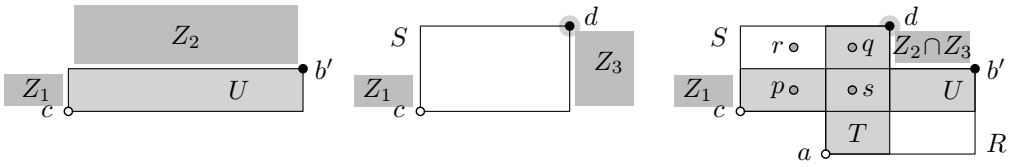


Figure 5-10: Positions of rectangles and zones defined in the proof of Lemma 5.4.2.

Assume by contradiction that the first claim does not hold. Then, there is a rectangle $U' = \Gamma(c',b') \in \mathcal{K}$ appearing before $U = \Gamma(c,b)$ and having corner-intersection with U. In particular the corner c' lies in the zone $Z_1 = (-\infty, c_x - 1] \times [c_y + 1, b_y - 1]$, and the corner b' lies in the zone $Z_2 = [c_x + 1, b_x - 1] \times [b_y + 1, \infty)$ as shown on the left of Figure 5-10.

Note that S and U' intersect since the top-left corner (c_x, b'_y) of U is in both rectangles. Since both S and U' are in $\mathcal{K}$, this intersection must be corner-free. Using that $c' \in Z_1$ we conclude that b' is either equal to d or it lies in the zone $Z_3 = [d_x + 1, \infty) \times [c_y + 1, d_y - 1]$. See the center of Figure 5-10.

We have $c' \in Z_1$ and $b' \in \{d\} \cup (Z_2 \cap Z_3)$ (see the right of Figure 5-10). Therefore, R and U' have corner-intersection contradicting the choice of S, since U' appears before S in right-top order. This completes the proof of the first claim. Note also that as $U \in \mathcal{K}$, it must be hit by a point $p \in H$.

In what follows assume that R is the *last* rectangle in $\mathcal{R}_\downarrow \setminus \mathcal{K}$ that is not hit by H. The second claim of the lemma follows by definition: since T appears after R in right-top order, it must be hit by a point $q \in H$.

Now we show the third and fourth claims. Since $p \in U \setminus R$ and $q \in T \setminus R$ we have $p_x < a_x \leq q_x$ and $p_y \leq b_y < q_y$. Let $r = (p_x, q_y)$, $s = (q_x, p_y)$ and suppose there is a rectangle $\hat{R} = \Gamma(\hat{a}, \hat{b}) \in \mathcal{R}_\downarrow$ hit by H but not by $H' = (H \setminus \{p,q\}) \cup \{r,s\}$. If $\hat{R}$ is hit by p (but not by r or s), then $\hat{b}$ must be in the region $[p_x, q_x - 1] \times [p_y, q_y - 1]$. In particular, $\hat{b} \in S \setminus \{c,d\}$. Therefore, $\Gamma(c, \hat{b})$ is a rectangle in $\mathcal{R}$ that contradicts the inclusion-wise minimality of S.

On the other hand, if $\hat{R}$ is hit by q (but not by r or s), then $\hat{a}$ must be in the region $[p_x + 1, q_x] \times [p_y + 1, q_y]$. As before, this means that $\hat{a} \in S \setminus \{c,d\}$, implying that $\Gamma(\hat{a}, d)$ is a rectangle in $\mathcal{R}$ contradicting the inclusion-wise minimality of S.

We have shown that $\text{hit}(H, \mathcal{R}_\downarrow) \subseteq \text{hit}(H', \mathcal{R}_\downarrow)$. Noting that R is also hit by $s \in H'$, we conclude the proof of the lemma. □

Let's go back to the definition of the hitting sets H_0 and H^* for $\mathcal{K}$. As a corollary of the previous lemma we have the following.

Lemma 5.4.2. *H^* is a hitting set for $\mathcal{R}_\downarrow$, and therefore, a hitting set for $\mathcal{R}$.*

Proof. If this is not the case then, by condition C3 in Lemma 5.4.1, H^* admits an admissible flip. This contradicts the construction of H^*. □

Using Lemma 5.4.2, we can find a maximum cross-free matching and a minimum biclique cover of a 2DORG $G = (A, B, \mathcal{R})$, where (A, B) is a bicolored rook representation, using the procedure depicted below as Algorithm 14.

Algorithm 14 to compute a maximum independent set and a minimum hitting set of $\mathcal{R}(A, B)$.

1: Compute $\mathcal{R}$, $\mathcal{R}_\downarrow$ and the c.f.i. greedy family $\mathcal{K}$.
2: Use an algorithm for the Dilworth chain-partitioning problem to compute a maximum independent set $\mathcal{R}_0$ and a minimum hitting set H_0 for $\mathcal{K}$.
3: Compute H^* from H_0 by flipping admissible flips while this is possible.
4: Return $\mathcal{R}_0$ and H^*.

Since H^* is a hitting set for $\mathcal{R}$, and $\mathcal{R}_0$ is an independent set of $\mathcal{R}$ with $|H^*| = |H_0| = |\mathcal{R}_0|$, we conclude they are both optima; therefore, they induce a maximum cross-free matching and a minimum biclique cover for G.

Theorem 5.4.3 (Soto and Telha [153])**.** *Algorithm 14 computes a maximum cross-free matching and a minimum biclique cover of the same size for any* 2DORG G *in polynomial time. In particular, this shows that* $\alpha^*(G) = \kappa^*(G)$.

It is necessary to remark that the ideas for this algorithm were already present in Frank's [62] algorithm for Györi's [80] problem. In fact, by applying Algorithm 14 to families of rectangles arising from convex graphs, we recover Frank's algorithm (with different notation). Benczúr et al. [12] have given an efficient implementation of Frank's algorithm. In the following section, we use the general lines of that implementation to obtain a faster version of Algorithm 14.

5.4.1 Running Time Improvement

Consider in this section a fixed 2DORG $G = (A, B, \mathcal{R})$ where (A, B) is a bicolored rook representation. We use $\mathcal{K}$ to denote the greedy c.f.i. subfamily of $\mathcal{R}_\downarrow$ obtained by processing the rectangles in $\mathcal{R}_\downarrow$ in right-top order, where a rectangle is added to $\mathcal{K}$ only if it does not have a corner intersection with previously added rectangles. Let $n = |A \cup B|$ be the number of vertices and $m = |\mathcal{R}|$ be the number of edges of G. Let $k = |\mathcal{K}|$ and ℓ be the number of vertices and of edges in the intersection graph $\mathcal{I}(\mathcal{K})$.

5.4.2 Overview

Consider the following naive implementation of Algorithm 14.

First step. Construct $\mathcal{R}, \mathcal{R}_\downarrow$ and $\mathcal{K}$: The family $\mathcal{R}$ can be constructed from (A, B) in $O(n^2)$. We can construct $\mathcal{R}_\downarrow$ naively in time $O(|\mathcal{R}|^2) = O(m^2)$ by checking every pair of rectangles. In order to add a rectangle to the family $\mathcal{K}$ we potentially have to check corner-intersection with all the previous rectangles. We can do this in time $O(|\mathcal{R}_\downarrow||\mathcal{K}|) = O(mk)$. The running time of this step is dominated by $O(m^2)$.

Second step. Construct $\mathcal{R}_0$ and H_0: Let $f(k, \ell)$ be the running time of an algorithm solving the Dilworth chain-partitioning problem on the poset $(\mathcal{K}, \preceq)$. By applying that algorithm, we recover a maximum antichain and a minimum chain partition of this poset. Obtaining $\mathcal{R}_0$ from the maximum antichain returned is direct, but to return the actual hitting set H_0 from the chain partition, we need to select for each chain, one point common to all the rectangles. Since each chain can have k rectangles, this can be done in $O(|H_0|k)$-time. The running time of this step is then dominated by $O(f(k, \ell) + nk)$.

Third step. Computing H^*: A very naive way to do this is the following. We can check if a pair of points is a flipping pair in time $O(|\mathcal{K}|) = O(k)$. Since there are $O(|H_0|^2) = O(n^2)$ possible flips, and the number of flips we require to do is bounded by n^3, the total running time of this step is $O(n^5 k)$.

By using the trivial bounds $m = O(n^2)$, we conclude that this implementation has running time $O(f(k, \ell) + n^5 k)$. The current best values for $f(k, \ell)$ are based on the deterministic algorithm of Hopcroft and Karp [86] for maximum bipartite matching, running in time $O(k+\ell\sqrt{k})$, and the randomized algorithm based on matrix multiplication of Mucha and Sankowski [120] which runs in randomized time $O(k^\omega)$ where $2 \leq \omega \leq 2.376$ is the best exponent for the matrix multiplication problem (See Lemma 3.2.3).

Using the bounds $\ell \leq k^2$ and $k \leq m \leq n^2$, we obtain that the total running time of this naive algorithm is dominated by $O(n^7)$. We can reduce vastly the running time by using the following strategy, which we describe in detail afterwards.

For the first step, we show how to construct an efficient data structure which allows us to construct $\mathcal{R}$, $\mathcal{R}_\downarrow$ and $\mathcal{K}$ in $O(n^2)$-time.

For the third step, we make a simple observation. To construct a hitting set for $\mathcal{R}_\downarrow$ starting from H_0 we do not need to perform every possible admissible flip: It is enough to perform flips for which the current set $\text{hit}(H, \mathcal{R}_\downarrow)$ of hit rectangles increases (one such flip is guaranteed by Lemma 5.4.1). As $|\mathcal{R}_\downarrow| \leq m \leq n^2$, we only need $O(n^2)$-flips. In fact, we can do better. Going over the list of rectangles in $\mathcal{R}_\downarrow$ in reverse top-right order, it is possible to find a sequence of at most k flips which yields a hitting set for $\mathcal{R}_\downarrow$. We modify the proposed data structure in order to find these flipping pairs efficiently. Finally, we show how to implement the flips, so that each one of them takes time $O(k \log\log n)$ by using van Emde Boas trees.

Using this approach, we get an $O(n^2 + (f(k, \ell) + nk) + (m + k^2 \log\log n))$ running time. To conclude, we give better bounds for k and ℓ in terms of n. Namely, we show that $\ell = O(n^2)$ and that $k = O(n \log n)$. Plugging in these bounds and the current

best bounds for $f(k,\ell)$, we get that our algorithm runs in $O(n^{5/2}\sqrt{\log n})$-deterministic time, or $O((n\log n)^\omega + (n\log n)^2 \log\log n)$-randomized time.

5.4.3 Data Structure

The following observation motivates a fast way to construct $\mathcal{R}_\downarrow$. For every point $a \in A$, let List(a) be the collection of points $v \in A \cup B \setminus \{a\}$ such that $\Gamma(a,v)$ is a nonempty rectangle not containing a point of $A \cup B \setminus \{a,v\}$. Observe that $b \in B$ is such that $\Gamma(a,b) \in \mathcal{R}_\downarrow$ if and only if $b \in \text{List}(a) \cap B$.

The points in List(a) form a staircase-shape as shown in Figure 5-11. They are therefore easy to construct by visiting all the points in $A \cup B \setminus \{a\}$ from bottom to top and remembering the leftmost visited one that is to the right of a.

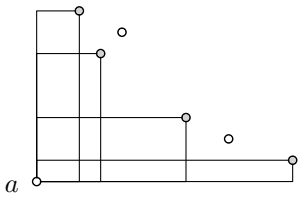


Figure 5-11: Staircase shape of List(a).

We now show a fast way to test if a rectangle in $\mathcal{R}_\downarrow$ is added to the greedy c.f.i. family $\mathcal{K}$. Consider the moment when a rectangle $R = \Gamma(a,b)$ is processed in the construction of $\mathcal{K}$. Let $L = \Gamma(a',b)$ be the *last* rectangle added to $\mathcal{K}$ before R is considered having the same top right corner as R, this is $\text{tr}(L) = b$. This rectangle may not exist, for instance in the case where R is the first rectangle considered having b as a corner. If L exists, we call it the **left-witness** of R, and we denote it as $L = \text{LW}(R)$.

If $L = \text{LW}(R) = \Gamma(a',b)$ exists, let $T = \Gamma(a',b')$ be the *first* rectangle in $\mathcal{K}$ added after L with same bottom left corner as L, this is $\text{bl}(T) = a'$. The rectangle T may not exist, for instance, if L is already the last rectangle with corner a'. If T exists we call it the **top-witness** of R, and we denote it as $T = \text{TW}(R)$.

Lemma 5.4.4. *The rectangle $R = \Gamma(a,b) \in \mathcal{R}_\downarrow$ is not added to $\mathcal{K}$ if and only if both rectangles $L = \text{LW}(R)$ and $T = \text{TW}(R)$ exist and the rectangle T has corner intersection with R.*

Proof. Sufficiency is direct, since the existence of a rectangle $T \in \mathcal{K}$ having corner intersection with R implies the latter rectangle is not added to $\mathcal{K}$.

To prove necessity, suppose that R is not added to $\mathcal{K}$. The rectangle R blames this to a rectangle $\text{blame}(R) = \Gamma(c,d)$ which is the first rectangle in $\mathcal{K}$ having corner intersection with R. By condition C1 of Lemma 5.4.1, $\Gamma(c,b)$ is a rectangle with right-top corner b, that is added to $\mathcal{K}$ before R is considered. This means that the left witness $L = \text{LW}(R) = \Gamma(a',b)$ exists (since it is the last rectangle added to $\mathcal{K}$ with the previous property). For the next part of the proof, see Figure 5-12 for reference.

Assume for sake of contradiction that the top witness $T = \text{TW}(R)$ does not exist or that T has no corner intersection with R. This is equivalent to saying that there

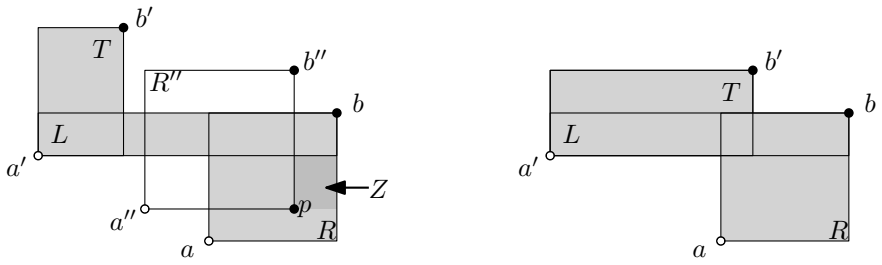


Figure 5-12: On the left, the situation if T has no corner intersection with R. On the right, the situation imposed by Lemma 5.4.4.

is no rectangle with bottom-left corner a' having corner intersection with R (if there is one, then T must also have corner-intersection with R).

Recall that R was not added to $\mathcal{K}$. This means that R must contain a *bottom-right corner* of a previous rectangle in $\mathcal{K}$ in its *topological interior* $\text{int}(R)$. Among all the possible such bottom-right corners, let $p = (p_x, p_y)$ be the one having larger x-coordinate, and, in case of ties, select the one with larger y-coordinate. Let $R'' = \Gamma(a'', b'') \in \mathcal{K}$ be the rectangle having p as its bottom-right corner (i.e. $p = (b''_x, a''_y)$).

Note that $p_y = a''_y$ must be strictly smaller than a'_y. Indeed, if $p_y > a'_y$, then p would be in the interior of L, contradicting the fact that L is in $\mathcal{K}$, and if $p_y = a'_y$ then R'' and L would have the same bottom-left corner a', contradicting our assumption for T. Also, we must have that $a'_x < a''_x$ since otherwise the point a' would be inside the rectangle R'', contradicting its minimality. See the left of Figure 5-12.

Let $U = \Gamma(a'', b) \in \mathcal{R}$. This rectangle is inclusion-wise minimal since otherwise, it would contain a point in $A \cup B \setminus \{a'', b\}$, contradicting the minimality of R or R''. The discussion of the previous paragraph implies that U appears after L in right-top order and has the same top-right corner b. We claim that U is also in $\mathcal{K}$. This would contradict the definition of L as left-witness and conclude the proof of the lemma.

Suppose that the claim does not hold. Then, U must contain the bottom-right corner q of a previous rectangle in $\mathcal{K}$ in its interior. The point q must be in $\text{int}(U) \setminus (\text{int}(L) \cup \text{int}(R''))$ as otherwise, L or R'' would not be in $\mathcal{K}$. This implies that q is in the region $Z = [b''_x, b_x) \times (a''_y, a'_y]$ (the dark area Z in the left of Figure 5-12). But then q is a bottom-right corner of a rectangle in $\mathcal{K}$, contained in $\text{int}(R)$ and having either higher x-coordinate than p, or having equal x-coordinate, but strictly larger y-coordinate. This contradicts the definition of p. □

The previous lemma states that in order to test if a rectangle R is added to $\mathcal{K}$ we only need to check if its top witness (provided it exists) has corner-intersection with R or not. This, together with the observation at the beginning of this section, motivates Algorithm 15 to construct $\mathcal{R}_\downarrow$ and $\mathcal{K}$ efficiently. This algorithm returns the entire lists $\mathcal{K}$ and $\overline{\mathcal{K}} = \mathcal{R}_\downarrow \setminus \mathcal{K}$. It also returns for every point $v \in A \cup B$, two lists $\mathcal{K}(v)$ and $\overline{\mathcal{K}}(v)$ containing the rectangles in $\mathcal{K}$ and $\overline{\mathcal{K}}$ respectively, having corner v. All these lists are given in right-top order. Furthermore, every rectangle in $\overline{\mathcal{K}}$ receives two pointers to its left and top witnesses. In the description of Algorithm 15, $\text{last}(\cdot)$ is a pointer to the last element of the corresponding list.

Algorithm 15 Data structure construction.

Require: Two subsets A and B in $[n]^2$, in rook representation.
Ensure: The associated collections $\mathcal{K}(\cdot)$, $\overline{\mathcal{K}}(\cdot)$ and the function LW and TW.

Initialize empty list $\mathcal{K}$ and $\overline{\mathcal{K}}$.
Initialize empty lists $\mathcal{K}(v), \overline{\mathcal{K}}(v)$, for all $v \in A \cup B$.
for each element $a \in A$, from left to right **do**
    List$(a) \leftarrow \emptyset$
    for each element $v \in A \cup B$, with $v_y > a_x$, from bottom to top **do**
        if $(v_x < a_x)$ or $(v_x >$ last(List$(a))_x)$ **then**
            continue to next element.
        else Add v at the end of List(a).
        end if
    end for
    for each element $b \in$ List(a) in the order they were included **do**
        if $b \in B$ **then**
            Let $R = \Gamma(a, b)$ and $L = \Gamma(a', b) \leftarrow$ last$(\mathcal{K}(b))$.
            if L is NULL **then**
                Add R to $\mathcal{K}$, $\mathcal{K}(a)$ and $\mathcal{K}(b)$.
            else
                Let $T = \Gamma(a', b')$ be the next rectangle in $\mathcal{K}(a')$ after L.
                if T is NULL or T has no corner-intersection with R **then**
                    Add R to $\mathcal{K}$, $\mathcal{K}(a)$ and $\mathcal{K}(b)$.
                else ▷ T has corner intersection with R
                    Add R to $\overline{\mathcal{K}}$, $\overline{\mathcal{K}}(a)$ and $\overline{\mathcal{K}}(b)$.
                    Set LW$(R) \leftarrow L$, TW$(R) \leftarrow T$.
                end if
            end if
        end if
    end for
end for

Since we only manipulate pointers and perform comparisons between points in the plane, we get the following corollary.

Lemma 5.4.5. *Given two subsets A and B in $[n]^2$ in rook representation, with $n = |A \cup B|$, Algorithm 15 runs in $O(n^2)$-time. Therefore, the first step of the refinement of Algorithm 14 runs in $O(n^2)$-time.*

5.4.4 Admissible Flips

We can use the left and top witnesses of rectangles in $\overline{\mathcal{K}}$ as a tool to find admissible flips. Let H be a hitting set of $\mathcal{K}$. For every rectangle $S \in \mathcal{K}$, let last$_H(S)$ be the point in H hitting S having maximum x-coordinate and, in case of ties, select the one with larger y-coordinate. We call this point the **last** point in H hitting S.

Let $R = \Gamma(a, b) \in \overline{\mathcal{K}}$ be such that all the rectangles in $\mathcal{R}_\downarrow$ appearing after R in right-top order are hit by H. Let $L = \mathrm{LW}(R) = \Gamma(a', b)$, $T = \mathrm{TW}(R) = \Gamma(a', b')$, $p = \mathrm{last}_H(L)$ and $q = \mathrm{last}_H(T)$.

Lemma 5.4.6. *Assume that p and q do not hit R. If $q_x < a_x$, then R is in $\mathrm{hit}(H, \mathcal{R}_\downarrow)$. Otherwise, the pair (p, q) is an admissible flip for H, and the set $H' = (H \setminus \{p, q\}) \cup \{(p_x, q_y), (p_y, q_x)\}$ is such that $\mathrm{hit}(H, \mathcal{R}_\downarrow) \cup \{S \in \overline{\mathcal{K}} \colon \mathrm{LW}(S) = \mathrm{LW}(R)\} \subseteq \mathrm{hit}(H', \mathcal{R}_\downarrow)$.*

Proof. Assume by contradiction that $q_x < a_x$ and R is not hit by H. By definition of q, we conclude that the area $[a_x, b'_x] \times [a'_y, b'_y]$ does not contain any point of H. In particular, as R is not hit by H, the entire rectangle $\Gamma(a, b')$ is not hit by H. But this is a contradiction since $\Gamma(a, b')$ is a rectangle in $\mathcal{R}_\downarrow$ appearing *after* R in right-top order, so it must be hit by H.

We have proved the first statement of the lemma. To prove the second one, suppose that $q_x \geq a_x$. Since p and q do not hit R, we must have that $p_x < a_x \leq q_x$ and $p_y \leq b_y < q_y$ as in Figure 5-13

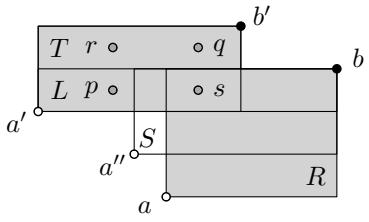


Figure 5-13: The admissible flip (p, q) and a rectangle S with $\mathrm{LW}(S) = \mathrm{LW}(R) = L$.

The proof that the flip (p, q) is admissible is similar to the proof of Lemma 5.4.2, but we give it here for clarity.

Let $r = (p_x, q_y)$, $s = (q_x, p_y)$ and suppose there is a rectangle $\hat{R} = \Gamma(\hat{a}, \hat{b}) \in \mathrm{hit}(H, \mathcal{R}_\downarrow)$ that is not hit by $H' = (H \setminus \{p, q\}) \cup \{r, s\}$. If $\hat{R}$ is hit by p (but not by r or s), then $\hat{b}$ must be in the region $[p_x, q_x - 1] \times [p_y, q_y - 1]$. In particular, $\hat{b} \in T \setminus \{a', b'\}$, contradicting the inclusion-wise minimality of T.

If on the other hand $\hat{R}$ is hit by q (but not by r or s), then $\hat{a}$ must be in the region $[p_x + 1, q_x] \times [p_y + 1, q_y]$. As before, this means that $\hat{a} \in T \setminus \{a', b'\}$, contradicting the inclusion-wise minimality of T. Therefore (p, q) is an admissible flip.

Finally we show that every rectangle $S \in \mathcal{R}_\downarrow \setminus \mathcal{K}$ with the same left witness as R is also hit by H'. For that, take one such $S = \Gamma(a'', b)$, with $a'' \notin \{a, a'\}$.

If the bottom-left corner of S is to the right of a, this is if $a''_x > a_x$, then S appears after R in right-top order. By hypothesis, S is hit by H and also by H'.

Suppose that the previous does not happen. Since $\mathrm{LW}(S) = \mathrm{LW}(R)$, we know that $a'_x < a''_x < a_x$ and $a'_y > a''_y > a_y$ as in Figure 5-13. This means that S contains the point $s \in H'$ defined above. Therefore, S is also in $\mathrm{hit}(H', \mathcal{R}_\downarrow)$. □

Consider the following procedure to find a sequence of flips that transforms any hitting set H_0 of $\mathcal{K}$ in a hitting set for $\mathcal{R}_\downarrow$. Go over the list of rectangles of $\overline{\mathcal{K}}$ in reverse top-right order. Check if the current rectangle R is hit by the current set H by checking the positions of $p = \mathrm{last}_H(\mathrm{LW}(R))$ and $q = \mathrm{last}_H(\mathrm{TW}(R))$. If the

rectangle is hit, continue. Otherwise, flip p and q. By Lemma 5.4.6, when a flip is performed every rectangle having the same left witness as the current observed one is immediately hit by the new set H. Therefore, the number of flips this procedure performs is at most the number of left witnesses $|\mathrm{LW}(\overline{\mathcal{K}})| \leq |\mathcal{K}|$. This procedure is depicted as Algorithm 16

Algorithm 16 Flipping Algorithm.

Require: Lists $\mathcal{K}$ and $\overline{\mathcal{K}}$. Left and top witness functions LW, TW. A hitting set H_0 for $\mathcal{K}$.
Ensure: A hitting set H for $\mathcal{R}_{\downarrow}$.
1: $H \leftarrow H_0$.
2: **for** each rectangle $R = \Gamma(a, b) \in \overline{\mathcal{K}}$, in reverse top-right order **do**
3: Let $p = \mathrm{last}_H(\mathrm{LW}(R))$ and $q = \mathrm{last}_H(\mathrm{TW}(R))$.
4: **if** R contains p or q, or if $q_x < a_x$ **then**
5: continue to next rectangle.
6: **else** Update H to $(H \setminus \{p, q\}) \cup \{(p_x, q_y), (p_y, q_x)\}$.
7: **end if**
8: **end for**

In order to run Algorithm 16 efficiently, we want to keep in each iteration and for every rectangle $R \in \mathcal{K}$, the collection of all points p in H hitting R. We also want to have quick access to the last point in H hitting R. We manage this as follows. Consider a collection of k priority queues, one per rectangle in $\mathcal{K}$, each one accepting priorities on the set $\{1, \ldots, n^2\}$. Every one of the n^2 possible points of the grid $[n]^2$ is given the priority $n(p_x - 1) + p_y$. Recovering $\mathrm{last}_H(R)$ is equivalent to finding the point $p \in R \cap H$ with the highest priority.

We start by populating the priority queue of R with the points in H contained in R, and every time we update H, we update all the priority queues by adding and removing at most two elements in each. There are different data structures for this task. For example, by using heaps or self-balancing binary trees we can perform insertions, deletions and maximum-priority search in time $O(\log |H|) = O(\log n)$. By using van Emde Boas trees [165], we can perform these operations faster, in time $O(\log \log n^2) = O(\log \log n)$.

By using the fact that we require $O(nk)$-operations to initially populate the priority queues and that in total we perform at most k flips, we obtain as corollary the following.

Lemma 5.4.7. *Using one priority queue for each $R \in \mathcal{K}$. Algorithm 16 perform at most $O(nk + k^2)$ insertions, deletions and maximum-priority search operations. It also requires to do constant work for each rectangle in $\overline{\mathcal{K}}$. Therefore, the second and third step of the refinement of Algorithm 14 runs in time*

$$O(f(k, \ell) + m + (nk + k^2) \log n),$$

using heaps or self-balancing binary trees. And it runs in time

$$O(f(k,\ell) + m + (nk + k^2)\log\log n),$$

using van Emde Boas trees.

5.4.5 Refined Algorithm

The improved version of Algorithm 14 is depicted below as Algorithm 17.

Algorithm 17 (improved version of Algorithm 14)

1: Run Algorithm 14 to compute $\mathcal{R}$, $\mathcal{R}_\downarrow$ and the c.f.i. greedy family $\mathcal{K}$.
2: Use an algorithm for the Dilworth chain-partitioning problem to compute a maximum independent set $\mathcal{R}_0$ and a minimum hitting set H_0 for $\mathcal{K}$.
3: Use Algorithm 16 to compute a flipping set H^* for $\mathcal{R}_\downarrow$.
4: Return $\mathcal{R}_0$ and H^*.

5.4.6 Bounds for c.f.i. Families

Our algorithm is, at the moment, fully described. The last ingredient we require to analyze its running time is to give better bounds for the number of vertices and edges of the intersection graph of the c.f.i. family $\mathcal{K}$.

For this section, let $\mathcal{R}_\downarrow$ be the set of inclusion-wise minimal rectangles coming from a 2DORG $G = (A, B, \mathcal{R})$, where (A, B) is a bicolored rook representation. Let also $\mathcal{K}$ be any c.f.i. subfamily of $\mathcal{R}_\downarrow$ (not necessarily a greedy one). Define the collections $\mathrm{bl}(\mathcal{K})$ and $\mathrm{tr}(\mathcal{K})$ of bottom-left corners of rectangles in $\mathcal{K}$ (the ones in A), and top-right corners of rectangles in $\mathcal{K}$ (the ones in B) respectively. Finally, let k and ℓ be the number of vertices and edges of the intersection graph $\mathcal{I}(\mathcal{K})$.

Lemma 5.4.8. *If all the rectangles of $\mathcal{K}$ intersect a fixed vertical (or horizontal) line λ, then*

$$k \leq |\mathrm{bl}(\mathcal{K})| + |\mathrm{tr}(\mathcal{K})| - 1. \tag{5.32}$$

Proof. Project all the rectangles in $\mathcal{K}$ onto the line λ to obtain a collection of intervals in the line. Since $\mathcal{K}$ is a c.f.i. family, all the intersections in $\mathcal{K}$ are skew-intersections (see (5.13)). Therefore, the collection of intervals forms a *laminar* family: if two intervals intersect, then one is contained in the other. Let X be the collection of extreme points of the intervals. Since by assumption (A, B) is a bicolored rook representation, $|X| = |\mathrm{bl}(\mathcal{K})| + |\mathrm{tr}(\mathcal{K})|$ and furthermore, every interval is a non-singleton interval. To conclude the proof of the lemma we only need to prove the following claim.

Claim: Any laminar family $\mathcal{I}$ of non-singleton intervals having extreme points in X has cardinality at most $|X| - 1$. The claim follows by induction in $|X|$. For $|X| = 2$, it holds trivially. For the general case, let $I = [p, q]$ be a inclusion-wise minimal interval in $\mathcal{I}$. Note that only one of p or q can be an extreme point of an interval in $\mathcal{I}\setminus\{I\}$ as otherwise we would contradict laminarity. Therefore, the laminar

family $\mathcal{I} \setminus \{I\}$ has at most $|X| - 1$ extreme points. By induction hypothesis we get that $|\mathcal{I} \setminus \{I\}| \leq (|X| - 1) - 1$, or equivalently, $|\mathcal{I}| \leq |X| - 1$. This concludes the proof of the claim and the lemma. □

Lemma 5.4.9. *Let $\mathcal{L} = \{\lambda_1, \ldots, \lambda_r\}$ be a collection of vertical lines sorted from left to right that intersect all the rectangles in $\mathcal{K}$, then*

$$k \leq n(1 + \lfloor \log_2(r) \rfloor) - r. \tag{5.33}$$

Proof. Consider the following collections of vertical lines:

$$\begin{aligned}
\mathcal{L}_0 &= \{\lambda_{2k+1} \colon k \geq 0\} = \{\lambda_1, \lambda_3, \lambda_5, \lambda_7, \lambda_9 \ldots\}. \\
\mathcal{L}_1 &= \{\lambda_{4k+2} \colon k \geq 0\} = \{\lambda_2, \lambda_6, \lambda_{10}, \lambda_{14}, \ldots\}. \\
\mathcal{L}_2 &= \{\lambda_{8k+4} \colon k \geq 0\} = \{\lambda_4, \lambda_{12}, \lambda_{20}, \lambda_{28}, \ldots\}. \\
&\vdots \\
\mathcal{L}_t &= \{\lambda_{2^t(2k+1)} \colon k \geq 0\}. \\
&\vdots
\end{aligned}$$

These collections of lines are disjoint and $\mathcal{L}_0, \ldots, \mathcal{L}_{\lfloor \log_2(r) \rfloor}$ is a partition of $\mathcal{L}$. Partition the family $\mathcal{K}$ of rectangles as $\mathcal{K}_0 \cup \mathcal{K}_1 \cup \cdots \cup \mathcal{K}_{\lfloor \log_2(r) \rfloor}$, where every rectangle is assigned to the set with largest index containing a vertical line that intersects it. See Figure 5-14 for an example illustrating this construction (in the example, $\mathcal{K}$ is a collection of disjoint rectangles).

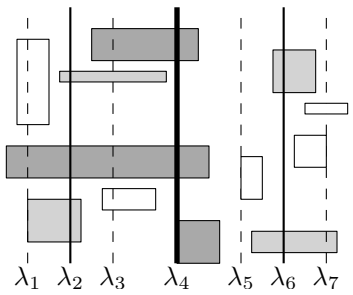


Figure 5-14: The dashed lines, thin lines and thick lines correspond to $\mathcal{L}_0$, $\mathcal{L}_1$ and $\mathcal{L}_2$ respectively. White, light-gray and dark-gray rectangles are assigned to $\mathcal{K}_0$, $\mathcal{K}_1$ and $\mathcal{K}_2$ respectively.

Fix $0 \leq t \leq \lfloor \log_2(r) \rfloor$. Every rectangle in $\mathcal{K}_t$ intersects a unique line in $\mathcal{L}_t$ (if it intersects two or more, then it would also intersect a line in $\mathcal{L}_{t+1}$). For a given line $\lambda \in \mathcal{L}_t$, let $\mathcal{K}_\lambda$ be the subset of rectangles in $\mathcal{K}_t$ intersecting λ. By Lemma 5.4.9, the number of rectangles in $\mathcal{K}_\lambda$ is at most $|\mathrm{bl}(\mathcal{K}_\lambda)| + |\mathrm{tr}(\mathcal{K}_\lambda)| - 1$. But it is easy to see that every point a in A belongs to at most one set $\{\mathrm{bl}(\mathcal{K}_\lambda) : \lambda \in \mathcal{L}_t\}$ (in fact, it belongs to the first line λ which is on or to the right of a). Therefore, $\sum_{\lambda \in \mathcal{L}_t} |\mathrm{bl}(\mathcal{K}_\lambda)| \leq |A|$,

and similarly, $\sum_{\lambda\in\mathcal{L}_t}|\mathrm{tr}(\mathcal{K}_\lambda)| \leq |B|$. Putting all together we get

$$|\mathcal{K}_t| = \sum_{\lambda\in\mathcal{L}_t} |\mathcal{K}_\lambda| \leq |A| + |B| - |\mathcal{L}_t| = n - |\mathcal{L}_t|. \tag{5.34}$$

Therefore,

$$|\mathcal{K}| = \sum_{t=0}^{\lfloor \log_2(r)\rfloor} |\mathcal{K}_\lambda| \leq (1 + \lfloor \log_2(r)\rfloor)n - |\mathcal{L}|. \tag{5.35}$$

This completes the proof. □

Since n vertical lines are enough to intersect all rectangles in $\mathcal{K}$, we conclude that in any case $k = O(n \log n)$. Now, let us bound the number ℓ of edges of the intersection graph $\mathcal{I}(\mathcal{K})$.

Lemma 5.4.10.

$$\ell = O(n^2). \tag{5.36}$$

Proof. Let $\Lambda(n)$ be the maximum value that ℓ can take, for a given value of n. Consider the vertical line $\lambda = \{(x, \lfloor n/2\rfloor) \colon x \in \mathbb{R}\}$. This line divides the grid in two roughly equal parts. Count the number of edges in $(\mathcal{K}, \preceq)$ as follows. Let E_1 be the edges connecting pairs of rectangles that are totally to the left of λ, E_2 be the edges connecting pairs of rectangles that are totally to the right of λ, and E_3 be the remaining edges. Then, $|E_1| \leq \Lambda(\lfloor n/2\rfloor)$ and $|E_2| \leq \Lambda(\lceil n/2\rceil)$. We bound the value of $|E_3|$ in a different way.

Let $\mathcal{K}_0$ be the set of rectangles intersecting the vertical line λ, then E_3 is exactly the collection of edges in $\mathcal{I}(\mathcal{K})$ having an endpoint in $\mathcal{K}_0$. By Lemma 5.4.9, $|\mathcal{K}_0| \leq n$. Now we bound the degree of each element in $\mathcal{K}_0$ in the graph $\mathcal{I}(\mathcal{K})$. Consider one rectangle $R = \Gamma(a, b) \in \mathcal{K}_0$. Every rectangle intersecting R must intersect one of the four lines defined by its sides. By using again Lemma 5.4.9, we conclude that the total number of rectangles in $\mathcal{K}$ intersecting R is at most $4n$. Therefore, $\Lambda(n)$ satisfies the recurrence

$$\Lambda(n) \leq \Lambda(\lfloor n/2\rfloor) + \Lambda(\lceil n/2\rceil) + 4n^2. \tag{5.37}$$

By solving this recurrence we get $\Lambda(n) = O(n^2)$, completing the proof. □

To finish this section, we note that there are c.f.i. families $\mathcal{K}$ for which $\ell = \Omega(n^2)$. See Figure 5-15 for an example. In this example, $\mathcal{K}$ was obtained using the greedy procedure on $\mathcal{R}_\downarrow$. Also, if we let $n' = |A| = |B| = n/2$, then $k = 2n' + 2$ and $\ell = (n')^2 + 4 = \Omega(n^2)$.

On the other hand, it is not clear if there are examples achieving $k = \Omega(n \log n)$. In particular, we give the following conjecture

Conjecture 5.4.11. *For any c.f.i. family, $k = O(n)$.*

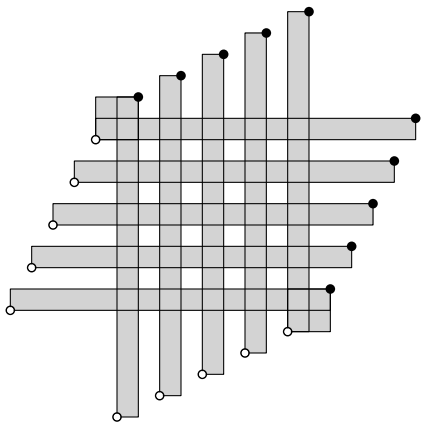


Figure 5-15: Example with $\ell = \Omega(n^2)$.

5.4.7 Conclusion

Combining Lemmas 5.4.5, 5.4.7, the bounds for k and ℓ obtained in the previous subsection and the discussion about the running time of the algorithm of Hopcroft and Karp and the one of Mucha and Sankowski, we obtain the following theorem.

Theorem 5.4.12. *Algorithm 17 finds a maximum independent set and a minimum hitting set for any collection of rectangles $\mathcal{R}(A, B)$ where (A, B) are in rook representation and $n = |A \cup B|$, in*

$$O\left(n^{2.5}\sqrt{\log n}\right)\text{-deterministic time, or in}$$
$$O((n \log n)^\omega + (n \log n)^2 \log\log n)\text{-randomized time,}$$

using Hopcroft and Karp's algorithm or the one of Mucha and Sankowski respectively, where $2 \leq \omega \leq 2.376$ is the best exponent for the matrix multiplication problem. If Conjecture 5.4.11 is true, these running times improve to

$$O(n^{2.5})\text{-deterministic time, and } O(n^\omega + n^2 \log\log n)\text{-randomized time.}$$

In particular, if a 2DORG *is given as $G = (A, B, \mathcal{R})$ with (A, B) in rook representation, then the jump-number of G, a maximum cross-free matching and a minimum biclique cover of G can be computed in the same running times as above.*

5.5 Relation to Frank and Jordan's Set-Pair Covering Problem

In a seminal paper, Frank and Jordán [63] extend Györi's result for independent point-interval pairs (which, by Theorem 4.3.7 is equivalent to the min-max relation between biclique covers and cross-free matchings for convex graphs) to *set-pairs*. We briefly describe a particular case of their result that concern us.

A collection of pairs of sets $\{(S_i, T_i)\}_i$ is **half-disjoint** if for every $i \neq j$, $S_i \cap S_j$ or $T_i \cap T_j$ is empty. A **directed-edge** (s,t) **covers** a set-pair (S,T) if $s \in S$ and $t \in T$. A family $\mathcal{S}$ of set-pairs is **crossing** if whenever (S,T) and (S',T') are in $\mathcal{S}$, so are $(S \cap T, S' \cup T')$ and $(S \cup T, S' \cap T')$. Frank and Jordán prove that for every crossing family $\mathcal{S}$, the *maximum size of a half-disjoint subfamily* is equal to the *minimum size of a collection of directed-edges covering* $\mathcal{S}$. They also give a linear programming based algorithm to compute both optimizers. Later, combinatorial algorithms for this result were also given (e.g. [11]). See Végh's Ph.D. thesis [166] for related references.

The min-max relation between biclique covers and cross-free matchings of 2DORGs stated in Theorem 5.4.3, can be seen as a non-trivial application of Frank and Jordán's result. Given a 2DORG $G = (A \cup B, \mathcal{R})$, consider the family of set-pairs $\mathcal{S} = \{(R_x, R_y) \colon R \in \mathcal{R}_\downarrow\}$. It is easy to check that this family is crossing, that half-disjoint families of $\mathcal{S}$ correspond to independent sets in $\mathcal{R}_\downarrow$ and that coverings of $\mathcal{S}$ by directed-edges correspond to hitting sets for $\mathcal{R}_\downarrow$. We remark that this reduction relies heavily on the geometric interpretation of 2DORGs we have presented in this work and also that the presented proofs are self-contained and simpler than the ones for Frank and Jordán's result.

5.6 Summary of Results and Open Problems

In this chapter, we have extended the class of bipartite graphs for which the jump number problem is efficiently solvable, devising algorithms running in $O(n^{2.5}\sqrt{\log n})$-deterministic time or $O((n \log n)^\omega + (n \log n)^2 \log\log n)$-randomized time. These algorithms compute not only the jump number of 2DORGs, but also a maximum cross-free matching and a minimum biclique cover.

Previously, the largest class of graphs where the jump number problem had been shown to be polynomially solvable was the class of convex graphs. In fact, for convex graphs we have improved considerably the running time to compute the jump number as before this work, only an $O(n^9)$-algorithm existed [40]. By relating our work with a previous result by Györi [80], we have shown in Chapter 4 the existence of an alternative $O(n^2)$-algorithm for convex graphs.

We have also shown a min-max relation between biclique covers and cross-free matchings of 2DORGs. As the minimum biclique cover problem of a graph is equivalent to the problem of computing the *boolean rank* of its adjacency matrix A, we have also expanded the class of matrices for which the boolean rank can be computed exactly in polynomial time.

We have also presented a min-max relation between hitting sets and independent sets for the families of rectangles arising from 2DORGs. This result may be of interest on its own.

Lastly, we have related the previous min-max relations to other relations arising from apparently unrelated problems in combinatorial optimization: the *minimum rectangle cover* and the *maximum antirectangle* of an orthogonal biconvex polygon, studied by Chaiken et al. [25], the *minimum base of a family of intervals* and the *maximum independent set of point-interval pairs*, studied by Györi [80]; and the *min-*

imum edge-cover and the *maximum half-disjoint family of set-pairs*, studied by Frank and Jordán [63]. Our min-max relations can be seen in a certain way as both a generalization of Győri's result and as a non-trivial application of Frank and Jordán's result.

We include Table 5.1 containing the current best running times for the studied problems on a graph $G = (A \cup B, \mathcal{R})$, with $n = |A \cup B|$. This table extends Table 4.1 presented in Chapter 4.

Table 5.1: Table of algorithmic results.

	Bip. Perm.	Biconvex	Convex	2DORG
Max. cross-free matching	$O(n)$ [16, 53]	$O(n^2)$ [16]	$O(n^9)$ [40]	-
(Jump number) **(new)**	-	-	$\boldsymbol{O(n^2)}$[a]	$\boldsymbol{\widetilde{O}(n^{2.5})}$[b] $\boldsymbol{\widetilde{O}(n^{\omega})}$[b]
Min. biclique-cover	$O(n)$ [16, 53]	$O(n^2)$[a]	$O(n^2)$[a]	-
(new)	-	-	-	$\boldsymbol{\widetilde{O}(n^{2.5})}$[b] $\boldsymbol{\widetilde{O}(n^{\omega})}$[b]

[a] These results follows from Theorem 4.3.7 and the algorithm of Franzblau and Kleitman [64] for geometric rectangle cover. The possibility of applying this algorithm to compute the biclique cover of convex graphs has been noticed by Amilhastre et al. [2].

[b] By Theorem 5.4.12.

Regarding unsolved questions in this area, the problem of determining the complexity of the jump number problem for permutation graphs (two dimensional graphs) remains open. Interestingly, as first observed by Ceroi [23], for two-dimensional posets we can interpret the jump number as the problem of finding a maximum *weighted* independent set of an associated family of rectangles, for a given assignment of weights. The fact that the rectangles are weighted makes solving this problem a very challenging task. In particular, the natural LP-relaxations, unlike the case of 2DORGs, have non-unit integrality gap. Bounding this gap is also an interesting question.

Chapter 6

Weighted Cross-free Matching

In this chapter we consider the problem of finding the *maximum weight cross-free matching* of a graph. We show this problem is NP-hard for 2DORGs, and give some subclasses for which we can solve this problem in polynomial time. The problem is defined as follows.

Problem 16 (Maximum Weight Cross-Free Matching Problem)**.** Given a graph $G = (V, E)$, and a nonnegative weight function $w\colon E \to \mathbb{R}_+$, find a cross-free matching M of maximum total weight.

For chordal bipartite graphs, this problem is equivalent to the *maximum weight jump number*, defined by Ceroi [24] and, in the case where $G = (A, B, \mathcal{R})$ is a 2DORG, to the *maximum weight independent set of rectangles* in $\mathcal{R}$.

6.1 NP-Hardness

Here we show the following result.

Theorem 6.1.1. *The maximum weight cross-free matching problem is* NP*-hard for* 2DORG*s, even if the weights are all in* $\{0, 1\}$.

Proof. To prove this result, we reduce from the maximum independent set of rectangles problem (MISRP), which is NP-hard *even if the vertices of the rectangles are all distinct* [60]. Given an instance $\mathcal{F}$ of MISRP having the previous property, let A (resp. B) be the set of bottom-left (resp. top-right) corners of rectangles in $\mathcal{F}$. The 2DORG $G = (A, B, \mathcal{R})$ satisfies $\mathcal{F} \subseteq \mathcal{R}$, so we can obtain the maximum independent set of $\mathcal{F}$ by finding a maximum weight cross-free matching in G, where we assign a weight of one to each $R \in \mathcal{F}$, and a weight of zero to every other rectangle in $\mathcal{R}$. □

6.2 Polynomial Time Algorithms

6.2.1 Bipartite Permutation

We now provide an efficient algorithm for the maximum weight cross-free matching of bipartite permutation graphs.

Recall that permutation graphs admit a rook representation induced by a permutation π: the vertex i of the graph is mapped to the point $(i, \pi(i))$. In particular, when the graph is bipartite, the previous representation is a bicolored rook representation (A, B) where all the points of A (resp. of B) form an antichain for $\leq_{\mathbb{R}^2}$. In what follows assume that $G = (A, B, \mathcal{R})$ is given to us in the way we just described.

We can solve the maximum weight independent set of $\mathcal{R}$ in $O(n^2)$ time using the fact that the complement of the intersection graph $\mathcal{I}(\mathcal{R})$ is a comparability graph. To see this, let us write $R \searrow S$ if R and S are disjoint and either $R_x < S_x$ or $R_y > S_y$ holds, where we use the notation $P_x < Q_x$ as shortcut for $p_x < q_x$ for all $p \in P, q \in Q$.

It is not hard to verify that $D = (\mathcal{R}, \searrow)$ is a partial order whose comparability graph is the complement of $\mathcal{I}(\mathcal{R})$, and that maximum weight cross-free matchings in G correspond to maximum weight paths in the digraph D, using w as a weight function on the vertex set $\mathcal{R}$. Since D has $|\mathcal{R}|$ vertices and $O(|\mathcal{R}|^2)$ arcs this optimal path Q^* can be found in $O(|\mathcal{R}|^2)$ time (see, e.g. [38]).

We can find the optimal path Q^* faster by exploiting the structure of D. For simplicity, assume first that all the weights are different.

Let $R \searrow S \searrow T$ be three consecutive rectangles in the optimal path Q^*, then we can extract information about the rectangle S. The following properties are easy to verify.

Right-Right Case. If $R_x < S_x$ and $S_x < T_x$, then
(i) Among all the rectangles located completely to the right of R and having the same top-right corner $\mathrm{tr}(S)$, S is the heaviest.

Right-Down Case. If $R_x < S_x$ and $S_y > T_y$, then
(ii) Among all the rectangles having the same bottom-left corner $\mathrm{bl}(S)$, S is the heaviest.

Down-Right Case. If $R_y > S_y$ and $S_x < T_x$, then
(iii) Among all the rectangles having the same top-right corner $\mathrm{tr}(S)$, S is the heaviest.

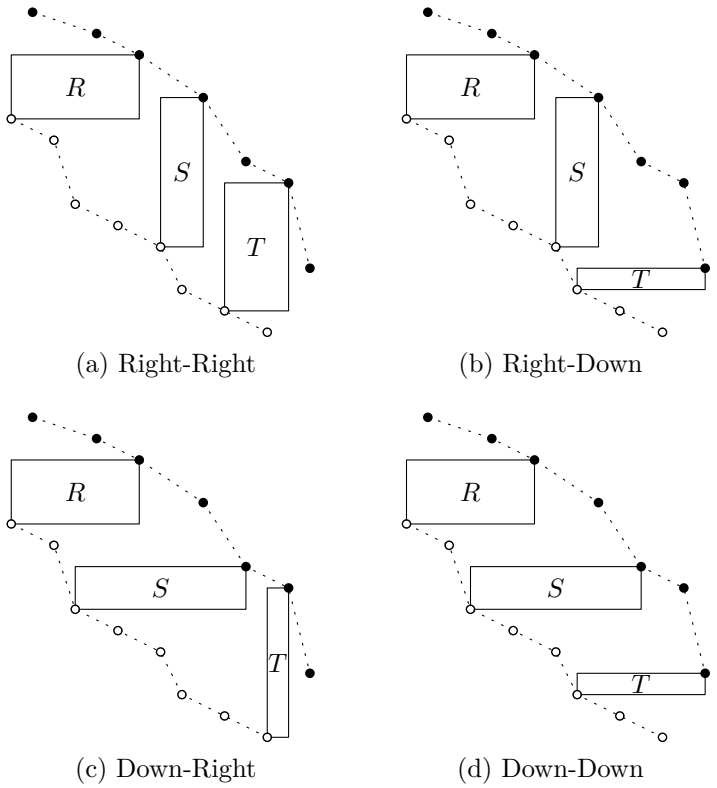
Down-Down Case. If $R_y > S_y$ and $S_y > T_y$, then
(iv) Among all the rectangles located completely below R and having the same bottom-left corner $\mathrm{bl}(S)$, S is the heaviest.

See Figure 6-1 for examples of each case.

The previous properties motivate the following recursion.

Define $T(v)$ for $v \in A \cup B$ as the rectangle of maximum weight (if any) having a corner in v. Also, define $T(b, b')$, for $b, b' \in B$, to be the rectangle (if any) of maximum weight being both strictly to the right of the vertical line passing through b and having top-right corner b'. Similarly, define $T(a, a')$, for $a, a' \in A$ to be the rectangle (if any) of maximum weight being strictly below the horizontal line passing through a and having bottom-left corner a'.

For $R \in \mathcal{R}$, let $V_{\rightarrow}(R)$ (resp. $V_{\downarrow}(R)$) be the maximum weight of a path in D starting on R and having its second rectangle (if any) strictly to the right of R (resp. strictly



(a) Right-Right (b) Right-Down

(c) Down-Right (d) Down-Down

Figure 6-1: Three consecutive rectangles in Q^*.

below R). Using properties (i)–(iv), we conclude that:

$$V(R) = \max\{V_{\rightarrow}(R), V_{\downarrow}(R)\}$$

$$V_{\rightarrow}(\Gamma(a,b)) = w(\Gamma(a,b)) + \max\left\{\max_{b' \,:\, b'_x > b_x} V_{\rightarrow}(T(b,b')), \max_{a' \,:\, a'_x > b_x} V_{\downarrow}(T(a'))\right\}.$$

$$V_{\downarrow}(\Gamma(a,b)) = w(\Gamma(a,b)) + \max\left\{\max_{a' \,:\, a'_y < a_y} V_{\downarrow}(T(a,a')), \max_{b' \,:\, b'_y < a_y} V_{\rightarrow}(T(b'))\right\}.$$

Claim 2. We can compute the entire set of values above in $O(n^2)$.

To prove this claim, note that $\{T(a)\}_{a\in A}$ and $\{T(b)\}_{b\in B}$ can be computed in time $O(|\mathcal{R}|) = O(n^2)$. Also, by visiting the points a' in A from bottom to top, we can find $T(a,a')$ for each $a \in A$ in time $O(|A|) = O(n)$. This means that the entire table $\{T(a,a')\}_{a,a'\in A}$ can be computed in $O(n^2)$-time and so can the table $\{T(b,b')\}_{b,b'\in B}$. Using those values, compute the quantities:

$$S(b) = \max_{b' \,:\, b'x > b_x} V_{\rightarrow}(T(b,b')). \qquad S'(b) = \max_{a' \,:\, a'_x > b_x} V_{\downarrow}(T(a')).$$

$$S(a) = \max_{a' \,:\, a'_y < a_y} V_{\downarrow}(T(a,a')). \qquad S'(a) = \max_{b' \,:\, b'y < a_y} V_{\rightarrow}(T(b')).$$

Above, there are $2|A \cup B| = O(n)$ quantities to compute, and since each one is

the maximum of at most n known values, the entire tables $S(\cdot)$ and $S'(\cdot)$ can be computed in $O(n^2)$ time.

After precomputing all the previous values, it takes constant time to output each entry of the tables $V(\cdot)$, $V_{\rightarrow}(\cdot)$, and $V_{\downarrow}(\cdot)$. As there are $3|\mathcal{R}| = O(n^2)$ of these values, we conclude the proof of the claim.

After computing $\{V(R)\}_{R\in\mathcal{R}}$, we can easily obtain the weight of the optimal path Q^* as the maximum over all these values. It is also very simple to recover the actual path Q^* from the tables using the recurrences above.

Theorem 6.2.1. *By using the dynamic algorithm described above, we can compute the maximum weight cross-free matching of a bipartite permutation graph in $O(n^2)$ time.*

6.2.2 Convex Graphs

By the discussion in Section 4.3.3 and especially by the proof of Theorem 4.3.7, cross-free matchings of convex graphs correspond to independent sets of a certain system of point-interval pairs. Lubiw [110] gives a polynomial time algorithm for the maximum weight independent set of a system of point-interval pairs. It is straightforward to implement her algorithm in $O(n^3)$ time.

We describe her algorithm in our terminology. Let G be a convex graph on $A \cup B$ with labeling $A = \{a_1, \ldots, a_k\}$, where the neighborhood of $b \in B$ is $\{a_{\ell(b)}, \ldots, a_{r(b)}\}$. We see G as an interval bigraph using $I_{a_i} = \{i\}$ and $I_b = [\ell(b), r(b)]$. Let $G = (A, B, \mathcal{R})$ where (A, B) is the natural bicolored representation for interval bigraphs described in Section 3.4.5. This representation puts the points $A \cup B$ on the triangular grid $U \equiv \{(i, -j) \in \mathbb{Z}^2 \colon 0 \leq j \leq i \leq |A|\}$ with the points of A lying in the line $y = -x$.

Since we are looking for a maximum weight independent set of rectangles, we can assume that no two rectangles in $\mathcal{R}$ are equal geometrically. If this is the case, only keep the one having maximum weight.

Given two disjoint rectangles R and S in $\mathcal{R}$, let us say that R *precedes* S if either $R_x < S_x$ and $R_y \cap S_y \neq \emptyset$ or if $R_y < S_y$ and $R_x \cap S_x \neq \emptyset$. It can be shown that this relation is acyclic; therefore, the maximum weight independent set $\mathcal{R}^*$ must contain a rectangle T that no one precedes. The algorithm finds $\mathcal{R}^*$ by guessing this rectangle T and then recursively finding the maximum weighted independent set of the rectangles lying strictly below T and the rectangles lying strictly to the left of T.

More precisely, for each point q in the grid U let $V(q)$ be the weight of a maximum independent set using only rectangles $R \in \mathcal{R}$ having bottom-left corner $a \leq_{\mathbb{R}^2} q$ and top-right corner $b \leq_{\mathbb{R}^2} q$. It is easy to check that

$$V(q) = \max_{a\in A\colon\, a\leq_{\mathbb{R}^2} q} \{w(a, q) + V(a_x - 1, q_y) + V(q_x, a_y - 1)\}.$$

where $w(a, q)$ is the weight of a maximum weight rectangle T contained in $\Gamma(a, q)$ or 0 if there is no such rectangle. See Figure 6-2.

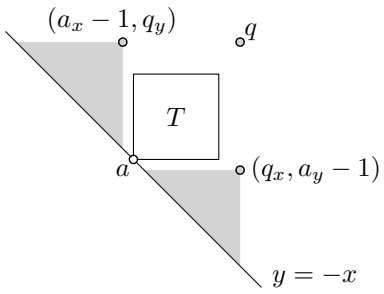


Figure 6-2: Example of Lubiw's algorithm. $V(q)$ equals the weight of rectangle T plus the weight of the maximum independent set in both shaded regions.

All values $\{w(a,q)\}_{a\in A, q\in U}$ can be precomputed in time $O(|A| \cdot |U|) = O(n^3)$. Once we know those values, we can also find all the values of $V(q)$ in time $O(|A| \cdot |U|) = O(n^3)$ by dynamic programming. Finally, once $\{V(q)\}_{q\in U}$ is known, the value $V(|A|, 0)$ holds the weight of the maximum independent set. As usual, we can recover the object achieving the maximum from the tables of values defined above.

Theorem 6.2.2 (Based on Lubiw's algorithm [110])**.** *By using the dynamic algorithm described above, we can compute the maximum weight cross-free matching of a convex graph in $O(n^3)$ time.*

6.3 Open Problems

We have shown that the weighted cross-free matching is NP-complete for 2DORGs and polynomially solvable for convex graphs. The most relevant open problem of this chapter is the complexity of the weighted cross-free matching of interval bigraphs.

Part III

Constrained Set Function Minimization

Chapter 7

Set Function Minimization under Hereditary Constraints

In this chapter we present an efficient algorithm to find nonempty minimizers of certain functions over any family of sets closed under inclusion. The function classes we consider include symmetric submodular functions. Our algorithm makes $O(n^3)$ oracle calls to the submodular function where n is the cardinality of the ground set. In contrast, the problem of minimizing a general submodular function under a cardinality constraint is known to be inapproximable within a factor of $o\left(\sqrt{n/\log n}\right)$ [159].

The results of this chapter are joint work with Michel Goemans. This chapter is organized as follows. First, we offer an introduction to the problem and some background on previous work. Later, we explore the unconstrained problem of finding a nontrivial minimizer of a set function. For the specific case of symmetric submodular functions, this problem can be efficiently solved by a pendant pair technique consolidated by Queyranne [141]; and Nagamochi and Ibaraki [127]. This technique finds a collection of at most n candidate sets, containing a minimizer of the problem. Later in the chapter, we define other classes of functions for which the pendant pair technique can also be used.

Afterwards, we move to the problem of minimizing set functions under hereditary constraints. We show how to modify the pendant pair technique in such a way that the candidate sets are always inside the hereditary family, and that at least one minimizer of the problem is declared as a candidate. By using ideas of Nagamochi and Ibaraki [127] we modify our algorithm so that it outputs all the inclusion-wise minimal minimizers of the problem.

After the completion of this work, we were informed of an independently discovered algorithm of Nagamochi [124] which is able to perform a stronger feat for certain classes of functions, including symmetric submodular functions. In the last part of this chapter we compare our results.

7.1 Introduction

Let V be a finite ground set of size n. A **set function** on V is a real valued function defined on all the subsets of V. A pair (V, f) where f is a set function on V is called a **(set function) system**. In this work we assume that the set function f is given through a **value oracle**, this is an oracle that given a set S returns $f(S)$.

A submodular function f is a set function on V, satisfying that for every pair of sets A and B,

$$f(A \cup B) + f(A \cap B) \leq f(A) + f(B). \tag{7.1}$$

Examples of submodular functions include weight functions, the cut function of a nonnegatively weighted graph and the entropy of a set of random variables. These functions have applications in many areas, including game theory, information theory and graph theory. Many important combinatorial optimization problems can be formulated as finding a minimizer of a submodular function. For this reason the following is considered a fundamental problem of the field.

Problem 17 (Submodular Function Minimization)**.** Given a submodular function $f\colon 2^V \to \mathbb{R}$, find a subset $X^* \subseteq V$ that minimizes $f(X^*)$.

Grötschel, Lovász and Schrijver [75, 76] show that the submodular function minimization problem can be solved using the ellipsoid method in strongly polynomial time and using a polynomial number of oracle calls. Later, a collection of combinatorial strongly polynomial algorithms have been developed by several authors [57, 89, 88, 134, 146, 91]. The current fastest combinatorial algorithms known, due to Iwata and Orlin [91] and Orlin [134] make $O(n^5 \log n)$ and $O(n^5)$ function oracle calls respectively, and run in $O(n^6 \log n)$ and $O(n^6)$ time respectively, where n is the size of the ground set.

There are faster algorithms available when the function f has more structure. The case where f is symmetric is of special interest. In this case, we also require the minimizer A^* of f to be a *nontrivial* subset of V, that is $\emptyset \subset A^* \subset V$, otherwise the problem becomes trivial since, by symmetry and submodularity, $f(\emptyset) = \frac{1}{2}(f(\emptyset) + f(V)) \leq \frac{1}{2}(f(A) + f(V \setminus A)) = f(A)$, for all $A \subseteq V$.

The canonical example of a symmetric submodular function is the cut function of a nonnegatively weighted graph. Finding a nontrivial minimizer corresponds in this case to the minimum cut problem. Nagamochi and Ibaraki [125, 126] give a combinatorial algorithm to solve this problem without relying on network flows. This algorithm has been improved and simplified independently by Stoer and Wagner [158] and Frank [61]. Queyranne [141] generalizes this and obtains a purely combinatorial algorithm that minimizes a symmetric submodular function using only $O(n^3)$ function oracle calls.

It is worth noting that if we impose simple additional constraints, minimizing general submodular functions becomes intractable. For example, the problem of minimizing a submodular function over all the sets of cardinality at most k is NP-hard to approximate within an $o\left(\sqrt{n/\log n}\right)$ factor, as shown by Svitkina and Fleischer [159].

In this work we show that this is not the case for symmetric submodular functions. In Section 7.3 we extend Queyranne's algorithm so that it returns a nontrivial minimizer of any *hereditary family*, this is, a family that it is closed under inclusion. In other words, the new algorithm solves the following problem.

Problem 18 (Hereditary Symmetric Submodular Function Minimization)**.** Given a symmetric submodular function $f\colon 2^V \to \mathbb{R}$, and an hereditary family $\mathcal{I} \subseteq 2^V$, find a nonempty subset $X^* \in \mathcal{I}$ that minimizes $f(X^*)$.

Common examples of hereditary families include

- **Cardinality families:** For $k \geq 0$, the family of all subsets with at most k elements: $\mathcal{I} = \{A \subseteq V\colon |A| \leq k\}$.
- **Knapsack families:** Given a weight function $w : V \to \mathbb{R}_+$, consider the family of all subsets of weight at most one unit: $\mathcal{I} = \{A \subseteq V\colon \sum_{v\in A} w(v) \leq 1\}$.
- **Matroid families:** Given a matroid $\mathcal{M}$ over V, consider the family of independent sets of $\mathcal{M}$.
- **Hereditary graph families:** Given a graph $G = (V, E)$, consider the family of sets S of vertices such that the induced subgraph $G[S]$ satisfies certain hereditary property such as being a clique, being triangle-free, being planar or exclude a given list of minors.
- **Matching families:** Given a hypergraph $H = (V, E)$, consider the family of matchings of H, that is sets of edges that are pairwise disjoint.

Since the intersection of hereditary families is hereditary, the hereditary minimization problem is very broad. This problem includes the following examples.

1. Find a minimum *unbalanced* cut in a graph; that is for given k, find among all nonempty sets of at most k vertices, the one inducing the minimum cut.
2. More generally, given a nonnegatively weighted graph, find a nonempty induced subgraph satisfying an hereditary graph property (e.g. triangle-free, clique, stable-set, planar) minimizing the weights of the edges having precisely one endpoint in the subgraph.

For the unrestricted problem (equivalently for the case where $\mathcal{I} = 2^V \setminus \{V\}$), Nagamochi and Ibaraki [127] present a modification of Queyranne's algorithm that finds all inclusion-wise minimal minimizers of a symmetric submodular function while still using a cubic number of oracle calls. Using similar ideas, we can also list all minimal solutions of an hereditary minimization problem.

The algorithms we present are not restricted to work on symmetric submodular functions. In Section 7.4 we explore other function classes where our methods can still be applied. For instance, we can solve the hereditary problem for functions f that are *restrictions* of a symmetric submodular function (also known as *submodular-posimodular functions*) or when $f(A)$ is defined as $d(A, V \setminus A)$ for a monotone and consistent symmetric bi-set map d in the sense of Rizzi [145].

Other related work. Constrained submodular function minimization problems, i.e. the minimization of a submodular function over *subfamilies* of 2^V, have been studied in different contexts. Padberg and Rao [138] show that the minimum odd cut problem obtained by restricting the minimization over all odd sets can be solved in polynomial time. This was generalized to submodular functions over larger families of sets (satisfying certain axioms) by Grötschel, Lovász and Schrijver [76] and by Goemans and Ramakrishnan [73]. This covers for example the minimization over all even sets, or all sets not belonging to a given antichain, or all sets excluding all minimizers (i.e. to find the second minimum). For the particular case of minimizing a *symmetric* submodular function under cardinality constraints the best previous result is a 2-approximation algorithm by Shaddin Dughmi [46]. Recently, Goel et al. [71] have studied the minimization of *monotone* submodular functions constrained to sets satisfying combinatorial structures on graphs, such as vertex covers, shortest paths, perfect matchings and spanning trees, giving inapproximability results and almost matching approximation algorithms for them. Independently, Iwata and Nagano [90] study both the vertex and the edge covering version of this problem.

The algorithm of Nagamochi and Ibaraki [127] also works with functions satisfying a less restrictive symmetry condition. Narayanan [130] shows that Queyranne's algorithm can be used to minimize a wider class of submodular functions, namely functions that are contractions or restrictions of symmetric submodular functions. Rizzi [145] has given further extension for a different class of functions.

Nagamochi [124] has recently given an algorithm to find all extreme sets of a symmetric submodular function, where a set is called extreme if its function value is strictly smaller than any one of its nontrivial subsets. As we show in Section 7.5, this algorithm can also be used to solve the hereditary minimization problem.

7.2 Unconstrained Minimization

In this section we review Queyranne's algorithm [141] to find nontrivial minimizers of certain systems (V, f). To simplify notation, we use the convention that for any set $A \subseteq V$, any $x \in A$, and any $y \in V \setminus A$; $A + y$ and $A - x$ stand for the sets $A \cup \{y\}$ and $A \setminus \{x\}$ respectively. Also, for $x \in V$ we use $f(x)$ to denote $f(\{x\})$.

A useful operation for set function systems is the following. Let Π be a partition of V. For every collection of parts $X \subseteq \Pi$, we use V_X to denote the set of elements contained in the union of the parts in X, this is

$$V_X = \bigcup_{S \in X} S \subseteq V. \tag{7.2}$$

The **fusion of f relative to Π**, denoted by f_Π is the set function on Π given by

$$f_\Pi(X) = f(V_X). \tag{7.3}$$

We say that (V', f') is a fusion of (V, f) if V' corresponds, up to renaming, to a partition of V and f' is the fusion of f relative to this partition.

Given a system (V, f), we say that we **fuse** a collection of elements $X \subseteq V$ into a single element s if we replace (V, f) by the system $((V \setminus X) + s, f_\Pi)$, where Π is the partition that has one part equal to X (which, by abusing notation we denote as s) and where all the other parts are singletons (which again, by abusing notation, keep the same name as the unique element they contain).

Queyranne's technique performs iterative fusions on the original system (V, f). To keep our explanation simple, we overload the notation above by saying that for any fusion (V', f') of (V, f) and any $a \in V'$, V_a is the set of elements in V that have been fused into a, and for every set $A \subseteq V'$, V_A is the union of all sets V_a with $a \in A$.

We say that a set $X \subseteq V$ **separates** two elements t and u of V, if X contains exactly one of t and u. We extend this notion to fusions by saying that two elements t and u in V' are separated by a set $X \subseteq V$ if $V_t \subseteq X$ and $V_u \subseteq V \setminus X$ or vice versa.

The following concept is crucial for the development of Queyranne's technique. An ordered pair (t, u) of elements of V is called a **pendant pair** for f in V if $\{u\}$ has the minimum f-value among all the subsets of V separating u and t, this is,

$$f(u) = \min\{f(U) \colon U \subset V,\ |U \cap \{u, v\}| = 1\}. \tag{7.4}$$

We say that a system (V, f) is **admissible for pendant pairs** (or simply, admissible) if for every fusion (V', f') with $|V'| \geq 2$ there exists a pendant pair (t, u) for f' in V'.

Suppose that (V, f) is an admissible system and that (t, u) is a pendant pair for f in V. Let X^* be a nontrivial minimizer of (V, f). Then we have two cases depending on whether X^* separates t and u or not.

If X^* separates t and u, then by definition of pendant pairs, $f(u) \leq f(X^*)$, and so $\{u\}$ is also a nontrivial minimizer.

If this is not the case, consider the set system (V', f') obtained by fusing u and v into a single element uv. Any nontrivial minimizer X' of this system induces a nontrivial minimizer $V_{X'}$ of (V, f).

By recursively applying the argument above $n - 1$ times (as all the fused systems have pendant pairs) we can find a nontrivial minimizer of (V, f). The procedure is described in Algorithm 18 below, which we call **Queyranne's algorithm**.

Algorithm 18 (Queyranne's algorithm).

Require: An admissible system (V, f).
Ensure: A nontrivial minimizer X^* for (V, f).

1: Let $(V', f') = (V, f)$, and $\mathcal{C} \leftarrow \emptyset$. ▷ $\mathcal{C}$ is the set of candidates.
2: **while** $|V'| \geq 2$ **do**
3: Find any pendant pair (t, u) for f' in V'.
4: Add V_u to $\mathcal{C}$. ▷ V_u is the set of elements of V that have been fused into u.
5: Update (V', f') by fusing $\{t, u\}$ into a single element tu.
6: **end while** ▷ $|V'| = 1$
7: Add V_u to $\mathcal{C}$, where u is the only element of V'.
8: Return the set X^* in $\mathcal{C}$ with minimum f-value.

The argument above implies that Queyranne's algorithm is correct. The only nontrivial step of the algorithm correspond to finding pendant pairs of fusions. Suppose in what follows that we have access to an algorithm $\mathcal{A}$ that computes pendant pairs for any fusion (V', f') in $O(T(|V'|))$-time and using $O(T(|V'|))$ calls to some value oracle, where $T(\cdot)$ is an increasing function.

Lemma 7.2.1. *By using $\mathcal{A}$ as a subroutine, Queyranne's algorithm returns a nontrivial minimizer of (V, f) in $O(nT(n))$-time and using the same asymptotic number of oracle calls.*

Queyranne originally devised this algorithm for symmetric submodular functions. He showed not only that these functions are admissible, a fact originally shown by Mader [111], but also that there is an efficient algorithm to compute pendant pairs.

Let f be a symmetric submodular function on V. Consider an ordering $(v_1, \ldots, v_n)$ of the elements of V, such that

$$f(W_{i-1} + v_i) - f(v_i) \leq f(W_{i-1} + v_j) - f(v_j), \text{ for all } 2 \leq i \leq j \leq n, \tag{7.5}$$

where v_1 can be chosen arbitrarily and W_i denotes the set $\{v_1, \ldots, v_i\}$. An ordering satisfying (7.5) is called a **maximum adjacency ordering**. Queyranne [141] shows the following result.

Lemma 7.2.2 (Queyranne [141])**.** *For a symmetric submodular function f on V, and an arbitrarily chosen element $v_1 \in V$, the last two elements (v_{n-1}, v_n) of a maximum adjacency ordering of V starting from v_1 constitute a pendant pair. Furthermore, this ordering can be found using $O(n^2)$ oracle calls and in the same running time.*

We do not prove Queyranne's lemma here as in Section 7.4 we show an extension to more general functions. By using the lemmas above and the fact that fusions of symmetric submodular functions are also symmetric submodular, Queyranne has shown the following.

Theorem 7.2.3 (Queyranne [141])**.** *The problem of finding a nontrivial minimizer of a symmetric submodular function f on V can be solved in $O(n^3)$ time and using $O(n^3)$ oracle calls to f.*

In the next section, we extend Queyranne's algorithm to the problem of finding nontrivial minimizers of set functions under hereditary constraints.

7.3 Constrained Minimization

A triple $(V, f, \mathcal{I})$ where f is a set function on V and $\mathcal{I}$ is an hereditary family of V, is called an **hereditary system**. Since hereditary families can be exponentially large in the size of the ground sets, we assume that we access $\mathcal{I}$ through an oracle that given a set A, answers whether A is in the family or not.

We say that a set A is a **minimal optimal** solution for $(V, f, \mathcal{I})$ if A is an inclusion-wise minimal minimizer of the function f over the nontrivial sets in $\mathcal{I}$.

Similarly, we say that A is a minimal optimal solution for (V, f) if A is minimal optimal for $(V, f, 2^V \setminus \{V\})$.

We extend the notion of fusion to hereditary systems as follows. Given a partition Π of V, the **fusion of $\mathcal{I}$ relative to Π**, denoted by $\mathcal{I}_\Pi$ is the family

$$\mathcal{I}_\Pi = \{I \subseteq \Pi \colon V_I \in \mathcal{I}\}.$$

It is easy to see that if $\mathcal{I}$ is hereditary then so is $\mathcal{I}_\Pi$. The fusions of an hereditary system $(V, f, \mathcal{I})$ are defined analogously to the fusions of (V, f). It is worth noting at this point that if $(V, f', \mathcal{I}')$ is a specific fusion of $(V, f, \mathcal{I})$ then we can test if $A \in \mathcal{I}'$ and evaluate $f'(A)$ using only one oracle call to $\mathcal{I}$ or f respectively.

A system (V, f) is **strongly admissible for pendant pairs** (or simply, strongly admissible) if for every fusion (V', f') where V' has at least 3 elements, and every $v \in V'$, there is a pendant pair (t, u) of V' avoiding v (this is, $t \neq v$ and $u \neq v$). We say that $(V, f, \mathcal{I})$ is strongly admissible if (V, f) is.

Queyranne's lemma (Lemma 7.2.2) implies that (V, f) is a strongly admissible system when f is a symmetric submodular function. In Section 7.4 we give other examples. The following lemma is of interest on its own.

Lemma 7.3.1. *The minimal optimal solutions of any strongly admissible system $(V, f, \mathcal{I})$ are pairwise disjoint.*

Proof. The lemma holds trivially if V has at most 2 elements. So assume $|V| \geq 3$. Suppose that two minimal optimal solutions A and B are not disjoint. Since no one can include the other, we have two cases, either $A \cup B = V$, or $V \setminus (A \cup B) \neq \emptyset$.

In the first case, consider the system $(V' = \{a, b, c\}, f')$ obtained from (V, f) by fusing $A \setminus B$ into a single element a, $B \setminus A$ into b and $A \cap B$ into c. Since (V, f) is strongly admissible there is a pendant pair for f' in V' avoiding c. Without loss of generality assume this pair is (a, b). Therefore $f'(b) \leq f'(\{a, c\})$ or equivalently, $f(B \setminus A) \leq f(A) = f(B)$, contradicting the inclusion-wise minimality of B.

For the second case, consider the system $(V' = \{a, b, c, d\}, f')$ obtained by fusing $A \setminus B$ into a, $B \setminus A$ into b, $A \cap B$ into c and $V \setminus (A \cup B)$ into d. Since (V, f) is strongly admissible there is a pendant pair for f' in V' avoiding d. If the pendant pair also avoids c, then without loss of generality we can assume this pair is (a, b). As above, this means that $f'(b) \leq f'(\{a, c\})$ or equivalently, $f(B \setminus A) \leq f(A) = f(B)$, contradicting the choice of B. Therefore the pendant pair contains c.

If c is the first element of the pair, we can assume without loss of generality that the pair is (c, b). Hence, $f'(b) \leq f'(\{a, c\})$ which is a contradiction as shown above.

On the other hand, if c is the second element of the pair, we can assume this pair is (b, c). Hence, $f'(c) \leq f'(\{a, c\})$ or equivalently $f(A \cap B) \leq f(A)$, contradicting the minimality of A.

As every case leads to a contradiction, we conclude that the minimal optimal solutions are disjoint. □

In what follows we present two algorithms: one to find a particular minimal optimal solution of a strongly admissible hereditary system $(V, f, \mathcal{I})$, and another to

find *all* minimal optimal solutions. The previous lemma implies that there are at most n of them.

Both of the mentioned algorithms are direct extensions of the algorithms presented by Nagamochi and Ibaraki [127]. In fact by setting $\mathcal{I}$ to be the hereditary family of sets not containing a particular element s, we recover their algorithms.

If we just use Queyranne's algorithm on f, we could introduce candidates that are not in the hereditary family. In order to avoid that, we first fuse all the loops[1], if any, of $\mathcal{I}$ into a single loop s, and proceed to find a pendant pair not containing it. In this way, we ensure that every selected candidate belongs to $\mathcal{I}$. If the hereditary family has no loops, then we just use Queyranne's algorithm until a loop s is created. From that point on we continue as before. The complete procedure is depicted below as Algorithm 19. We defer the problem of finding pendant pairs to the next section. For now we assume the existence of an algorithm $\mathcal{A}$ that given an element s is able to find a pendant pair avoiding s.

Algorithm 19 FindMinimalOptimal $(V, f, \mathcal{I})$

Require: A strongly admissible hereditary system $(V, f, \mathcal{I})$.
Ensure: A minimal optimal set X^* for the hereditary minimization problem.

Set $(V', f', \mathcal{I}') \leftarrow (V, f, \mathcal{I})$, and $\mathcal{C} \leftarrow \emptyset$. ▷ $\mathcal{C}$ is the set of candidates.
while $\mathcal{I}'$ has no loops **do**
    Find any pendant pair (t, u) of f'.
    Add V_u to $\mathcal{C}$. ▷ V_u is the set of elements of V that have been fused into u.
    Update $(V', f', \mathcal{I}')$ by fusing $\{t, u\}$ into a single element tu.
end while ▷ $\mathcal{I}'$ has at least one loop.
Let $(V' + s, f', \mathcal{I}')$ be the system obtained by fusing all the loops of $\mathcal{I}$ into a single element denoted s (keep s as an element *outside* V').
while $|V'| \geq 2$ **do**
    Find a pendant pair (t, u) of f' not containing s.
    Add V_u to $\mathcal{C}$.
    if $\{t, u\} \in \mathcal{I}'$ **then**
        Update $(V' + s, f', \mathcal{I}')$ by fusing $\{t, u\}$ into a single element tu.
    else
        Update $(V' + s, f', \mathcal{I}')$ by fusing $\{s, t, u\}$ into a single element called s.
    end if
end while
if $|V'| = 1$ (say $V' = \{u\}$) **then**
    Add V_u to $\mathcal{C}$.
end if
Find the sets in $\mathcal{C}$ of minimum f-value. Among those, return the set X^* that was added earlier to $\mathcal{C}$.

Theorem 7.3.2. *Algorithm 19 outputs an optimal solution of any strongly admissible hereditary system* $(V, f, \mathcal{I})$.

[1] A loop of an hereditary family is a singleton that is not in $\mathcal{I}$.

Proof. By induction, we can check that at the beginning of each iteration, either $\mathcal{I}'$ is loopless or s is its only loop. Every candidate V_u introduced in $\mathcal{C}$ is obtained from a non-loop u. Therefore, the output set X^* is in $\mathcal{I}$.

Suppose for contradiction that there is a nonempty set $Y \in \mathcal{I}$ such that $f(Y) < f(X^*)$. Assume first, that this set does not separate any pendant pair found in the execution of the algorithm. Since the algorithm only fuses pendant pairs and loops we have that at every iteration and for every element w in the current ground set, the associated set $V_w \subseteq V$ is either completely inside Y or completely outside Y. In particular, since $V_s \notin \mathcal{I}$, V_s is always completely outside Y. Therefore, at the end of the algorithm, Y must be equal to the set V_u in line 18 and so it is included in the set of candidates, contradicting the definition of X^*.

Consider then the first pendant pair (t, u) found by the algorithm that is separated by Y. By the property of pendant pairs, $f'(u) \leq f(Y)$ for the function f' at that iteration. But then, the set $V_u \in V$ of elements that were fused into u is a candidate considered by the algorithm. Thus, $f(X^*) \leq f(V_u) = f'(u) \leq f(Y)$, which contradicts our assumption. Therefore X^* has minimum f-value among the nontrivial sets of $\mathcal{I}$.

Furthermore, since we choose X^* as the set that is introduced first into the family of candidates $\mathcal{C}$ (among the ones of minimum value), then this set X^* is also an inclusion-wise minimal minimizer of $(V, f, \mathcal{I})$. Indeed, if there is a set $Y \in \mathcal{I}$ such that $f(Y) = f(X^*)$, with $\emptyset \neq Y \subset X^*$, then Y separates two elements of X^*. This means that at some moment before the introduction of X^* as a candidate, the algorithm finds a pendant pair (t, u) separated by the set Y with both t and u in X^*. At this iteration, the candidate V_u is such that $f(V_u) = f(Y) = f(X^*)$, which is a contradiction since V_u is introduced before X^* to the set of candidates. □

Suppose that a given algorithm $\mathcal{A}$ computes pendant pairs of (V', f') avoiding a given element in $O(T(|V'|))$-time and using $O(T(|V'|))$ oracle calls, where $T(\cdot)$ is an increasing function. Then we have the following.

Theorem 7.3.3. *By using $\mathcal{A}$ as a subroutine, Algorithm 19 returns a minimal optimal solution of the strongly admissible hereditary system $(V, f, \mathcal{I})$ in $O(nT(n))$-time and using the same asymptotic number of oracle calls.*

Proof. Direct from the fact that each iteration decreases the cardinality of V' by one or two units. □

We can use the fact that the minimal optimal solutions are disjoint to find all of them. We first compute one particular minimal optimal solution X^* of the system and fuse it into a single element s which we will consider a loop for the new family. Then we run the algorithm again in such a way that, every time a optimal solution X is found we fuse $X + s$ into s in order to avoid finding solutions having nonempty intersection with X. A naive implementation of this procedure requires $O(n)$ calls to Algorithm 19. A better implementation is described in Algorithm 20.

Theorem 7.3.4. *Algorithm 20 outputs all minimal optimal solutions of the strongly admissible hereditary system $(V, f, \mathcal{I})$.*

Algorithm 20 FindAllMinimalOptimals $(V, f, \mathcal{I})$

Require: A strongly admissible hereditary system $(V, f, \mathcal{I})$.
Ensure: The family $\mathcal{F}$ of minimal optimal solutions for the hereditary minimization problem.

Using FindMinimalOptimal, compute a minimal optimal solution X^* for $(V, f, \mathcal{I})$. Set $\lambda^* \leftarrow f(X^*)$.
Let $(V' + s, f', \mathcal{I}')$ be the system obtained by fusing X^* and all the loops of $\mathcal{I}$ into a single element, denoted s. (During the execution of the algorithm, keep s as an element *outside* V'.)
$\mathcal{I}' \leftarrow \mathcal{I}' \setminus \{A \in \mathcal{I}' : s \in A\}$. ▷ If s is not a loop, consider it as one.
Let $\mathcal{F} = \{X^*\}$.
for each $v \in V$ with $f'(v) = \lambda^*$ **do**
    Add $\{v\}$ to $\mathcal{F}$.
    Update $(V' + s, f', \mathcal{I}')$ by fusing $\{s, v\}$ into s.
end for
while $|V'| \geq 2$ **do** ▷ $f'(v) > \lambda^*$ for all $v \in V'$, and s is the only loop of $\mathcal{I}'$.
    Find a pendant pair (t, u) of f' not containing s.
    if $\{t, u\} \in \mathcal{I}'$ and $f'(\{t, u\}) = \lambda^*$ **then**
        Add $V_{\{t,u\}}$ to $\mathcal{F}$.
        Update $(V' + s, f', \mathcal{I}')$ by fusing $\{s, t, u\}$ into s.
    else if $\{t, u\} \in \mathcal{I}'$ and $f'(\{t, u\}) > \lambda^*$ **then**
        Update $(V' + s, f', \mathcal{I}')$ by fusing $\{t, u\}$ into tu.
    else ▷ $\{t, u\} \notin \mathcal{I}'$.
        Update $(V' + s, f', \mathcal{I}')$ by fusing $\{s, t, u\}$ into s.
    end if
end while
Return the family $\mathcal{F}$.

Proof. Since all the solutions added to $\mathcal{F}$ are disjoint and have minimum f-value, it is enough to show that every minimal optimal solution is eventually added to $\mathcal{F}$. Suppose that this is not the case, and let $Y \in \mathcal{I}$ be an inclusion-wise minimal minimizer of $(V, f, \mathcal{I})$ with $Y \notin \mathcal{F}$.

We first claim that at every moment and for every $w \in V' + s$, the associated set V_w is always completely inside or completely outside Y. We prove this by induction. The claim is true at the beginning of the algorithm and after the fusion of all loops and X^*. Since the minimal optimal solutions are disjoint, this also holds after all the optimal singletons are added to $\mathcal{F}$ and fused into s.

Suppose that the claim holds at the beginning of an iteration in the while-loop and let (t, u) be the pendant pair found at that moment. Note that Y can not separate t from u, since in that case we would have $f'(u) = f(Y) = \lambda^*$. But, by construction, the algorithm ensures that at every iteration the singletons are not optimal, i.e., $f'(v) > \lambda^*$ for every $v \in V'$, contradicting the previous sentence. It follows that V_t and V_u are both either completely inside or both completely outside Y. If all the elements participating in a fusion at this iteration are completely inside or completely

outside Y then the claim still holds at the end of the iteration. The only case left to consider is when V_t and V_u are inside Y, V_s is outside and we fuse $\{s,t,u\}$ together into s. We only do this when $\{t,u\} \in \mathcal{I}'$ and $f'(\{t,u\}) = \lambda^*$ or when $\{t,u\} \notin \mathcal{I}'$. As $V_{\{t,u\}} \subseteq Y \in \mathcal{I}$, we must be in the first case. Then, according to the algorithm, $V_{\{t,u\}} = V_t \cup V_u$ is added to $\mathcal{F}$. By minimality of Y we obtain $Y = V_{\{t,u\}}$ which contradicts the fact that $Y \notin \mathcal{F}$ and proves the claim.

Since Y is never added to $\mathcal{F}$, and at every moment $V_s \supseteq X^*$ is completely outside Y, the previous claim implies that after the while-loop, the set Y corresponds to the unique element in V', say $Y = V_u$, for $V' = \{u\}$. As we maintained that the singletons cannot be optimal, we get a contradiction. □

Similar to the case of Algorithm 19, if algorithm $\mathcal{A}$ is able to compute pendant pairs of (V', f') avoiding a given element in $O(T(|V'|))$-time and using $O(T(|V'|))$ calls to some oracle, for some increasing function $T(\cdot)$, we conclude that.

Theorem 7.3.5. *By using $\mathcal{A}$ as a subroutine, Algorithm 20 returns all minimal optimal solutions of the strongly admissible hereditary system $(V, f, \mathcal{I})$ in $O(nT(n))$-time and using the same asymptotic number of oracle calls.*

Proof. Direct from the fact that each iteration decreases the cardinality of V' by at least one unit. □

7.4 Set and Bi-set Functions

In this section, we study different families of strongly admissible set functions and show how to find pendant pairs in all of them. The families presented are not new and can be found in different articles related to Queyranne's symmetric submodular function algorithm [127, 145, 130].

Let V be a finite ground set of size n and $\mathcal{Q}(V)$ be the collection of disjoint pairs of subsets of V,

$$\mathcal{Q}(V) = \{(A,B) \colon A, B \subseteq V, A \cap B = \emptyset\}.$$

A **bi-set function** on V is a function $d \colon \mathcal{Q}(V) \to \mathbb{R}$. Just like the case of set functions, we assume that bi-set functions are given through a value oracle, that is an oracle that given a pair of disjoint sets (A,B), returns $d(A,B)$.

A bi-set function d on V is **symmetric** if for all $A \subseteq V$, $d(A,B) = d(B,A)$.

Every symmetric bi-set function admits a canonical symmetric set function and vice versa. Given a symmetric bi-set function d on V, the symmetric set function $f^{(d)}$ is defined as:

$$f^{(d)}(A) = d(A, V \setminus A), \text{ for all } A \subseteq V.$$

Given a (not necessarily symmetric) set function f on V, the symmetric bi-set function

$d^{(f)}$ is defined[2] as:

$$d^{(f)}(A,B) = \frac{1}{2}\left(f(A) + f(B) + f(\emptyset) - f(A \cup B)\right), \text{ for all } A \neq B.$$

Lemma 7.4.1. *If f is a symmetric function on V, then $f = f^{(d^{(f)})}$.*

Proof. Let $d = d^{(f)}$, and $\hat{f} = f^{(d)}$, then for all $A \subseteq V$ we have

$$\hat{f}(A) = d(A, V \setminus A) = \frac{1}{2}\left(f(A) + f(V \setminus A) + f(\emptyset) - f(V)\right) = f(A). \quad \square$$

However, even for symmetric d, the functions d and $d^{(f^{(d)})}$ can be extremely different. For example, consider the non constant function

$$d(A,B) = |A| + |B|, \text{ for all } (A,B) \in \mathcal{Q}(V).$$

then $f^{(d)}$ is a constant function equal to n, and $d^{(f^{(d)})}$ is also constant equal to n.

We conclude this section with a natural example of a set and bi-set function. Let $G = (V, E, w)$ be a weighted graph with $w\colon E \to \mathbb{R}$. For all pairs (A, B) of disjoint sets, define $E(A : B)$ as the set of edges having one endpoint in A and one endpoint in B. The **cut between sets A and B** is defined as

$$d_G(A,B) = w(E(A : B)), \text{ for all } (A,B) \in \mathcal{Q}(V);$$

and the **cut function** of G (also called the **weighted degree function** of G) is $f_G\colon V \to \mathbb{R}$ defined as

$$f_G(A) = w(E(A : V \setminus A)), \text{ for all } A \subseteq V.$$

It is easy to see that $f^{(d_G)} = f_G$ and $d^{(f_G)} = d_G$.

7.4.1 Fusions and Minors

We can extend the notion of fusion to bi-set functions. Let Π be a partition of V into a collection of sets. The **fusion of d relative to Π**, denoted by d_Π is the bi-set function on Π given by

$$d_\Pi(X,Y) = d(V_X, V_Y), \tag{7.6}$$

where we recall that $V_X = \bigcup_{S \in X} S$. Note that if d is symmetric then so are all their fusions.

We focus now on certain operations on set functions. Consider a set function f on V and let $S \subseteq V$ and $\overline{S} = V \setminus S$. The function obtained from f by **deleting** $\overline{S}$,

[2]It is also possible to remove the term $f(\emptyset)$ or change its sign in the definition. But this definition is better behaved.

also known as the **restriction** of f to S, is defined as

$$f \setminus \overline{S} = f.S \colon 2^S \to \mathbb{R}, \text{ where } f.S(A) = f(A), \text{ for all } A \subseteq S. \tag{7.7}$$

The function obtained from f by **contracting** $\overline{S}$, also known as the **contraction** of f to S, is defined as

$$f/\overline{S} = f \times S \colon 2^S \to \mathbb{R}, \text{ where } f \times S(A) = f(A \cup \overline{S}) - f(\overline{S}), \text{ for all } A \subseteq S. \tag{7.8}$$

We say that a restriction or contraction is **non-trivial** if the associated set S above satisfies $\emptyset \subset S \subset V$.

It is easy to see that deletion and contraction commute, that is, if S, T are disjoint subsets of V, then $(f/S) \setminus T = (f \setminus T)/S$. Any function obtained from f by deleting and contracting subsets is called a **minor** of f.

Let f be an arbitrary set function on V and s be an element outside V. The **anti-restriction** of f with extra element s is the function g on $V + s$ such that

$$g(A) = \begin{cases} f(A), & \text{if } s \notin A, \\ f(V \setminus A), & \text{if } s \in A. \end{cases} \tag{7.9}$$

The **anti-contraction** of f with extra element s is the function h on $V + s$ such that

$$h(A) = \begin{cases} f(V \setminus A), & \text{if } s \notin A, \\ f(A - s), & \text{if } s \in A. \end{cases} \tag{7.10}$$

Lemma 7.4.2.

1. *Let g be the anti-restriction with extra element s of a set function f on V, then $g \setminus \{s\} = f$*
2. *Let h be the anti-contraction with extra element s of a set function f on V with $f(\emptyset) = 0$, then $h/\{s\} = f$.*

Proof. The first item follows since by definition, for all $A \subseteq V$, $g \setminus \{s\}(A) = g(A) = f(A)$. The second item holds since for all $A \subseteq V$, $h/\{s\}(A) = h(A + s) - h(s) = f(A) - f(\emptyset)$. □

An application of anti-restrictions and anti-contractions is the following.

Lemma 7.4.3. *Let g and h be the anti-restriction and anti-contraction with extra element s of a set function f on V and let P be a property of set functions closed by taking fusions and by adding constant functions. Then,*

1. *Both functions g and h are symmetric;*
2. *f is a nontrivial restriction of a symmetric function with property P if and only if its anti-restriction g has property P;*

3. *f is a nontrivial contraction of a symmetric function with property P if and only if $f(\emptyset) = 0$ and its anti-contraction h has property P.*

Proof. The first item follows directly from the definitions of g and h. The sufficiency of the second and third items are due to Lemma 7.4.2. We only need to check necessity.

For the second item, suppose that $f = f' \setminus T$ for some symmetric set function f' with property P. Let $\hat{f}$ be the function obtained from f' by fusing T into a single element which we denote as s. The function $\hat{f}$ is symmetric and has property P, and furthermore it is easy to see that $f = \hat{f} \setminus \{s\}$. To conclude this part, we show that $\hat{f}$ is equal to the anti-restriction g of f. Indeed, for all $A \subseteq V$,

$$\begin{aligned} g(A) = f(A) = \hat{f}(A), \text{ and} \\ g(A + s) = g(V \setminus A) = f(V \setminus A) = \hat{f}(V \setminus A) = \hat{f}(A + s). \end{aligned}$$

For the third item, let $f = f'/T$ for some symmetric set function f' with property P. Similar to above, let $\hat{f}$ be the function obtained from f' by fusing T into a single element s. Then f is equal to $\hat{f}/\{s\}$. In particular, $\hat{f}$ is symmetric, has property P and

$$f(\emptyset) = \hat{f}(\emptyset + s) - \hat{f}(s) = 0.$$

Moreover, for all $A \subseteq V$, we have

$$\begin{aligned} h(A) = f(V \setminus A) = \hat{f}(V \setminus A + s) - \hat{f}(s) = \hat{f}(A) - \hat{f}(s), \text{ and} \\ h(A + s) = f(A) = \hat{f}(A + s) - \hat{f}(s). \end{aligned}$$

Therefore, the anti-contraction h of f is equal to $\hat{f}$ shifted by the constant $\hat{f}(s)$. We conclude the proof by recalling that P is closed under the addition of constants. □

We also have the following property.

Lemma 7.4.4. *Let g be the anti-restriction with extra element s of a set function f on V. Given an hereditary system $\mathcal{I}$ on V, the solutions for the hereditary minimization problem on $(V, f, \mathcal{I})$ coincide with the solutions for the hereditary system $(V + s, g, \mathcal{I})$.*

Proof. This follows since $f = g \setminus \{s\}$ and s is a loop of $\mathcal{I}$ when viewed as an hereditary system on $V + s$. □

7.4.2 Submodular Functions

Consider two different sets $A, B \subseteq V$. We say that A and B are **intersecting** if $A \setminus B$, $B \setminus A$, and $A \cap B$ are all nonempty. We further say that A and B are **crossing** if they are intersecting and the set $V \setminus (A \cup B)$ is also nonempty.

Consider the following submodular inequality,

$$f(A \cup B) + f(A \cap B) \leq f(A) + f(B). \tag{7.11}$$

A set function $f\colon 2^V \to \mathbb{R}$ is called **fully (resp., intersecting, crossing) submodular** if inequality (7.11) is satisfied for every pair A and B in V (resp., for every pair of intersecting sets or crossing sets). Fully submodular functions are what we usually denote as submodular functions. However, from this point on we keep the adjective "fully" to avoid confusion. Note that every fully submodular function is intersecting submodular and every intersecting submodular function is crossing submodular.

The function f is called **fully supermodular** if $-f$ is fully submodular, and **fully modular** if it is both fully submodular and fully supermodular. We extend these definitions to their intersecting and crossing versions.

Consider the following examples.

1. The cut function of a nonnegatively weighted undirected or directed graph is fully submodular.

2. The function $f\colon \{1,2,3\} \to \mathbb{R}$ given by

$$f(A) = \begin{cases} 1, & \text{if } |A| = 2; \\ 0, & \text{otherwise.} \end{cases}$$

   The only possibility for two sets A and B to be intersecting is that both of them have cardinality 2 and intersect in a single element. In this case $2 = f(A) + f(B) \geq f(A \cup B) - f(A \cap B) = 0$. Hence f is intersecting submodular. But it is not fully submodular since $0 = f(1) + f(2) \leq f(\{1,2\}) + f(\emptyset) = 1$.

3. The function $g\colon \{1,2,3,4\} \to \mathbb{R}$ given by

$$g(A) = \begin{cases} -1, & \text{if } A \in E_1 = \{\{1,2\},\{3,4\}\}; \\ 1, & \text{if } A \in E_2 = \{\{1,3\},\{1,4\},\{2,3\},\{2,4\}\}; \\ 0, & \text{otherwise.} \end{cases}$$

   The only possibility for two sets A and B to be crossing is that both of them have cardinality 2, intersect in a single element and leave one element outside their union. At least one of A or B must then be in E_2, from here we have that $g(A) + g(B) \geq 0 = g(A \cup B) + g(A \cap B)$. This implies that g is crossing submodular. However, g is not intersecting submodular since $-1 = g(\{1,2,3\}) + g(\{3,4\}) \leq g(\{1,2,3,4\}) + g(3) = 0$.

The following lemma states properties of the fusions, restrictions and contractions of submodular functions.

Lemma 7.4.5. *Let $f\colon 2^V \to \mathbb{R}$ be a set function.*

1. *If f is (fully, intersecting, crossing) submodular then so are all its fusions.*

2. *If f is (fully, intersecting, crossing) submodular then so are all its restrictions and contractions.*

3. *If f is crossing submodular then its non-trivial restrictions are intersecting submodular.*

Proof. The proof of the first item is straightforward from the definitions. For the second item note that if f is (fully, intersecting, crossing) submodular function on V and T is a nontrivial subset of V then for all $A, B \subseteq V \setminus T$,

$$\begin{aligned} f \setminus T(A) + f \setminus T(B) - f \setminus T(A \cup B) - f \setminus T(A \cap B) \\ = f(A) + f(B) - f(A \cup B) - f(A \cap B), \end{aligned} \tag{7.12}$$

implying that $f \setminus T$ is also (fully, intersecting, crossing) submodular. On the other hand,

$$\begin{aligned} &f/T(A) + f/T(B) - f/T(A \cup B) - f/T(A \cap B) \\ &= f(A \cup T) + f(B \cup T) - f((A \cup T) \cup (B \cup T)) - f((A \cap T) \cap (B \cap T)). \end{aligned} \tag{7.13}$$

Note that if A and B have nonempty intersection, then they are intersecting (resp. crossing) in $V \setminus T$ if and only if $A \cup T$ and $B \cup T$ are intersecting (resp. crossing) in V. Therefore, the function f/T is (fully, intersecting, crossing) submodular.

The third item follows from the fact that if A and B are two intersecting sets in $V \setminus T$ such that $A \cup B = V \setminus T$ then the sets A and B are crossing in V, therefore by (7.12) the submodular inequality holds for $f \setminus T$. □

7.4.3 Posimodular Functions

A set function $f \colon 2^V \to \mathbb{R}$ is called **fully posimodular** if for all $A, B \subseteq V$,

$$f(A \setminus B) + f(B \setminus A) \leq f(A) + f(B). \tag{7.14}$$

We say that f is **intersecting posimodular** or **crossing posimodular** if inequality (7.14) is satisfied for every pair A and B of intersecting sets or crossing sets respectively. A function f is **(fully, intersecting, crossing) negamodular** if $-f$ is (fully, intersecting, crossing) posimodular.

Lemma 7.4.6. *Every symmetric fully submodular function is fully posimodular, and every symmetric crossing submodular function is intersecting posimodular.*

Proof. Let f be a symmetric set function on V. Consider first the case where f is fully submodular. By symmetry,

$$\begin{aligned} f(A) + f(B) = f(V \setminus A) + f(B) \geq f((V \setminus A) \cup B) + f((V \setminus A) \cap B) \\ = f(A \setminus B) + f(B \setminus A), \end{aligned} \tag{7.15}$$

for all $A, B \subseteq V$. This implies fully posimodularity.

Now assume that f is only crossing submodular. Let A and B be two intersecting sets. We have two cases: either $A \cup B = V$ or A and B are crossing.

In the first case, $V \setminus A = B \setminus A$ and $V \setminus B = A \setminus B$, then we have

$$f(A) + f(B) = f(V \setminus A) + f(V \setminus B) = f(B \setminus A) + f(A \setminus B).$$

In the second case, $V \setminus A$ and B are also crossing, hence (7.15) also holds. □

The converse of the previous lemma does not hold. For example, the sum of a fully modular function and a symmetric fully submodular function is always fully submodular and fully posimodular but it is not necessarily symmetric. It is also easy to construct examples of functions that are fully posimodular but not fully submodular. One such example is the indicator function of the ground set V.

Since (fully, intersecting, crossing) submodularity is preserved under taking fusions and adding constants, Lemma 7.4.3 in the previous section allows us to test whether a function is a restriction or a contraction of a symmetric (fully, intersecting, crossing) submodular function by looking at its anti-restriction or anti-contraction. As a corollary, we have the following.

Lemma 7.4.7 (Implicit in Nagamochi and Ibaraki [127] and Narayanan [130])**.** *The nontrivial restrictions of symmetric crossing submodular functions are exactly those functions that are both intersecting submodular and intersecting posimodular.*

Proof. Let f be a set function on V and g be its anti-restriction with extra element s. By Lemma 7.4.3 we need to check that the following two statements are equivalent:

(i) g is symmetric crossing submodular.

(ii) f is intersecting submodular and intersecting posimodular.

Suppose that g is symmetric crossing submodular. Then condition (ii) holds by using Lemmas 7.4.5 and Lemma 7.4.6.

Suppose then that f satisfies condition (ii). We already know, by definition, that g is symmetric. Let A and B be two crossing sets in $V + s$.

If $s \notin A \cup B$, then we have

$$g(A) + g(B) - g(A \cup B) - g(A \cap B) = f(A) + f(B) - f(A \cup B) - f(A \cap B) \geq 0,$$

where the last inequality holds by intersecting submodularity of f.

If $s \in A \cap B$, then let $A' = A - s$ and $B' = B - s$. We have

$$\begin{aligned} g(A) + g(B) &= f(V \setminus A') + f(V \setminus B') \\ &\geq f((V \setminus A') \cup (V \setminus B')) + f((V \setminus A') \cap (V \setminus B')) \\ &= f(V \setminus (A' \cap B')) + f(V \setminus (A' \cup B')) \\ &= g((A' \cap B') + s) + g((A' \cup B') + s) \\ &= g(A \cap B) + g(A \cup B), \end{aligned}$$

where the inequality holds by intersecting submodularity of f.

For the remaining cases assume that $s \in A \setminus B$ without loss of generality and let $A' = A \setminus s$. By intersecting posimodularity of f we have

$$\begin{aligned} g(A) + g(B) &= f(V \setminus A') + f(B) \\ &\geq f((V \setminus A') \setminus B) + f(B \setminus (V \setminus A')) \\ &= f(V \setminus (A' \cup B)) + f(A' \cap B) \\ &= g(A \cup B) + g(A \cap B). \end{aligned}$$

Hence, g is crossing submodular. □

7.4.4 Rizzi Functions

Rizzi [145] introduces a nice family of bi-set functions, which we describe below. We say that a symmetric bi-set function d on V is a **Rizzi bi-set function** if the following properties hold.

1. (**Consistency**) For all $A, B, C \subseteq V$ disjoint:

$$d(A, B) \leq d(A, C) \text{ implies } d(A \cup C, B) \leq d(A \cup B, C).$$

2. (**Monotonicity**) For all nonempty disjoint $A, B, C \subseteq V$:

$$d(A, B) \leq d(A, B \cup C).$$

The associated **Rizzi set function** is $f = f^{(d)}$.

Lemma 7.4.8. *For every set function f, the bi-set function $d^{(f)}$ is symmetric and consistent.*

Proof. The function $d^{(f)}(A, B) = \frac{1}{2}(f(A) + f(B) + f(\emptyset) - f(A \cup B))$ is symmetric by definition. Consistency holds since

$$\begin{aligned} &d^{(f)}(A \cup C, B) - d^{(f)}(A \cup B, C) \\ &\quad = \frac{1}{2}(f(A \cup C) + f(B) - f(A \cup B) - f(C)) \\ &\quad = d^{(f)}(A, B) - d^{(f)}(A, C). \end{aligned}$$

□

Rizzi observed the following.

Lemma 7.4.9 (Rizzi [145])**.** *If f is an intersecting submodular function on V, then the associated bi-set function[3] $d^{(f)}$ is a Rizzi bi-set function. Furthermore, if f is symmetric, f is the associated Rizzi set function of $d^{(f)}$.*

[3]Rizzi considered the function $d(A, B) = f(A) + f(B) - f(A \cup B)$ instead, but the function we consider satisfies $f = f^{(d^{(f)})}$.

Proof. To show monotonicity. Let A,B,C be nonempty disjoint sets. Then

$$2\Big(d^{(f)}(A, B \cup C) - d^{(f)}(A, B)\Big) \\ = f(B \cup C) - f(A \cup B \cup C) - f(B) + f(A \cup B). \tag{7.16}$$

Let $D = B \cup C$ and $E = A \cup B$, so that $D \cup E = A \cup B \cup C$ and $D \cap E = B$. Since all A, B and C are nonempty, the sets D and E are intersecting. Hence, (7.16) above is nonnegative implying monotonicity.

We have already seen in Lemma 7.4.1 that for symmetric f, $f = f^{(d^{(f)})}$. □

Rizzi [145] mistakenly states that the lemma above holds if we replace intersecting submodularity by the weaker condition of crossing submodularity. However, for the symmetric crossing submodular function $f\colon V \to \mathbb{R}$ satisfying $f(\emptyset) = f(V) = 1$, and $f(X) = 0$ for all $X \notin \{\emptyset, V\}$, where $|V| \geq 3$ this does not holds. Indeed, if $\{A, B, C\}$ is a partition of V in nonempty parts, then

$$2d^{(f)}(A, B \cup C) = f(A) + f(B \cup C) + f(\emptyset) - f(A \cup B \cup C) = 0, \text{ and} \\ 2d^{(f)}(A, B) = f(A) + f(B) + f(\emptyset) - f(A \cup B) = 1.$$

Hence, we do not have monotonicity. One way to fix this problem is to define a weaker version of monotonicity. Consider the following property.

1'. (**Weak Monotonicity**) $d(A, B) \leq d(A, B \cup C)$, for all nonempty disjoint sets $A, B, C \subseteq V$ with $A \cup B \cup C \neq V$.

We call d a **weak Rizzi bi-set function** if it is symmetric, weak monotone and consistent. The associated function $f^{(d)}$ is called a **weak Rizzi set function**.

Lemma 7.4.10. *If f is a crossing submodular function on V, then the associated bi-set function $d^{(f)}$ is a weak Rizzi bi-set function. Furthermore if f is symmetric, f is the associated weak Rizzi set function of $d^{(f)}$.*

Proof. We only need to show weak monotonicity. Indeed, for A, B and C satisfying $A \cup B \cup C \neq \emptyset$, the right hand side of (7.16) is nonnegative by crossing submodularity of f. □

We remark here that recognizing that a function f is a (weak) Rizzi function without having access to the bi-set Rizzi function d for which $f = f^{(d)}$ is not simple. This follows since it is not always the case that $d = d^{(f^{(d)})}$. It is also easy to find Rizzi functions that are not crossing submodular. The following example can be found in Rizzi [145].

Given a weighted connected graph $G = (V, E, w)$, with $w\colon E \to \mathbb{R}_+$, define the **maximum separation** between two disjoint sets of vertices A and B as

$$d(A, B) = \max_{ab \in E(A:B)} w(ab).$$

This function is symmetric, monotone and consistent. However, for the complete graph in $\{a, b, c, d\}$ where all the edges have weight 1, except ac and bd which have weight 2. The corresponding function $f = f^{(d)}$ given by $f(A) = d(A, V \setminus A)$, for $A \subseteq V$ is not crossing submodular since

$$3 = f(\{a, c\}) + f(\{a, d\}) < f(\{a, c, d\}) + f(a) = 4.$$

The function f is not even crossing posimodular since

$$3 = f(\{a, c\}) + f(\{a, d\}) < f(c) + f(d) = 4.$$

In particular, we have the following result.

Lemma 7.4.11. *The class of restrictions of weak Rizzi functions strictly contains the restrictions of symmetric crossing submodular functions.*

Proof. The containment follows from Lemma 7.4.10. The fact that the containment is strict follows from the example above and Lemma 7.4.5. □

A useful property of (weak) Rizzi bi-set functions is the following

Lemma 7.4.12. *If d is a (weak) Rizzi bi-set function then so are all its fusions.*

Proof. Direct from the definitions. □

Rizzi has shown that a version of the maximum adjacency ordering, defined by Queyranne allows us to find pendant pairs for Rizzi set functions. His proof, which we describe below, naturally extends to weak Rizzi set functions.

Let d be a weak Rizzi bi-set function. An ordering $(v_1, \ldots, v_n)$ of the elements of V, such that

$$d(v_i, W_{i-1}) \geq d(v_j, W_{i-1}), \text{ for all } 2 \leq i \leq j \leq n, \tag{7.17}$$

where v_1 can be chosen arbitrarily, and W_i denotes the set $\{v_1, \ldots, v_i\}$ is called a **maximum adjacency ordering** for d. The origin of the name comes from its interpretation for the cut function on a graph. If $d(A, B)$ represents the cut between A and B in a given graph G, then the i-th vertex of a maximum adjacency ordering of d is exactly the one having maximum number of edges going towards the previous $i - 1$ vertices.

Lemma 7.4.13 (Essentially in Rizzi [145])**.** *For a weak Rizzi bi-set function d on V, and an arbitrary element $v_1 \in V$, the last two elements (v_{n-1}, v_n) of a maximum adjacency ordering of V starting from v_1 constitute a pendant pair for $f^{(d)}$. Furthermore, this order can be found by using $O(n^2)$ oracle calls to d and in the same running time.*

Proof. It is easy to check the claim regarding the running time since to construct the i-th element of the ordering we only require to find the maximum of the $n - i$ different values $\{d(x, W_{i-1})\}_{x \in V \setminus W_{i-1}}$. We show the rest by induction on the number of elements.

The lemma holds trivially for $n = 2$ since the only set separating v_1 from v_2 are the singletons and the function $f^{(d)}$ is symmetric. For $n = 3$, the only sets separating v_2 from v_3 are $\{v_3\}$, $\{v_1, v_3\}$ and their complements. By definition of the ordering, $d(v_2, v_1) \geq d(v_3, v_1)$. Consistency implies that

$$f^{(d)}(\{v_1, v_3\}) = d(\{v_1, v_3\}, v_2) \geq d(\{v_1, v_2\}, v_3) = f^{(d)}(v_3).$$

Consider $n \geq 4$ and let S be any set separating v_n and v_{n-1}. We must show

$$d(S, V \setminus S) \geq d(v_n, V - v_n). \tag{7.18}$$

It is easy to see that $(v_{\{1,2\}}, v_3, \ldots, v_n)$ is a maximum adjacency ordering for the function $d_{1,2}$ obtained by fusing v_1 and v_2 into $v_{\{1,2\}}$. If S does not separate v_1 and v_2 then (7.18) holds since (v_{n-1}, v_n) is a pendant pair of $d_{1,2}$ by induction. So assume that S separates v_1 and v_2.

We claim that the ordering $(v_1, v_{\{2,3\}}, \ldots, v_n)$ is a maximum adjacency ordering for the function $d_{2,3}$ obtained by fusing v_2 and v_3 into $v_{\{2,3\}}$. Indeed, we only need to prove that $d_{2,3}(v_{\{2,3\}}, v_1) \geq d_{2,3}(v_j, v_1)$ for all $j \geq 4$. This follows since, by hypothesis and weak monotonicity, we have

$$d(v_j, v_1) \leq d(v_2, v_1) \leq d(\{v_2, v_3\}, v_1).$$

If S does not separate v_2 and v_3 then (7.18) holds by induction, since (v_{n-1}, v_n) is a pendant pair for $d_{2,3}$.

The only possibility left is that S separates v_1 from v_2 and v_2 from v_3. This means that S does not separate v_1 and v_3. To conclude (7.18) it suffices to show that $(v_2, v_{\{1,3\}}, \ldots, v_n)$ is a maximum adjacency ordering for the function $d_{1,3}$ obtained by fusing v_1 and v_3 into $v_{\{1,3\}}$. Assume that this is not the case, then we must have

$$d_{1,3}(v_{\{1,3\}}, v_2) < d_{1,3}(v_j, v_2)$$

for some $j \geq 4$. Since $(v_1, \ldots, v_n)$ is a maximum adjacency ordering of d, we have $d(v_2, v_1) \geq d(v_3, v_1)$ and $d(v_3, \{v_1, v_2\}) \geq d(v_j, \{v_1, v_2\})$. By consistency we also have $d(\{v_1, v_3\}, v_2) \geq d(\{v_1, v_2\}, v_3)$. Combining the inequalities above and using weak monotonicity we get

$$\begin{aligned} d(v_3, \{v_1, v_2\}) \geq d(v_j, \{v_1, v_2\}) \geq d(v_j, v_2) &= d_{1,3}(v_j, v_2) \\ &> d_{1,3}(v_{\{1,3\}}, v_2) = d(\{v_1, v_3\}, v_2) \geq d(v_3, \{v_1, v_2\}), \end{aligned}$$

which is a contradiction. □

7.4.5 Main Results

By combining Lemmas 7.4.13 and 7.4.12 we conclude that weak Rizzi functions are strongly admissible for pendant pairs. By Lemma 7.4.10, symmetric crossing submodular functions have the same property.

Summarizing, we obtain the following results.

Theorem 7.4.14. *We can compute all the minimal optimal solutions of the hereditary system $(V, f, \mathcal{I})$ in time $O(n^3)$ and using $O(n^3)$ oracle calls for the following cases:*

1. *If $f = f^{(d)}$, for a weak Rizzi bi-set function d on V, and we have access to an oracle for d.*
2. *If f is a symmetric (fully, crossing) submodular function on V and we have access to an oracle for f.*
3. *If f is an intersecting submodular and intersecting posimodular function on V and we have access to an oracle for f.*
4. *If f is defined as*

$$f(A) = d(A, (V \setminus A) + s), \text{ for all } A \subseteq V$$

for some weak Rizzi bi-set function d on $V + s$, and we have access to an oracle for d.

Proof. For all results we use Algorithm 20. The first item follows by Lemma 7.4.13 and the fact that all the fusions of d can be evaluated using oracle calls to d. Item number two holds since by Lemma 7.4.10, the bi-set function $d^{(f)}$ is a weak Rizzi function and $f = f^{(d^{(f)})}$. Therefore, we can use the result of the first item to conclude.

For the third item, we can construct a value oracle for the anti-restriction g of f. This value oracle uses one call to the oracle for f. Let s be the extra element added by the anti-restriction and consider the system $(V + s, g, \mathcal{I})$. By Lemma 7.4.7, g is crossing submodular on $V + s$ so we can use the result of the second item to find all the minimal optimal solutions of $(V + s, g, \mathcal{I})$. By Lemma 7.4.4, these solutions coincide with the ones of the system $(V, f, \mathcal{I})$.

Finally for item number four, consider the system $(V + s, g, \mathcal{I})$ where $g = f^{(d)}$ is the weak Rizzi function on $V + s$ obtained from d. We can find the minimal optimal solutions for this system using the first item in this theorem. We claim that g is the anti-restriction of f with extra element s. Indeed, for all $A \subseteq V$,

$$g(A) = f^{(d)}(A) = d(A, (V + s) \setminus A) = f(A), \text{ and}$$
$$g(A + s) = f^{(d)}(A + s) = d(A + s, V \setminus A) = d(V \setminus A, (V \setminus (V \setminus A)) + s) = f(V \setminus A).$$

By Lemma 7.4.4, the solutions found for $(V + s, g, \mathcal{I})$ coincide with the ones of the system $(V, f, \mathcal{I})$. □

It is worth noting at this point that we can also use our methods to find all the inclusion-wise maximum minimizers of some functions constrained to **co-hereditary families**, that is families of sets closed under union. Given a family of sets $\mathcal{I}$ on V, its **dual** with respect to V is the family $\mathcal{I}^{*V} = \{A \colon V \setminus A \in \mathcal{I}\}$. Then the co-hereditary families are exactly the duals of hereditary family. The set function dual f^{*V} of f is defined as $f^{*V}(A) = f(V \setminus A)$ for all $A \subseteq V$. A triple $(V, f, \mathcal{J})$ is

a co-hereditary system if $(V, f^{*V}, \mathcal{J}^{*V})$ is an hereditary system. A set $X \subseteq V$ is a **maximal optimal** solution for a co-hereditary system $(V, f, \mathcal{I})$ if X is a nontrivial inclusion-wise *maximal* set minimizing f over all the sets in $\mathcal{I}$. Note that the maximal optimal solutions for a co-hereditary system $(V, f, \mathcal{J})$ are exactly the complements of the minimal optimal solutions of the hereditary system $(V, f^{*V}, \mathcal{J}^{*V})$. As a corollary of the previous theorem we have the following.

Theorem 7.4.15. *We can compute all the maximal optimal solutions of the co-hereditary system $(V, f, \mathcal{J})$ in time $O(n^3)$ and using $O(n^3)$ oracle calls for the following cases:*

1. *If $f = f^{(d)}$, for a weak Rizzi bi-set function d on V, and we have access to an oracle for d.*
2. *If f is a symmetric (fully, crossing) submodular function on V and we have access to an oracle for f.*
3. *If the anti-contraction of f is a symmetric crossing submodular function on V and we have access to an oracle for f.*
4. *If f is defined as*

$$f(A) = d(A + s, V \setminus A), \text{ for all } A \subseteq V$$

for some weak Rizzi bi-set function d on $V + s$, and we have access to an oracle for d.

Proof. The theorem follows since the co-hereditary systems $(V, f, \mathcal{J})$ for each case are exactly the duals of the hereditary systems considered in Theorem 7.4.14 □

7.5 Nagamochi's Flat Pair Based Algorithm

In this section we briefly describe some of the ideas in a recent algorithm of Nagamochi [124] that can also be used to solve the hereditary constrained minimization problem. We need the following definitions.

Consider a set function f on V. A nonempty proper subset A of V is called an **extreme set** of f if

$$f(A) < f(B), \text{ for all } \emptyset \subset B \subset A. \tag{7.19}$$

Let $\mathcal{X}(f)$ be the collection of all extreme sets of f.

A **flat pair** of f is an unordered pair of elements $\{t, u\}$ such that

$$f(X) \geq \min_{x \in X} f(x), \text{ for all } X \subseteq V \text{ separating } t \text{ from } u. \tag{7.20}$$

Call a function f **admissible for flat pairs**[4] if f and all its fusions admit flat pairs.

[4] In [124], this class of functions is not named.

As noted by Nagamochi, a non-singleton extreme set A can not separate a flat pair $\{t, u\}$. This fact alone implies that the extreme sets of functions admissible for flat pairs form a laminar family. Nagamochi shows this in an algorithmic way, we give a direct proof below.

Lemma 7.5.1. *The set $\mathcal{X}(f)$ of extreme sets of a function f admissible for flat pairs is laminar.*

Proof. Suppose by contradiction that two extreme sets A and B are intersecting. Let (V', f') be the system obtained by fusing all the elements of $A \setminus B$ into a, all the elements of $B \setminus A$ into b, all the elements of $A \cap B$ into c, and if $V \setminus (A \cup B)$ is nonempty, all the elements of this set into d. Every pair of elements in V' is separated by either A or B. Hence none of them can be a flat pair. □

The above lemma implies that the number of extreme sets of a flat pair admissible function is $O(n)$. Nagamochi gives an algorithm that outputs all the extreme sets of $\mathcal{X}(f)$ for any function f admissible for flat pairs provided we have access to an algorithm that finds a flat pair of any fusion of f. The algorithm is similar to Queyranne's algorithm in the sense that at every iteration we fuse a flat pair together. In what follows we give a high level overview of the algorithm.

Initialize a set $\mathcal{X}$ with the singletons of V. This set contains at every iteration all the extreme sets of f that are contained inside the singletons of the current fused system (in other words, $\mathcal{X}$ contains all the extreme sets completely inside V_x for some $x \in V'$, where V' is the current ground set). In every iteration, the algorithm finds a flat pair $\{t, u\}$, and fuse them into a single element tu. After that, it tests if the set V_{tu} containing all the elements of V that have been fused into the new element tu is extreme by using the information in $\mathcal{X}$. If V_{tu} is extreme, the algorithm adds it to $\mathcal{X}$.

Nagamochi gives an efficient implementation of the above algorithm that runs in $O(nT(n))$-time, where $T(n)$ is the time needed to find a flat pair of a function over a ground set of n elements.

We can use Nagamochi's algorithm to solve the hereditary minimization problem on $(V, f, \mathcal{I})$ where f is admissible for flat pairs as follows. Any minimal optimal solution of $(V, f, \mathcal{I})$ is an extreme solution of f. Hence, to find all the minimal optimal solutions of $(V, f, \mathcal{I})$ we can compute the extreme sets of f, keep the ones contained in $\mathcal{I}$ and find among these the ones having minimum f-value. This algorithm has the same asymptotic time complexity as the algorithm we have presented for the same problem (see Theorem 7.3.5).

It is an interesting question to decide when a function is admissible for flat pairs. Nagamochi has shown that symmetric submodular functions (and also their restrictions) admit flat pairs, and that we can find them by using a similar technique to the one used by Queyranne to find pendant pairs.

More precisely, consider an ordering $(v_1, \ldots, v_n)$ of the elements of V, such that

$$f(v_i) + f(W_{i-1} + v_i) \leq f(v_j) + f(W_{i-1} + v_j), \text{ for all } 1 \leq i \leq j \leq n, \tag{7.21}$$

where W_i denotes the set $\{v_1, \ldots, v_i\}$. An order satisfying (7.21) is called a **minimum degree ordering**. Nagamochi [124] shows the following result:

Lemma 7.5.2 (Nagamochi [124]). *For a symmetric crossing submodular function f on V, the last two elements $\{v_{n-1}, v_n\}$ of a minimum degree ordering of V constitute a flat pair. Furthermore, this ordering can be found by using $O(n^2)$ value oracle calls to f and in the same running time.*

He also extends the above lemma to intersecting submodular intersecting posimodular functions. Given one such function f, modify it so that $f(\emptyset) = f(V) = -\infty$. This does not affect its intersecting submodularity or its intersecting posimodularity. Consider the anti-restriction g of this modified function with extra element s. This function is symmetric crossing submodular. As $g(s) = f(V) = -\infty$, the element s is always the first element of a minimum degree ordering; therefore, by Lemma 7.5.2, g has a flat pair $\{t, u\}$ avoiding s. This pair $\{t, u\}$ is also a flat pair for the original function f since if there was a subset $X \subseteq V$ separating t and u for which $f(X) < \min_{x \in X} f(x)$, we would also have $g(X) < \min_{x \in X} g(x)$ contradicting the fact that $\{t, u\}$ is a flat pair for g.

In what follows, we show a small modification to the lemma above that allows us to find flat pairs for a slightly larger family of functions.

Consider a symmetric bi-set function d on V satisfying the following property.

$$\begin{aligned} &d(A \cup B, C) \geq d(A, B \cup C) \text{ implies } d(A \cup B, C \cup D) \geq d(A, B \cup C \cup D), \\ &\text{for all nonempty and disjoint } A, B, C, D \subseteq V. \end{aligned} \tag{7.22}$$

Let $(v_1, \ldots, v_n)$ be an ordering of the elements of V, such that

$$d(v_i, (V \setminus W_{i-1}) - v_i) \leq d(v_j, (V \setminus W_{i-1}) - v_j), \text{ for all } 1 \leq i \leq j \leq n, \tag{7.23}$$

where W_i denotes the set $\{v_1, \ldots, v_i\}$. As before, call an ordering satisfying (7.23) a **minimum degree ordering** of d. This name also comes from its interpretation for the case where $d(A, B)$ represents the cut between A and B. In this case, the i-th vertex of the ordering is selected as the one having minimum degree on the graph obtained by removing the first $i - 1$ vertices.

Lemma 7.5.3. *Let d be a symmetric bi-set function on V satisfying* (7.22)*. The last two elements $\{v_{n-1}, v_n\}$ of a minimum degree ordering of d constitute a flat pair of the set function f defined by $f(A) = d(A, V \setminus A)$.*

Proof. The proof is essentially the same as the one given by Nagamochi [124] to prove Lemma 7.5.2. We include it here for completeness.

Define for every $i = n - 2, \ldots, 0$, the symmetric function $f_i \colon (V \setminus W_i) \to \mathbb{R}$ by

$$f_i(X) = d(X, (V \setminus W_i) \setminus X), \text{ for all } X \subseteq V \setminus W_i.$$

We claim that for all $i \in \{n-2, n-3, \ldots, 0\}$, $\{v_n, v_n\}$ is a flat pair of f_i. This is,

$$f_i(X) \geq \min_{x \in X} f_i(x), \text{ for all } X \subseteq V \setminus W_i \text{ separating } v_{n-1} \text{ and } v_n. \tag{7.24}$$

Note that $f_0(X) = f(X)$, hence to prove the lemma we need to show the above claim holds for $i = 0$.

We prove the claim by induction. The case $i = n-2$ is trivial as $\{v_{n-1}\}$ and $\{v_n\}$ are the only sets in $V \setminus W_{n-2}$ separating v_{n-1} and v_n. Suppose that (7.24) holds for $i = j$, we prove that it also holds for $i = j-1$. Let X be subset of $V \setminus W_{j-1}$ separating v_{n-1} from v_n. If $|X| = 1$ then (7.24) holds trivially, so assume that $|X| \geq 2$.

We have two cases

Case 1: $v_j \notin X$. By hypothesis, there is an element $x^* \in X$ such that $f_j(X) \geq f_j(x^*)$. Set $A = \{x^*\}$, $B = X - x^*$, $C = ((V \setminus W_j) \setminus X)$ and $D = \{v_j\}$. Thus A, B, C, D is a partition of $V \setminus W_{j-1}$ in nonempty sets.

Note that $f_j(X) \geq f_j(x^*)$ is equivalent to

$$d(A \cup B, C) \geq d(A, B \cup C).$$

By (7.22), this implies that

$$d(A \cup B, C \cup D) \geq d(A, B \cup C \cup D),$$

or equivalently, $f_{j-1}(X) \geq f_{j-1}(x^*)$, completing the proof of this case.

Case 2: $v_j \in X$. By choice of v_j, $f_{j-1}(v_j) \leq f_{j-1}(x)$ for all $x \in V \setminus W_{j-1}$. Consider the set $Y = (V \setminus W_j) \setminus X$ not containing v_j, which separates v_{n-1} and v_n. By the first case $f_{j-1}(Y) \geq \min_{y \in Y} f_{j-1}(y)$.

Then we have

$$f_{j-1}(X) = f_{j-1}(Y) \geq \min_{y \in Y} f_{j-1}(y) \geq f_{j-1}(v_j),$$

completing the proof. □

We now show that the functions obtained from bi-set functions d satisfying (7.22) include symmetric crossing submodular functions. Indeed, let f be a symmetric crossing submodular function. As f is symmetric, $f = f^{(d^{(f)})}$. We show that $d^{(f)}$ satisfies (7.22). Indeed suppose that A, B, C, D are disjoint nonempty subsets of V. Then $d^{(f)}(A \cup B, C) \geq d^{(f)}(A, B \cup C)$ is equivalent to

$$f(A \cup B) + f(C) \geq f(A) + f(B \cup C). \tag{7.25}$$

Since $B \cup C$ and $C \cup D$ are crossing sets, we also have

$$f(B \cup C) + f(C \cup D) \geq f(B \cup C \cup D) + f(C). \tag{7.26}$$

Summing (7.25) and (7.26) we get

$$f(A \cup B) + f(C \cup D) \geq f(A) + f(B \cup C \cup D) \tag{7.27}$$

which is equivalent to $d^{(f)}(A \cup B, C \cup D) \geq d^{(f)}(A, B \cup C \cup D)$.

Recall the maximum separation function $d(A, B) = \max_{e \in E(A:B)} w(e)$ for a given weighted graph $G = (V, E, w)$. As shown in Section 7.4.4, the function $f = f^{(d)}$ is a Rizzi function that is not necessarily crossing submodular. Note that f also satisfies (7.22). This shows that Lemma 7.4.10 is a strict generalization of Lemma 7.5.2. In

fact, even though our notion of minimum degree ordering (7.23) finds a flat pair, Nagamochi's minimum degree ordering (7.21) does not necessarily find one for the function f defined in this paragraph.

As an example of the last assertion, consider the complete graph G on 6 vertices $\{a, b, c, a', b', c'\}$, assign a weight of one to every edge in the triangles $T_1 = \{a, b, c\}$ and $T_2 = \{a', b', c'\}$ and zero weight to all the other edges. See Figure 7-1.

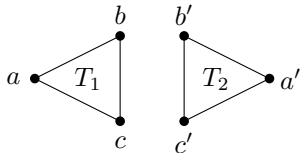


Figure 7-1: Present edges have unit weight. Non-edges have zero weight.

We have

$$d(A, B) = \begin{cases} 0 & \text{if } A \subseteq T_1 \text{ and } B \subseteq T_2 \text{ or vice versa;} \\ 1 & \text{otherwise.} \end{cases}$$

Therefore, $f(A) = 0$ if and only if $A \in \{T_1, T_2\}$.

Under Nagamochi's notion (7.21) of minimum degree ordering, the i-th vertex of the ordering is selected as the one minimizing the function $g_i(x) = f(x)+f(W_{i-1}+x)$, where W_j stands for the first j selected vertices. Note that $g_i(x)$ has value 2 unless, $W_{i-1} + x \in \{T_1, T_2\}$, where $g_i(x)$ has unit value. From here it is easy to see that (a, a', b, b', c, c') is a minimum degree ordering of f in the sense of Nagamochi. But the last two vertices $\{c, c'\}$ do not form a flat pair since T_1 separates them and $0 = f(T_1) < \min_{x \in T_1} f(x) = 1$.

Under our notion (7.23) of minimum degree ordering, the i-th vertex is selected as the one minimizing $h_i = d(x, (V \setminus W_{i-1}) - x)$. In this situation, (a, a', b, b', c, c') is not a minimum degree ordering since for $i = 4$,

$$\begin{aligned} h_i(b') &= d(b', (V \setminus W_{i-1}) - b') = d(b', \{c, c'\}) = 1, \text{ but} \\ h_i(c') &= d(c', (V \setminus W_{i-1}) - c') = d(c', \{b, c\}) = 0. \end{aligned}$$

We remark that Nagamochi's notion (7.21) and our notion (7.23) of minimum degree ordering coincide whenever f is symmetric and $d = d^{(f)}$ (in particular, this holds for symmetric crossing submodular functions). In this situation, for the functions g_i and h_i defined above

$$\begin{aligned} 2h_i(x) &= 2d(x, (V \setminus W_{i-1}) - x) = f(x) + f((V \setminus W_{i-1}) - x) - f(\emptyset) + f(V \setminus W_{i-1}) \\ &= f(x) + f(W_{i-1} + x) - f(\emptyset) + f(W_{i-1}) \\ &= g_i(x) - f(\emptyset) + f(W_{i-1}). \end{aligned}$$

Therefore, the minimizers of $h_i(x)$ and $g_i(x)$ on $V \setminus W_{i-1}$ coincide.

7.6 Discussion

In this chapter we have given polynomial time algorithms to solve the hereditary minimization problem for a class of functions strictly containing symmetric crossing submodular functions and their restrictions.

We can compare our results to the ones of Nagamochi [124]. On the one hand, his algorithm can solve the more general problem of finding extreme sets of symmetric crossing submodular functions and their restrictions. On the other hand, our algorithms (Algorithms 19 and 20) work for a strictly larger class of functions.

We have also presented a slight extension of Nagamochi's minimum degree ordering that allows us to find extreme sets of a more general class of functions. Namely, the ones arising from bi-set functions satisfying (7.22), and their restrictions. We have not studied how do these functions compare to weak Rizzi functions and their restrictions, and it would be very interesting to find a superclass of both for which we can find extreme sets efficiently.

A related question is the following. We have seen that the extreme sets of functions admissible for flat pairs form a laminar family (Lemma 7.5.1). The same holds for functions that are strongly admissible for pendant pairs: the proof of this fact is the same as the one of Lemma 7.3.1 stating that the minimal optimal solutions are disjoint. Is it possible to devise an efficient algorithm to compute extreme set of a function given an oracle to find pendant pairs on every fusion?

Nagamochi [124] gives an example of a cut function of a graph for which the pendant pairs and the flat pairs are different. An interesting question is to find natural functions admitting one type of pairs but not the other. Similarly, it is open to find the relation between functions strongly admissible for pendant pairs and functions that are admissible for flat pairs.

Bibliography

[1] M. Ajtai, N. Megiddo, and O. Waarts. Improved algorithms and analysis for secretary problems and generalizations. *SIAM J. Discret. Math.*, 14(1):1–27, 2001. [43]

[2] J. Amilhastre, M. C. Vilarem, and P. Janssen. Complexity of minimum biclique cover and minimum biclique decomposition for bipartite domino-free graphs. *Discrete Applied Mathematics*, 86(2-3):125–144, 1998. [109, 118, 119, 150]

[3] A. von Arnim and C. de la Higuera. Computing the jump number on semi-orders is polynomial. *Discrete Applied Mathematics*, 51(1–2):219–232, 1994. [109]

[4] B. Aronov, E. Ezra, and M. Sharir. Small-size ε-nets for axis-parallel rectangles and boxes. *SIAM Journal on Computing*, 39(7):3248–3282, 2010. [127]

[5] M. Babaioff, M. Dinitz, A. Gupta, N. Immorlica, and K. Talwar. Secretary problems: Weights and discounts. In *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'09, pages 1245–1254, 2009. [13, 42, 43]

[6] M. Babaioff, N. Immorlica, D. Kempe, and R. Kleinberg. A knapsack secretary problem with applications. In M. Charikar, K. Jansen, O. Reingold, and J. D. P. Rolim, editors, *APPROX-RANDOM*, volume 4627 of *Lecture Notes in Computer Science*, pages 16–28. Springer, 2007. [13, 30, 44]

[7] M. Babaioff, N. Immorlica, D. Kempe, and R. Kleinberg. Online auctions and generalized secretary problems. *SIGecom Exchanges*, 7(2):1–11, 2008. [32, 41]

[8] M. Babaioff, N. Immorlica, and R. Kleinberg. Matroids, secretary problems, and online mechanisms. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'07, pages 434–443, 2007. [12, 13, 14, 23, 32, 35, 41, 42, 70, 80, 81]

[9] K. A. Baker, P. C. Fishburn, and F. S. Roberts. Partial orders of dimension 2. *Networks*, 2(1):11–28, 1972. [91, 92]

[10] M. Bateni, M. Hajiaghayi, and M. Zadimoghaddam. Submodular secretary problem and extensions. In M. J. Serna, R. Shaltiel, K. Jansen, and J. D. P.

Rolim, editors, *APPROX-RANDOM*, volume 6302 of *Lecture Notes in Computer Science*, pages 39–52. Springer, 2010. [44]

[11] A. A. Benczúr. Pushdown-reduce: an algorithm for connectivity augmentation and poset covering problems. *Discrete Applied Mathematics*, 129(2-3):233–262, 2003. [149]

[12] A. A. Benczúr, J. Förster, and Z. Király. Dilworth's theorem and its application for path systems of a cycle - implementation and analysis. In J. Nešetřil, editor, *ESA*, volume 1643 of *Lecture Notes in Computer Science*, pages 498–509. Springer, 1999. [138]

[13] C. Berge. Färbung von Graphen, deren sämtliche bzw. deren ungerade Kreise starr sind. *Wissenchaftliche Zeitschrift der Martin-Luther-Universität Halle-Wittenberg Mathematisch-Naturwissenschaftliche Reihe*, 10:114–115, 1961. [103]

[14] K. S. Booth and G. S. Lueker. Testing for the consecutive ones property, interval graphs, and graph planarity using PQ-tree algorithms. *Journal of Computer and System Sciences*, 13(3):335–379, 1976. [99]

[15] V. Bouchitte and M. Habib. The calculation of invariants of ordered sets. In I. Rival, editor, *Algorithms and Order*, volume 255 of *NATO Science Series C*, pages 231–279. Kluwer Academic Publishers, 1989. [15, 109]

[16] A. Brandstädt. The jump number problem for biconvex graphs and rectangle covers of rectangular regions. In J. Csirik, J. Demetrovics, and F. Gécseg, editors, *Fundamentals of Computation Theory*, volume 380 of *Lecture Notes in Computer Science*, pages 68–77. Springer, 1989. [15, 109, 118, 119, 130, 150]

[17] A. Brandstädt, V. B. Le, and J. P. Spinrad. *Graph classes: a survey*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1999. [89]

[18] H. Brönnimann and M. Goodrich. Almost optimal set covers in finite vc-dimension. *Discrete Computational Geometry*, 14(1):463–479, 1995. [127]

[19] J. Bruno and L. Weinberg. The principal minors of a matroid. *Linear Algebra and its Applications*, 4(1):17–54, 1971. [61]

[20] F. T. Bruss. What is known about Robbins' problem? *Journal of Applied Probability*, 42(1):108–120, 2005. [43]

[21] N. Buchbinder, K. Jain, and M. Singh. Secretary problems via linear programming. In F. Eisenbrand and F. B. Shepherd, editors, *IPCO*, volume 6080 of *Lecture Notes in Computer Science*, pages 163–176. Springer, 2010. [44]

[22] P. A. Catlin, J. W. Grossman, A. M. Hobbs, and H.-J. Lai. Fractional arboricity, strength, and principal partitions in graphs and matroids. *Discrete Applied Mathematics*, 40(3):285–302, 1992. [63]

[23] S. Ceroi. *Ordres et Géométrie Plane: Application au Nombre de Sauts*. PhD thesis, Université Montpellier II, 2000. (In French). [109, 150]

[24] S. Ceroi. A weighted version of the jump number problem on two-dimensional orders is NP-complete. *Order*, 20(1):1–11, 2003. [109, 151]

[25] S. Chaiken, D. J. Kleitman, M. Saks, and J. Shearer. Covering regions by rectangles. *SIAM Journal on Algebraic and Discrete Methods*, 2(4):394–410, 1981. [17, 114, 115, 118, 149]

[26] P. Chalermsook and J. Chuzhoy. Maximum independent set of rectangles. In *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'09, pages 892–901, 2009. [127]

[27] G. Chaty and M. Chein. Ordered matchings and matchings without alternating cycles in bipartite graphs. *Utilitas Math.*, 16:183–187, 1979. [106, 108]

[28] M. Chein and M. Habib. Jump number of dags having dilworth number 2. *Discrete Applied Mathematics*, 7(3):243 – 250, 1984. [108]

[29] M. Chein and P. Martin. Sur le nombre de sauts d'une forêt. *C. R. Acad. Sci. Paris*, 275:159–161, 1972. [108]

[30] L. Chen and Y. Yesha. Efficient parallel algorithms for bipartite permutation graphs. *Networks*, 23(1):29–39, 1993. [100]

[31] Y. Chow, S. Moriguti, H. Robbins, and S. Samuels. Optimal selection based on relative rank (the "secretary problem"). *Israel Journal of Mathematics*, 2(2):81–90, 1964. [43]

[32] M. Chudnovsky, N. Robertson, P. Seymour, and R. Thomas. The strong perfect graph theorem. *Annals of Mathematics*, 164(1):51–229, 2006. [103, 131, 133]

[33] V. Chvátal. On certain polytopes associated with graphs. *J. Comb. Theory, Ser. B*, 18(2):138–154, 1975. [104]

[34] J. Cibulka, J. Hladký, A. Kazda, B. Lidický, E. Ondráčková, M. Tancer, and V. Jelínek. The pinning number of overlapping rectangles. Dimacs REU Project (unpublished manuscript), 2006. [126]

[35] O. Cogis and M. Habib. Nombre de sauts et graphes série-parallèles. *RAIRO, Informatique théorique*, 13(1):3–18, 1979. [108]

[36] C. J. Colbourn and W. R. Pulleyblank. Minimizing setups in ordered sets of fixed width. *Order*, 1(3):225–229, 1985. [108]

[37] D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progressions. *J. Symb. Comput.*, 9(3):251–280, 1990. [87]

[38] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, third edition, 2009. [86, 152]

[39] J. C. Culberson and R. A. Reckhow. Covering polygons is hard. *J. Algorithms*, 17(1):2–44, 1994. [118]

[40] E. Dahlhaus. The computation of the jump number of convex graphs. In V. Bouchitté and M. Morvan, editors, *ORDAL*, volume 831 of *Lecture Notes in Computer Science*, pages 176–185. Springer, 1994. [15, 17, 105, 109, 119, 149, 150]

[41] S. Das, M. Sen, A. Roy, and D. West. Interval digraphs: an analogue of interval graphs. *Journal of Graph Theory*, 13(2):189–202, 1989. [97]

[42] M. Dawande. A notion of cross-perfect bipartite graphs. *Information Processing Letters*, 88(4):143–147, 2003. [110, 111]

[43] R. P. Dilworth. A decomposition theorem for partially ordered sets. *The Annals of Mathematics*, 51(1):161–166, 1950. [86]

[44] N. B. Dimitrov and C. G. Plaxton. Competitive weighted matching in transversal matroids. In L. Aceto, I. Damgård, L. A. Goldberg, M. M. Halldórsson, A. Ingólfsdóttir, and I. Walukiewicz, editors, *ICALP*, volume 5125 of *Lecture Notes in Computer Science*, pages 397–408. Springer, 2008. [13, 42]

[45] D. Duffus, I. Rival, and P. Winkler. Minimizing setups for cycle-free ordered sets. *Proceedings of the American Mathematical Society*, 85(4):509–513, 1982. [107, 108]

[46] S. Dughmi. Submodular functions: Extensions, distributions, and algorithms a survey. PhD Qualifying Exam Report, Department of Computer Science, Stanford University. ArXiv version in `http://arxiv.org/abs/0912.0322`, 2009. [162]

[47] B. Dushnik and E. W. Miller. Partially ordered sets. *American Journal of Mathematics*, 63(3):600–610, 1941. [88, 92]

[48] E. B. Dynkin. The optimum choice of the instant for stopping a markov process. *Soviet Mathematics – Doklady*, 4:627–629, 1963. [12, 13, 27]

[49] J. Edmonds. Minimum partition of a matroid into independent subsets. *Journal of research of the National Bureau of Standards. Section B*, 69:67–72, 1965. [77]

[50] J. Edmonds. Matroids and the greedy algorithm. *Mathematical Programming*, 1(1):127–136, 1971. [13, 39, 40]

[51] M. H. El-Zahar and J. H. Schmerl. On the size of jump-critical ordered sets. *Order*, 1(1):3–5, 1984. [110]

[52] S. Even, A. Pnueli, and A. Lempel. Permutation graphs and transitive graphs. *J. ACM*, 19(3):400–410, 1972. [91]

[53] H. Fauck. Covering polygons with rectangles via edge coverings of bipartite permutation graphs. *Elektronische Informationsverarbeitung und Kybernetik*, 27(8):391–409, 1991. [15, 109, 118, 119, 130, 150]

[54] S. Felsner. A 3/2-approximation algorithm for the jump number of interval orders. *Order*, 6(4):325–334, 1990. [109]

[55] T. S. Ferguson. Who solved the secretary problem? *Statistical Science*, 4(3):282–289, 1989. [12, 43]

[56] P. C. Fishburn and P. L. Hammer. Bipartite dimensions and bipartite degrees of graphs. *Discrete Mathematics*, 160(1–3):127–148, 1996. [110]

[57] L. Fleischer and S. Iwata. A push-relabel framework for submodular function minimization and applications to parametric optimization. *Discrete Applied Mathematics*, 131(2):311–322, 2003. [160]

[58] D. G. Fon-Der-Flaass and A. V. Kostochka. Covering boxes by points. *Discrete Mathematics*, 120(1–3):269–275, 1993. [126]

[59] L. R. Ford and D. R. Fulkerson. *Flow in networks*. Princeton University Press Princeton, 1962. [87]

[60] R. J. Fowler, M. Paterson, and S. L. Tanimoto. Optimal packing and covering in the plane are NP-complete. *Inf. Process. Lett.*, 12(3):133–137, 1981. [125, 151]

[61] A. Frank. On the edge-connectivity algorithm of Nagamochi and Ibaraki, 1994. Laboratoire Artemis, IMAG, Université J. Fourier, Grenoble. [160]

[62] A. Frank. Finding minimum generators of path systems. *J. Comb. Theory, Ser. B*, 75(2):237–244, 1999. [135, 138]

[63] A. Frank and T. Jordán. Minimal edge-coverings of pairs of sets. *J. Comb. Theory, Ser. B*, 65(1):73–110, 1995. [17, 18, 118, 121, 148, 150]

[64] D. S. Franzblau and D. J. Kleitman. An algorithm for covering polygons with rectangles. *Information and Control*, 63(3):164–189, 1984. [118, 119, 150]

[65] P. Freeman. The secretary problem and its extensions: A review. *International Statistical Review/Revue Internationale de Statistique*, 51(2):189–206, 1983. [12]

[66] S. Fujishige. Theory of principal partitions revisited. In *Research Trends in Combinatorial Optimization*, pages 127–162, 2009. [60, 63]

[67] D. Gale. Optimal assignments in an ordered set: An application of matroid theory. *Journal of Combinatorial Theory*, 4(2):176–180, 1968. [13, 39, 40]

[68] T. Gallai. Transitiv orientierbare graphen. *Acta Mathematica Hungarica*, 18(1):25–66, 1967. [89]

[69] G. Gierz and W. Poguntke. Minimizing setups for ordered sets: A linear algebraic approach. *SIAM J. Algebraic Discrete Methods*, 4(1):132–144, 1983. [108]

[70] J. P. Gilbert and F. Mosteller. Recognizing the maximum of a sequence. *Journal of the American Statistical Association*, 61(313):35–73, 1966. [27, 43]

[71] G. Goel, C. Karande, P. Tripathi, and L. Wang. Approximability of combinatorial problems with multi-agent submodular cost functions. In *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science*, FOCS'09, pages 755–764, 2009. [162]

[72] G. Goel and A. Mehta. Online budgeted matching in random input models with applications to adwords. In *Proceedings of the Nineteenth Annual ACM-SIAM Symposium on Discrete algorithms*, SODA'08, pages 982–991, 2008. [42]

[73] M. X. Goemans and V. S. Ramakrishnan. Minimizing submodular functions over families of sets. *Combinatorica*, 15(4):499–513, Dec. 1995. [162]

[74] D. A. Gregory, N. J. Pullman, K. F. Jones, and J. R. Lundgren. Biclique coverings of regular bigraphs and minimum semiring ranks of regular matrices. *J. Comb. Theory Ser. B*, 51(1):73–89, 1991. [112]

[75] M. Grötschel, L. Lovász, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1(2):169–197, 1981. [18, 160]

[76] M. Grötschel, L. Lovász, and A. Schrijver. *Geometric algorithms and combinatorial optimization.* Springer-Verlag, second edition, 1993. [18, 104, 160, 162]

[77] U. I. Gupta, D. T. Lee, and J. Y.-T. Leung. Efficient algorithms for interval graphs and circular-arc graphs. *Networks*, 12(4):459–467, 1982. [125]

[78] S. M. Gusein-Zade. The problem of choice and the sptimal stopping rule for a sequence of independent trials. *Theory of Probability and its Applications*, 11(3):472–476, 1966. [43]

[79] A. Gyárfás and J. Lehel. Covering and coloring problems for relatives of intervals. *Discrete Mathematics*, 55(2):167–180, 1985. [126]

[80] E. Györi. A minimax theorem on intervals. *J. Comb. Theory, Ser. B*, 37(1):1–9, 1984. [17, 114, 115, 116, 117, 118, 135, 138, 149]

[81] M. Habib. *Partitions en chemins des sommets et sauts dans les graphes sans circuit.* PhD thesis, Université Pierre et Marie Curie, 1975. Thèse de 3e cycle. [108]

[82] M. Habib. Comparability invariants. In M. Pouzet and D. Richard, editors, *Ordres: Description et Rôles*, volume 99 of *North-Holland Mathematics Studies*, pages 371–385. North-Holland, 1984. [15, 108]

[83] R. Hegde and K. Jain. The hardness of approximating poset dimension. *Electronic Notes in Discrete Mathematics*, 29:435–443, 2007. European Conference on Combinatorics, Graph Theory and Applications. [89]

[84] T. Hiraguchi. On the dimension of partially ordered sets. *Sci. Rep. Kanazawa Univ.*, 1:77–94, 1951. [89]

[85] A. J. Hoffman, A. W. J. Kolen, and M. Sakarovitch. Totally-balanced and greedy matrices. *SIAM Journal on Algebraic and Discrete Methods*, 6(4):721–730, 1985. [93]

[86] J. E. Hopcroft and R. M. Karp. An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM Journal on Computing*, 2(4):225–231, 1973. [87, 139]

[87] S. Im and Y. Wang. Secretary problems: Laminar matroid and interval scheduling. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'11, pages 1265–1274, 2011. [13, 42, 44]

[88] S. Iwata. A fully combinatorial algorithm for submodular function minimization. *J. Comb. Theory, Ser. B*, 84(2):203–212, Mar. 2002. [160]

[89] S. Iwata, L. Fleischer, and S. Fujishige. A combinatorial strongly polynomial algorithm for minimizing submodular functions. *Journal of the ACM*, 48(4):761–777, 2001. [160]

[90] S. Iwata and K. Nagano. Submodular function minimization under covering constraints. In *Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science*, FOCS'09, pages 671–680, 2009. [162]

[91] S. Iwata and J. B. Orlin. A simple combinatorial algorithm for submodular function minimization. In *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'09, pages 1230–1237, 2009. [160]

[92] D. S. Johnson. The NP-completeness column: An ongoing guide. *Journal of Algorithms*, 8(3):438–448, 1987. [118]

[93] D. R. Karger. Random sampling and greedy sparsification for matroid optimization problems. *Mathematical Programming*, 82(1):41–81, 1998. [42]

[94] G. Kishi and Y. Kajitani. Maximally distant trees and principal partition of a linear graph. *IEEE Transactions on Circuit Theory*, 16(3):323–330, 1969. [60, 61]

[95] R. Kleinberg. A multiple-choice secretary algorithm with applications to online auctions. In *Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'05, pages 630–631, 2005. [13, 30]

[96] D. E. Knuth. Irredundant intervals. *ACM Journal of Experimental Algorithmics*, 1, 1996. Article 1. [117, 118]

[97] N. Korula and M. Pál. Algorithms for secretary problems on graphs and hypergraphs. In S. Albers, A. Marchetti-Spaccamela, Y. Matias, S. E. Nikoletseas, and W. Thomas, editors, *ICALP*, volume 5556 of *Lecture Notes in Computer Science*, pages 508–520. Springer, 2009. [13, 14, 42, 44, 75, 81]

[98] G. Károlyi and G. Tardos. On point covers of multiple intervals and axis-parallel rectangles. *Combinatorica*, 16(2):213–222, 1996. [126]

[99] E. Kushilevitz and N. Nisan. *Communication complexity*. Cambridge University Press, New York, NY, USA, 1997. [16, 112]

[100] H.-J. Lai and H. Lai. Every matroid is a submatroid of a uniformly dense matroid. *Discrete Applied Mathematics*, 63(2):151–160, 1995. [70]

[101] D. V. Lindley. Dynamic programming and decision theory. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 10(1):39–51, 1961. [12, 13, 27, 43]

[102] L. Lovász. A characterization of perfect graphs. *J. Comb. Theory, Ser. B*, 13(2):95 – 98, 1972. [104]

[103] L. Lovász. Normal hypergraphs and the perfect graph conjecture. *Discrete Mathematics*, 2(3):253–267, 1972. [103, 104]

[104] L. Lovász. Submodular functions and convexity. In M. G. A. Bachem and B. Korte, editors, *Mathematical Programming – The State of the Art (Bonn 1982)*, pages 235–257. Springer Verlag, 1983. [18]

[105] V. V. Lozin. E-free bipartite graphs. *Diskretn. Anal. Issled. Oper., Ser. 1*, 7(1):49–66, 2000. [110]

[106] V. V. Lozin and M. U. Gerber. On the jump number problem in hereditary classes of bipartite graphs. *Order*, 17(4):377–385, 2000. [109]

[107] A. Lubiw. *Orderings and some combinatorial optimization problems with geometric applications*. PhD thesis, University of Toronto, Toronto, Ont., Canada, 1986. [118]

[108] A. Lubiw. Doubly lexical orderings of matrices. *SIAM J. Comput.*, 16(5):854–879, 1987. [93]

[109] A. Lubiw. The Boolean Basis Problem and How to Cover Some Polygons by Rectangles. *SIAM Journal on Discrete Mathematics*, 3(1):98, 1990. [113, 118]

[110] A. Lubiw. A weighted min-max relation for intervals. *J. Comb. Theory, Ser. B*, 53(2):151–172, 1991. [117, 118, 154, 155]

[111] W. Mader. Über n-fach zusammenhängende eckenmengen in graphen. *J. Comb. Theory, Ser. B*, 25(1):74–93, 1978. [164]

[112] R. B. Manfrino, J. A. G. Ortega, and R. V. Delgado. *Inequalities: A Mathematical Olympiad Approach*. Birkhäuser, 2009. [51]

[113] W. J. Masek. Some NP-complete set covering problems. (unpublished manuscript), 1979. [118]

[114] C. McCartin. An improved algorithm for the jump number problem. *Information Processing Letters*, 79(2):87–92, 2001. [110]

[115] R. M. McConnell and J. P. Spinrad. Linear-time transitive orientation. In *Proceedings of the Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'97, pages 19–25, 1997. [92]

[116] L. Mirsky. A dual of Dilworth's decomposition theorem. *The American Mathematical Monthly*, 78(8):876–877, 1971. [86]

[117] J. Mitas. Tackling the jump number of interval orders. *Order*, 8(2):115–132, 1991. [109]

[118] R. Motwani and P. Raghavan. *Randomized algorithms*. Cambridge University Press, New York, NY, USA, 1995. [73]

[119] A. G. Mucci. Differential equations and optimal choice problems. *The Annals of Statistics*, 1(1):104–113, 1973. [43]

[120] M. Mucha and P. Sankowski. Maximum matchings via gaussian elimination. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science*, FOCS'04, pages 248–255, 2004. [87, 139]

[121] H. Müller. Alternating cycle-free matchings. *Order*, 7(1):11–21, 1990. [109, 110, 118]

[122] H. Müller. On edge perfectness and classes of bipartite graphs. *Discrete Mathematics*, 149(1–3):159–187, 1996. [110, 111, 118, 130]

[123] H. Müller. Recognizing interval digraphs and interval bigraphs in polynomial time. *Discrete Applied Mathematics*, 78(1–3):189–205, 1997. [96, 97]

[124] H. Nagamochi. Minimum degree orderings. *Algorithmica*, 56(1):17–34, 2010. [19, 159, 162, 181, 182, 183, 186]

[125] H. Nagamochi and T. Ibaraki. Computing Edge-Connectivity in multigraphs and capacitated graphs. *SIAM Journal on Discrete Mathematics*, 5(1):54–66, 1992. [18, 160]

[126] H. Nagamochi and T. Ibaraki. A linear-time algorithm for finding a sparse k-connected spanning subgraph of a k-connected graph. *Algorithmica*, 7(1):583–596, 1992. [18, 160]

[127] H. Nagamochi and T. Ibaraki. A note on minimizing submodular functions. *Information Processing Letters*, 67(5):239–244, 1998. [19, 159, 161, 162, 166, 169, 175]

[128] H. Narayanan. *Theory of matroids and network analysis.* PhD thesis, Department of Electrical Engineering, Indian Institute of Technology, Bombay, 1974. [62]

[129] H. Narayanan. *Submodular Functions and Electrical Networks*, volume 54 of *Annals of Discrete Mathematics*. Elsevier, 1997. [61]

[130] H. Narayanan. A note on the minimization of symmetric and general submodular functions. *Discrete Applied Mathematics*, 131(2):513–522, 2003. [162, 169, 175]

[131] H. Narayanan and M. N. Vartak. An elementary approach to the principal partition of a matroid. *Transactions of the Institute of Electronics and Communication Engineers of Japan. Section E*, E64(4):227–234, 1981. [60, 61, 63]

[132] O. Ore. *Theory of graphs*. American Mathematical Society, 1962. [88]

[133] J. Orlin. Contentment in graph theory: Covering graphs with cliques. *Indagationes Mathematicae (Proceedings)*, 80(5):406–424, 1977. [112, 117]

[134] J. B. Orlin. A faster strongly polynomial time algorithm for submodular function minimization. *Mathematical Programming*, 118(2):237–251, 2007. [18, 160]

[135] Y. Otachi, Y. Okamoto, and K. Yamazaki. Relationships between the class of unit grid intersection graphs and other classes of bipartite graphs. *Discrete Applied Mathematics*, 155(17):2383–2390, 2007. [16]

[136] S. Oveis Gharan and J. Vondrák. On variants of the matroid secretary problem. Manuscript. ArXiv version in `http://arxiv.org/abs/1007.2140`, 2011. [14, 41, 55, 70, 80]

[137] J. G. Oxley. *Matroid theory*. Oxford University Press, USA, 2006. [36, 58, 60, 74, 79]

[138] M. W. Padberg and M. R. Rao. Odd minimum cut-sets and b-matchings. *Mathematics of Operations Research*, 7:67–80, 1982. [162]

[139] W. R. Pulleyblank. Alternating cycle free matchings. Technical Report CORR 82-18, University of Waterloo - Dept. of Combinatorics and Optimization, 1982. [15, 108]

[140] W. R. Pulleyblank. On minimizing setups in precedence constrained scheduling. (unpublished), 1982. [106]

[141] M. Queyranne. Minimizing symmetric submodular functions. *Mathematical Programming*, 82(1):3–12, 1998. [18, 19, 159, 160, 162, 164]

[142] R. Rado. Note on independence functions. *Proceedings of the London Mathematical Society*, 3(1):300–320, 1957. [13]

[143] W. T. Rasmussen and S. R. Pliska. Choosing the maximum from a sequence with a discount function. *Applied Mathematics & Optimization*, 2(3):279–289, 1975. [43]

[144] I. Rival. Optimal linear extensions by interchanging chains. *Proceedings of the American Mathematical Society*, 89(3):387–394, 1983. [108]

[145] R. Rizzi. On minimizing symmetric set functions. *Combinatorica*, 20(3):445–450, 2000. [19, 161, 162, 169, 176, 177, 178]

[146] A. Schrijver. A combinatorial algorithm minimizing submodular functions in strongly polynomial time. *J. Comb. Theory, Ser. B*, 80(2):346–355, 2000. [160]

[147] A. Schrijver. *Combinatorial Optimization - Polyhedra and Efficiency*. Springer, 2003. [36, 61, 75, 76]

[148] A. Sharary. The jump number of Z-free ordered sets. *Order*, 8(3):267–273, 1991. [109]

[149] A. H. Sharary and N. Zaguia. On minimizing jumps for ordered sets. *Order*, 7(4):353–359, 1990. [109]

[150] A. M. S. Shrestha, S. Tayu, and S. Ueno. On orthogonal ray graphs. *Discrete Applied Mathematics*, 158(15):1650–1659, 2010. [95]

[151] A. M. S. Shrestha, S. Tayu, and S. Ueno. On two-directional orthogonal ray graphs. In *Proceedings of 2010 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 1807–1810, 2010. [16, 94, 95]

[152] J. A. Soto. Matroid secretary problem in the random assignment model. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA'11, pages 1275–1284, 2011. ArXiv version in `http://arxiv.org/abs/1007.2152`. [13, 14, 41, 55, 69, 80]

[153] J. A. Soto and C. Telha. Jump number of two-directional orthogonal ray graphs. To appear in the proceedings of the 15th Conference on Integer Programming and Combinatorial Optimization, 2011. [16, 105, 115, 129, 138]

[154] J. Spinrad, A. Brandstädt, and L. Stewart. Bipartite permutation graphs. *Discrete Applied Mathematics*, 18(3):279–292, 1987. [100]

[155] G. Steiner. On finding the jump number of a partial orber by substitution decomposition. *Order*, 2(1):9–23, 1985. [108]

[156] G. Steiner and L. K. Stewart. A linear time algorithm to find the jump number of 2-dimensional bipartite partial orders. *Order*, 3(4):359–367, 1987. [15, 109, 130]

[157] T. J. Stewart. Optimal selection from a random sequence with learning of the underlying distribution. *Journal of the American Statistical Association*, 73(364):775–780, 1978. [43]

[158] M. Stoer and F. Wagner. A simple min-cut algorithm. *Journal of the ACM*, 44(4):585–591, 1997. [160]

[159] Z. Svitkina and L. Fleischer. Submodular approximation: Sampling-based algorithms and lower bounds. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 697–706, 2008. [18, 159, 160]

[160] M. M. Syslo. The jump number problem on interval orders: A approximation algorithm. *Discrete Mathematics*, 144(1–3):119–130, 1995. [109]

[161] E. Szpilrajn. Sur l'extension de l'ordre partiel. *Fundamenta mathematicae*, 16:386–389, 1930. [88]

[162] N. Thain. On the transversal matroid secretary problem. Master's thesis, McGill University, Canada, 2008. [13, 42]

[163] N. Tomizawa. Strongly Irreducible Matroids and Principal Partition of a Matroid into Strongly Irreducible Minors. *Electronics & Communications In Japan*, 59(A):1–10, 1976. [62]

[164] W. T. Trotter. *Combinatorics and Partially Ordered Sets: Dimension theory*. Johns Hopkins University Press, 2001. [85]

[165] P. van Emde Boas. Preserving order in a forest in less than logarithmic time. In *16th Annual Symposium on Foundations of Computer Science*, FOCS'75, pages 75–84, 1975. [144]

[166] L. A. Végh. *Connectivity Augmentation Algorithms*. PhD thesis, Eötvös Loránd University, 2010. [149]

[167] G. Wegner. Über eine kombinatorisch-geometrische frage von hadwiger und debrunner. *Israel Journal of Mathematics*, 3(4):187–198, 1965. [126]

[168] M. Yannakakis. The complexity of the partial order dimension problem. *SIAM Journal on Algebraic and Discrete Methods*, 3(3):351–358, 1982. [89]