On the dualization problem in graphs, hypergraphs, and lattices

Oscar Defrain

Université Clermont Auvergne

École Doctorale Sciences pour l'Ingénieur, Clermont-Ferrand

Thèse présentée par

Oscar Defrain

pour obtenir le grade de

Docteur d'Université

Spécialité Informatique

On the dualization problem in graphs, hypergraphs, and lattices

---

Soutenue publiquement le 02/09/2020 devant le jury constitué de :

Nadia CREIGNOU	
Professeur, Université Aix Marseille	*Rapporteur*
Sergei KUZNETSOV	
Professeur, Higher School of Economics, Moscow	*Rapporteur*
Kazuhisa MAKINO	
Professeur, Reasearch Institute of Math. Sciences, Kyoto	*Rapporteur*
Victor CHEPOI	
Professeur, Université Aix Marseille	*Examinateur*
Arnaud DURAND	
Professeur, Université Paris Diderot	*Examinateur*
Aurélie LAGOUTTE	
Maître de Conférence, Université Clermont Auvergne	*Examinatrice*
Lhouari NOURINE	
Professeur, Université Clermont Auvergne	*Directeur*

Contents

Acknowledgements

My first thanks naturally go to my PhD advisor Lhouari Nourine for these three years of research in Clermont-Ferrand. Thank you for your trust, and for having given me all these opportunities to initiate new collaborations in France and abroad. I am also thankful to the reviewers Nadia Creignou, Sergei Kuznetsov, and Kazuhisa Makino, and to the examiners Victor Chepoi, Arnaud Durand, and Aurélie Lagoutte, for accepting to be in my defense committee. Thank you for having read my manuscript, and for your valuable remarks. Many thanks to Marthe Bonamy, Kazuhisa Makino, Jean-Florent Raymond, and Takeaki Uno, for inviting me to spend some time on nice problems in Bordeaux, Kyoto, Berlin, and Tokyo, and thanks to Meike Hatzel, Marc Heinrich, Tereza Klimošová, Jonathan Narboni, Michał Pilipczuk, Jocelyn Thiebaut, and Kunihiro Wasa, for the nice blackboard sessions there. Thanks also to my coauthors Pierre Charbit, Gwenaël Joret, Aurélie Lagoutte, Vincent Limouzy, Piotr Micek, Lucas Pastor, Jean-Sébastien Sereni, and Simon Vilmin, for the nice projects, and to Mamadou Kanté, Andrea Marino, Arnaud Mary, and Yann Strozecki, for the interesting discussions on enumeration problems and algorithms. At last, thanks to you Marthe Bonamy for the many opportunities you gave me, for your guidance, your support, and to you Marcin Pilipczuk for allowing me to continue my research as a postdoc at the University of Warsaw.

I would also like to thank Aurélie, Bruno, Jean-Florent, Laurent, Lucas, Olivier, Vincent, Yannick, and the other colleagues from the Building D and beyond for maintaining such a great and inclusive working place in Clermont-Ferrand. Thanks also to the PhD students and postdocs I (almost) shared an office with, Benjamin, Giacomo, Simon, Alexey, Caroline, Matthieu, and Mozhgan, for these very nice moments, and to you Béatrice and Séverine for making everything easier here. A special thank to Nofar for having given me the opportunity to publicly express how essential Simon was as an office mate during these last two years of PhD!

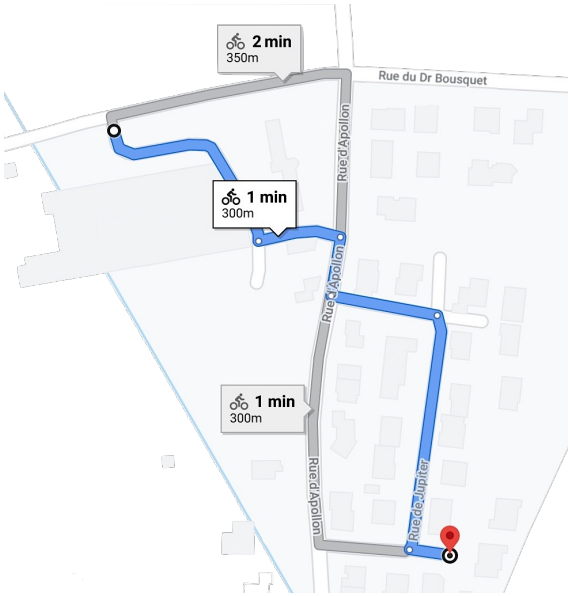
I would finally like to thank my family for their support. Many thanks to Arthur, Anaïs, Éric, Odile, and to my mother, for doing the trip to Clermont-Ferrand, and for helping me preparing the defense. Thanks also to my CUC family here in Auvergne, to Apostolos, Godefroy, Vincent, and to Alexandra, for these great moments together!

Introduction

Many algorithmic problems in discrete mathematics and computer science require to find a best solution among all feasible ones. These problems, known as optimization problems, are widely encountered in practice. A notable example is the problem of finding a shortest path from a source to a destination in a given network. Another example concerns the computation of a best answer to a given query in a database. In everyday life, problems of that kind naturally arise when using GPS navigation, or when booking flights to a destination; see Figure A for visual examples. In addition to these practical aspects, optimization problems are well studied, and raise important fundamental questions. Research has been focusing on characterizing their computational complexity, either by exhibiting algorithms that run in polynomial time in the size of the input, or by showing that a problem is NP-complete, hence that it is most certainly intractable.

Rather than finding one solution, it is sometimes more desirable to generate all of them. This is for instance useful in certain applications to database search [YYH05], network analysis [GK07], and bioinformatics [Dam06]. Problems of that kind are also encountered in everyday life, typically when several solutions may be of interest for a user, as in Figure A. In graph theory, enumeration problems seem to have been first mentioned in the early 70's with the pioneer works of Tiernen [Tie70] and Tarjan [Tar73] on cycles in directed graphs, and of Akkoyunlu [Akk73] on maximal cliques in undirected graphs. However, they already appeared in disguise in earlier works [PU59, Mar64]. In hypergraph theory, the most notable enumeration problem is the one of listing minimal transversals, which plays a fundamental role in many areas [EG95, GMKT97]. In lattice theory, enumeration problems naturally arise when building lattices [NR99, HMNS01], or when translating between their different representations [Wil94, Kha95, BMN17, HN18]. It is maybe with the newly discovered techniques for enumeration [RT75, TIAS77, AF96], and with the landmark paper of Johnson, Yannakakis and Papadimitriou [JYP88], that enumeration theory started to gain interest. This last decade, several PhD theses were devoted to enumeration theory, e.g. [Bag09, Str10, Mar13, Mar15], and the first edition of the Workshop on Enumeration Problems and their Applications (WEPA) was launched in Clermont-Ferrand, France. This, together with the surveys on enumeration complexity [CKP+19, Str19], the two Dagstuhl Seminars 18421 and 19211 on algorithmic enumeration [FGS19, BKPS19], and the creation of a Wikipedia page on enumeration algorithms [WIK], illustrate the recent gain of interest for the field.

It has certainly come to the reader that the number of solutions to an enumeration problem may grow exponentially in the size of the input. For example in Figure A.(a), the biker is given two choices in order to reach Rue d'Apollon, and then two choices



(a) Bike itineraries.

CSA	20:10 – 08:35+1 CSA · Smartwings	12 h 25 min CDG–WAW	1 escale 9 h 10 min PRG	99 €
CSA	09:45 – 13:35 CSA	3 h 50 min CDG–WAW	1 escale 35 min PRG	117 €
SAS	14:30 – 22:30 SAS	8 h 0 min CDG–WAW	1 escale 4 h 55 min CPH	122 €
AF	13:00 – 15:20 Air France	2 h 20 min CDG–WAW	Sans escale	203 €
	14:25 – 18:05 Lufthansa	3 h 40 min CDG–WAW	1 escale 55 min FRA	285 €
LOT	19:45 – 22:00 LOT	2 h 15 min CDG–WAW	Sans escale	367 €
	114 autres vols			

(b) Flights from CDG to WAW.

Figure A: Everyday expressions of optimization and enumeration problems. A shortest bike itinerary is given in blue (a), and a cheapest flight in green (b). Alternative solutions are proposed in both situations.

in order to reach the final destination. This leads to a total number of four possible distinct routes, which clearly, when generalized, is exponential in the number of roads in the network. Therefore when devising enumeration algorithms, and in stark contrast to decision or optimization problems, looking for a running time polynomially bounded by the size of the input is not a reasonable—let alone meaningful—efficiency criterion. Rather, we aim for so-called output-polynomial time algorithms whose running time is polynomially bounded by the sizes of both the input and output data. An important research direction is then to bound, in addition, the delay between two consecutive output solutions. Bounds may be expressed by a polynomial in the size of the input, or by a polynomial in the sizes of the input plus already output solutions. These different notions of complexity lead to a number of intriguing open questions in the field [JYP88, CKP+19, Str19].

When dealing with the space used by an enumeration algorithm, only the working space is considered: it is assumed that the solutions are flashed, and then immediately discarded. However in order to avoid producing repetitions, some algorithms rely on storing the already generated solutions. This inexorably requires space that is linear in the number of solutions, thus potentially exponential in the size of the input. It is then another important question whether an enumeration algorithm requiring exponential space in order to avoid repetitions may be turned into one that uses only polynomial space, without increasing "too much" its complexity. Recent papers raising the question include [MS19, CU19, CMG+19].

This thesis focuses on graphs, hypergraphs, and lattices. It is mainly concerned with the different shapes that takes a problem on these structures: the dualization of monotone Boolean functions. In its most simple terms, this problem asks, given a pos-

itive[1] CNF φ, and a positive DNF ψ, whether $\varphi = \psi$. In its generation versions, only one of the two formulas is provided, and the other (of minimum size) is to be computed. It is easily conceived that such a fundamental question may be encountered in practice. The problem plays in fact a crucial role in database theory, Boolean switching theory, logic, Artificial Intelligence (AI), machine learning, data mining, and many other concrete fields [EG95, GMKT97, NP12]. Implementations are numerous [Uno], but are not the object of this manuscript.

In addition to the important role the dualization plays in practice, the problem appears to be of great fundamental interest. Firstly, it is ubiquitous in many areas of theoretical computer science, as we will see in a moment. It is for example equivalent to the aforementioned problem of enumerating the minimal transversals of a hypergraph, or to the problem of enumerating the minimal dominating sets of a graph [EMG08, KLMN14]. Secondly, its precise complexity status remains open after more than forty years of research, despite continuous attempts to solve it [EG95, EMG08]. To date, it is not known whether the problem can be solved in polynomial time (and its generation version in output-polynomial time): this arguably constitutes one of the most important open problems in enumeration theory. It has however been showed in a landmark paper by Fredman and Khachiyan [FK96] that the problem admits a quasi-polynomial $N^{o(\log N)}$ time algorithm, where $N = |\varphi| + |\psi|$. In consequence, the problem is probably not coNP-hard, and is believed to lie somewhere in the frontier between P and coNP.

In this thesis, both positive and negative results are exhibited for the dualization, under different restrictions, formulations, and generalizations. The manuscript is organized as follows. In Chapter 1 we give a brief introduction to enumeration problems, algorithms, and their complexity, and formally define the dualization of monotone Boolean functions through different formulations. Chapter 2 is devoted to the study of one equivalent formulation: the enumeration of minimal dominating sets in graphs. We obtain new output-polynomial time algorithms in graph classes related to K_t-free graphs and to posets of bounded dimension. Chapter 3 is devoted to the dualization in lattices given by implicational bases, a first generalization of the dualization. Both tractability and intractability results are obtained under various restrictions concerning notions of width and premises' size in the implicational base. Chapter 4 is devoted to the problem of translating between the representations of a lattice, a second generalization of the dualization. Tractability and intractability results are obtained under some restrictions concerning acyclicity. Each chapter ends with a list of open issues that are summarized in the conclusion of this manuscript.

[1] Only containing non-negated literals, e.g. $\varphi = 1 \wedge (2 \vee 3)$ and $\psi = 12 \vee 23$.

Chapter 1

Enumeration problems and algorithms

This chapter contains a brief introduction to enumeration problems, algorithms, and their complexity. Other sources of interest on this topic include the PhD thesis of Bagan [Bag09], the ones of Strozecki [Str10], Mary [Mar13], Marino [Mar15] and the recent survey [Str19].

We saw in the introduction how enumeration problems gained interest these last decades, with several questions on the topic remaining unsolved. We give here the necessary background for the definition of the open problem that is at the heart of this dissertation: the dualization of monotone Boolean functions.

The chapter is organized as follows. In Section 1.1 we give a rudimentary definition of what an enumeration problem is, and state general assumptions on the enumeration problems that are considered in this manuscript. Section 1.2 deals with the complexity of enumeration algorithms. Monotone dualization is presented through different formulations in Section 1.3.

1.1 Definitions and examples

Let Σ be a finite—say binary—alphabet and Σ^* denote the set of words constructed on alphabet Σ. Let $R \subseteq \Sigma^* \times \Sigma^*$ be a binary predicate. We shall note $R(x, y)$ if $(x, y) \in R$, and define $R(x) = \{y \in \Sigma^* \mid R(x, y)\}$. In the following, we respectively call *instance* and *solution* the elements x and y such that $R(x, y)$ holds; x will also be called the *input*, and each such y an *output*.

1.1.1 Problem definitions

Decision problems are maybe the most encountered problems in computer science. They can be defined as follows.

Definition 1.1.1. *The decision problem δ_R associated to a binary predicate R is the function which maps to every instance $x \in \Sigma^*$ the bit 0 if $R(x) = \emptyset$, the bit 1 otherwise.*

Decision problems basically ask whether a given instance admits a solution, and are also referred to as "*yes–no questions*". An algorithm solving a decision problem δ_R is one that computes its associated function, i.e., it outputs given any instance x, the bit 0 if $R(x) = \emptyset$, the bit 1 otherwise. We call *algorithm for δ_R* such an algorithm.

For example in the SUBSET SUM problem, one is given a finite subset of negative and non-negative integers X, and must decide whether there exists a subset $Y \subseteq X$ such that the sum of all the integers in Y is zero. This problem is easily seen to fall within the formalism of Definition 1.1.1 by considering any binary encoding x of X, any binary encoding y of Y, and defining the binary predicate R accordingly: $R(x, y)$ holds if and only if x codes a set X and y a subset of $Y \subseteq X$ such that the sum of all the integers in Y is zero. Observe more generally that a suitable binary encoding allows to consider complex objects as instances and solutions of a problem, e.g., words, graphs, hypergraphs, posets, or the concatenation of several such objects.

Many decision problems require to detect a structure with prescribed properties. On the other hand, rather that checking the existence of a solution, it is sometimes more desirable to generate all of them. For example, in a generation version of SUBSET SUM, one may be asked to output all subsets $Y \subseteq X$ such that the sum of all integers in Y is zero. Problems of that kind are called enumeration problems and may be defined as follows.

Definition 1.1.2. *The enumeration problem Π_R associated to a binary predicate R is the function which maps to every instance $x \in \Sigma^*$ the set $R(x)$ of its solutions.*

Enumeration problems are also known as *listing algorithms*, or *generation algorithms*. An algorithm solving an enumeration problem Π_R is one that computes its associated function, i.e., it outputs given any instance x, a sequence $y_1, \ldots, y_\ell$ of solutions such that $R(x) = \{y_1, \ldots, y_\ell\}$, and for all $i \neq j$, $y_i \neq y_j$. This last condition says that no solution is repeated. We call *enumeration algorithm for* Π_R (or simply *algorithm* for Π_R) such an algorithm.

1.1.2 Predicate assumptions

We state common assumptions on binary predicates that are shared by every enumeration problem considered in this manuscript. We chose not to elaborate on these assumptions and refer to [Str19] for further details on these points.

If x is a word on alphabet Σ, then $|x|$ denote its length. We say that predicate R is *polynomially balanced* if $|y| \leq \text{poly}(|x|)$ for every two words x, y such that $R(x, y)$, i.e., if every solution of R is of polynomial size in the size of the input. This is clearly the case of SUBSET SUM as defined previously. Note that this does not prevent $R(x)$ to be of exponential size in the size of x, a key point that is developed in Section 1.2. We furthermore assume that checking whether $R(x, y)$ holds, given any couple $(x, y) \in \Sigma^* \times \Sigma^*$, can be done in polynomial time in the sizes of x and y. Again, note that this holds for SUBSET SUM as the sum of integers can be computed in polynomial time. Problems that satisfy these properties constitute a class analogous to NP for enumeration problems; see [Str19].

1.1.3 Computational model

The computational model we consider is this manuscript is the random access machine model (RAM) equipped with comparison, addition, subtraction and multiplication, together with an additional operation `Output(i,j)` which outputs the con-

catenation of its registers $R_i, R_{i+1}, \ldots, R_j$. This model has been considered in various previous works [Bag09, Str10, Str19] and is useful to access an exponential amount of memory only using polynomial time, as it is implicitely assumed in various works such as [GHKV15, CU19] and used in Chapter 2.

1.2 Complexity of enumeration problems

We assume that the reader is familiar with classical complexity theory. We refer to the book [GJ79] if not. If f is a function, we write $f(n) = \mathrm{poly}(n)$ when there is a constant $c \in \mathbb{N}$ such that $f(n) \in O(n^c)$, and $f(n) = \mathrm{polylog}(n)$ when there is a constant $c \in \mathbb{N}$ such that $f(n) \in O((\log n)^c)$.

As pointed out earlier, while all predicates R considered in this manuscript are polynomially balanced, the set $R(x)$ of solutions to output may be of exponential size in the size of x. Regarding this, two different approaches—*input-sensitive* and *output-sensitive*—have been considered in the literature in order to measure the complexity of an enumeration algorithm. While this thesis only deals with the second approach, we overview here the first approach for completeness.

1.2.1 Input-sensitive approach

In the input-sensitive approach, one measures the complexity of an enumeration algorithm in term of input size. For most problems Π_R—in which the size of $R(x)$ may be exponential in the size of x—, this approach aims to get the fastest exponential-time algorithm, typically lowering the base of the exponent as much as possible. This leads to results of the following kind.

Theorem 1.2.1 ([MM65]). *There is an $O(1.4423^n)$ time algorithm enumerating maximal cliques in n-vertex graphs.*

In fact, Theorem 1.2.1 was not originally stated as an enumeration result. It was initially stated as an upper bound of $3^{n/3} \approx 1.4423$ on the number of maximal independent sets an n-vertex graph admits, which proof yields a straightforward enumeration algorithm listing them within that time. Such an upper-bound is attained by considering the complementary graph of disjoint unions of triangles. Most of the time, input-sensitive algorithms consist of showing upper bounds on the number of objects to enumerate [FGPS08, CHvHK13]. Obtaining tight bounds like the one of $3^{n/3}$ for maximal cliques remains however open for other combinatorial objects, such as the minimal dominating sets [AKH16], the objects that are considered in Chapter 2.

1.2.2 Output-sensitive approach

In the output-sensitive approach, one measures the complexity of an enumeration algorithm in term of input plus output sizes. The goal, here, is to achieve polynomial time bounds. This leads to results of the following kind.

Theorem 1.2.2 ([TIAS77]). *There is an $O(nm \cdot d)$ time algorithm enumerating the maximal cliques in n-vertex m-edge graphs, with d the number of maximal cliques in the input graph.*

Additional output-sensitive algorithms for maximal cliques enumeration in graphs may be found in [JYP88, MU04].

In fact, Theorem 1.2.2 is originally stated in a much more refined way, saying that in addition to the set of solutions being output in $O(nm \cdot d)$ time, the time spent by the algorithm between any two consecutive outputs is bounded by $O(nm)$. These complexity refinements constitute a significant proportion of the research that is dedicated to enumeration problems and output-sensitive algorithms, and certainly contribute to the appeal of ouput-sensitive enumeration for practical uses.

1.2.3 Three main notions of (output-sensitive) complexity

Many notions of complexity exist for output-sensitive algorithms, and are for example surveyed in [CKP+19, Str19]. We only review here the three historical ones that will be considered in this manuscript.

In the following if $\mathtt{A}$ is an enumeration algorithm for Π_R, we denote by $T_{\mathtt{A}}(x)$ its execution time on input x. Assume now that $y_1, \ldots, y_\ell$ are the elements of $R(x)$ enumerated in the order in which they are output by $\mathtt{A}$. We denote by $T_{\mathtt{A}}(x, i)$ the time $\mathtt{A}$ requires until it outputs y_i, and put $T_{\mathtt{A}}(x, 0) = 0$ and $T_{\mathtt{A}}(x, \ell + 1) = T_{\mathtt{A}}(x)$. We simply note $T(x)$ and $T(x, i)$ when $\mathtt{A}$ is clear from the context.

The next three important notions of complexity were first made explicit by Johnson, Yannakakis and Papadimitriou in [JYP88].

Definition 1.2.3. *An enumeration algorithm for Π_R is running in output-polynomial time if, for any instance x, $T(x) = \mathrm{poly}(|x| + |R(x)|)$.*

Following this definition, we will also say that an algorithm is running in output quasi-polynomial time if $T(x) = 2^{\mathrm{polylog}(N)}$, and that it runs in output sub-exponential time if $T(x) = 2^{o(N)}$, where $N = |x| + |R(x)|$.

In the following, we call *output-polynomial time algorithm* for Π_R any algorithm solving Π_R in output-polynomial time, and say that an enumeration problem Π_R is *tractable* if it admits one such algorithm. Observe here that the time $T(x, i + 1) - T(x, i)$ spent by an output-polynomial time algorithm before outputting a first solution may not be bounded by a polynomial in the size of x, as long as $|R(x)|$ is superpolynomial in $|x|$. Intuitively, it could be in fact that all solutions are "retained" by an output-polynomial time algorithm and output at the very last moment, just before halting. A more restricted notion of tractability is then defined as follows.

Definition 1.2.4. *An enumeration algorithm for Π_R is running in incremental-polynomial time if, for any instance x and every $1 \leq i \leq |R(x)| + 1$, $T(x, i) = \mathrm{poly}(|x| + i)$.*

In the following, we call *incremental-polynomial time algorithm* any algorithm which is running in incremental-polynomial time. Clearly by definition, any incremental-polynomial time algorithm defines an output-polynomial time algorithm.

Observe that the time $T(x, i)$ spent by an incremental-polynomial time algorithm between two consecutive outputs $y_i, y_{i+1} \in R(x)$ may not be bounded by a polynomial in the size of x, as long as i is superpolynomial in the size of x. This finally leads to the next most restricted notion of tractability, and last complexity notion that we consider in this manuscript.

Definition 1.2.5. *An enumeration algorithm for Π_R is running with polynomial delay if, for any instance x and every $1 \leq i \leq |R(x)| + 1$, $T(x, i) - T(x, i-1) = \text{poly}(|x|)$.*

This last complexity notion bounds the times spent by the algorithm before the first output, between any two consecutive outputs, and after the last output. Clearly, an algorithm running with polynomial delay is an incremental-polynomial time algorithm, hence an output-polynomial time algorithm. Algorithms of that kind constitute the more efficient algorithms one may seek in enumeration theory. The delay is sometimes lowered to linear time, or even constant time after polynomial preprocessing. Such extreme cases, however, will not be encountered in this manuscript.

1.2.4 Exponential output size

Recall that free to disregard several objects as coded by a single word $x \in \Sigma^*$, enumeration problems can contain complex inputs. The enumeration problems we consider in this thesis are all combinatorial problems that require, given subsets $X_1, \ldots, X_p \subseteq X$ where X is called *ground set*, to enumerate subsets $Y_1, \ldots, Y_\ell \subseteq X$ with desired properties. Recall also by the assumptions of Section 1.1.2 that checking whether a given set $Y \subseteq X$ is one of $Y_1, \ldots, Y_\ell$ can be done in polynomial time for these problems. This leads to a trivial output-polynomial time algorithm solving these problems on instances that are known to have exponentially many solutions. The algorithm proceeds as follows. For every subset $Y \subseteq X$, it tests in polynomial time whether Y is one of $Y_1, \ldots, Y_\ell$, and discard it otherwise, in a time bounded by $2^{|X|}$. The difficulty then lies in showing that non-trivial instances have an exponential lower bound on their number of solutions, or in the fact that some instances may have sub-exponentially—yet not polynomially—many solutions.

Despite the fact that this basic observation will not be used in the upcoming chapters, we felt that mentioning this aspect could enlighten the reader.

1.2.5 Memory

When considering the space complexity of an enumeration algorithm, the space of solutions, or *output space*, is not counted. It is assumed that the solutions are only flashed and then immediately discarded. We point out that it is still an option, e.g. for practical reasons, to store or not to store the solutions when they are flashed.

1.2.6 Decision and enumeration

Observe that to any enumeration problem Π_R for $R \subseteq \Sigma^* \times \Sigma^*$ corresponds a decision problem asking, given an instance $x \in \Sigma$ and a set $Y \subseteq \Sigma^*$, whether $R(x) = Y$. Such a decision problem is obtained by defining $R' \subseteq \Sigma^* \times \Sigma^*$ such that $R'(x') \neq \emptyset$ if and only if x' codes $\{x\}, Y \subseteq \Sigma^*$ with $R(x) = Y$. Informally, problems of that kind ask whether every solution to an enumeration problem is obtained. We call *decision version* of Π_R, denoted π_R, such a problem. Most of the enumeration problems considered in this manuscript come along with their decision version. The following is folklore and is proved by running an algorithm for Π_R for a number of steps that is proportional in x and Y being the inputs of π_R.

Proposition 1.2.6. *Let Π_R be an enumeration problem and π_R be its decision version. Then π_R can be solved in polynomial time whenever Π_R can be solved in output-polynomial time.*

Conversely, observe that to any decision problem π_R of the form "given word $x \in \Sigma$ and set $Y \subseteq \Sigma^*$, decide whether $Y = R(x)$ holds" for some arbitrary binary relation R, corresponds an enumeration problem Π_R^1 asking, given x, to enumerate the set $R(x)$, and one Π_R^2 asking, given $R(x)$, to compute x. Yet proven to hold for the dualization of monotone Boolean functions introduced in Section 1.3, it is however not clear in general whether the existence of a polynomial-time algorithm for π_R yields an output-polynomial time algorithm for Π_R^1, and Π_R^2. Additional conditions are usually added to π_R in order to get an equivalence, such as exhibiting a counter example to $Y = R(x)$ in case when $Y \neq R(x)$. These variations will not be encountered in this manuscript.

1.2.7 Reductions and intractability

Several reductions between enumeration problems can be defined, depending on the different complexities that they may preserve. They will naturally appear in this manuscript, starting with the next section, and were formally defined in [Mar13] for the complexities considered in this chapter. We chose here to only give the commonly employed notions of hardness that will be used in the manuscript, and let the reader encounter the reductions over the chapters through particular cases.

Definition 1.2.7. *Let A, B be two binary predicates and Π_A, Π_B be their associated enumeration problems. We say that Π_A is polynomially harder[1] than Π_B (or that Π_A is Π_B-hard) if there exists an output-polynomial time algorithm for Π_B whenever there is one for Π_A. Then, Π_A and Π_B are polynomially equivalent if Π_A is polynomially harder than Π_B, and vice versa.*

Note that the same notions can be defined for incremental-polynomial (resp. polynomial-delay) time complexities. We will always mention explicitly the complexity that is preserved in that case.

In enumeration theory, hardness of a problem Π_A is either shown by proving that Π_A is harder than another tough (often open) problem Π_B, or by showing that getting an output-polynomial time algorithm for Π_A would yield a polynomial-time algorithm for some NP-hard problem. For example, the problems we consider in Chapters 2, 3, and 4 are all known to be harder than the dualization of monotone Boolean functions. As for some generalizations of this problem, considered in Chapter 3, it is known that they cannot be solved in output-polynomial time unless P=NP. In that second framework, we usually reduce the decision version π_A of the enumeration problem Π_A to a well-known NP-hard problem δ by means of classical reductions. By Proposition 1.2.6, this is sufficient to show that the original enumeration problem Π_A cannot be solved in output-polynomial time, unless P=NP. The next corollary follows.

Corollary 1.2.8. *Let Π_R be an enumeration problem and π_R be its decision version. Then Π_R cannot be solved in output-polynomial time if π_R is NP-hard, unless P=NP.*

[1] The reader may be more confortable with the formulation "polynomially (at least) as hard as".

1.3 Dualization of monotone Boolean functions and hypergraph dualization

We end the preliminaries by defining the dualization of monotone Boolean functions, and several equivalent formulations in hypergraphs. These problems are ubiquitous in enumeration theory and are the starting point of this thesis. The original formulation is presented for historicity, while the different hypergraph formulations will be continuously referred to in the rest of this manuscript.

1.3.1 Dualization of monotone Boolean functions

A *Boolean function* is a mapping $f : \{0,1\}^n \to \{0,1\}$. We call *arity* of f the integer n, call *Boolean vector* a tuple v in $\{0,1\}^n$, and note v_i the value of vector v on coordinate i. We note $u \leq v$ if $u_i \leq v_i$ for all i, and call *Boolean algebra of dimension* n the set of all vectors $v \in \{0,1\}^n$ ordered by $\leq$. An example of a Boolean algebra is given in Figure 1.1. A Boolean function f is said to be *monotone* if $u \leq v$ implies $f(u) \leq f(v)$ for all $u, v \in \{0,1\}^n$. It is well known that a Boolean function f is monotone if and only if it admits a conjunctive normal form (CNF)

$$\varphi = \bigwedge_{I \in F} \bigvee_{i \in I} x_i$$

such that $f = \varphi$,[2] and where no *literal* x_i is negated. Such a formula is called *prime* if in addition $I_1 \not\subseteq I_2$ for any two distinct *clauses* $I_1, I_2 \in F$. To every monotone Boolean function corresponds a unique prime CNF. An example of a monotone Boolean function and its prime CNF is given in Figure 1.1. Another essential representation of a monotone Boolean function f consists of its prime disjunctive normal form (DNF)

$$\psi = \bigvee_{J \in G} \bigwedge_{j \in J} x_j$$

such that $f = \psi$, where no literal x_j is negated, and where $J_1 \not\subseteq J_2$ for any two distinct *terms* $J_1, J_2 \in G$; this representation is also unique.

The dual f^d of a Boolean function f is defined by $f^d(v) = \overline{f}(\overline{v})$, where $\overline{f}, \overline{v}$ respectively denote the complement of f and v, i.e., $\overline{f}(v) = 1 - f(v)$ and $\overline{v} = (1 - v_1, \ldots, 1 - v_n)$. It is monotone as well, and by definition, it satisfies the equality $(f^d)^d = f$. The dual of a monotone Boolean function is given in Figure 1.1. It is well known that if φ is a prime CNF formula for f, then a prime DNF formula ψ for f^d is obtained by exchanging $\vee$ and $\wedge$, as well as the constants 0 and 1. For example, the dual prime DNF of $\varphi = (1 \vee 2) \wedge (1 \vee 3)$ is $\psi = 12 \vee 13$. Then, expanding ψ into a CNF using De Morgan's laws yields a prime CNF for the dual. Note that the size of the prime CNF of f^d may be exponential in that of f. A canonical example is the formula $\varphi = (1 \vee 2) \wedge (3 \vee 4) \wedge (5 \vee 6)$, generalized to any arity. In that case, expanding the dual function is still reasonable, a point that was discussed in Section 1.2.4. This is however not always the case, and the expanding approach may result, unfortunately, into an exponential blowup in the sizes of the two functions at hand. A canonical example of this phenomenon is the formula $\varphi = (1 \vee 2) \wedge (2 \vee 3) \wedge (3 \vee 4)$ and its dual, generalized to any arity.

[2] We shall note $f = \varphi$ if $f(x_1, \ldots, x_n) = 1$ if and only if the assignment $x_1, \ldots, x_n$ satisfies φ.

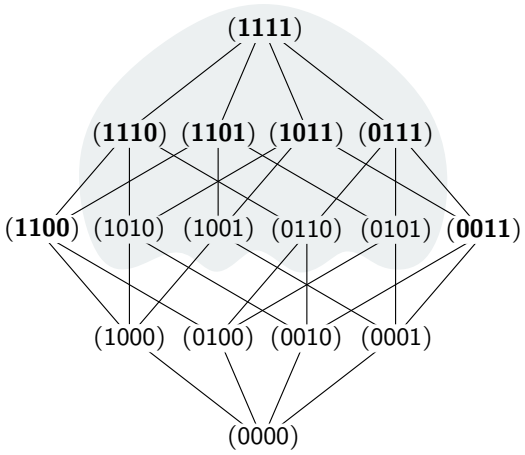


Figure 1.1: The Boolean algebra of dimension four, the monotone Boolean function f of prime CNF $\varphi = (1 \vee 2) \wedge (3 \vee 4)$, and its dual. Vectors v such that $f(v) = 1$ are in the gray area, vectors such that $f^d(v) = 1$ are in bold. It is easily verified that $f(v) = \overline{f}(\overline{v})$.

The next problem naturally arises.[3]

Dualization of Monotone Conjunctive Normal Forms (DUALIZATION)
Input: A pair f, g of monotone Boolean functions given by their prime CNF.
Question: Are f and g dual?

Generation version of DUALIZATION
Input: A monotone Boolean function f given by its prime CNF.
Output: The prime CNF of the dual function f^d.

Observe that the problem could have been equivalently stated as the dualization of monotone *disjunctive* normal forms, by exchanging $\wedge$ and $\vee$. This is the representation considered in the landmark paper [FK96], while the one presented here can be found in the important work [EGM03] and survey [EMG08]. Observe in fact that if f^d has prime CNF $\varphi' = \bigwedge_{I \in F} \bigvee_{i \in I} x_i$, then f has for prime DNF $\psi = \bigvee_{I \in F} \bigwedge_{i \in I} x_i$. Hence, computing the prime CNF of f^d from the prime CNF of f amounts to computing the DNF of f from its prime CNF. As $(f^d)^d = f$, computing the prime CNF of f from its prime DNF is equivalent to computing the prime DNF of f from its prime CNF. This yields the following alternative—and perhaps more easily defined—version of DUALIZATION, and its two (equivalent) generation versions that we shall retain in the following.

Dualization of Monotone Boolean Functions (MONOTONE DUAL)
Input: A pair f, g of monotone Boolean functions—f given by its prime CNF, and g by its prime DNF.
Question: Are f and g equal?

First generation version of MONOTONE DUAL
Input: A monotone Boolean function f given by its prime CNF.
Output: The prime DNF of f.

[3] In the remaining of the document, decision problems will be defined in "input–question" environments, while enumeration problems will be defined in "input–output" environments. Here in the enumeration version, the prime CNF of the dual function f^d is seen as a family of clauses to be output.

Second generation version of MONOTONE DUAL
Input: A monotone Boolean function f given by its prime DNF.
Output: The prime CNF of f.

As stated in the introduction, testing the duality of two monotone Boolean functions is ubiquitous in many areas of computer science [EG95, GMKT97, DMP99]. The problem has become so central in enumeration theory that it is now common, in order to characterize the complexity of an enumeration problem, either to exhibit an output-polynomial time algorithm, or to show that the problem is at least as hard as the dualization [MS19, CGK+19, CKMU19], a line of research that emerged from [KLMN14]. For harder problems, it is often desirable to obtain an output quasi-polynomial time algorithm by reduction to the dualization [MR92, Wil00, NP14, AN17, BMN17, AN19]. Hence depending on the problem, the dualization of monotone Boolean functions either plays a tractable, or an intractable frontier, concerning the complexity of a problem. Its precise complexity status is thereby one of the most challenging open question in enumeration theory.

To date, the best known algorithm is due to Fredman and Khachiyan [FK96] and runs in quasi-polynomial time $N^{o(\log N)}$ where $N = |\varphi| + |\psi|$, with φ and ψ being the prime CNF and DNF of two functions f and g, and where $|\varphi|$ denotes the number of clauses in φ. In consequence, the problem is unlikely to be NP-hard. Furthermore, it is known from [BI95] that there is a polynomial time algorithm solving MONOTONE DUAL if and only if there is an incremental-polynomial time algorithm solving its generation version. Such an incremental quasi-polynomial time algorithm for the generation version of MONOTONE DUAL is made explicit in [FK96].

Since the algorithm of Fredman and Khachiyan, progress has been made on various subclasses of monotone functions. Most importantly, output-polynomial time algorithms were obtained in degenerate CNFs, read-k CNFs, acyclic CNFs and formulas of bounded treewidth [EGM03].

1.3.2 Hypergraph dualization

A *hypergraph* $\mathcal{H}$ is a couple $(V(\mathcal{H}), \mathcal{E}(\mathcal{H}))$ where $V(\mathcal{H})$ is called *vertex set* (or *ground set*), and where $\mathcal{E}(\mathcal{H}) \subseteq 2^{V(\mathcal{H})}$ is a family of subsets called *hyperedges*. A hypergraph is called *Sperner* if $E_1 \not\subseteq E_2$ for any two distinct hyperedges $E_1, E_2 \in \mathcal{E}(\mathcal{H})$. Examples of Sperner hypergraphs are given in Figure 1.2. A *transversal* of $\mathcal{H}$ is a subset $T \subseteq V(\mathcal{H})$ of vertices that intersects every hyperedge $E \in \mathcal{E}(\mathcal{H})$. It is (inclusion-wise) minimal if $T \setminus \{x\}$ is no longer a transversal for any $x \in T$. The set of all minimal transversals of $\mathcal{H}$ is denoted by $Tr(\mathcal{H})$.

Note that if $\mathcal{H}$ is a hypergraph, then $(V(\mathcal{H}), Tr(\mathcal{H}))$ also defines a hypergraph, which is Sperner. In this manuscript, and as is custom, we will sometimes refer to a hypergraph by its set of hyperedges only, when the ground set is clear from the context, or more generally when it is obtained by union of the hyperedges (i.e., no vertex is isolated). We shall for example refer to $Tr(\mathcal{H})$ as the hypergraph $(V(\mathcal{H}), Tr(\mathcal{H}))$. Then, two hypergraphs $\mathcal{H}$ and $\mathcal{G}$ on same ground set $V(\mathcal{H}) = V(\mathcal{G})$ are called *dual* if $\mathcal{G} = Tr(\mathcal{H})$. An example of two dual hypergraphs is given in Figure 1.2. It can be observed that if $\mathcal{H}$ is a Sperner hypergraph, then $Tr(Tr(\mathcal{H})) = \mathcal{H}$. It was also observed in [Ber84] that any hypergraph $\mathcal{H}$ that is not Sperner may always be reduced in poly-

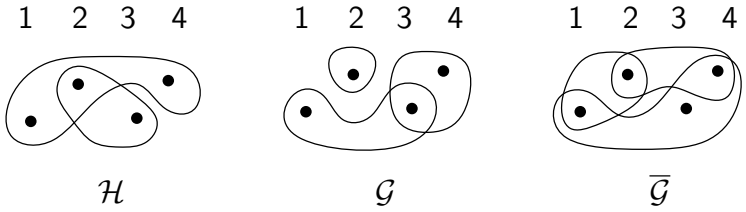


Figure 1.2: Two dual hypergraphs $\mathcal{H}$ and $\mathcal{G}$, as well as the complementary hypergraph $\overline{\mathcal{G}}$ of $\mathcal{G}$ defined by $V(\overline{\mathcal{G}}) = V(\mathcal{G})$ and $\mathcal{E}(\overline{\mathcal{G}}) = \{V(\mathcal{G}) \setminus E \mid E \in \mathcal{G}\}$. Then $\mathcal{G} = Tr(\mathcal{H})$ and $\overline{\mathcal{G}} = MIS(\mathcal{H})$.

nomial time into a Sperner hypergraph $\mathcal{H}'$, with no incidence on the set of its minimal transversals, i.e., such that $Tr(\mathcal{H}) = Tr(\mathcal{H}')$.

The next problem follow.

Hypergraph Dualization (HYPERGRAPH DUAL)
Input: Two hypergraphs $\mathcal{H}$ and $\mathcal{G}$.
Question: Are $\mathcal{H}$ and $\mathcal{G}$ dual?

Generation version of HYPERGRAPH DUAL (TRANS-ENUM)
Input: A hypergraph $\mathcal{H}$.
Output: The set $Tr(\mathcal{H})$ of all minimal transversals of $\mathcal{H}$.

It is well known that the prime DNF ψ of a monotone Boolean function f corresponds to its minimal true vectors. Hence that each term of ψ minimally intersects each clause of the prime CNF φ of f. Free to disregard CNFs and DNFs as hypergraphs, this establishes the equivalence of HYPERGRAPH DUAL with MONOTONE DUAL, hence of HYPERGRAPH DUAL with DUALIZATION. This equivalence has brought many new results to the dualization problem. Most notably, the dualization was shown to be solvable in polynomial time on hypergraphs of bounded conformality by Boros, Elbassioni, Gurvich, and Khachiyan [BEGK04, KBEG07]. This class of hypergraph will be defined in Chapter 2 and contains hypergraphs of bounded edge intersection, and of bounded edge size. This algorithm will be used as a black box in three of the algorithms presented in this manuscript. Other tractable cases that will not be used in the following include degenerate hypergraphs [EGM03], α and β acyclic hypergraphs [EG95, EGM03], or hypergraphs without small holes [KKP18].

As for DUALIZATION, several forms of hypergraph dualization exist. We introduce another generation version that will be used in Chapter 4. An *independent set* of $\mathcal{H}$ is a subset $I \subseteq V(\mathcal{H})$ of vertices such that $E \not\subseteq I$ for any $E \in \mathcal{E}(\mathcal{H})$. It is called (inclusion-wise) *maximal* if $I \cup \{x\}$ is no longer an independent set for any $x \in V(\mathcal{H}) \setminus T$. The set of all maximal independent sets of $\mathcal{H}$ is denoted by $MIS(\mathcal{H})$, and the corresponding enumeration problem defined as follows.

Hypergraph Maximal Independent Sets Enumeration (MIS-ENUM)
Input: A hypergraph $\mathcal{H}$.
Output: The set $MIS(\mathcal{H})$ of all maximal independent sets of $\mathcal{H}$.

The maximal independent sets of a hypergraph are given in Figure 1.2. It is easily seen that a subset I of $V(\mathcal{H})$ is a maximal independent set of $\mathcal{H}$ if and only if its com-

plementary $T = V(\mathcal{H}) \setminus I$ is a minimal transversal of $\mathcal{H}$. We immediately deduce that TRANS-ENUM and MIS-ENUM are equivalent. Other equivalent forms of hypergraph dualization may be found in the survey [EMG08] and include the enumeration of all minimal set coverings (minimal sets of hyperedges whose union is the ground set) of a hypergraph. These other forms will not be encoutered in this manuscript.

The next three chapters are devoted to the study of monotone dualization through the other shapes it takes on graphs and lattices: minimal dominating sets enumeration, lattice dualization, and meet-irreducible enumeration.

Chapter 2

Minimal dominating sets enumeration

In this chapter we present new algorithms for the problem of enumerating all minimal dominating sets in graphs. Parts of these results appeared in joint works with Marthe Bonamy, Marc Heinrich, Michał Pilipczuk, and Jean-Florent Raymond [BDHR19, BDH+20]. Other parts were obtained with Marthe Bonamy, Piotr Micek, and Lhouari Nourine, and may be found in the preprint [BDMN20].

We saw in the previous chapter how the monotone dualization problem could be formulated in term of transversality in hypergraphs. It has recently been proved by Kanté, Limouzy, Nourine, and Mary [KLMN14] that the problem could also be formulated as the enumeration of all (inclusion-wise) minimal dominating sets in graphs. A dominating set in a graph G is a set D of vertices such that every vertex of G is either in D, or adjacent to a vertex of D. It is (inclusion-wise) minimal if no strict subset is a dominating set. Since [KLMN14], this problem has gained lots of interest, and characterizing its complexity in particular graph classes has become a fruitful line of research. We review here the main contributions of the last ten years. Incremental-polynomial time algorithms were found for chordal bipartite graphs [GHK+16], $\{C_6, C_8\}$-free bipartite graphs [KKP18], and unit square graphs [GHK+18]. Polynomial-delay algorithms were obtained in chordal and line graphs [KLM+15a, KLM+15b]. Linear-delay algorithms were given for permutation and interval graphs [KLM+13], graphs of bounded clique width [Cou09], LMIM-width [GHK+18], split and P_6-free chordal graphs [KLMN14].

In addition to these recent results, the problem inherited from previously known algorithms on hypergraphs and CNFs. Most notably, the problem is known to be solvable in output-polynomial time in $\log(n)$-degenerate graphs [EGM03], and with polynomial delay in planar and degenerate graphs [EGM03], as well as in graphs of bounded conformality [BEGK04, KLMN12].

When starting this thesis, two important graph classes were notably open: bipartite graphs, and co-bipartite graphs [KN14]. We show in this chapter that the first class admits an output-polynomial time algorithm, by proving the case of graphs without big cliques. Our result is based on a technique called *ordered generation* on reducing the enumeration to smaller pieces of the initial instance. This result puts the light on a super class of bipartite graphs that remains open: the class of comparability graphs. We show this later class to be tractable whenever the underlying partial order (or poset) is of bounded dimension. This result is based on a recent technique technique called *flip-*

ping method on enumerating minimal dominating sets from (inclusion-wise) maximal independent sets. Concerning co-bipartite graphs, it is known that the problem in that class is as hard as the general case [KLMN14]. This therefore holds for super classes of co-bipartite graphs such as incomparability graphs. For this later class, we nevertheless show the problem to be tractable whenever the underlying poset is, again, of bounded dimension. This is based on *flashlight search*, a different technique, and relies on the geometrical representation of incomparability graphs of bounded dimension, which was given by Golumbic et al. in [GRU83].

The chapter is organized as follows. In Section 2.1 we introduce necessary concepts and definitions, and recall a result from [KLMN14] on the equivalence of TRANS-ENUM and the enumeration of minimal dominating sets. Output-polynomial, incremental-polynomial, and polynomial-delay algorithms are respectively given for several graph classes related to K_t-free graphs, and for the comparability and incomparability graphs of posets of bounded dimension, in Sections 2.2, 2.3, and 2.4. Future research directions are discussed in Section 2.5.

2.1 Preliminaries

We define here notions related to graphs, posets, and domination that will be used in this chapter.

2.1.1 Graphs

A graph G is a pair $(V(G), E(G))$ with $V(G)$ its set of vertices (or *ground set*) and $E(G) \subseteq \{\{u, v\} \mid u, v \in V(G),\ u \neq v\}$ its set of edges. An example of a graph is given in Figure 2.1. Edges are denoted by uv (or vu) instead of $\{u, v\}$. Two vertices u, v of G are called *adjacent* if $uv \in E(G)$. In this chapter, and as is common, we will usually denote $|V(G)|$ by n, and $|E(G)|$ by m. A *clique* (respectively an *independent set*) in a graph G is a set of pairwise adjacent (respectively non-adjacent) vertices. A *biclique* is a set of vertices that can be partitioned into two independent sets A, B such that every vertex in A is adjacent to every vertex in B. We note K_t the clique on t elements, and $K_{t,t}$ the biclique on $2t$ elements of partition A, B such that $|A| = |B| = t$. We note $K_t - e$ the clique on t elements minus an edge, and $K_t + K_2$ the disjoint union of K_t and an edge. The subgraph of G *induced* by $X \subseteq V(G)$, denoted by $G[X]$, is the graph $(X, E(G) \cap \{\{u, v\} \mid u, v \in X,\ u \neq v\})$; $G - X$ is the graph $G[V(G) \setminus X]$. For every graph H, we say that G is H-free if no induced subgraph of G is isomorphic to H.

If the vertex set of a graph G can be partitioned into one part inducing a clique and one part inducing an independent set (respectively two independent sets, two cliques), we say that G is a *split* (respectively *bipartite*, *co-bipartite*) graph. A *complete multipartite graph* is a graph which vertex set can be partitionned into several independent sets such that there is an edge between every pair of vertices from different independent sets.

Some small graphs we consider in this manuscript are P_3, which is the path on three vertices (i.e., $G =$ •–•–•), its complementary $\overline{P_3}$ (i.e., $G =$ •–• •), the *paw*, which is the graph obtained by adding a vertex of degree one to K_3 (i.e., $G =$ ), and $K_4 - e$, also known as *diamond* graph (i.e., $G =$ ).

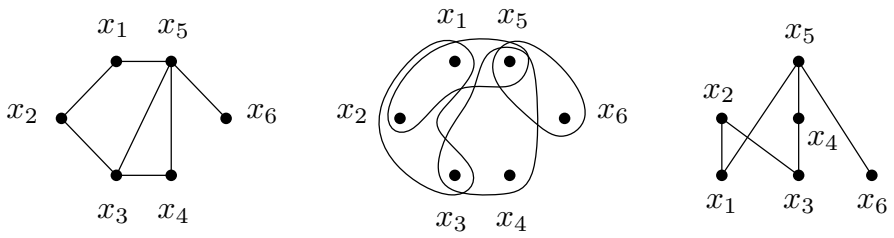


Figure 2.1: A graph G (left), its Sperner hypergraph $\mathcal{N}(G)$ of closed neighborhoods (middle), and a poset P such that G is the comparability graph of P (right). On this example, the set $D = \{x_2, x_4, x_6\}$ is both a minimal dominating set of G, and a minimal transversal of $\mathcal{N}(G)$. In the poset, it satisfies $\uparrow D \cup \downarrow D = V(G)$.

2.1.2 Neighborhood and domination

We define notions related to domination. Let G be a graph and u be a vertex of G. The *neighborhood* of u is the set $N(u) = \{v \in V(G) \mid uv \in E(G)\}$. The *closed neighborhood* of u is the set $N[u] = N(u) \cup \{u\}$. For example in Figure 2.1, the closed neighborhood of x_2 is the set $N[x_2] = \{x_1, x_2, x_3\}$. For a subset $X \subseteq V(G)$ we define $N[X] = \bigcup_{x \in X} N[x]$ and $N(X) = N[X] \setminus X$. Let $D, X \subseteq V(G)$ be two subsets of vertices of G. We say that D *dominates* X if $X \subseteq N[D]$. It is (inclusion-wise) minimal if $X \not\subseteq N[D \setminus \{x\}]$ for any $x \in D$. A (minimal) *dominating set* of G is a (minimal) dominating set of $V(G)$. The set of all minimal dominating sets of G is denoted by $\mathcal{D}(G)$, and the problem of enumerating $\mathcal{D}(G)$ given G defined as follows.

Minimal Dominating Sets Enumeration (DOM-ENUM)
Input: A graph G.
Output: The set $\mathcal{D}(G)$ of all minimal dominating sets of G.

Observe that a set D of vertices is a minimal dominating set of G if it minimally intersects all the closed neighborhoods of G. We immediately deduce that DOM-ENUM is a particular case of TRANS-ENUM by considering the so-called *hypergraph of closed neighborhoods of* G, usually denoted $\mathcal{H} = \mathcal{N}(G)$, and defined by $V(\mathcal{H}) = V(G)$ and $\mathcal{E}(\mathcal{H}) = \{N[x] \mid x \in V(G)\}$. Furthermore, this hypergraph can be considered Sperner by defining $\mathcal{E}(\mathcal{H}) = \mathrm{Min}_{\subseteq}\{N[x] \mid x \in V(G)\}$, where $\mathrm{Min}_{\subseteq}$ denotes the inclusion-wise minimal sets of the family. An example of one such hypergraph is given in Figure 2.1.

In [KLMN14] the authors in fact show that there is a polynomial delay algorithm for DOM-ENUM if and only if there is one for TRANS-ENUM. Since DOM-ENUM is a particular case of TRANS-ENUM, we only need to show how TRANS-ENUM reduces to DOM-ENUM. We briefly review here the construction. For any hypergraph $\mathcal{H}$ we denote by $I(\mathcal{H})$ the *bipartite incidence graph* of $\mathcal{H}$ with bipartition $X = V(\mathcal{H})$ and $Y = \{y_E \mid E \in \mathcal{E}(\mathcal{H})\}$, and where there is an edge between $x \in X$ and $y_E \in Y$ if x belongs to E in $\mathcal{H}$. The construction of a bipartite incidence graph is given in Figure 2.2. Consider now the co-bipartite graph $B(\mathcal{H})$ obtained from $I(\mathcal{H})$ by adding a special vertex v to X, and by completing both X and Y into cliques. Then, it is easily seen that minimal dominating sets of $B(\mathcal{H})$ are either of size two, or consist of minimal subsets of X being adjacent to every $y_E \in Y$. Minimal dominating sets of the second type are exactly the minimal transversals of $\mathcal{H}$. As they is a quadratic number in $n = |V(G)|$ of

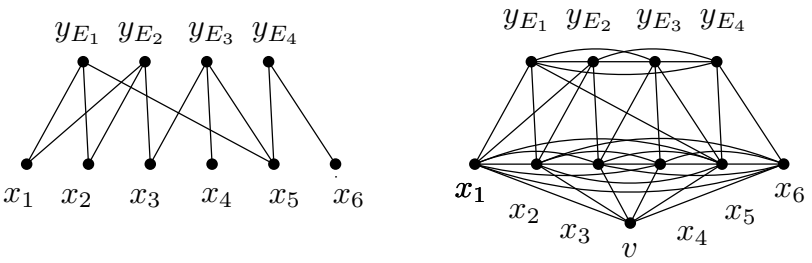


Figure 2.2: The incidence graph $I(\mathcal{H})$ (left) of the hypergraph $\mathcal{H}$ given in Figure 2.1 and defined by $V(\mathcal{H}) = \{x_1, \ldots, x_6\}$ and $\mathcal{E}(\mathcal{H}) = \{E_1, \ldots, E_4\}$, where $E_1 = \{x_1, x_2, x_5\}$, $E_2 = \{x_1, x_2, x_3\}$, $E_3 = \{x_3, x_4, x_5\}$, and $E_4 = \{x_5, x_6\}$. A co-bipartite construction is obtained from $I(\mathcal{H})$ by completing the partitions into cliques and adding a new vertex v adjacent to every x_i (right).

minimal dominating sets of the first type, the next theorem follows.

Theorem 2.1.1 ([KLMN14]). DOM-ENUM *restricted to co-bipartite graphs is polynomially equivalent to* TRANS-ENUM *and* DOM-ENUM.

2.1.3 Maximal independent sets

Recall that an *independent set* in a graph G is a set of pairwise non-adjacent vertices. For a graph G, we denote by $MIS(G)$ the set of all its (inclusion-wise) maximal independent sets, and by MIS-ENUM the problem of generating $MIS(G)$ from G. It is easily observed that every maximal independent set is a minimal dominating set, hence that $MIS(G) \subseteq \mathcal{D}(G)$. However, MIS-ENUM appears to be much more tractable than DOM-ENUM, as witnessed by the many efficient algorithms that are known for the problem [TIAS77, JYP88, MU04]. These last two observations are the starting point of the flipping method introduced in [GHKV15] and that is at the heart of Section 2.3.

2.1.4 Private neighbors

We define notions that are useful when dealing with domination. Let $u \in D$. We call *private neighbor* of u with respect to D a vertex $v \in V(G)$ that is only dominated by u in D, i.e., that is such that $N[v] \cap D = \{u\}$. Note that u can be its own private neighbor (we say that u is *self-private*). The set of private neighbors of $u \in D$ is denoted by $\mathrm{Priv}(D, u)$. A set $I \subseteq V(G)$ is called *irredundant* if $\mathrm{Priv}(I, x) \neq \emptyset$ for all $x \in I$. The following observation is folklore.

Observation 2.1.2. *A set $D \subseteq V(G)$ is a minimal dominating set of G if and only if it is both a dominating set, and an irredundant set.*

2.1.5 Red blue domination

We define a slightly different setting of domination that is of interest in Section 2.3. A *red-blue graph* $G(R, B)$ is a graph G together with two disjoint subsets $R, B \subseteq V(G)$

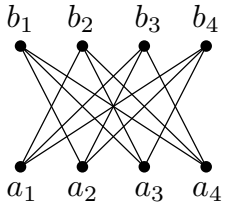


Figure 2.3: The standard example of order four.

where vertices in R are called *red* and those in B are *blue*. For simplicity in the following, we do not require R and B to partition $V(G)$. A red dominating set of $G(R, B)$ is a red set $D \subseteq R$ that dominates B, i.e., that is such that $B \subseteq N[D]$. It is (inclusion-wise) minimal if $B \not\subseteq N[D \setminus \{x\}]$ for any $x \in D$. We note $\mathcal{D}(R, B)$ the set of all minimal red dominating sets of $G(R, B)$, and RED-BLUE-DOM-ENUM the problem of enumerating $\mathcal{D}(R, B)$ given G, R and B. In the context of red-blue domination, dominating sets are always assumed to be red, and private neighbors assumed to be blue.

It is easily seen in Figure 2.2 that the minimal transversals of $\mathcal{H}$ are the red dominating sets of $G(R, B)$ where $G = I(\mathcal{H})$, $R = X$ and $B = Y$, no matter the edges in X and Y. In fact, as the edges induced by R and B may be ignored when considering red-blue domination, and by definition of bipartite incidence graphs, RED-BLUE-DOM-ENUM and TRANS-ENUM may be seen as two formulations of the same problem. We in particular deduce the following folklore theorem in the line of Theorem 2.1.1.

Theorem 2.1.3 (Folklore). RED-BLUE-DOM-ENUM *restricted to bipartite and co-bipartite graphs is polynomially equivalent to* TRANS-ENUM *(hence to* DOM-ENUM*).*

2.1.6 Posets and (in)comparability graphs

We conclude the section with notions from posets that will be used in Section 2.3.

A *partially ordered set* (or *poset*) $P = (V, \leq)$ is a binary relation $\leq$ on a ground set V which is reflexive, anti-symmetric and transitive. Two elements u and v of P are said to be *comparable* if $u \leq v$ or $v \leq u$, otherwise they are said to be *incomparable*, denoted $u \parallel v$. We note $x < y$ if $x \leq y$ and $x \neq y$. Posets are represented by their *Hasse diagram* in which each element is a vertex in the plane, and where there is a line segment or curve that goes upward from x to y whenever $x < y$ and there is no z such that $x < z < y$. A *chain* (respectively an *antichain*) in a poset P is a set of pairwise comparable (respectively incomparable) vertices; P is called a *total order* (or *linear order*) if V is a chain.

The *comparability graph* of a poset $P = (V, \leq)$ is the graph G defined on same ground set $V(G) = V$ and where two vertices u and v are made adjacent if they are comparable in P. An example of a poset and its comparability graph is given in Figure 2.1. The *incomparability graph* (or *co-comparability graph*) of P is the complementary: u and v are made adjacent if they are incomparable in P.

We now define notations from order theory that will be used in the context of a poset and its comparability graph only. Let G be the comparability graph of a poset $P = (V, \leq)$. The *ideal of* u is the set $\downarrow u = \{v \in V \mid v \leq u\}$, and the *filter of* u is the dual $\uparrow u = \{v \in V \mid u \leq v\}$. Then $N[u] = \downarrow u \cup \uparrow u$. These notions extend to subsets $S \subseteq V$ as follows: $\downarrow S = \bigcup_{u \in S} \downarrow u$, $\uparrow S = \bigcup_{u \in S} \uparrow u$. Then a (minimal) dominating set of

G is a (minimal) set $D \subseteq V$ such that $\uparrow D \cup \downarrow D = V$, and a (maximal) independent set of G is a (maximal) antichain of P. We note $\mathrm{Min}(S)$ and $\mathrm{Max}(S)$ the sets of minimal and maximal elements in S with respect to $\leq$. Clearly, $\mathrm{Min}(S)$ and $\mathrm{Max}(S)$ define antichains of P for every $S \subseteq V$.

We call poset induced by $X \subseteq V$, denoted $P[X]$, the suborder restricted on the elements of X only. A poset $P = (V, \leq)$ is *bipartite* if V can be partitioned into two sets A, B such that $a < b$ implies $a \in A$ and $b \in B$. We note S_t the *standard example* with bipartition $A = \{a_1, \ldots, a_t\}$ and $B = \{b_1, \ldots, b_t\}$ such that $a_i < b_j$ for all $i, j \in \{1, \ldots, t\}$, $i \neq j$. See Figure 2.3 for an example of such a poset.

The dimension of a poset $P = (V, \leq)$ is the least t such that P is the intersection of t linear orders $<_1, \ldots, <_t$ on V, i.e., $x \leq y$ if and only if $x <_1 y, \ldots, x <_t y$. It is well known that S_t has dimension t [DM41]. Note that dimension is monotone under taking induced suborders. Therefore, posets containing a large standard example as a suborder have large dimension. The converse is however not true, as there are posets of unbounded dimension with no S_2 as an induced suborder, see [Tro92] for a comprehensive study and references.

2.2 Ordered generation in K_t-free graphs and variants

We are now ready to present the first contributions of this thesis, giving output-polynomial time algorithms for DOM-ENUM in graph classes related to K_t-free graphs.

2.2.1 Ordered generation in bicolored graphs

We describe a general procedure that will be used throughout Section 2.2. This procedure will construct minimal dominating sets one neighborhood at a time, in a variant of what is known as the *backtrack search technique* in [RT75, FLM97, MS19], and referred to as *ordered generation* in [EGM03].

In what follows, we find it more convenient to deal with the slightly more general setting of domination in bicolored graphs. A *bicolored graph* is a graph together with a subset of its vertex set. For a graph G and a subset $A \subseteq V(G)$, we denote by $G(A)$ the bicolored graph G with prescribed set A. We also say that G has *bicoloring* $(A, V(G) \setminus A)$. Then, a *dominating set of* $G(A)$ is a set $D \subseteq V(G)$ that dominates A, i.e., such that $A \subseteq N[D]$. It is (inclusion-wise) minimal if it does not contain any dominating set of $G(A)$ as a proper subset. Intuitively, the vertices of $G - A$ may be used in the dominating set, but do not need to be dominated. For every graph G and subset $A \subseteq V(G)$, we denote by $\mathcal{D}(G, A)$ the set of minimal dominating sets of $G(A)$. Then $\mathcal{D}(G, A) = \mathcal{D}(G)$ whenever $A = V(G)$.

A *peeling* of a bicolored graph $G(A)$ is a sequence of vertex sets $(V_0, \ldots, V_p)$ such that $V_p = A$, $V_0 = \emptyset$, and for every $i \in \{1, \ldots, p\}$, there is a vertex $v_i \in V_i$ such that

$$V_{i-1} = V_i \setminus N[v_i].$$

We call $(v_1, \ldots, v_p)$ the *vertex sequence* of the peeling. It is straightforward to see that given a bicolored graph $G(A)$, any peeling of $G(A)$ can be computed in $O(n^2)$ time and space: start with the whole set A, and as long as A remains non-empty, pick a vertex v in it and remove $N[v]$ from A. The representation of a peeling is given in Figure 2.4.

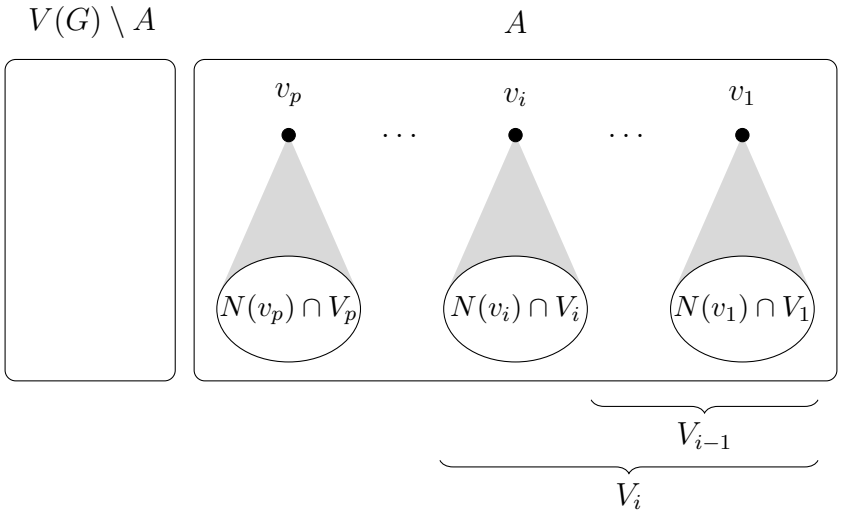


Figure 2.4: Representation of a peeling of a bicolored graph $G(A)$ constructed by iteratively removing v_i's and their neighborhoods, for i from p to 1. Note that vertices that are effectively removed at step i are those of $N[v_i] \cap V_i$, as vertices in $N[v_i] \setminus V_i$ have already been removed at a previous step. A crucial property is that v_i has no neighbor in V_{i-1}.

In the remaining of this section, we consider a bicolored graph $G(A)$, together with a fixed peeling $(V_0, \ldots, V_p)$ of $G(A)$ with vertex sequence $(v_1, \ldots, v_p)$. Observe that $\mathcal{D}(G, V_p) = \mathcal{D}(G, A)$. We now define the relation that will be used by our algorithm to enumerate the minimal dominating sets of $G(A)$ without repetition. Recall that the sets of $\mathcal{D}(G, V_i)$ may contain vertices of $G - V_i$, which is a crucial point.

Definition 2.2.1. *Let $i \in \{0, \ldots, p-1\}$ and $D \in \mathcal{D}(G, V_{i+1})$. We define $\mathrm{Parent}(D, i+1)$ as the pair (D^*, i) where D^* is obtained from D by exhaustively applying the following operation: as long as there exists a vertex x in D satisfying $\mathrm{Priv}(D, x) \cap V_i = \emptyset$, remove from D the vertex of smallest index with this property.*

Clearly, there is a unique way to build $\mathrm{Parent}(D, i+1)$ given D and i. By construction, the obtained set D^* is a minimal dominating set of $G(V_i)$. Hence, every set in $\mathcal{D}(G, V_{i+1})$ can be obtained by completing some D^* in $\mathcal{D}(G, V_i)$; we develop this point below.

Proposition 2.2.2. *Let $i \in \{0, \ldots, p-1\}$ and $D^* \in \mathcal{D}(G, V_i)$. Then:*

(i) if D^* dominates V_{i+1} then $D^* \in \mathcal{D}(G, V_{i+1})$ and $\mathrm{Parent}(D^*, i+1) = (D^*, i)$;

(ii) otherwise, $D^* \cup \{v_{i+1}\} \in \mathcal{D}(G, V_{i+1})$ and $\mathrm{Parent}(D^* \cup \{v_{i+1}\}, i+1) = (D^*, i)$.

Proof. First note that since $D^* \in \mathcal{D}(G, V_i)$, for all $x \in D^*$ we have $\mathrm{Priv}(D^*, x) \cap V_i \neq \emptyset$, implying also $\mathrm{Priv}(D^*, x) \cap V_{i+1} \neq \emptyset$. Hence, if D^* dominates V_{i+1} then in fact D^* is a minimal dominating set of $G(V_{i+1})$ and thus $D^* \in \mathcal{D}(G, V_{i+1})$. Since $\mathrm{Priv}(D^*, x) \cap V_i \neq \emptyset$ for all $x \in D^*$, we then have that $\mathrm{Parent}(D^*, i+1) = (D^*, i)$ directly from the definition.

Suppose now that D^* does not dominate V_{i+1}, and observe that then $D = D^* \cup \{v_{i+1}\}$ does. Moreover, $\mathrm{Priv}(D, v_{i+1}) \cap V_{i+1} \neq \emptyset$. Since v_{i+1}, by the definition of the

peeling, is not adjacent to any vertex in V_i, it cannot steal any private neighbor from the elements of D^*, i.e., $\mathrm{Priv}(D^*, x) \cap V_i \neq \emptyset$ implies $\mathrm{Priv}(D^* \cup \{v_{i+1}\}, x) \cap V_i \neq \emptyset$ for any $x \in D^*$. Hence $\mathrm{Priv}(D, x) \cap V_{i+1} \neq \emptyset$ for all $x \in D$. Now, note that since v_{i+1} does not steal private neighbors from the elements of D^*, it is indeed the only node in D with no private neighbors in V_i, and it is removed when constructing $\mathrm{Parent}(D, i+1)$. Hence $\mathrm{Parent}(D, i+1) = (D^*, i)$, as claimed. □

The Parent function as introduced in Definition 2.2.1 defines a tree on vertex set

$$\{(D, i) \mid i \in \{0, \ldots, p\}\,,\ D \in \mathcal{D}(G, V_i)\},$$

with leaves $\{(D, p) \mid D \in \mathcal{D}(G, A)\}$ and root $(\emptyset, 0)$ (the empty set being the only minimal dominating set of the empty vertex set V_0). Our algorithms will search this tree in order to enumerate the minimal dominating sets of $G(A)$. Proposition 2.2.2 guarantees that for every $i < p$ and every $D^* \in \mathcal{D}(G, V_i)$, the pair (D^*, i) is the parent of some $(D, i+1)$ with $D \in \mathcal{D}(G, V_{i+1})$ (possibly $D = D^*$). Consequently, every branch of the tree leads to a different minimal dominating set of $G(A)$. In particular, for every $i \in \{0, \ldots, p-1\}$ we have

$$|\mathcal{D}(G, V_i)| \leq |\mathcal{D}(G, V_{i+1})| \leq |\mathcal{D}(G, A)|. \tag{2.1}$$

Given a set $D^* \in \mathcal{D}(G, V_i)$, we now focus on the enumeration of all $D \in \mathcal{D}(G, V_{i+1})$ such that $\mathrm{Parent}(D, i+1) = (D^*, i)$. Any (inclusion-wise) minimal set $X \subseteq V(G)$ such that $V_{i+1} \subseteq N[D^* \cup X]$ will be called a *candidate extension* of (D^*, i). In other words, X is a candidate extension of (D^*, i) if and only if it is a minimal dominating set of the bicolored graph G with prescribed set $V_{i+1} \setminus N[D^*]$. Then, we denote by $\mathcal{C}(D^*, i)$ the set of all candidate extensions of (D^*, i), i.e.,

$$\mathcal{C}(D^*, i) \stackrel{\text{def}}{=} \mathcal{D}(G, V_{i+1} \setminus N[D^*]). \tag{2.2}$$

Observe that if $(D, i+1)$ has (D^*, i) as its parent, then $D \setminus D^*$ is candidate extension of (D^*, i). From Proposition 2.2.2, we also know that one of $(D^*, i+1)$ and $(D^* \cup \{v_{i+1}\}, i+1)$ has (D^*, i) as its parent, hence that either $\emptyset$ or $\{v_{i+1}\}$ is a candidate extension of (D^*, i). Note that we have no guarantee that any other candidate extension forms a minimal dominating set of V_{i+1}, together with D^*. We show that it is still reasonable to test each of the candidate extensions even though D^* might have a unique child.

Lemma 2.2.3. *Let $H(B)$ be a bicolored graph and $D \subseteq V(H)$. Then*

$$|\mathcal{D}(H, B \setminus N[D])| \leq |\mathcal{D}(H, B)|.$$

Proof. We argue that for every $X \in \mathcal{D}(H, B \setminus N[D])$ we can find a set $D_X \in \mathcal{D}(H, B)$ so that the sets D_X are pairwise different for different X; this assertion immediately implies the desired inequality. For this, we define D_X as any minimal dominating set of $H(B)$ that is a subset of $D \cup X$; such a set exists as $D \cup X$ dominates B. By definition, every vertex of X has a private neighbor in $B \setminus N[D]$ so we have $X \subseteq D_X$. Moreover, since X is a minimal dominating set of $B \setminus N[D]$, X is disjoint with D. We conclude that $X = D_X \setminus D$, and hence that the sets D_X are pairwise different for different X. □

As a consequence of Lemma 2.2.3 and Inequality (2.1), we have the following.

Corollary 2.2.4. *Let $i \in \{0, \dots, p-1\}$ and $D^* \in \mathcal{D}(G, V_i)$. Then $|\mathcal{C}(D^*, i)| \leq |\mathcal{D}(G, A)|$.*

We conclude the ordered generation with the following statement, which reduces the existence of an output-polynomial time algorithm enumerating $\mathcal{D}(G, A)$ to the existence of one enumerating $\mathcal{C}(D^*, i)$ for any $i \in \{0, \dots, p-1\}$ and $D^* \in \mathcal{D}(G, V_i)$.

Theorem 2.2.5. *Let $f \colon \mathbb{N}^2 \to \mathbb{N}$ and $s \colon \mathbb{N} \to \mathbb{N}$ be two functions. Assume that there is an algorithm that, given a bicolored graph $G(A)$ on n vertices, a peeling $(V_0, \dots, V_p)$ of $G(A)$, $i \in \{0, \dots, p-1\}$, and $D^* \in \mathcal{D}(G, V_i)$, enumerates $\mathcal{C}(D^*, i)$ in time at most $f(n, |\mathcal{D}(G, A)|)$ and space at most $s(n)$. Then there is an algorithm that, given a bicolored graph $G(A)$ on n vertices, enumerates the set $\mathcal{D}(G, A)$ in time*

$$O(n^4 d^2 + f(n, d) \cdot nd)$$

and space $O(n \cdot s(n))$, where $d = |\mathcal{D}(G, A)|$.

Proof. Let us assume that there exists an algorithm $\mathtt{B}$ that, given a bicolored graph $G(A)$ on n vertices, a peeling $(V_0, \dots, V_p)$ of $G(A)$, $i \in \{0, \dots, p-1\}$ and $D^* \in \mathcal{D}(G, V_i)$, enumerates $\mathcal{C}(D^*, i)$ in time at most $f(n, |\mathcal{D}(G, A)|)$ and space at most $s(n)$. Note that we may assume that $s(n) \in \Omega(n)$, as $\mathtt{B}$ needs to store its input. We describe an algorithm $\mathtt{A}$ that enumerates $\mathcal{D}(G, A)$ within the specified time and space complexities.

The algorithm first checks whether $A = \emptyset$ and, if so, returns $\{\emptyset\}$. Otherwise, it computes a peeling $(V_0, \dots, V_p)$ of $G(A)$ in time $O(n^2)$ and using $O(n^2)$ space. Recall that the Parent relation defines a tree T on vertex set

$$\{(D, i) \mid i \in \{0, \dots, p\}\,,\ D \in \mathcal{D}(G, V_i)\},$$

with leaves $\{(D, p) \mid D \in \mathcal{D}(G, A)\}$ and root $(\emptyset, 0)$. Therefore, in order to enumerate $\mathcal{D}(G, A)$, it is enough for $\mathtt{A}$ to enumerate the leaves of T. To do so, the algorithm performs a depth-first search (DFS) of T outputting each visited leaf. For each node (D^*, i), $i \in \{0, \dots, p-1\}$ of T, the algorithm runs $\mathtt{B}$ on input $(G(A), (V_0, \dots, V_p), i, D^*)$ to generate $\mathcal{C}(D^*, i)$ in time $f(n, d)$ and space $s(n)$. For every $X \in \mathcal{C}(D^*, i)$ generated by $\mathtt{B}$, the algorithm tests whether $D^* \cup X$ is a minimal dominating set of V_{i+1}, and whether $\mathrm{Parent}(D^* \cup X, i+1) = (D^*, i)$. This requires $O(n^3)$ steps per candidate extension, and a total working space of $O(n)$, disregarding the space needed to store the (globally fixed) graph G. As by Corollary 2.2.4 we have $|\mathcal{C}(D^*, i)| \leq |\mathcal{D}(G, A)| = d$, the total time spent by $\mathtt{A}$ at each node of T is bounded by $O(n^3 d + f(n, d))$. By Inequality (2.1) we have $|V(T)| \leq pd$ and clearly $p \leq n$, so the total running time of $\mathtt{A}$ is bounded by

$$O(n^4 d^2 + f(n, d) \cdot nd).$$

Regarding the space, we observe that whenever we visit a node of T, we do not need to compute the whole set of its children. Instead, it is enough in order to continue the DFS to compute the next unvisited child only, which can be done using $\mathtt{B}$ and pausing it afterward. Therefore, when we visit some $(D, i) \in V(T)$, we only need to store the data of the $i - 1$ (paused) executions of $\mathtt{B}$ enumerating the children of the ancestors of (D, i), plus the data of the algorithm enumerating the children of D, i.e., $i \cdot (O(n) + s(n))$ space. As $s(n) \in \Omega(n)$, the described algorithm uses $O(n \cdot s(n))$ space, as claimed. □

2.2.2 Candidate extensions in triangle-free graphs

We show that candidate extensions can be enumerated in output-polynomial time in triangle-free graphs, which by Theorem 2.2.5 leads to an output-polynomial time algorithm enumerating minimal dominating sets in this class of graphs. In fact, our result holds in the more general context where only the graph induced by the set that needs to be dominated is required to be triangle-free, and not necessarily the whole graph, a point that is discussed in Section 2.2.5.

In the following, we consider a bicolored graph $G(A)$ on n vertices. Moreover, we have a fixed peeling $(V_0, \ldots, V_p)$ of $G(A)$ with vertex sequence $(v_1, \ldots, v_p)$. Then, we consider

$$i \in \{0, \ldots, p-1\}, \qquad D^* \in \mathcal{D}(G, V_i),$$

and define $\mathcal{C}(D^*, i)$ as in Equality (2.2) in Section 2.2.1. We will show how to enumerate $\mathcal{C}(D^*, i)$ in output-polynomial time whenever $G[A]$ is triangle-free.

Kanté, Limouzy, Mary, and Nourine gave the following characterization of minimal dominating sets in split graphs.

Proposition 2.2.6 ([KLMN14]). *Let H be a split graph with vertices partitioned into an independent set S and a clique C, where S is taken to be (inclusion-wise) maximal. Then, for every $D \in \mathcal{D}(H)$ the following holds:*

(i) $D \cap S = S \setminus N(D \cap C)$, so in particular D is uniquely determined by its intersection with the clique; and

(ii) for every $x \in D$, $\mathrm{Priv}(D, x) \cap S \neq \emptyset$.

Furthermore, $\mathcal{D}(H)$ can be enumerated with delay $O(n^2)$ and using $O(n^2)$ space.

We can now use Proposition 2.2.6 to establish the following understanding of candidate extensions in terms of minimal dominating sets of an auxiliary split graph. The set S in the next lemma corresponds to the elements that, together with v_{i+1}, must be dominated by the candidate extensions of (D^*, i). This situation is depicted in Figure 2.5.

Lemma 2.2.7. *Suppose that $N(v_{i+1})$ is an independent set. Let $S = V_{i+1} \setminus (N[D^*] \cup \{v_{i+1}\})$, $C = N(S) \setminus \{v_{i+1}\}$, and let H be the split graph obtained from $G[S \cup C]$ by completing C into a clique. Then, every set X in $\mathcal{C}(D^*, i)$ either belongs to $\mathcal{D}(H)$, or belongs to $\mathcal{D}(H)$ after removing one vertex (i.e., $X \setminus \{u\} \in \mathcal{D}(H)$ for some $u \in X$), or is such that $X = \{v_{i+1}\}$. Moreover, $|\mathcal{D}(H)| \leq n \cdot |\mathcal{C}(D^*, i)| + 1$.*

Proof. Consider any $X \in \mathcal{C}(D^*, i)$, $X \neq \{v_{i+1}\}$. Then, by definition, X is a minimal dominating set of $V_{i+1} \setminus N[D^*]$. By assumption as $V_{i+1} \setminus N[D^*] \subseteq N[v_{i+1}]$, we have $v_{i+1} \notin X$. Observe then that $X \subseteq C \cup S$.

We first consider the case when $v_{i+1} \in N[D^*]$. Then $V_{i+1} \setminus N[D^*] = S$ and, as H is a supergraph of $G[S \cup C]$ and S remains an independent set in H, it follows that X minimally dominates S in H. Note that either X contains a vertex of C, and then this vertex dominates C in H, or $X = S$, and then X dominates C in H as well. We conclude that X is a minimal dominating set of H, i.e., that $X \in \mathcal{D}(H)$ in this case.

We now consider the remaining case when $v_{i+1} \notin N[D^*]$; then $V_{i+1} \setminus N[D^*] = S \cup \{v_{i+1}\}$. Observe that now either X is a minimal dominating set of S, or there exists $u \in N(v_{i+1}) \cap X$ such that $X \setminus \{u\}$ is a dominating set of $S \setminus N[u]$, i.e., of $S \setminus \{u\}$ as $N(v_{i+1})$ is an independent set. Denote $D = X$ in the former case and $D = X \setminus \{u\}$ in the latter case; we now apply a reasoning similar to that from the previous paragraph. Since $v_{i+1} \notin D$ and D is a minimal dominating set of S or $S \setminus \{u\}$, it follows that $D \subseteq C \cup S$. If $D \subseteq S$, then either $D = S \setminus \{u\}$ (if u is defined and $u \in S$) or $D = S$ (otherwise), because $N(v_{i+1})$ (hence in particular S) is an independent set. If $D \not\subseteq S$, then $D \cap C \neq \emptyset$. In both cases, D dominates C in H, and we conclude that X (if u is defined and $u \in S$) or D (otherwise) is a dominating set of H. Moreover, since $\mathrm{Priv}(D, x) \cap S \neq \emptyset$ for each $x \in D$ it follows that X or D is a minimal dominating set of H. As $X = D$ or $X = D \cup \{u\}$ for some vertex u, we conclude that either X belongs to $\mathcal{D}(H)$, or it belongs to $\mathcal{D}(H)$ after removing one vertex, a vertex that was here only to dominate v_{i+1}.

Having considered both cases, the claimed property of elements of $\mathcal{C}(D^*, i)$ follows. We are left with proving the claimed upper bound on $|\mathcal{D}(H)|$. We first show that

$$|\{D \in \mathcal{D}(H) \mid D \cap S = \emptyset\}| \leq (n-1) \cdot |\{D \in \mathcal{D}(H) \mid D \cap S \neq \emptyset\}| + 1. \tag{2.3}$$

Indeed, consider the map f that, given $D \in \{D \in \mathcal{D}(H) \mid D \cap S = \emptyset\}$, $D \neq \emptyset$, removes one arbitrary vertex from D, and completes the dominating set by adding all the vertices in the independent set which are no longer dominated. Then, f maps non-empty elements of $\{D \in \mathcal{D}(H) \mid D \cap S = \emptyset\}$ to the set $\{D \in \mathcal{D}(H) \mid D \cap S \neq \emptyset\}$. Moreover, every element in this second set is the image of at most $|C| \leq n-1$ elements by f. This implies the desired bound.

From Inequality (2.3) we immediately obtain that

$$|\mathcal{D}(H)| \leq n \cdot |\{D \in \mathcal{D}(H) \mid D \cap S \neq \emptyset\}| + 1,$$

so it suffices to prove that

$$|\{D \in \mathcal{D}(H) \mid D \cap S \neq \emptyset\}| \leq |\mathcal{C}(D^*, i)|.$$

To see this, we observe that in fact we have $\{D \in \mathcal{D}(H) \mid D \cap S \neq \emptyset\} \subseteq \mathcal{C}(D^*, i)$. Indeed, by Proposition 2.2.6 we have that every $D \in \mathcal{D}(H)$ is a minimal dominating set of S in G, and it moreover dominates v_{i+1} provided $D \cap S \neq \emptyset$. □

We now show how to efficiently enumerate the candidate extensions.

Lemma 2.2.8. *There is an algorithm enumerating $\mathcal{C}(D^*, i)$ in total time $O(n^4 \cdot |\mathcal{D}(G, A)|)$ and $O(n^2)$ space whenever $N(v_{i+1})$ is an independent set.*

Proof. First, observe that given any set B of vertices, we can test in $O(n^2)$ time and space whether $B \in \mathcal{C}(D^*, i)$. Hence, it suffices to enumerate in time $O(n^4 \cdot |\mathcal{D}(G, A)|)$ and $O(n^2)$ space a superset $\mathcal{F}$ of $\mathcal{C}(D^*, i)$ of size $O(n^2 \cdot |\mathcal{D}(G, A)|)$, and for each element of $\mathcal{F}$ to test whether it belongs to $\mathcal{C}(D^*, i)$. By Lemma 2.2.7 we can use

$$\mathcal{F} = \{v_{i+1}\} \cup \mathcal{D}(H) \cup \{D \cup \{u\} \mid D \in \mathcal{D}(H),\ u \in V(G)\},$$

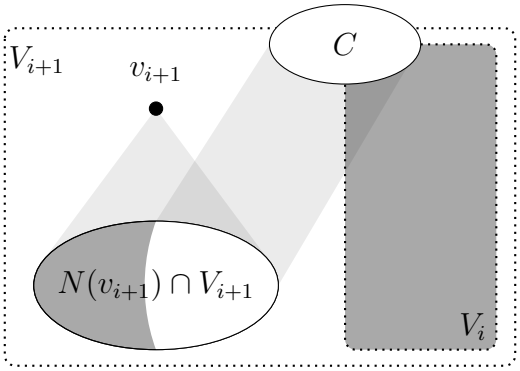


Figure 2.5: The situation of Lemma 2.2.7. The set $N[D^*] \cap V_{i+1}$ is depicted in gray (except v_{i+1}, in the first case of the proof), and the set $S = V_{i+1} \setminus (N[D^*] \cup \{v_{i+1}\}) \subseteq N(v_{i+1}) \cap V_{i+1}$ is in white. Note that $C = N(S) \setminus \{v_{i+1}\}$ may intersect $V(G) \setminus V_{i+1}$.

where H is the split graph defined in the statement of Lemma 2.2.7. Observe that

$$|\mathcal{F}| \leq (n+1) \cdot |\mathcal{D}(H)| + 1$$

and, using Proposition 2.2.6, we can enumerate $\mathcal{F}$ in total time $O(n^3 \cdot |\mathcal{D}(H)|)$ and space $O(n^2)$. It now remains to observe that by Lemma 2.2.7 and Corollary 2.2.4 we have

$$|\mathcal{D}(H)| \leq n \cdot |\mathcal{C}(D^*, i)| + 1 \leq n \cdot |\mathcal{D}(G, A)| + 1,$$

so the claimed time complexity follows. □

We conclude with the following theorem which is a consequence of Theorem 2.2.5, Lemma 2.2.8, and of the fact that when $G[A]$ is triangle-free, the neighborhood of any vertex is an independent set.

Theorem 2.2.9. *There is an algorithm that, given a bicolored graph $G(A)$ on n vertices such that $G[A]$ is triangle-free, enumerates the set $\mathcal{D}(G, A)$ in time*

$$O(\text{poly}(n) \cdot |\mathcal{D}(G, A)|^2)$$

and $O(n^3)$ space.

When $A = V(G)$, we have $\mathcal{D}(G) = \mathcal{D}(G, A)$. Hence, Theorem 2.2.9 implies the existence of an algorithm enumerating the minimal dominating sets in triangle-free graphs in output-polynomial time and polynomial space.

2.2.3 Minimal dominating sets in K_t-free graphs

In this section, we generalize the characterization of Lemma 2.2.7 and show how to use it to enumerate minimal dominating sets in K_t-free graphs, at the cost of an increased complexity (see Theorem 2.2.13).

We start with a general lemma that, roughly, implies that any output-polynomial time algorithm that may repeat outputs can be turned into one without repetition, without increasing space.

Lemma 2.2.10. *Let $\Sigma_{\text{in}}, \Sigma_{\text{out}}$ be two sets and $R \subseteq \Sigma_{\text{in}} \times \Sigma_{\text{out}}$ be a relation. Let $f, s \colon \Sigma_{\text{in}} \to \mathbb{N}$ be two functions. Suppose that there is a deterministic algorithm enumerating, given any $x \in \Sigma_{\text{in}}$, the set $\{y \in \Sigma_{\text{out}} \mid xRy\}$ in time at most $f(x)$ and space at most $s(x)$, possibly with repetition. Then there is an algorithm that, on the same input, returns the same output without repetition, in time $O(f(x)^2)$ and space $O(s(n))$.*

Proof. Let `B'` be the algorithm that on input $x \in \Sigma_{\text{in}}$ outputs $\{y \in \Sigma_{\text{out}} \mid xRy\}$, possibly with repetition, in time at most $f(x)$ and space at most $s(x)$. Elements $y \in \Sigma_{\text{out}}$ satisfying xRy will be called *solutions*. We now give an algorithm `B` that, on the same input x, outputs all solutions without repetition. Algorithm `B` simulates `B'` while counting its number of output calls. Every time `B'` outputs a solution y, `B` runs a new simulation of `B'` to verify whether y was not output by `B'` before. This new simulation is terminated at the first attempt of outputting y, and for the verification, `B` simply checks the output solution counts in both simulations against each other. If y is indeed output by `B'` for the first time, then `B` also outputs y, and otherwise `B'` ignores this output and proceeds with the simulation. Thus, `B` outputs every solution exactly once: at the first moment when `B'` outputs it. The time complexity of `B` is $O(f(x)^2)$, because for every step of `B'` we run a second simulation of `B'` that takes time at most $f(x)$. The space complexity of `B` is at most $2 \cdot s(x) + O(1) = O(s(x))$, because we need to store the internal data of two simulations of `B'` at any time. □

By combining Lemma 2.2.10 and Theorem 2.2.5, we get the following corollary.

Corollary 2.2.11. *Let $f \colon \mathbb{N}^2 \to N$ and $s \colon \mathbb{N} \to \mathbb{N}$ be two functions. Suppose that there is an algorithm that, given a bicolored graph $G(A)$ on n vertices, a peeling $(V_0, \ldots, V_p)$ of $G(A)$, $i \in \{0, \ldots, p-1\}$ and $D^* \in \mathcal{D}(G, V_i)$, enumerates the set $\mathcal{C}(D^*, i)$ in time at most $f(n, |\mathcal{D}(G, A)|)$ and space at most $s(n)$, possibly with repetition. Then there is an algorithm that, given a bicolored graph $G(A)$ on n vertices, enumerates the set $\mathcal{D}(G, A)$ in time*

$$O(n^4 d^2 + f(n, d)^2 \cdot nd)$$

and space $O(n \cdot s(n))$, where $d = |\mathcal{D}(G, A)|$.

The aforementioned generalization of Lemma 2.2.7 is the following.

Lemma 2.2.12. *Let $G(A)$ be a bicolored graph and $(V_0, \ldots, V_p)$ be a fixed peeling of $G(A)$ with vertex sequence $(v_1, \ldots, v_p)$. Let $i \in \{0, \ldots, p-1\}$, $D^* \in \mathcal{D}(G, V_i)$ and $S = V_{i+1} \setminus (N[D^*] \cup \{v_{i+1}\})$. Then:*

- *if $v_{i+1} \in N[D^*]$ then $\mathcal{C}(D^*, i) = \mathcal{D}(G, S)$;*
- *otherwise, every element of $\mathcal{C}(D^*, i)$ is of the form $Q \cup \{w\}$ for some $w \in N[v_{i+1}]$ and $Q \in \mathcal{D}(G, S \setminus N[w])$. Furthermore, in this case $|\mathcal{D}(G, S \setminus N[w])| \leq |\mathcal{C}(D^*, i)|$ for each $w \in N[v_{i+1}]$.*

Proof. By definition, $\mathcal{C}(D^*, i) = \mathcal{D}(G, V_{i+1} \setminus N[D^*])$. Note that if $v_{i+1} \in N[D^*]$ then $V_{i+1} \setminus N[D^*] = S$, so we immediately get $\mathcal{C}(D^*, i) = \mathcal{D}(G, S)$. This resolves the first case.

Suppose then that $v_{i+1} \notin N[D^*]$ and consider any $X \in \mathcal{C}(D^*, i)$. Since X minimally dominates $V_{i+1} \setminus N[D^*] = S \cup \{v_{i+1}\}$, there exists some $w \in N[v_{i+1}] \cap X$. Then $X \setminus \{w\}$

dominates $S \setminus N[w]$ and for every element of $X \setminus \{w\}$, its private neighbor in $S \cup \{v_{i+1}\}$ has to actually belong to $S \setminus N[w]$. We conclude that $X \setminus \{w\} \in \mathcal{D}(G, S \setminus N[w])$, proving the characterization of the elements of $\mathcal{C}(D^*, i)$ in this case.

We are left with proving the claimed upper bound on $|\mathcal{D}(G, S \setminus N[w])|$, for each $w \in N[v_{i+1}]$. Take any $Q \in \mathcal{D}(G, S \setminus N[w])$; clearly $w \notin Q$. If Q dominates $S \cup \{v_{i+1}\}$, then Q is also a minimal dominating set of $S \cup \{v_{i+1}\}$, because every vertex of Q has a private neighbor in $S \setminus N[w] \subseteq S \cup \{v_{i+1}\}$. Otherwise, $Q \cup \{w\}$ is a minimal dominating set of $S \cup \{v_{i+1}\}$: v_{i+1} is the private neighbor of w, and w could not steal any private neighbors in $S \setminus N[w]$ from any vertices from Q. We conclude that either Q or $Q \cup \{w\}$ belongs to $\mathcal{C}(D^*, i)$, which proves that $|\mathcal{D}(G, S \setminus N[w])| \leq |\mathcal{C}(D^*, i)|$. □

Let us point out the key difference between the statements of Lemma 2.2.12 and of Lemma 2.2.7. In Lemma 2.2.7, we reduced the enumeration of $\mathcal{C}(D^*, i)$ to the enumeration of $\mathcal{D}(H)$ for a single split graph H. In Lemma 2.2.12, to obtain larger generality we need to separately consider sets $\mathcal{D}(G, S \setminus N[w])$ for each $w \in N[v_{i+1}]$. When enumerating $\mathcal{C}(D^*, i)$ via enumerating these sets, we will unavoidably obtain repetitions of elements of $\mathcal{C}(D^*, i)$. These will be handled using Lemma 2.2.10 at the cost of an increased complexity.

Observe that by Lemma 2.2.12, to be able to enumerate the candidate extensions in general (and thus the minimal dominating sets, using Theorem 2.2.5) in output-polynomial time, it suffices to be able to enumerate the minimal dominating sets of $G(S)$ and of $G(S \setminus N[w])$ for every $w \in N[v_{i+1}]$. For bicolored graphs $G(A)$ such that $G[A]$ is K_t-free, this can be done by exploiting the fact that $G[S]$ is K_{t-1}-free and running the same algorithm on $G(S)$, as we shall describe now. We recall that $\omega(G)$ denotes the size of a largest clique in G.

Theorem 2.2.13. *There is a function $p\colon \mathbb{N} \to \mathbb{N}$ and an algorithm that, given a bicolored graph $G(A)$ on n vertices, enumerates the set $\mathcal{D}(G, A)$ in time at most*

$$p(t) \cdot n^{2^{t+1}} \cdot |\mathcal{D}(G, A)|^{2^t}$$

and space at most $p(t) \cdot n^{t+1}$, where $t = \omega(G[A]) + 1$.

When $A = V(G)$, we have $\mathcal{D}(G) = \mathcal{D}(G, A)$. Hence, Theorem 2.2.13 implies the existence of an algorithm enumerating, for every integer $t \geq 1$, the minimal dominating sets in K_t-free graphs in output-polynomial time and polynomial space. We stress that we provide a single algorithm for all values of t—note that as stated, the algorithm does not require knowledge of t.

Proof of Theorem 2.2.13. In this proof we consider two algorithms `A` and `B` that recursively call each other in order to enumerate the minimal dominating sets of a bicolored graph. We first give their specifications, then describe them, and finally prove that they perform as specified. Let $f\colon \mathbb{N}^3 \to \mathbb{N}$ be defined by $f(n, d, t) = n^{2^{t+1}-3} \cdot d^{2^t-1}$, for every $n, d, t \in \mathbb{N}$.

Specifications of `A` and `B`. We will show that the algorithms `A` and `B` have the following properties P and Q, for every $t \geq 1$ in case of $P(t)$ and every $t \geq 2$ in case of $Q(t)$.

$P(t)$: There is a constant $p(t) \in \mathbb{N}$ such that given an n-vertex graph G and a set $A \subseteq V(G)$ such that $G[A]$ is K_t-free, A outputs $\mathcal{D}(G, A)$ in time at most $p(t) \cdot f(n, |\mathcal{D}(G, A)|, t)$ and space at most $p(t) \cdot n^{t+1}$.

$Q(t)$: There is a constant $q(t) \in \mathbb{N}$ such that given a n-vertex graph G, a set $A \subseteq V(G)$ such that $G[A]$ is K_t-free, a peeling $(V_0, \ldots, V_p)$ of $G(A)$ with vertex sequence $(v_1, \ldots, v_p)$, $i \in \{0, \ldots, p-1\}$, and $D^* \in \mathcal{D}(G, V_i)$, B outputs $\mathcal{C}(D^*, i)$ in time at most $q(t) \cdot n^2 \cdot f(n, |\mathcal{D}(G, A)|, t-1)^2$ and space at most $q(t) \cdot n^t$.

The statement of Theorem 2.2.13 is implied by $P(t)$ holding for all $t \geq 1$. In order to prove it, we will also show that $Q(t)$ holds for every $t \geq 2$. Let us first describe A.

Description of A. The algorithm A is the one given by Theorem 2.2.5 that takes as input a bicolored graph $G(A)$, using B as a routine to enumerate candidate extensions. We will show below that B indeed does so.

Description of B. Recall that B takes as input a bicolored graph $G(A)$, a peeling $(V_0, \ldots, V_p)$ of $G(A)$ with vertex sequence $(v_1, \ldots, v_p)$, an integer $i \in \{0, \ldots, p-1\}$, and a set $D^* \in \mathcal{D}(G, V_i)$.

We first describe an auxiliary routine B'. Let $S = V_{i+1} \setminus (N[D^*] \cup \{v_{i+1}\})$. Then, Lemma 2.2.12 above allows us to consider two cases depending on whether D^* dominates v_{i+1} or not:

(i) if $v_{i+1} \in N[D^*]$, we call algorithm A on $G(S)$ to enumerate $\mathcal{D}(G, S)$ and we give the same output;

(ii) otherwise, we iterate over all $w \in N[v_{i+1}]$ and $Q \in \mathcal{D}(G, S \setminus N[w])$ (where the latter is obtained via a call to A) and output $Q \cup \{w\}$ if and only if it is a candidate extension of D^*.

We are now done with B'. As we will show later, B' enumerates $\mathcal{C}(D^*, i)$, however each element may be repeated, up to n times. Then B is obtained from B' using Lemma 2.2.10. This concludes the description of B.

Correctness of A and B. Now that we described the algorithms A and B, we show that they conform to their specifications, i.e., we prove that $P(t)$ holds for every $t \geq 1$ and that $Q(t)$ holds for every $t \geq 2$. The proof by induction on t is split in lemmas.

Lemma 2.2.14. *$P(1)$ holds.*

Proof. The statement $P(1)$ deals with pairs (G, A) such that $G[A]$ is K_1-free, so $A = \emptyset$. In these cases we clearly have $\mathcal{D}(G, A) = \{\emptyset\}$. Notice that such inputs are correctly handled by algorithm A. Checking whether A is empty and returning $\{\emptyset\}$ takes $O(n)$ time and $O(n^2)$ space. We define $p(1)$ as an integer such that these steps take at most $p(1) \cdot n$ time and at most $p(1) \cdot n^2$ space on an input graph of order n. As $f(n, |\mathcal{D}(G, A)|, t) = n$ in this case, $P(1)$ holds. □

Lemma 2.2.15. *For every integer $t \geq 1$, $P(t) \Rightarrow Q(t+1)$.*

Proof. Let $t \geq 1$ and let us assume that the statement $P(t)$ holds (in particular, $p(t)$ is defined). Let $\mathcal{I} = (G, A, V_0, \ldots, V_p, v_1, \ldots, v_p, i, D^*)$ be an input of $\mathtt{B}$ such that $G[A]$ is K_{t+1}-free. Let us define $n = |G|$ and $d = |\mathcal{D}(G, A)|$. We review the description of $\mathtt{B}$ to show that $Q(t+1)$ holds. We first consider the auxiliary routine $\mathtt{B'}$.

Claim 1. *Given $\mathcal{I}$, the algorithm $\mathtt{B'}$ enumerates $\mathcal{C}(D^*, i)$ with each output repeated up to n times, in time at most $k \cdot n \cdot f(n, d, t)$ and space at most $k \cdot n^{t+1}$, for some constant k.*

Proof. Let $S = V_{i+1} \setminus (\{v_{i+1}\} \cup N[D^*])$. Note that as D^* dominates V_i, we have $S \subseteq N(v_{i+1}) \cap V_{i+1}$. Also, S can be computed in $O(n^2)$ time and space.

Since $i < p$, we have $V_{i+1} \subseteq A$, from the definition of a peeling. In particular, $G[V_{i+1}]$ is K_{t+1}-free. As $S \subseteq N(v_{i+1}) \cap V_{i+1}$, we get that $G[S]$ is K_t-free. Hence, when applying the algorithm recursively to enumerate $\mathcal{D}(G, S')$ for any $S' \subseteq S$, we may use the already established property $P(t)$, yielding the following:

Remark 2.2.16. For any $S' \subseteq S$, a call to $\mathtt{A}$ on (G, S') returns $\mathcal{D}(G, S')$ in time at most $p(t) \cdot f(n, |\mathcal{D}(G, S')|, t)$ and space at most $p(t) \cdot n^{t+1}$.

If $v_{i+1} \in N[D^*]$, then, by Lemma 2.2.12, we enumerate $\mathcal{C}(D^*, i)$ without repetitions simply by enumerating $\mathcal{D}(G, S)$. By Remark 2.2.16, this takes time

$$\begin{aligned} & p(t) \cdot f(n, |\mathcal{D}(G, S)|, t) & \\ = \; & p(t) \cdot f(n, |\mathcal{C}(D^*, i)|, t) & \text{(by Lemma 2.2.12)} \\ \leq \; & p(t) \cdot f(n, d, t) & \text{(by Corollary 2.2.4)} \end{aligned}$$

and space at most $p(t) \cdot n^{t+1}$.

Now suppose that $v_{i+1} \notin N[D^*]$. Then, by Lemma 2.2.12, we enumerate all elements of the set $\mathcal{C}(D^*, i)$, however each of them is enumerated once per every form $Q \cup \{w\}$ it can take, where $w \in N[v_{i+1}]$ and $Q \in \mathcal{D}(G, S \setminus N[w])$. Every such occurrence is characterized by the choice of w, hence there are at most n of them and, consequently, every member of $\mathcal{C}(D^*, i)$ is enumerated at most n times.

Regarding time and space complexity we perform at most n times (once for every choice of $w \in N[v_{i+1}]$) the following operations:

- constructing $S \setminus N[w]$, in $O(n)$ time and space;
- calling $\mathtt{A}$ on $(G, S \setminus N[w])$, in time at most $p(t) \cdot f(n, |\mathcal{D}(G, S \setminus N[w])|, t)$ and space at most $p(t) \cdot n^{t+1}$, by Remark 2.2.16;
- checking, for each set Q among the outputs of $\mathtt{A}$, whether $Q \cup \{w\}$ belongs to $\mathcal{C}(D^*, i)$, in $O(n^2)$ time and space. There are at most d outputs.

By Lemma 2.2.12 and Corollary 2.2.4, we have

$$|\mathcal{D}(G, S \setminus N[w])| \leq |\mathcal{C}(D^*, i)| \leq d.$$

Therefore, in total the time complexity of these steps adds up to:

$$\begin{aligned} & n \cdot \left[O(n) + p(t) \cdot f(n, d, t) + O\left(n^2 \cdot d\right)\right] & \\ = \; & O\left(p(t) \cdot n \cdot f(n, d, t)\right) & \text{(as } t \geq 2 \text{ in this case).} \end{aligned} \tag{2.4}$$

Similarly, the space complexity can be upper-bounded by $O(p(t) \cdot n^{t+1})$. We conclude by setting k as $p(t)$ times the maximum of the constants hidden in the $O(\cdot)$ notation above. ⌟

As proved in Lemma 2.2.10, the algorithm of Claim 1 can be turned into an algorithm B that does not repeat outputs. That is, there is a constant $q(t+1)$ (depending on k) such that given $\mathcal{I}$, B runs in time at most $q(t+1) \cdot n^2 \cdot f(n,d,t)^2$ and space at most $q(t+1) \cdot n^{t+1}$. Hence $Q(t+1)$ holds, as desired. □

Lemma 2.2.17. *For every integer $t \geq 2$, $Q(t) \Rightarrow P(t)$.*

Proof. Let us assume that for some integer $t \geq 2$, the statement $Q(t)$ holds (and in particular $q(t)$ is defined). Let G be a graph and $A \subseteq V(G)$ be such that $G[A]$ is K_t-free. We set $n = |G|$ and $d = |\mathcal{D}(G,A)|$. By $Q(t)$, the enumeration of candidate extensions in $G(A)$ can be carried out by B in total time at most

$$q(t) \cdot n^2 \cdot f(n,d,t-1)^2$$

and space at most $q(t) \cdot n^t$. According to Theorem 2.2.5, A then enumerates $\mathcal{D}(G,A)$ in time

$$\begin{aligned} & O(n^4 \cdot d^2 + q(t) \cdot n^3 \cdot f(n,d,t-1)^2 \cdot d) \\ = \; & O(n^4 \cdot d^2 + q(t) \cdot f(n,d,t)) && \text{(by the definition of } f\text{)} \\ = \; & O(q(t) \cdot f(n,d,t)) && \text{(as } t \geq 2\text{)} \end{aligned}$$

and space $O(q(t) \cdot n^{t+1})$. Therefore, there is a constant $p(t)$ (depending on $q(t)$) such that A runs on this input in time at most $p(t) \cdot f(n,d,t)$ and space at most $p(t) \cdot n^{t+1}$. This proves $P(t)$. □

Concluding the proof. We proceed by induction on t. The base case $P(1)$ follows from Lemma 2.2.14. The induction step that, for every integer $t \geq 1$, $P(t)$ implies $P(t+1)$, is obtained by combining Lemmas 2.2.15 and 2.2.17. We conclude that $P(t)$ holds for every integer $t \geq 1$. That is, the algorithm A has the properties claimed in the statement of the theorem. □

We note that the complexity of the algorithm of Theorem 2.2.13 for K_t-free graphs could be slightly improved when $t \geq 3$, using Theorem 2.2.9 as a base case, however that would not remove the exponential contribution of t to the degree of the polynomial.

2.2.4 Variants of K_t-free graphs

We give output-polynomial time algorithms for classes related to K_t-free graphs relying on the algorithms and characterizations of candidate extensions given in Sections 2.2.1, 2.2.2, and 2.2.3.

Forbidding $K_t + K_2$

In this section we show how the algorithm of Theorem 2.2.13 on K_t-free graphs can be extended to the setting of $(K_t + K_2)$-free graphs.

Theorem 2.2.18. *There is an algorithm that, for every fixed $t \in \mathbb{N}$, enumerates minimal dominating sets in $(K_t + K_2)$-free graphs in output-polynomial time and polynomial space.*

Proof. Let $t \in \mathbb{N}$ and let G be a $(K_t + K_2)$-free graph. It is well-known that the minimal dominating sets of G that induce edgeless subgraphs are exactly the maximal independent sets of G. We can therefore enumerate these using the polynomial delay algorithms of Tsukiyama et al. [TIAS77] for maximal independent sets. In the sequel we may thus focus on those minimal dominating sets of G that induce at least one edge.

We show how to enumerate, for every edge uv of G, the minimal dominating sets of G that contain both u and v. Let $A_{uv} = V(G) \setminus N[\{u, v\}]$ and observe that $G[A_{uv}]$ is K_t-free. First, we enumerate $G(A_{uv})$ using the algorithm of Theorem 2.2.13, which runs in output-polynomial time and polynomial space, as t is fixed. For every $D \in \mathcal{D}(G, A_{uv})$ obtained from the aforementioned call, we output $D \cup \{u, v\}$ if it is a minimal dominating set of G, and discard D otherwise. By Lemma 2.2.3 (applied for $H = G$ and $B = V(G)$) we have $|\mathcal{D}(G, A_{uv})| \leq |\mathcal{D}(G)|$. Hence, enumerating $\mathcal{D}(G, A_{uv})$ produces all those minimal dominating sets of G that at least induce the edge uv in time $\mathrm{poly}(n \cdot |\mathcal{D}(G)|)$ and space $\mathrm{poly}(n)$, where the degrees of these polynomials depend on t (see Theorem 2.2.13).

Now that we know how to enumerate minimal dominating sets that induce at least one particular edge, we can run the above routine for every edge of G to enumerate all minimal dominating sets of G, possibly with repetitions. Observe that the same output can be repeated at most $|E(G)|$ times. Then, repetitions are avoided using Lemma 2.2.10 with Σ_{in} being the set of all graphs, Σ_{out} the set of all vertex sets, and R the relation that associates every graph to its minimal dominating sets. □

Forbidding $K_t - e$

Another interesting case is the one of $(K_t - e)$-free graphs. In this section we show how the characterization of Lemma 2.2.12 can be used to enumerate candidate extensions in diamond-free graphs (which are $(K_t - e)$-free for $t = 4$), which by Theorem 2.2.5 gives an output-polynomial time algorithm enumerating minimal dominating sets in this class. We leave open the existence of such an algorithm in the case when $t \geq 5$.

In what follows, we consider a bicolored graph $G(A)$ on n vertices such that G is diamond-free, together with a fixed peeling $(V_0, \ldots, V_p)$ of $G(A)$ with vertex sequence $(v_1, \ldots, v_p)$. Then, we consider

$$i \in \{0, \ldots, p-1\}, \qquad D^* \in \mathcal{D}(G, V_i),$$

and define $S = V_{i+1} \setminus (N[D^*] \cup \{v_{i+1}\})$ and $\mathcal{C}(D^*, i)$ as in Sections 2.2.1, 2.2.2, and 2.2.3. Note that contrarily to the triangle-free case and the K_t-free case, we here require the whole graph G to be diamond-free and not only $G[A]$. We start with an easy observation.

Observation 2.2.19. *For every vertex u of G, $G[N(u)]$ is P_3-free. Then $G[N(v_{i+1})]$, hence also $G[S]$, can be partitioned into a disjoint union of cliques (i.e., it is a cluster graph).*

We will show how to minimally dominate one clique of S, then a disjoint union of cliques of S, and will conclude with the enumeration of $\mathcal{C}(D^*, i)$.

Lemma 2.2.20. *Let K be a clique of $G[S]$ and u be a vertex in $G - S$, $u \neq v_{i+1}$, that is adjacent to some vertex of K. If u is adjacent to v_{i+1}, then it is complete to K. Otherwise u has exactly one neighbor in K.*

Proof. If $u \in N(v_{i+1})$ then, as $G[N(v_{i+1})]$ is P_3-free and $K \subseteq N(v_{i+1})$, u is complete to K. If u is not adjacent to v_{i+1}, then it has exactly one neighbor in K, as otherwise $\{a, b, u, v_{i+1}\}$ would induce a diamond in G, for any two neighbors $a, b \in K$ of u. □

Lemma 2.2.21. *Let K be a clique in $G[S]$. Then $\mathcal{D}(G, K)$ can be enumerated in total time $O(n^2 + n \cdot |\mathcal{D}(G, K)|)$ and $O(n^2)$ space.*

Proof. We describe an algorithm enumerating $\mathcal{D}(G, K)$ in the specified time and space bounds. We first output $\{v_{i+1}\}$ as it is complete to K. We then output all vertices $u \in N(v_{i+1})$ such that $u \in K$ or u is adjacent to some vertex of K. By Lemma 2.2.20, these vertices are also complete to K. Then, for every $x \in K$, we compute the neighborhood of x outside of $N(v_{i+1})$ in total time $O(n^2)$. By Lemma 2.2.20, these neighborhoods are disjoint. At last, we enumerate the unordered Cartesian products of these neighborhoods. This can clearly be done in total time of $O(n \cdot |\mathcal{D}(G, K)|)$ using $O(n)$ space as they are disjoint. Clearly, every element in such an unordered Cartesian product is a minimal dominating set of K, and the described algorithm performs within the specified time and space bounds. The correctness of the algorithm follows from Lemma 2.2.20. □

Lemma 2.2.22. *Let W be a subset of S. Then $\mathcal{D}(G, W)$ can be enumerated in total time $O(n^7 \cdot |\mathcal{D}(G, A)|^3)$ and $O(n^3)$ space.*

Proof. We use the ordered generation described in Section 2.2.1. The algorithm first computes a peeling $(U_1, \ldots, U_q)$ of $G(W)$ with vertex sequence $(u_1, \ldots, u_q)$, in $O(n^2)$ time and space. Note that $N[u_1] \cap W, \ldots, N[u_q] \cap W$ is exactly the disjoint clique partition of $G[W]$; denote these sets by $W_1, \ldots, W_q$. Given $j \in \{0, \ldots, q-1\}$ and $D^\circ \in \mathcal{D}(G, U_j)$, we define $\mathcal{C}'(D^\circ, j)$ as the set of candidate extensions of (D°, j) with respect to the chosen peeling of $G(W)$ and we show how to enumerate $\mathcal{C}'(D^\circ, j)$ in time $O(n^6 \cdot |\mathcal{D}(G, A)|^2)$ and using $O(n^2)$ space.

We rely on the same characterization of candidate extensions that we use in the proof of Theorem 2.2.13, i.e., Lemma 2.2.12. Recall that this lemma allows us to consider two cases depending on whether D° dominates u_{j+1} or not. Let $Y = W_{j+1} \setminus N[D^\circ]$.

If D° dominates u_{j+1}, then we can enumerate $\mathcal{C}'(D^\circ, j)$ by just enumerating $\mathcal{D}(G, Y)$ as these sets coincide. As Y is a clique in $G[S]$, by Lemma 2.2.21 we can enumerate $\mathcal{D}(G, Y)$ in time $O(n^2 + n \cdot |\mathcal{D}(G, Y)|)$ and space $O(n^2)$. By Lemma 2.2.3 we have $|\mathcal{D}(G, Y)| \leq |\mathcal{D}(G, A)|$, hence the procedure runs within the required time and space complexity.

In the remaining case when D° does not dominate u_{j+1}, we iterate over all $w \in N[u_{j+1}]$ and $Q \in \mathcal{D}(G, Y \setminus N[w])$ (obtained via a call to the algorithm of Lemma 2.2.21)

and output $Q \cup \{w\}$ if and only if this set belongs to $\mathcal{C}'(D^\circ, j)$, which can be checked in $O(n^2)$ time and space. Again, by Lemma 2.2.3 we have $|\mathcal{D}(G, Y \setminus N[w])| \leq |\mathcal{D}(G, A)|$ for all w as above. Hence, in total, the described algorithm enumerates $\mathcal{C}'(D^\circ, i)$, possibly with repetitions, in time $O(n^3 \cdot |\mathcal{D}(G, A)|)$ and using $O(n^2)$ space. Using Corollary 2.2.11, we obtain an algorithm enumerating $\mathcal{D}(G, W)$ in time $O(n^7 \cdot |\mathcal{D}(G, A)|^3)$, and using $O(n^3)$ space. □

Lemma 2.2.23. *There is an algorithm enumerating $\mathcal{C}(D^*, i)$, possibly with repetitions, in total time $O(n^8 \cdot |\mathcal{D}(G, A)|^3)$ and using $O(n^3)$ space.*

Proof. We apply the same argument as in the previous lemma, and in the proof of Theorem 2.2.13. Lemma 2.2.12 allows us to consider two cases depending on whether D^* dominates v_{i+1} or not. If D^* dominates v_{i+1}, we call the algorithm of Lemma 2.2.22 to enumerate $\mathcal{D}(G, S)$ without repetitions in total time $O(n^7 \cdot |\mathcal{D}(G, A)|^3)$ and $O(n^3)$ space. If D^* does not dominate v_{i+1}, we iterate over all $w \in N[v_{i+1}]$ and $Q \in \mathcal{D}(G, S \setminus N[w])$ (obtained via a call to the algorithm of Lemma 2.2.22 as $S \setminus N[w] \subseteq S$) and output $Q \cup \{w\}$ if and only if this set belongs to $\mathcal{C}(D^*, i)$. An analogous complexity analysis shows that this algorithm runs in time $O(n^8 \cdot |\mathcal{D}(G, A)|^3)$ and uses $O(n^3)$ space, and it enumerates $\mathcal{C}(D^*, i)$ possibly with repetitions. □

As a consequence of Corollary 2.2.11 and Lemma 2.2.23, we get the following.

Theorem 2.2.24. *There is an algorithm that, given a bicolored graph $G(A)$ on n vertices such that G is diamond-free, enumerates the set $\mathcal{D}(G, A)$ in time*

$$O(\mathrm{poly}(n) \cdot |\mathcal{D}(G, A)|^7)$$

and $O(n^4)$ space.

Note that when $A = V(G)$, we have $\mathcal{D}(G) = \mathcal{D}(G, A)$. Hence, Theorem 2.2.24 implies the existence of an algorithm enumerating the minimal dominating sets in diamond-free graphs in output-polynomial time and using polynomial space.

Paw-free graphs

We now consider the exclusion of a specific graph, the paw, and show that DOM-ENUM admits an output-polynomial time algorithm in paw-free graphs.

In what follows, we consider a bicolored graph $G(A)$ on n vertices such that G is paw-free, together with a fixed peeling $(V_0, \ldots, V_p)$ of $G(A)$ with vertex sequence $(v_1, \ldots, v_p)$. Then, we consider

$$i \in \{0, \ldots, p-1\}, \qquad D^* \in \mathcal{D}(G, V_i),$$

and define $S = V_{i+1} \setminus (N[D^*] \cup \{v_{i+1}\})$ and $\mathcal{C}(D^*, i)$ as in Sections 2.2.1, 2.2.2 and 2.2.3. As in the previous section we stress that we require the whole graph G to be paw-free, and not only $G[A]$. We start with an easy observation.

Observation 2.2.25. *For every vertex u of G, $G[N(u)]$ is $\overline{P_3}$-free. Hence $G[S]$ is a complete multipartite graph (also called a co-cluster graph).*

Note that when enumerating $\mathcal{C}(D^*, i)$ (i.e., the minimal dominating sets of $G(S)$), we may safely ignore the edges between two vertices of $G - S$. Therefore, if S is an independent set, we can delete all edges of $G - S$ (in $O(n^2)$ time) to obtain that $N(v_{i+1})$ is an independent set and then apply the algorithm enumerating $\mathcal{C}(D^*, i)$ in this setting given by Lemma 2.2.8. In the next lemma, we consider the case where S contains at least one edge.

Lemma 2.2.26. *Assume that $G[S]$ contains at least one edge, and let u be a vertex of G, $u \neq v_{i+1}$, that has a neighbor in S. If u is not adjacent to v_{i+1}, then it is complete to S. Otherwise, u is complete to $S \setminus I_j$, for some $j \in \{1, \ldots, q\}$ where $I_1, \ldots, I_q$ is the complete multipartition of $G[S]$ (every I_j induces an independent set in G, while vertices from different I_j's are adjacent).*

Proof. As by assumption $G[S]$ contains an edge, we have $q \geq 2$. Assume first that u is not adjacent to v_{i+1}, but has a neighbor in S; in particular $u \notin S$. Suppose for contradiction that u is not complete to S. Hence there are vertices $x \in S \cap N(u)$ and $y \in S \setminus N(u)$. Note that $xy \notin E(G)$ as otherwise $\{u, v_{i+1}, x, y\}$ induces a paw in G. Then $x, y \in I_j$ for some $j \in \{1, \ldots, q\}$. Let $z \in S \setminus I_j$; such a vertex exists as $q \geq 2$ and it is complete to $\{v_{i+1}, x, y\}$ by definition of the I_k's. Then, either $uz \in E(G)$ and $\{u, x, y, z\}$ induces a paw, or $uz \notin E(G)$ and $\{u, v_{i+1}, y, z\}$ does, a contradiction.

Assume now that u is adjacent to v_{i+1}. If u belongs to S, then it belongs to some I_j, $j \in \{1, \ldots, q\}$ and is complete to $S \setminus I_j$, by definition of the I_k's. We now assume $u \in N(v_{i+1}) \setminus S$. If there is no $j \in \{1, \ldots, q\}$ such that u is complete to $S \setminus I_j$, then, as $q \geq 2$, u has at least two non-neighbors $x \in I_{j'}$ and $y \in I_{j''}$ for two different $j', j'' \in \{1, \ldots, q\}$. Then $\{u, v_{i+1}, x, y\}$ induces a paw in G, a contradiction. □

Lemma 2.2.27. *There is an algorithm enumerating $\mathcal{C}(D^*, i)$ in total time $O(n^5 \cdot |\mathcal{D}(G, A)|)$ and $O(n^2)$ space.*

Proof. In the case where S induces an independent set, we use the algorithm of Lemma 2.2.8 to enumerate $\mathcal{C}(D^*, i)$ in time

$$O(n^4 \cdot |\mathcal{D}(G, A)|)$$

and $O(n^2)$ space. Otherwise, we deduce from Lemma 2.2.26 that minimal dominating sets of S are either of size at most two, or of the form I_j for some $j \in \{1, \ldots, q\}$. If $v_{i+1} \in N[D^*]$, that is if $S = V_{i+1} \setminus N[D^*]$, we try each of these sets and output those that minimally dominate S; this can be done in total time $O(n^4)$. This enumerates $\mathcal{C}(D^*, i)$ by definition. If $v_{i+1} \notin N[D^*]$, we first output I_j for every $j \in \{1, \ldots, q\}$. Then, we iterate over all vertex subsets of size at most three and output those that minimally dominate S; this can be done in total time $O(n^5)$. This will enumerate $\mathcal{C}(D^*, i)$, for the following reason implied by Lemma 2.2.26. If $X \in \mathcal{C}(D^*, i)$, then either $X = I_j$ for some $j \in \{1, \ldots, j\}$, or X contains at most three vertices: one with v_{i+1} as a private neighbor and at most 2 with private neighbors in S. □

As a consequence of Theorem 2.2.5 and Lemma 2.2.27, we get the following.

Theorem 2.2.28. *There is an algorithm that, given a bicolored graph $G(A)$ on n vertices such that G is paw-free, enumerates the set $\mathcal{D}(G, A)$ in time*

$$O(\mathrm{poly}(n) \cdot |\mathcal{D}(G, A)|^2)$$

and $O(n^3)$ space.

Note that when $A = V(G)$, we have $\mathcal{D}(G) = \mathcal{D}(G, A)$. Hence, Theorem 2.2.28 implies the existence of an algorithm enumerating the minimal dominating sets in paw-free graphs in output-polynomial time and using polynomial space.

2.2.5 Technique limitations

In this section, we discuss various obstacles that we detected in our attempts to improve our results or proofs.

A standard technique fails for bipartite graphs

A natural technique (referred to as *flashlight search*, investigated in Section 2.4) to enumerate valid solutions to a given problem such as, for instance, sets of vertices satisfying a given property, is to build them element by element. If during the construction one detects that the current partial solution cannot be extended into a valid one, then it can be discarded along with all the other partial solutions that contain it. Note that in order to apply this technique, one should be able to decide whether a given partial solution can be completed into a valid one. It turns out that for minimal dominating sets, this problem (called the *extension problem* problem) is NP-complete [KLMN11], even when restricted to split graphs [KLM+15a]. We show that it remains NP-complete in bipartite graphs, so in particular on $(K_t + K_2)$-free graphs for every $t \geq 3$. This stands in contrast with Theorem 2.2.13 and suggests that, indeed, a more involved technique was needed to obtain our results.

The extension problem, denoted DCS, is formally defined as follows. Given a graph G and a set $A \subseteq V(G)$ of vertices, is there a minimal dominating set $D \in \mathcal{D}(G)$ such that $A \subseteq D$.

Theorem 2.2.29. DCS *restricted to bipartite graphs is* NP*-complete.*

Proof. Since DCS is NP-complete in the general case, it is clear that DCS is in NP even when restricted to bipartite graphs. Let us now present a hardness reduction from SAT.

Given an instance φ of SAT with variables $x_1, \ldots, x_n$ and clauses $C_1, \ldots, C_m$, we construct a bipartite graph G and a set $A \subseteq V(G)$ such that there exists a minimal dominating set containing A if and only if there exists a truth assignment to the variables of φ that satisfies all the clauses. The graph G has vertex bipartition (X, Y), defined as follows.

The first part X contains two special vertices u and w, and for every variable x_i, one vertex for each of the literals x_i and $\neg x_i$. The second part Y contains one vertex y_{C_j} per clause C_j, one vertex neg_{x_i} per variable x_i, and two special vertices v and z. For every $i \in \{1, \ldots, n\}$ we make neg_{x_i} adjacent to the two literals x_i and $\neg x_i$ and for

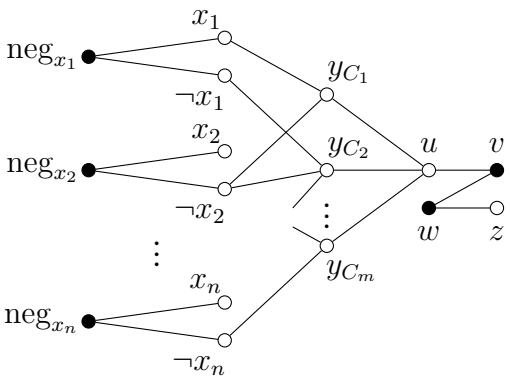


Figure 2.6: A bipartite graph G and a set $A \subseteq V(G)$ constructed from an instance of SAT with variables $x_1, \ldots, x_n$ and clauses $C_1, \ldots, C_m$. Black vertices constitute the set A. Then A can be extended into a minimal dominating set D of G if and only if there is a truth assignment of the variable satisfying all the clauses.

every $j \in \{1, \ldots, m\}$ we make y_{C_j} adjacent to u and to every literal C_j contains. Finally, we add edges to form the path $uvwz$ and set $A = \{\text{neg}_{x_1}, \ldots, \text{neg}_{x_n}, v, w\}$. Clearly this graph can be constructed in polynomial time from φ. The construction is illustrated in Figure 2.6.

Let us show that A can be extended into a minimal dominating set of G if and only if φ has a truth assignment that satisfies all the clauses. The proof is split into two claims. A *partial assignment* of φ is a truth assignment of a subset of the variables $x_1, \ldots, x_n$. Observe that a partial assignment may satisfy all the clauses (i.e., the values of the non-assigned variables do not matter). A partial assignment that satisfies all the clauses is called a *minimal assignment* if none of its proper subsets satisfies all the clauses.

Claim 2. *Let $S \subseteq \{x_1, \neg x_1, \ldots, x_n, \neg x_n\}$ be a set containing at most one literal for each variable. Then S minimally dominates $\{y_{C_1}, \ldots, y_{C_m}\}$ if and only if its elements form a minimal assignment of φ.*

Proof. Suppose S is as above and S minimally dominates $\{y_{C_1}, \ldots, y_{C_m}\}$. Consider any $j \in \{1, \ldots, m\}$. Since $y_{C_j} \notin S$, the set S contains a neighbor x of y_{C_j}. By construction, x is a literal appearing in C_j. Hence, the literals present in S form a partial assignment of the variables of φ satisfying all its clauses. Moreover, this partial assignment is minimal by the minimality of S. The proof in the other direction is analogous. ⌟

Claim 3. *If D is a minimal dominating set of G containing A, then $D \setminus A \subseteq \{x_1, \neg x_1, \ldots, x_n, \neg x_n\}$ and D contains at most one literal of each variable.*

Proof. Notice that $\text{Priv}(A, v) = \{u\}$. If y_{C_j} belongs to D for some $j \in \{1, \ldots, m\}$, then $\text{Priv}(D, v) = \emptyset$, a contradiction to the minimality of D. For similar reasons $u, z \notin D$. Hence $D \cap \{u, z, y_{C_1}, \ldots, y_{C_m}\} = \emptyset$. Besides, for every $i \in \{1, \ldots, m\}$, D contains at most one of x_i and $\neg x_i$, as otherwise $\text{Priv}(D, \text{neg}_{x_i})$ would be empty, again contradicting the minimality of D. This proves the claim. ⌟

If A can be extended into a minimal dominating set D of G, then by combining the two claims above, we deduce that φ has a truth assignment that satisfies all clauses. Conversely, if φ has a truth assignment satisfying all the clauses, then it also has a minimal truth assignment satisfying all the clauses, so there is a set S as in the statement of Claim 2. In $S \cup A$, every element of S has a private neighbor, as a consequence of the minimality of S and the fact that no element of A has a neighbor among the clause vertices. Besides, each of $\mathrm{neg}_{x_1}, \ldots, \mathrm{neg}_{x_n}$ has a private neighbor (because S contains at most one of the two literals for each variable) and it is easy to see that the same holds for v and w. Hence $S \cup A$ is a minimal dominating set of G.

Given an instance φ of SAT, we constructed in polynomial time an instance (G, A) of DCS that is equivalent to satisfiability of φ. This proves that DCS is NP-hard. □

Limitations of the bicolored argument

Let us present a brief argument of why enumerating the minimal dominating sets in a bicolored graph $G(A)$ is DOM-ENUM-hard if A can contain an arbitrarily large clique and no restriction is put on the structure of $G - A$ nor its interactions with A. In other words, we argue that DOM-ENUM can be reduced to the problem of enumerating the minimal dominating sets in a bicolored graph $G(A)$ where A is a clique.

Because of Theorem 2.1.1, we know that enumerating minimal dominating sets of a co-bipartite graph G is DOM-ENUM-hard. However, note that free to disregard the minimal dominating sets consisting of exactly one vertex in each clique of the partition, every minimal dominating set is included in one of the two cliques. Let A_1 and A_2 be the two sides of this partition. Observe that as both A_1 and A_2 induce cliques, they satisfy any property that does not limit the size of the largest clique. Combined with the fact that minimal dominating sets consisting of exactly one vertex in each side of the partition are easy to enumerate, we obtain the desired conclusion.

Note however that this obstacle was circumvented in Theorems 2.2.24 and 2.2.28 by keeping track of what the forbidden structures in G imply for the interactions between $G - A$ and A. Unfortunately, the arguments were quite ad hoc in nature and it is unclear how far they can be generalized.

This obstacle was bypassed in a different way in Theorem 2.2.18, simply by first enumerating all the minimal dominating sets without a given structure, then using the fact that the structure appears in any remaining dominating set to guess where it does, and finally arguing that the vertices that remain to be dominated cannot induce an arbitrarily large clique. We now show that this technique is in fact very limited.

Limitations of enumerating all minimal dominating sets with a certain structure

We present now a brief argument on why enumerating all H-free minimal dominating sets in a graph is DOM-ENUM-hard unless H is a clique of size at most 2. Here, a minimal dominating set D is *H-free* if $G[D]$ does not contain H as an induced subgraph.

The case when H is not a clique is directly implied by the argument in Section 2.2.5. We now focus on the case when H is a clique on at least 3 vertices; it suffices to handle the case when H is a triangle. In other words, we argue that DOM-ENUM can be reduced to the question of enumerating all triangle-free minimal dominating sets.

Consider a graph G. We build an auxiliary graph G' by creating two copies A and B of $V(G)$, creating a vertex u, and setting $V(G') = A \cup B \cup \{u\}$. We set A to be an independent set, B to be a clique, and the vertex u to be adjacent to all of A and none of B. We set the edges between A and B as follows: a vertex in A and a vertex in B are adjacent if and only if the vertices of G they originate from are the same or are adjacent.

Let us consider what the structure of a minimal dominating set D of G' can be, and how easy it is to generate all minimal dominating sets of a given type. We consider three cases.

1. $u \notin D$. We generate all minimal dominating sets of the split graph $G'[A \cup B]$: this can be done in output-polynomial time according to Proposition 2.2.6. For each such minimal dominating set, either the intersection with A is non-empty and it is a minimal dominating set of G', or it is empty and we can generate in polynomial-time all additions of a vertex of A that would result in a minimal dominating set of G', if any. Since the number of minimal dominating sets of $G'[A \cup B]$ with empty intersection with A is polynomially bounded by the number of those with non-empty intersection (see Lemma 2.2.8, Inequality (2.3)), we can generate all minimal dominating sets of G' not containing u in output-polynomial time.

2. $u \in D$ and $D \cap B \neq \emptyset$. Then $|D \cap B| = 1$, and for any $v \in B$, the set $\{u, v\}$ is a minimal dominating set of G'.

3. $u \in D$ and $D \cap B = \emptyset$. All these minimal dominating sets are triangle-free. We observe that there is a bijection between the minimal dominating sets of this type and the minimal dominating sets of G.

Summarizing, the first two types of minimal dominating sets are easy to generate in output-polynomial time. We note that, free again to disregard minimal dominating sets that are easy to generate, enumerating all triangle-free minimal dominating sets of G' boils down to enumerating all minimal dominating sets of G' that are included in $A \cup \{u\}$ and contain u. This is equivalent to enumerating all minimal dominating sets of G, hence the conclusion.

Note, however, that there is still hope for this technique when we assume some structure on the whole graph.

2.3 Flipping method in comparability graphs

Golovach, Heggernes, Kratsch and Villanger introduced in [GHKV15] the so-called *flipping method* to efficiently enumerate minimal dominating sets in line graphs. This method was later used with much success [GHK+16, GHK+18, KKP18]. We recall it below and in the process, we show that the flipping method can be improved to run with polynomial space (in contrast to exponential space from the original version). We then show how it yields an incremental-polynomial time algorithm for DOM-ENUM in the comparability graphs of posets of bounded dimension.

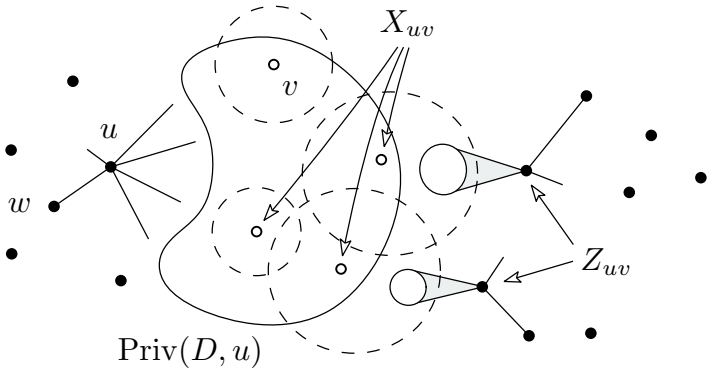


Figure 2.7: An illustration of the flipping operation on a dominating set D such that $G[D]$ contains at least one edge incident to some vertex u, here depicted by uw. Black vertices are elements of D, white vertices are some elements of $\mathrm{Priv}(D,u)$. Dashed discs represent closed neighborhoods, and plain disks represent private neighborhoods.

2.3.1 Polynomial-space flipping method

A key step of the flipping method is the *flipping operation*, that we describe now.

The flipping operation Let G be an n-vertex graph and $v_1,\dots,v_n$ be any ordering of vertices in G. We note that this order induces a lexicographical order on the family $2^{V(G)}$. Let D be a minimal dominating set of G such that $G[D]$ contains at least one edge incident to some vertex u. The following procedure is illustrated in Figure 2.7. Since D is a minimal dominating set, the set $\mathrm{Priv}(D,u)$ is not empty. Let $v \in \mathrm{Priv}(D,u)$. Since u is adjacent with another vertex from D, we have $u \notin \mathrm{Priv}(D,u)$, so $v \neq u$. We want to replace u with v (*flip* u,v) and obtain another minimal dominating set. Let $X_{uv} \subseteq \mathrm{Priv}(D,u) \setminus N[v]$ be the lexicographically smallest maximal independent set in $G[\mathrm{Priv}(D,u) \setminus N[v]]$. In other words, X_{uv} is obtained from the empty set by iteratively adding a vertex of smallest index in $\mathrm{Priv}(D,u) \setminus N[\{v\} \cup X_{uv}]$, until no such vertex exists. Consider the set $D' = (D \setminus \{u\}) \cup X_{uv} \cup \{v\}$. Note that D' is a (not necessarily minimal) dominating set of G. Some vertices of D' may have no private neighbors, however every vertex of $X_{uv} \cup \{v\}$ is self-private. Let Z_{uv} be the lexicographically smallest set which has to be removed from D' in order to make it minimal. In other words, Z_{uv} is obtained from the empty set by iteratively adding a vertex z of smallest index in $D' \setminus Z_{uv}$ such that z has no private neighbor with respect to $D' \setminus Z_{uv}$, until no such vertex exists. Since the elements of $X_{uv} \cup \{v\}$ are self-private in D', the sets $X_{uv} \cup \{v\}$ and Z_{uv} are disjoint, and non-adjacent. Let us finally set $D^* = ((D \setminus \{u\}) \cup X_{uv} \cup \{v\}) \setminus Z_{uv}$. Then D^* is a minimal dominating set of G.

Observe that since X_{uv} and Z_{uv} are selected greedily with respect to $v_1,\dots,v_n$, this procedure is deterministic. Therefore, the procedure assigns to every minimal dominating set D of G, and to every two vertices u,v such that u is in D and is not isolated in $G[D]$, and $v \in \mathrm{Priv}(D,u)$, a unique set D^*. We call D^* the *parent of D with respect to flipping u and v*, and denote it by $\mathrm{Parent}_{uv}(D)$. Conversely, we denote by $\mathrm{Children}(D^*)$ the set of all minimal dominating sets D such that $D^* = \mathrm{Parent}_{uv}(D)$ for some edge uv, and call *child of D^** any element of $\mathrm{Children}(D^*)$. Note that, in the procedure, every

edge in $G[D^*]$ is also an edge in $G[D]$, while there is at least one edge incident with u that appears in $G[D]$ and not in $G[D^*]$. This simple but important observation was formalized as follows.

Proposition 2.3.1 ([GHKV15]). *Let $D, D^* \in \mathcal{D}(G)$ be such that $D^* = \mathrm{Parent}_{uv}(D)$ for some edge uv. Then $E(G[D^*]) \subsetneq E(G[D])$ and v is an isolated vertex of $G[D^*]$.*

The *flipping operation* is then defined to be the reverse of how $D^* = \mathrm{Parent}_{uv}(D)$ was generated from D. This means, given D^* with an isolated element $v \in D^*$ and u a neighbor of v, the operation removes X_{uv} and adds back Z_{uv}, to obtain a child D of D^* with respect to flipping u and v. Obviously, the difficulty is to guess appropriate sets for X_{uv} and Z_{uv} when we are given only D^*, u, and v.

The flipping method We now describe the flipping method as originally introduced in [GHKV15]. Assume that there exists an algorithm $\mathtt{A}$ that, given $D^* \in \mathcal{D}(G)$, enumerates a family $\mathcal{D}$ of minimal dominating sets of G such that $\mathrm{Children}(D^*) \subseteq \mathcal{D} \subseteq \mathcal{D}(G)$. We stress the fact that $\mathcal{D}$ may contain minimal dominating sets that are not actual children of D^*. The *flipping method*, then, consists of a depth-first search (DFS) on a directed supergraph[1] $\mathcal{G}$ whose nodes are minimal dominating sets of G, with one additional special node r, called the root, which has no in-neighbors. The out-neighbors of the root are the maximal independent sets of G (which are minimal dominating sets), and there is an arc from a minimal dominating set $D^* \in V(\mathcal{G})$ to another one $D \in V(\mathcal{G})$ if $\mathtt{A}$ generates D from D^*. At first, the DFS is initiated at the root. Its out-neighbors are generated with polynomial delay using the algorithm of Tsukiyama et al. [TIAS77]. The out-neighbors of the other nodes are generated using $\mathtt{A}$. Since $\mathtt{A}$ outputs (in particular) every child of a given node, we can argue using Proposition 2.3.1 that every minimal dominating set is reachable from r. More solutions may however be output by $\mathtt{A}$, and all the difficulty lies in handling the inherent repetitions.

In [GHKV15] and later papers [GHK+16, GHK+18], a list of already visited nodes of $\mathcal{G}$ is maintained in order to handle repetitions, inexorably requiring space that is linear in $\mathcal{D}(G)$, thus potentially exponential in n. The stack of an arbitrary DFS from the root r to the current visited node D may also require exponential space. The achieved time complexity, on the other hand, is incremental-polynomial.

Lemma 2.3.2 ([GHKV15]). *Let G be a graph. Suppose that there is an algorithm $\mathtt{A}$ that, given $D^* \in \mathcal{D}(G)$, enumerates with polynomial delay a family $\mathcal{D}$ of minimal dominating sets of G such that $\mathrm{Children}(D^*) \subseteq \mathcal{D} \subseteq \mathcal{D}(G)$. Then there is an algorithm that enumerates with incremental delay the set $\mathcal{D}(G)$ of all minimal dominating sets of G.*

We would like to mention that a similar proof allows for "incremental delay" instead of "polynomial delay" in the hypothesis of this statement. We further strengthen the statement in the following.

A polynomial-space flipping method We show here that guiding the DFS toward the children, together with the trick of Lemma 2.2.10 on running the algorithm again

[1] While this is the standard term in this context, one may be more comfortable thinking of it as an "auxiliary graph".

at each output, allows us to handle repetitions with polynomial space at the cost of an increased—but still incremental-polynomial—complexity.

The next lemma is central to the next section and may be regarded as a space improvement of Lemma 2.3.2. It is also of general interest as far as the flipping method is concerned.

Lemma 2.3.3. *Let G be an n-vertex graph, let $p\colon \mathbb{N} \to \mathbb{N}$ and $s \in \mathbb{N}$. Suppose that there is an algorithm $\mathtt{A}$ that, given $D^* \in \mathcal{D}(G)$, enumerates a family $\mathcal{D}$ with delay $p(t)$ and space s, where $\mathrm{Children}(D^*) \subseteq \mathcal{D} \subseteq \mathcal{D}(G)$, p is a non-decreasing function, and t is the number of elements of $\mathcal{D}$ already generated. Then there is an algorithm that enumerates the set $\mathcal{D}(G)$ of all minimal dominating sets of G with delay*

$$O(n^9) \cdot i^3 \cdot p(i)^2$$

and space $O(n^2) \cdot s$, where i is the number of already generated minimal dominating sets.

Proof. In the following, let $\mathcal{G}'$ be the directed graph[2] on vertex set $V(\mathcal{G}') = \mathcal{D}(G) \cup \{r\}$ and edge set $E(\mathcal{G}') = \{(r, D) \mid D \in \mathrm{MIS}(G)\} \cup \{(D^*, D) \mid D \in \mathrm{Children}(D^*)\}$, where r is a special vertex referred to as the *root*.

Let us first argue that every minimal dominating set D is reachable from r in $\mathcal{G}'$, by induction on the number of edges in it. If D contains no edge, it is a maximal independent set, thus an out-neighbor of r. If D contains an edge uw, we can flip u and one of its private neighbors v. Let $D^* = \mathrm{Parent}_{uv}(D)$. By Proposition 2.3.1, D^* has fewer edges than D and is thus reachable from r. There is an arc from D^* to D in $\mathcal{G}'$, hence the conclusion. Therefore, a DFS of $\mathcal{G}'$ initiated at r visits all minimal dominating sets of G.

Furthermore, for every minimal dominating set D and every directed path from r to D the length of the path is at most $|E(G[D])| \leq n^2$.

We now describe an algorithm $\mathtt{B}$ that enumerates, possibly with repetitions, the set $\mathcal{D}(G)$ of all minimal dominating sets of G. When $\mathtt{B}$ outputs a set that was not output before we call this output a *first occurrence*. The algorithm $\mathtt{B}$ will output first occurrences with a delay

$$O(n^7) \cdot i \cdot p(i)$$

and space $O(n^2) \cdot s$, where i is the number of first occurrences output so far. Algorithm $\mathtt{B}$ proceeds as follows. First, it outputs every out-neighbor of r without duplication using the algorithm of Tsukiyama et al. in [TIAS77]. Then, it proceeds with what boils down to a DFS of $\mathcal{G}'$ initiated at r, as follows. When visiting a node $D^* \in V(\mathcal{G}')$, $\mathtt{B}$ seeks the children of D^* by running $\mathtt{A}$. Each set D returned by $\mathtt{A}$ is then output by $\mathtt{B}$. Then $\mathtt{B}$ checks if D is a child of D^*. If so, $\mathtt{B}$ "pauses" the execution of $\mathtt{A}$ on D^*, and launches $\mathtt{A}$ on D. When the execution of $\mathtt{A}$ on D is complete, $\mathtt{B}$ "resumes" the execution of $\mathtt{A}$ on the node D^*.

Before, we discuss the delays between consecutive first occurrences output by $\mathtt{B}$, we take a pause to determine the following: given D and D^* in $\mathcal{G}'$, how fast can we determine if D is a child of D^*? The brute force approach we choose goes as follows: (1) guess the vertex u in D (such that there is an edge incident to u in $G[D]$); (2) guess the vertex v in $\mathrm{Priv}(D, u)$; (3) perform the flipping operation along the uv edge; (4)

[2]This forms a subgraph of $\mathcal{G}$ as defined in Section 2.3.1: Informally, the directed graph $\mathcal{G}'$ is what $\mathcal{G}$ would be if $\mathtt{A}$ was reliable, i.e., only generated children of D^*.

check if the resulting set is D^*. Note that the number of possible guesses in (1) and (2) is at most n^2. In order to make a flip when u and v are fixed, we need to compute the sets X_{uv} and Z_{uv}. The straightforward approach does it in $O(n^3)$ time. Therefore, we can determine if D is a child of D^* in $O(n^5)$ time and $O(n^2)$ space (as for convenience, we work with the adjacency matrix).

We now examine the delay of B between two consecutive first occurrences. The outputs generated by the algorithm of Tsukiyama et al., so the maximal independent sets of G, are produced within $O(n^3)$ time and $O(n^2)$ space, see [TIAS77].

Let D be a first occurrence output by B that is produced by a call of A on a node D^* in $\mathcal{G}'$. Say that D is the i-th first occurrence in order output by B. Thus D is a child of D^*. To obtain the next first occurrence output by B, we consider the path from r to D in $\mathcal{G}'$. The algorithm B continues launching A on D and for each node of the path, except r and D, B has on the stack a paused execution of A called on the node. In the worst case scenario, all the executions will be resumed and each of them will output at most i sets, all of them being already output by B before. Since the length of the path from r to D^* is bounded by n^2 and since p is a non-decreasing function, there are at most $n^2 \cdot i \cdot p(i)$ sets output by the executions of A. Each set is checked by B to see if it is a child of the respective node of $\mathcal{G}'$. A single check takes $O(n^5)$ time, so in total in $O(n^7) \cdot i \cdot p(i)$ time the algorithm B outputs the next first occurrence.

We note that since the time spent between the i-th and $(i+1)$-th first occurrence produced by B is $O(n^7) \cdot i \cdot p(i)$, and since the total number of first occurrences is $|\mathcal{D}(G)| \leq 2^n$, there is a small constant c such that B runs for at most $\mathrm{poly}(n + 2^n) \leq 2^{cn}$ time.

The space consumed by B is dominated by the space taken by at most n^2 paused executions of A and the adjacency matrix of G. Thus B runs in $O(n^2) \cdot s$ space.

We are now ready to describe an algorithm C that enumerates $\mathcal{D}(G)$ without repetitions and within the desired time and space constraints. Algorithm C proceeds as follows. First, it launches a master instance of B. It also maintains a counter keeping track of the number of steps (i.e., elementary steps counted by the time complexity) of the master instance of B. Since B runs for at most 2^{cn} steps, a cn-bits long counter suffices. Whenever the master instance outputs a new set D, and the counter of steps indicates i, the algorithm C launches a new instance of B, runs it for $i-1$ steps, and compares each of its output with D. The new instance of B is killed after exactly $i-1$ steps. If D did not appear as the output of the new instance, then we conclude that it is a first occurrence of the master instance, and the algorithm C outputs D. If D has appeared as one of the outputs of the new instance, then C ignores it and continues the simulation of the master instance. In that way, every set of $\mathcal{D}(G)$ is output by C without repetitions.

We now examine the delay of C between the output of the i-th and $(i+1)$-th minimal dominating sets. Consider the simulation of the master instance of B from the i-th to the $(i+1)$-th first appearance. During that time, recall that at most $n^2 \cdot i \cdot p(i)$ sets are output by the executions of A. Thus, the number of new instances of B launched by C between the two outputs is bounded by $n^2 \cdot i \cdot p(i)$. Every such instance runs for at most $i \cdot O(n^7) \cdot i \cdot p(i)$ time (as p is non-decreasing). In total, C runs for at most

$$n^2 \cdot i \cdot p(i) \cdot i \cdot O(n^7) \cdot i \cdot p(i) = O(n^9) \cdot i^3 \cdot p(i)^2$$

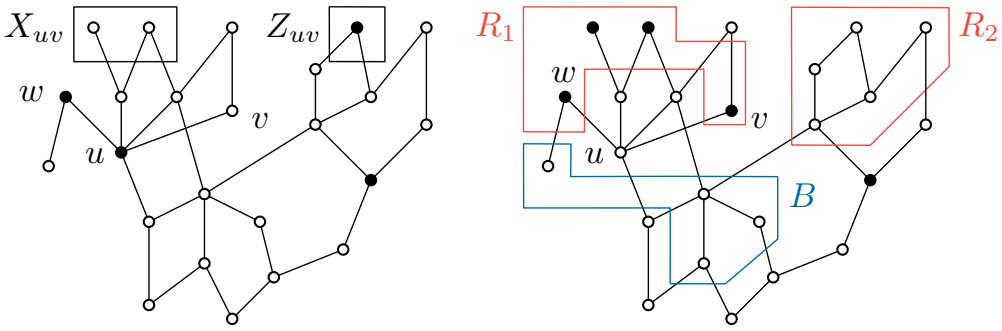


Figure 2.8: A minimal dominating set D (on the left) and its parent $D^* = \mathrm{Parent}_{uv}(D)$ (on the right) represented by black vertices in the underlying poset of a comparability graph. The edge induced by D and incident to u is depicted by uw.

time steps.

The space consumed by C is determined by the space required by at most two independent instances of B running in the same time which is $2 \cdot O(n^2) \cdot s$, and the size of the counter which is $O(n)$. Thus C runs in $O(n^2) \cdot s$ space. □

The algorithms given in [GHKV15, GHK+16, GHK+18] for line graphs, graphs of girth at least 7, chordal bipartite graphs, and unit-square graphs rely on the flipping method and run in incremental-polynomial time and exponential space. By directly plugging in Lemma 2.3.3 instead of Lemma 2.3.2 in the procedure, we obtain the following.

Corollary 2.3.4. *There is an incremental-polynomial time and polynomial-space algorithm enumerating minimal dominating sets in line graphs, graphs of girth at least 7, chordal bipartite graphs, and unit-square graphs.*

2.3.2 Flipping method in comparability graphs

We now show how the flipping method, and more particularly the existence of an algorithm as required in Lemma 2.3.3, can be reduced to red-blue domination in comparability graphs.

We point that Lemma 2.3.3 is stated for general graphs and that the family $\mathcal{D}$ to be constructed can contain arbitrarily many solutions that are not actual children. In[GHKV15], [GHK+16] and [GHK+18], the authors were able to provide such an algorithm A in line graphs, graphs of girth seven, chordal bipartite graphs and unit square graphs. In these last two cases, they proved that to obtain an efficient A, it suffices to design an efficient algorithm that enumerates all the minimal red dominating sets of an appropriate subgraph within the same class. We conduct a similar analysis to show that, in comparability graphs, it suffices to design an efficient algorithm that enumerates all the minimal red dominating sets of a subgraph in which blue vertices are minimal with respect to the associated poset.

Lemma 2.3.5. *Let G be the comparability graph of a poset $P = (V, \leq)$. Suppose that there is an algorithm B that, given an antichain B of P and a set $R \subseteq \uparrow B \setminus B$, enumerates with polynomial delay and polynomial space the set $\mathcal{D}(R, B)$ of minimal red dominating sets of*

$G(R, B)$. Then there is an algorithm $\mathtt{A}'$ that, given $D^* \in \mathcal{D}(G)$ and $u, v \in V(G)$, enumerates with polynomial delay a family $\mathcal{D} \subseteq \mathcal{D}(G)$ of minimal dominating sets of G with the property that $\mathcal{D}$ contains all minimal dominating sets D such that $D^* = \mathrm{Parent}_{uv}(D)$.

Proof. The proof is conducted in the fashion of [GHK+16]. We are given a minimal dominating set D^* of G, an isolated vertex v of $G[D^*]$, and a neighbor u of v. Let us assume in the following that $u \leq v$ in P. The dual situation is handled by flipping upside-down the poset. This situation is depicted in Figure 2.8 (right). We aim to compute using $\mathtt{B}$ a family of sets $\mathcal{D}$ such that $\{D \in \mathcal{D}(G) \mid D^* = \mathrm{Parent}_{uv}(D)\} \subseteq \mathcal{D} \subseteq \mathcal{D}(G)$.

Let $R_1 \subseteq \uparrow u \cap D^*$ be the set of upper-neighbors x of u in D^* such that $x \in \mathrm{Priv}(D^*, x)$, i.e., these vertices are self-private in D^*. Note that in particular, R_1 is an antichain of P, and it contains v. Let $B = V(G) \setminus N[(D^* \setminus R_1) \cup \{u\}]$ be the set of all vertices that are not dominated anymore when replacing R_1 with u in D^*. Note that $B \subseteq \downarrow R_1 \setminus R_1$. Finally, let $R_2 = \uparrow B \setminus N[R_1]$. In particular, there are no edges between R_1 and R_2. Let us finally set $R = (R_1 \setminus \{v\}) \cup R_2$. Informally, R forms the set of vertices we can use to dominate B. We exclude v from that set, since the whole point of the operation is to delete v. We obtain $R \subseteq \uparrow B \setminus B$, and in fact $B \subseteq \downarrow R \setminus R$. Note that B is not necessarily an antichain, so we restrict our attention to the maximal elements of B.

The notation $\mathcal{D}(R \cap \uparrow \mathrm{Max}(B), \mathrm{Max}(B))$ is blatantly cumbersome. To simplify the upcoming arguments, we first prove that $\mathcal{D}(R \cap \uparrow \mathrm{Max}(B), \mathrm{Max}(B))$ is in fact equal to the conceptually simpler $\mathcal{D}(R, B)$.

Claim 4. *The sets $\mathcal{D}(R, B)$ and $\mathcal{D}(R \cap \uparrow \mathrm{Max}(B), \mathrm{Max}(B))$ are equal.*

Proof. We recall that $B \subseteq \downarrow R \setminus R$. As a consequence, $R \subseteq \uparrow \mathrm{Max}(B)$, and it suffices to argue that $\mathcal{D}(R, B) = \mathcal{D}(R, \mathrm{Max}(B))$.

We first note that every dominating set of $\mathrm{Max}(B)$ in $\uparrow \mathrm{Max}(B)$ is a dominating set of B. Indeed, for any $x \in B$, $y \in \mathrm{Max}(B)$, and $z \in \uparrow \mathrm{Max}(B)$, if xy and yz are both edges, then $x \leq y$ and $y \leq z$, so that xz is an edge. This guarantees $\mathcal{D}(R, \mathrm{Max}(B)) \subseteq \mathcal{D}(R, B)$.

The converse is straightforward in the sense that every dominating set of B is in particular a dominating set of $\mathrm{Max}(B)$. As we argued above, any subset that dominates $\mathrm{Max}(B)$ dominates B. Therefore, by minimality, no proper subset of a set in $\mathcal{D}(R, B)$ dominates $\mathrm{Max}(B)$. Every minimal dominating set of B in R is a minimal dominating of $\mathrm{Max}(B)$, hence $\mathcal{D}(R, B) \subseteq \mathcal{D}(R, \mathrm{Max}(B))$ and the conclusion. ⌟

Let us now describe $\mathtt{A}'$. We enumerate all minimal red dominating sets in $\mathcal{D}(R, B)$ using $\mathtt{B}$ with Claim 4. For each minimal red dominating set $X \in \mathcal{D}(R, B)$, we consider the set $D' = (D^* \setminus R_1) \cup \{u\} \cup X$ of vertices of G. Note that X may be empty, in which case $B = \emptyset$ and $\mathcal{D}(R, B) = \{\emptyset\}$; that is not an issue. We greedily reduce D' into an irredundant set D of G, and output D.

This may seem counter-intuitive in an enumeration context, as a greedy reduction typically does not explore all options. However, we will argue later (see Claim 8) that D' is already irredundant in all relevant cases, so $D = D'$ and the greedy reduction does not affect the pool of children.

Let $\mathcal{D}$ be the set of all generated sets; we prove in the following four claims that $\mathcal{D}$ has the desired properties. Namely, the correctness of $\mathtt{A}'$ follows from Claims 5 and 8. We conclude the proof with the complexity analysis of $\mathtt{A}'$.

Claim 5. *All elements of $\mathcal{D}$ are minimal dominating sets of G. Furthermore, there is no repetitions in $\mathcal{D}$, and $|\mathcal{D}| = |\mathcal{D}(R, B)|$.*

Proof. There is a natural bijection between minimal red dominating sets $\mathcal{D}(R, B)$ and outputs $\mathcal{D}$ (taken with multiplicity). We only need to argue two things: that every output is a minimal dominating set, and that there is no repetitions.

There is nothing to argue in the case where $\mathcal{D}(R, B) = \emptyset$, and we assume from now on that $\mathcal{D}(R, B)$ is non-empty. Let $X \in \mathcal{D}(R, B)$. To argue that its corresponding output is a minimal dominating set, it suffices to argue that $(D^* \setminus R_1) \cup \{u\} \cup X$ is a dominating set. Let w be a vertex not dominated by $(D^* \setminus R_1) \cup \{u\}$. By definition, it belongs to B, so $w \in N[X]$. Therefore, $(D^* \setminus R_1) \cup \{u\} \cup X$ is a dominating set. Since we output an irredundant subset of $(D^* \setminus R_1) \cup \{u\} \cup X$, it follows that the output is a minimal dominating set.

Finally, observe that when greedily reducing $(D^* \setminus R_1) \cup \{u\} \cup X$ into a minimal dominating set D, we maintain $X \subseteq D$ as each element of X has a private neighbor in B. In fact, we have $D \cap R = X$, which guarantees that a different choice of X would yield a different output D. ⌟

Claim 6. *For any set $D \in \mathcal{D}(G)$ such that $D^* = \mathrm{Parent}_{uv}(D)$, let X_{uv} and Z_{uv} be the disjoint sets defined in the* Parent *relation, so that $D = (D^* \cup \{u\} \cup Z_{uv}) \setminus (X_{uv} \cup \{v\})$. We have $X_{uv} \subseteq R_1 \setminus \{v\}$ and $Z_{uv} \subseteq R_2$. Additionally, for $Y_{uv} = \bigcup_{z \in Z_{uv}} \mathrm{Priv}(D, z)$, we have $Y_{uv} \subseteq B$.*

Proof. Recall that Z_{uv} is defined as a set of vertices that lose their private neighbors with respect to D when adding $X_{uv} \cup \{v\}$ to $D \setminus \{u\}$. These private neighbors are the elements of the set Y_{uv}.

By definition of the Parent relation, $G[D]$ contains an edge uw, and v is selected in the set $\mathrm{Priv}(D, u)$. Since uw is an edge, one of $u \leq w$ and $w \leq u$ holds. As $v \in \mathrm{Priv}(D, u)$ and $w \in D$, the vertices v and w are incomparable. Since $u \leq v$, the case $w \leq u$ would lead to a contradiction, and we derive $u \leq w$.

Let us first argue that $X_{uv} \subseteq R_1 \setminus \{v\}$. We have $v \notin X_{uv}$, so we focus on proving $X_{uv} \subseteq R_1$. Recall that $X_{uv} \subseteq \mathrm{Priv}(D, u) \setminus N[v]$ by definition. Since $u \leq v$, we derive $X_{uv} \subseteq \uparrow u \cap D^*$. It remains to argue that $x \in \mathrm{Priv}(D^*, x)$ for every $x \in X_{uv}$. Since X_{uv} is an independent set by construction, we have $x \in \mathrm{Priv}(X_{uv}, x)$. Since $X_{uv} \cap N[v] = \emptyset$, we have $x \in \mathrm{Priv}(X_{uv} \cup \{v\}, x)$. Since $X_{uv} \subseteq \mathrm{Priv}(D, u)$, we derive $x \in \mathrm{Priv}(X_{uv} \cup \{v\} \cup (D \setminus \{u\}), x)$, hence $x \in R_1$. It follows that $X_{uv} \subseteq R_1$.

Let us now argue that $Z_{uv} \subseteq R_2$ and $Y_{uv} \subseteq B$. Consider $z \in Z_{uv}$, and a private neighbor y of z with respect to D. Note that $y \in Y_{uv}$ and that y is considered without loss of generality since every element of Y_{uv} is the private neighbor of some element in Z_{uv} with respect to D. Therefore, it suffices to argue that $z \in R_2$ and $y \in B$. Recall that z has no private neighbor with respect to $(D \setminus \{u\}) \cup X_{uv} \cup \{v\}$, though y is a private neighbor of z with respect to D, which contains u. It follows that y is in the neighborhood of $X_{uv} \cup \{v\}$, but not in that of u. Since $X_{uv} \cup \{v\} \subseteq \uparrow u$ and $y \notin \uparrow u$, we have $y \notin \uparrow (X_{uv} \cup \{v\})$. We derive that $y \in \downarrow (X_{uv} \cup \{v\}) \setminus N[u]$. If $y \geq z$, then $z \in \downarrow (X_{uv} \cup \{v\})$, which contradicts the fact that the vertices in $X_{uv} \cup \{v\}$ are private neighbors of u with respect to D. Consequently, $y \leq z$. Note that $z \notin N[R_1]$. Since $R_2 = \uparrow B \setminus N[R_1]$, the fact that $z \in R_2$ follows from $y \in B$, which we argue below.

We have $N(y) \cap D^* \subseteq X_{uv} \cup \{v\}$, as the only neighbor of y in D is z. As shown earlier, $X_{uv} \subseteq R_1$, hence $N(y) \cap D^* \subseteq R_1$. Since $u \notin N(y)$ and $B = V(G) \setminus N[(D^* \setminus R_1) \cup \{u\}]$, we derive $y \in B$, as desired. It follows that $Y_{uv} \subseteq B$ and $Z_{uv} \subseteq R_2$. ⌟

Claim 7. *For any set $D \in \mathcal{D}(G)$ such that $D^* = \mathrm{Parent}_{uv}(D)$, let X_{uv} and Z_{uv} be the disjoint sets defined in the* Parent *relation, so that $D = (D^* \cup \{u\} \cup Z_{uv}) \setminus (X_{uv} \cup \{v\})$. Then the set $R_1 \cup Z_{uv} \setminus (X_{uv} \cup \{v\})$ is a minimal red dominating set of $G(R, B)$.*

Proof. Let $X = R_1 \cup Z_{uv} \setminus (X_{uv} \cup \{v\})$. By Claim 6, we obtain that $X \subseteq (R_1 \setminus \{v\}) \cup R_2 = R$. Therefore, it only remains to argue two things: that X dominates B, and that X is minimal, i.e., that every vertex in X has a private neighbor in B with respect to X.

Let $y \in B$. By definition of B, we have $N(y) \cap D^* \subseteq R_1 \cup \{v\}$ and $y \notin N[u]$. Since D is a dominating set and given how D and D^* relate, the vertex y has a neighbor either in $R_1 \setminus (X_{uv} \cup \{v\})$ or in Z_{uv}. In either case, the vertex y has a neighbor in X. We conclude that X dominates B.

Let us now argue that every vertex x in X has a private neighbor in B with respect to X. Note that $x \in D$ and $\mathrm{Priv}(D, x) \neq \emptyset$.

Let us first consider the case $x \in R_1 \setminus (X_{uv} \cup \{v\})$. Since $u \in D$, we have $\mathrm{Priv}(D, x) \subseteq N[x] \setminus N[u]$. Since moreover $x \in \uparrow u$ we have that $\mathrm{Priv}(D, x) \subseteq \downarrow x \setminus N[u]$. In particular $x \notin \mathrm{Priv}(D, x)$. Let $y \in \mathrm{Priv}(D, x)$. Then $N(y) \cap D = \{x\}$ and so $N(y) \cap D^* \subseteq \{x\} \cup X_{uv} \cup \{v\} \subseteq R_1 \cup \{v\}$. Hence $y \in B$. Therefore, every vertex in $X \cap (R_1 \setminus (X_{uv} \cup \{v\}))$ has a private neighbor in B with respect to X.

We now consider the case $x \in Z_{uv}$. Let $y \in \mathrm{Priv}(D, x)$. Recall that $X_{uv} \cup \{v\}$ and Z_{uv} are non-adjacent. Also y is dominated by D^* but not by $D^* \setminus (X_{uv} \cup \{v\})$, and so $y \neq x$. Hence $y \in N(X_{uv} \cup \{v\})$ and so $y \in N(R_1 \cup \{v\})$. As $N(y) \cap D = \{x\}$ and $x \notin D^*$, we have $N(y) \cap D^* \subseteq X_{uv} \cup \{v\} \subseteq R_1 \cup \{v\}$. Hence $y \in B$. Consequently every $x \in X$ has a private neighbor in B, and so $X \in \mathcal{D}(R, B)$. ⌟

The core statement now follows easily.

Claim 8. *The set $\mathcal{D}$ contains every $D \in \mathcal{D}(G)$ such that $D^* = \mathrm{Parent}_{uv}(D)$.*

Proof. Let $D \in \mathcal{D}(G)$ be such that $D^* = \mathrm{Parent}_{uv}(D)$. Then $D^* = ((D \setminus \{u\}) \cup X_{uv} \cup \{v\}) \setminus Z_{uv}$, where X_{uv} and Z_{uv} are the disjoint sets defined in the Parent relation. Consider the set $Y_{uv} = \bigcup_{z \in Z_{uv}} \mathrm{Priv}(D, z)$.

From Claim 7, we obtain that $R_1 \cup Z_{uv} \setminus (X_{uv} \cup \{v\})$ is a minimal dominating set of $G(R, B)$. Consequently, B outputs $R_1 \cup Z_{uv} \setminus (X_{uv} \cup \{v\})$, which prompts A′ to consider the set $(D^* \setminus R_1) \cup \{u\} \cup (R_1 \cup Z_{uv} \setminus (X_{uv} \cup \{v\})) = (D^* \cup \{u\} \cup Z_{uv}) \setminus (X_{uv} \cup \{v\}) = D$ as a candidate to be output after being greedily reduced into an irredundant set. Note that the set is already irredundant, so D is generated. In other words, we have $D \in \mathcal{D}$ as desired. ⌟

Each of the sets R_1, R_2, R and B can be constructed in polynomial time in n. The same goes for computing and reducing D' into a minimal dominating set given $X \in \mathcal{D}(R, B)$. In addition, $R \cap \uparrow \mathrm{Max}(B)$ and $\mathrm{Max}(B)$ can be computed in polynomial time in n, and by Claim 4 the set $\mathcal{D}(R, B)$ can be generated with polynomial delay using B.

This concludes the proof. □

We conclude this section with the following corollary of Lemmas 2.3.3 and 2.3.5, observing that the antichain B of a poset P is minimal in $P[\uparrow B]$. Note that while the algorithm $\mathbb{A}'$ only computes the children for a fixed pair u, v, there are at most n^2 such pairs. By running $\mathbb{A}'$ for every pair consecutively, we output all children, with each child being repeated possibly n^2 times. We repeat the trick of Section 2.3.1 and get rid of repetitions, to the cost of squaring the time complexity. The obtained algorithm now performs in incremental-polynomial time and polynomial space, as desired.

Theorem 2.3.6. *Let $\mathcal{G}$ be a graph class where every graph is comparability. If there is an incremental-polynomial time and polynomial-space algorithm enumerating minimal red dominating sets in red-blue graphs of $\mathcal{G}$ whose blue vertices are minimal with respect to the associated poset, then there is one enumerating minimal dominating sets in graphs of $\mathcal{G}$.*

2.3.3 Red-blue domination in comparability graphs

We mentioned in the preliminaries of this chapter that RED-BLUE-DOM-ENUM is already as hard as TRANS-ENUM even restricted to bipartite graphs, hence to comparability graphs. We show nevertheless that the problem can be solved in incremental-polynomial time under various restrictions on the red and blue sets (satisfying those of Lemma 2.3.5), as well as on the underlying poset. More precisely, we show that, for any fixed integer t, RED-BLUE-DOM-ENUM is tractable in the comparability graph of S_t-free posets, whenever the blue elements are minimal in the poset. Since posets of bounded dimension do not contain any S_p for some large enough p, we can derive the same for bounded dimension posets.

The key observation is that instances of red-blue domination in that case are of bounded conformality. As a corollary, we can use the algorithm of Khachiyan et al. in [KBEG07] to solve them in incremental-polynomial time. This yields by Theorem 2.3.6 an incremental-polynomial time algorithm enumerating minimal dominating sets in the comparability graphs of these posets.

Let us recall the notion of conformality introduced by Berge in [Ber84]. Informally, a hypergraph has small conformality when the property of not being contained in a hyperedge is witnessed by small subsets, in the sense that if a set is not contained in any hyperedge, then some small subset of it is not either. More formally, let c be an integer and $\mathcal{H}$ be a hypergraph. We say that $\mathcal{H}$ is of *conformality* c if the following property holds for every subset $X \subseteq V(\mathcal{H})$: X is contained in a hyperedge of $\mathcal{H}$ whenever each subset of X of cardinality at most c is contained in a hyperedge of $\mathcal{H}$. Remember from Section 2.1 that a hypergraph $\mathcal{H}$ is Sperner if $E_1 \not\subseteq E_2$ for any two distinct hyperedges E_1, E_2 in $\mathcal{H}$.

Khachiyan, Boros, Elbassioni, and Gurvich proved the following.

Theorem 2.3.7 ([KBEG07])**.** *The minimal transversals can be enumerated in incremental-polynomial time but using exponential space in Sperner hypergraphs of bounded conformality.*

Our result is a corollary of the following, which basically says that in our setting, hypergraphs with large conformality induce large S_t in the underlying poset.

Lemma 2.3.8. *Let $P = (V, \leq)$ be a poset and $B = \mathrm{Min}(P)$. Let $\mathcal{H}$ be the Sperner hypergraph defined by $V(\mathcal{H}) = P - B$ and $\mathcal{E}(\mathcal{H}) = \mathrm{Min}_{\subseteq}\{\uparrow x \setminus \{x\} \mid x \in B\}$. If $\mathcal{H}$ is not of conformality $t-1$ for some integer t, then P contains S_t as a suborder.*

Proof. Assume that $\mathcal{H}$ is not of conformality $t-1$, i.e., there is a red subset $X \subseteq V(\mathcal{H})$ that is not contained in a hyperedge of $\mathcal{H}$, and such that every subset $Y \subseteq X$ of size at most $t-1$ is contained in a hyperedge of $\mathcal{H}$. We consider $X = \{x_1, \ldots, x_p\}$ of minimum cardinality. Then $p \geq t$ and to every $x_i \in X$ corresponds a hyperedge E_i of $\mathcal{H}$ such that $E_i \cap X = \{X \setminus \{x_i\}\}$. Indeed, if no such E_i exists for some $x_i \in X$, then $X' = X \setminus \{x_i\}$ is not contained in a hyperedge of $\mathcal{H}$, and still every subset $Y \subseteq X'$ of size at most $t-1$ is, contradicting the minimality of X.

Let us show that X is an antichain of P. Suppose toward a contradiction that X is not an antichain and contains two elements x_i, x_j such that $x_i < x_j$. As $R \subseteq \uparrow B \setminus B$, every hyperedge of $\mathcal{H}$ that contains x_i contains x_j. We conclude that $X \subseteq E_i$, a contradiction. Hence X is an antichain.

Consider now the antichain $\{e_1, \ldots, e_p\} \subseteq B$ corresponding to $E_1, \ldots, E_p$ in the poset P, i.e., such that $E_i = \uparrow e_i \setminus \{e_i\}$ for $i \in \{1, \ldots, p\}$. Then the set $\{e_1, x_1, \ldots, e_p, x_p\}$ induces S_p as suborder, $p \geq t$. □

Lemmas 2.3.3, 2.3.5, and 2.3.8 together yield the following corollary.

Corollary 2.3.9. *There is an algorithm enumerating, for every fixed integer t, the minimal dominating sets in comparability graphs of S_t-free posets.*

The next theorem follows from Corollary 2.3.9 and the observation that a poset containing S_t has dimension at least t.

Theorem 2.3.10. *For any fixed integer d, there is an output-polynomial time algorithm enumerating minimal dominating sets in the comparability graphs of posets with dimension at most d.*

Unfortunately, as the algorithm in [KBEG07] requires exponential space, Corollary 2.3.9 does not yield a polynomial-space algorithm.

Finally, we note that while it is not clear whether the comparability graphs of bounded dimension posets are of bounded LMIM-width (and hence covered by the algorithm in [GHK+18] using similar methods), comparability graphs of S_t-free posets are not.

2.4 Flashlight search in incomparability graphs

We give a polynomial-delay algorithm enumerating minimal dominating sets in the incomparability graphs of bounded dimension posets, given with linear extensions witnessing the dimension.

2.4.1 Geometrical representation

Let $P = (V, \leq)$ be a poset on n elements and of dimension at most d. Let $\leq_1, \ldots, \leq_d$ be a sequence of linear orders witnessing it. Thus, we have $x \leq y$ in P if and only if $x \leq_i y$

for each $i \in \{1, \ldots, d\}$. Consider d distinct vertical lines $\mathbf{L}_1, \ldots, \mathbf{L}_d$ in the plane, sorted from left to right in that order. For each $i \in \{1, \ldots, d\}$, we distinguish n points on $\mathbf{L}_i$ and label them bottom-up with elements of P sorted by $\leq_i$. Now for each element v in P, we define a piecewise linear curve $\overline{v}$ consisting of $d-1$ segments and connecting points labelled v on consecutive lines. It is a folklore observation, see e.g. [GRU83], that the incomparability graph of P is the intersection graph of this family of curves. An example for $d = 4$ is given in Figure 2.9.

In the remaining of this section, we assume that a poset $P = (V, \leq)$ of dimension d is given and we are also given the total orders $\leq_1, \ldots, \leq_d$ witnessing the dimension of P. As described above we fix the lines $\mathbf{L}_1, \ldots, \mathbf{L}_d$, and the piecewise linear curves representing each element of P. Let G be the incomparability graph of P.

For a non-empty subset S of elements of P and $i \in \{1, \ldots, d\}$, we define $\mathbf{L}_i(S)$ to be the maximum element of S in $\leq_i$. We call *vertices upwards from* S the elements of the set

$$\mathbf{U}(S) = \{v \in V \setminus S \mid \mathbf{L}_i(S) <_i v \text{ for some } i \in \{1, \ldots, d\}\}.$$

A set D is an *upward extension* of S if $S \subseteq D$ and $D \setminus S \subseteq \mathbf{U}(S)$.

Let S be a subset of elements of P of size at least $3d$. We call the *first layer* of S the tuple $\mathbf{A}(S) = (a_1, \ldots, a_d)$ so that $a_1 = \mathbf{L}_1(S)$, and for every $i \in \{2, \ldots, d\}$,

$$a_i = \mathbf{L}_i\left(S \setminus \{a_1, \ldots, a_{i-1}\}\right).$$

Note that by this definition it might happen that $a_i \neq \mathbf{L}_i(S)$. This is in particular the case if $\mathbf{L}_i(S) = \mathbf{L}_j(S)$ for some $i, j \in \{1, \ldots, d\}$ with $j < i$. The *second layer* of S is the set $\mathbf{B}(S) = \mathbf{A}(S \setminus \mathbf{A}(S))$. The *third layer* of S is the set $\mathbf{C}(S) = \mathbf{A}(S \setminus (\mathbf{A}(S) \cup \mathbf{B}(S))$. Note that since $|S| \geq 3d$, the three layers are well-defined. We call the *border* of S the concatenation

$$\mathbf{T}(I) = (a_1, \ldots, a_d, b_1, \ldots, b_d, c_1, \ldots, c_d)$$

of these three layers.

We say that I can be *extended upwards into a minimal dominating set* whenever there is an upward extension of I that is a minimal dominating set of G. In the following, we aim to decide in polynomial time whether a given irredundant set I in G can be extended upwards into a minimal dominating set. When the given set I is of size at most $3d$, say $I = \{x_1, \ldots, x_p\}$ and $p \leq 3d$, then this can be done efficiently by checking for all the tuples $(y_1, \ldots, y_p) \in \mathrm{Priv}(I, x_1) \times \cdots \times \mathrm{Priv}(I, x_p)$ whether $\mathbf{U}(I) \setminus N[y_1, \ldots, y_p]$ dominates $G - N[I]$ or not. A single tuple like that could be verified in $O(n^2)$ time. Since the number of tuples is no more than n^p, the total time is $O(n^{3d+2})$. If there is such a tuple $(y_1, \ldots, y_p)$, then $\mathbf{U}(I) \setminus N[y_1, \ldots, y_p]$ can be greedily reduced into a minimal set X so that $I \cup X$ is a minimal dominating set of G. Otherwise, we know that I cannot be extended as we explored all the possibilities for I to keep its private neighbors in an upward extension. We show that the same technique can be applied for irredundant sets of arbitrary size. The key insight is that it is enough to check whether we can extend I into a dominating set so that all elements in the border $\mathbf{T}(I)$ keep a private neighbor.

Theorem 2.4.1. *Let I be an irredundant set of G of size at least $3d$ and let $\mathbf{T}(I) = (t_1, \ldots, t_{3d})$. Then I can be extended upwards into a minimal dominating set of G if and only if there exists*

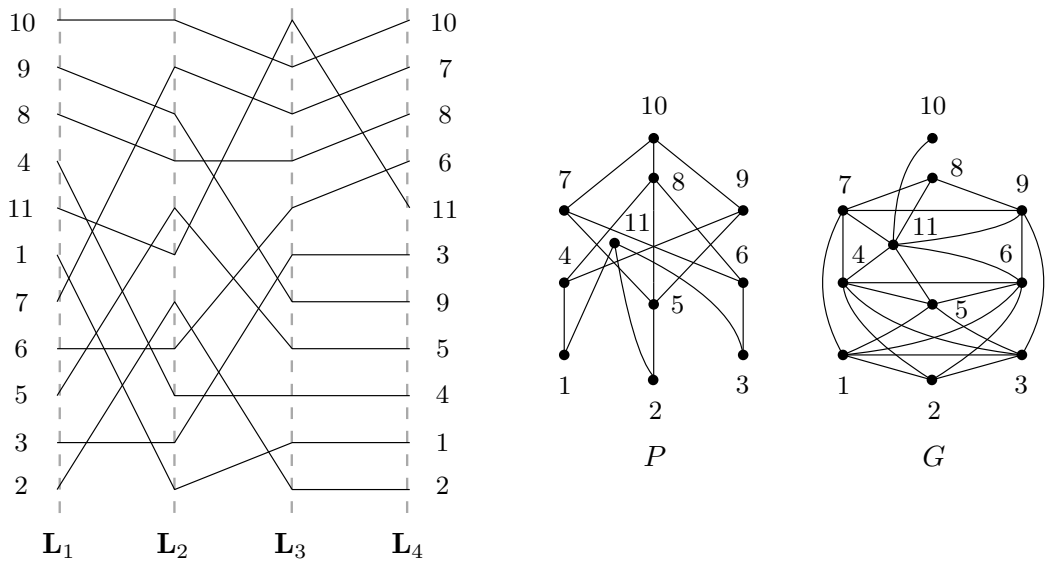


Figure 2.9: A poset P and its incomparability graph G as an intersection graph of curves induced by four linear extensions witnessing the dimension.

a tuple $(w_1, \ldots, w_{3d}) \in \mathrm{Priv}(I, t_1) \times \cdots \times \mathrm{Priv}(I, t_{3d})$ *such that* $\mathbf{U}(I) \setminus N[w_1, \ldots, w_{3d}]$ *is a dominating set of* $G - N[I]$.

Proof. First, we prove the forward implication. Suppose that I can be extended upwards into a minimal dominating set of G and let X be an upward extension of I such that $D = I \cup X$ is a minimal dominating set of G. Then X dominates $G - N[I]$ and $\mathrm{Priv}(D, u) \neq \emptyset$ for every $u \in D$. Since D is a minimal dominating set there exists $(w_1, \ldots, w_{3d})$ in $\mathrm{Priv}(D, t_1) \times \cdots \times \mathrm{Priv}(D, t_{3d})$. Note that $\mathrm{Priv}(D, t_i) \subseteq \mathrm{Priv}(I, t_i)$ for each $i \in \{1, \ldots, 3d\}$. This completes the proof of the forward implication.

We turn to argue the backward implication. Suppose that there exists $(w_1, \ldots, w_{3d})$ in $\mathrm{Priv}(I, t_1) \times \cdots \times \mathrm{Priv}(I, t_{3d})$ such that $X := \mathbf{U}(I) \setminus N[w_1, \ldots, w_{3d}]$ dominates $G - N[I]$. Thus $D = I \cup X$ dominates G. In order to conclude that I extends upwards into a minimal dominating set of G, all we need to see is that each element u in I has a private neighbor with respect to D, i.e., $\mathrm{Priv}(D, u) \neq \emptyset$. Since X avoids $N[w_1, \ldots, w_{3d}]$, we have that $w_i \in \mathrm{Priv}(D, t_i)$ for each $i \in \{1, \ldots, 3d\}$. Consider any element u in $I \setminus \mathbf{T}(I)$. Since I is irredundant, we can fix $v \in \mathrm{Priv}(I, u)$. We shall show that $v \in \mathrm{Priv}(D, u)$.

For convenience, we split the sequence $(t_1, \ldots, t_{3d})$ into $(a_1, \ldots, a_d)$, $(b_1, \ldots, b_d)$, and $(c_1, \ldots, c_d)$, so it relates to the initial layers $\mathbf{A}(I)$, $\mathbf{B}(I)$, and $\mathbf{C}(I)$, respectively.

In order to get a contradiction, suppose that there is $x \in X$ such that x and v are adjacent in G, i.e., $\overline{x}$ and $\overline{v}$ intersect. In particular, there is some $q \in \{1, \ldots, d\}$ such that $x <_q v$.

We claim that

$$v < b_k \text{ in } P \text{ for every } k \in \{1, \ldots, d\}.$$

Since $v \in \mathrm{Priv}(I, u)$ and $b_k \in I$, $\overline{v}$ and $\overline{b_k}$ do not intersect. Therefore, either $v < b_k$ in P or $v > b_k$ in P. Assume toward a contradiction that $b_k < v$ in P. Since $\overline{u}$ and $\overline{v}$ intersect, we can fix $p \in \{1, \ldots, d\}$ such that $v <_p u$. Recall that $u \in I \setminus \mathbf{T}(I)$. Thus, by the definition of the third layer $\mathbf{C}(I)$ we have $u <_p c_p$. Hence $v <_p u <_p c_p$. Since $v \in \mathrm{Priv}(I, u)$ and

$c \in I$, we have that $\overline{c_p}$ and $\overline{v}$ stay disjoint. We deduce that $v < c_p$ in P. This contradicts our assumption as $b_k < v < c_p$ in P but no element of the second layer can be below an element of the third layer. This completes the proof of the claim.

In particular, we have $x <_q v <_q b_k$ for every $k \in \{1, \ldots, d\}$.

Consider the tuple $(s_1, \ldots, s_d) = (w_{d+1}, \ldots, w_{2d})$ of private neighbors of the elements of $(b_1, \ldots, b_d)$. We claim that there exist indices $\alpha, \beta \in \{1, \ldots, d\}$ such that

$$b_\alpha <_q s_\beta.$$

Towards the contradiction, assume that $s_i <_q b_j$ for all $i, j \in \{1, \ldots, d\}$. Since s_i stays disjoint from b_j for $i \neq j$, we conclude that $s_i < b_j$ in P for every $i \neq j$. Since $\overline{s_i}$ intersects $\overline{b_i}$, there is an index $t(i)$ so that $b_i <_{t(i)} s_i$, for each $i \in \{1, \ldots, d\}$. Clearly, the values $t(i)$ must be all distinct for $i \in \{1, \ldots, d\}$. This way, we need to take d distinct values for t_i's and because of our assumption all of them are in $\{1, \ldots, d\} \setminus \{q\}$, a contradiction. This proves the claim.

We concluded so far that

$$x <_q v <_q b_\alpha <_q s_\beta.$$

Now recall that $x \in X \subseteq \mathbf{U}(I)$. Thus, there must be some $p \in \{1, \ldots, d\}$ with

$$\mathbf{L}_p(I) <_p x.$$

Recall also that $\overline{s_\beta}$ intersects $\overline{b_\beta}$, so we can fix $t \in \{1, \ldots, d\}$ such that $s_\beta <_t b_\beta$. By the definition of the first two layers, we have that $b_\beta <_t a_t$. Thus, $s_\beta <_t a_t$. Since $s_\beta \in \mathrm{Priv}(I, b_\beta)$ and $a_t \in I$, we get that s_β and a_t are disjoint. Therefore, $s_\beta < a_t$ in P. In particular, we get $s_\beta <_p a_t$ and

$$s_\beta <_p a_t \leq_p \mathbf{L}_p(I) <_p x.$$

The two inequalities $s_\beta >_q x$ and $s_\beta <_p x$ imply that $\overline{x}$ intersects $\overline{s_\beta}$. Thus x and s_β are adjacent in G. This contradicts the assumption that $X \subseteq \mathbf{U}(I) \setminus N[s_\beta]$ and completes the proof of the backward implication. □

We deduce the next corollary, by guessing a good tuple $(w_1, \ldots, w_{3d})$ as in Theorem 2.4.1 in case when $|I| \geq 3d$, and checking for such a tuple whether the set $\mathbf{U}(I) \setminus N[w_1, \ldots, w_{3d}]$ is a dominating set of $G - N[I]$. Space is polynomial as we only iterate through neighborhoods.

Corollary 2.4.2. *There is an algorithm that, given an irredundant set I of G, decides in $O(n^{3d+2})$ time and polynomial space whether I can be extended upwards into a minimal dominating set.*

2.4.2 Flashlight search

We are now ready to describe an algorithm based on *flashlight search* enumerating minimal dominating sets in the incomparability graphs of bounded dimension posets. We refer the reader to [CS18, MS19] for more details on this classical technique. The following Parent relation will be used by the algorithm to construct the minimal dominating sets one vertex at a time, starting from the emptyset.

Definition 2.4.3. *Let I be a non-empty irredundant set of G that can be extended upwards into a minimal dominating set. We call parent of I the unique irredundant set $I^* = \mathrm{Parent}(I)$ obtained by removing vertex $\mathbf{L}_1(I)$ from I, i.e., the greatest vertex v in I with respect to $<_1$.*

Observe that every minimal dominating set D of G is an irredundant set of G that can be extended upwards into a minimal dominating set (the extension being D itself), with no children. Conversely, an irredundant set that can be extended upwards into a minimal dominating set, with no children, is necessarily a minimal dominating set.

Consequently, the Parent relation as introduced in Definition 2.4.3 defines a tree T whose nodes are irredundant sets of G that can be extended upwards into minimal dominating sets, root is the empty set, leaves are minimal dominating sets of G, and where there is an edge between two irredundant sets I^* and I if $I^* = \mathrm{Parent}(I)$. As in Section 2.3, the enumeration proceeds with what boils down to a DFS of T initiated at the empty set. When visiting a node $I^* \in V(T)$, the algorithm seeks the children of I as follows. It checks for every candidate vertex $v \in \mathbf{U}(I^*)$ whether $I^* \cup \{v\}$ can be extended upwards into a minimal dominating set, using Corollary 2.4.2, and whether the obtained set is a child of I^*, using Definition 2.4.3. Whenever it is the case, the algorithm "pauses" the generation of children of I^*, and generates children of $I^* \cup \{v\}$. When the generation on $I^* \cup \{v\}$ is complete, the algorithm "resumes" the generation on I^*. During this procedure, only the leaves of T, hence the minimal dominating sets of G, are output by the algorithm. Duplications are implicitly avoided by the tree structure of T.

The delay time complexity is bounded by twice the depth of the tree (the maximal distance between two leaves in T), times the time complexity of solving the extension problem and checking the Parent relation for every candidate vertex v. This sums up to

$$2n \cdot O(n^{3d+2} + n^2) \cdot n = O(n^{3d+4})$$

Space complexity is polynomial as we only need to store for each node $W \in T$ from the root to the current node $I^* \in T$ the data of the (paused) execution of the children generation on node W.

We conclude to the following.

Theorem 2.4.4. *There is an algorithm that, given the incomparability graph G of an n-element poset P of dimension d, together with d linear orders witnessing the dimension of P, enumerates all minimal dominating sets of G with delay $O(n^{3d+4})$ and using polynomial space.*

2.5 Further work and open problems

We conclude the chapter by stating open problems and reviewing some perspectives for further research on DOM-ENUM.

2.5.1 Graphs with a forbidden induced subgraph

We recall that a graph is H-free if it does not contain H as an induced subgraph. In Section 2.2 we investigated DOM-ENUM in graph classes forbidding an induced subgraph H. We gave algorithms that run in output-polynomial time and polynomial

space when H is a clique, or more generally when $H = K_t + K_2$, and when H is the paw or the diamond. For simplicity, let us here denote by DOM-ENUM(H) the problem DOM-ENUM restricted to H-free graphs.

The most natural continuation of our work is to seek output-polynomial time algorithms for DOM-ENUM(H) for other choices of the graph H. We discuss a possible classification of the graphs H depending on whether DOM-ENUM(H) admits an output-polynomial time algorithm, is DOM-ENUM-hard, or is not known to belong to one of these two cases. We stress that the first two cases may not be disjoint as it is currently open whether DOM-ENUM admits an output-polynomial time algorithm in general. However, in the current state of the art, such a classification will highlight specific graph classes where the problem could be attacked more easily than in the general case.

Because of Theorem 2.1.1, if H is such that co-bipartite graphs form a subclass of H-free graphs then DOM-ENUM(H) is DOM-ENUM-hard. This includes the cases $H = C_t$ or $H = P_t$ with $t \geq 5$. This is also true for any graph H that has an independent set of size at least three, in particular all graphs H that have at least three connected components and graphs with two connected components where one component has one non-edge. Therefore, all the graphs H with more than one connected component for which DOM-ENUM(H) is not known to be DOM-ENUM-hard are of the form $H = K_p + K_q$ (where by $+$ we denote the disjoint union), for integers $p, q \geq 1$. We gave an output-polynomial time algorithm for the case where $p \leq 2$ or $q \leq 2$ in Theorem 2.2.18 and leave open the existence of such algorithms for $p, q \geq 3$.

Let us now focus on connected choices of H. Besides the case where H is a clique, which we addressed with Theorem 2.2.13, we settled the case where $H = K_t - e$ for $t = 4$ (Theorem 2.2.24). For $t \in \{2, 3\}$, DOM-ENUM(H) is output-polynomial time solvable since $(K_t - e)$-free graphs then are, respectively, cliques and disjoint unions of cliques. To the best of our knowledge, it is currently unknown whether DOM-ENUM($K_t - e$) for $t \geq 5$ is DOM-ENUM-hard and whether it is output-polynomial time solvable. We also considered graphs H of the form $(K_t - \{uv, vw\})$ for $t \geq 3$, i.e., graphs obtained from a clique on t vertices by removing two incident edges. When $t = 3$, $(K_t - \{uv, vw\})$-free graphs are exactly the complete multipartite graphs, for which an output-polynomial time algorithm can be obtained as in the proof of Lemma 2.2.27. We dealt with the case $t = 4$ in Theorem 2.2.28 and leave open the cases of larger t.

Regarding the exclusion of specific graphs, note that the status of DOM-ENUM(P_t) is completely explored: either $t \leq 4$ and an output-polynomial time algorithm is known, or $t \geq 5$ and the problem is DOM-ENUM-hard, as noted above. Among graph classes defined by forbidding an induced cycle, we proved that DOM-ENUM(C_3) is output-polynomial time solvable by Theorem 2.2.9 and noted that DOM-ENUM(C_t) is DOM-ENUM-hard for $t \geq 5$, so only DOM-ENUM(C_4) remains to be classified. The graph C_4 is also the only graph on at most 4 vertices for which DOM-ENUM(H) has not been classified yet. Other graph classes that are closed by taking induced subgraphs and where no output-polynomial time algorithm for DOM-ENUM nor DOM-ENUM-hardness proof are known to include unit-disk graphs [KN14, GHK+16].

Another natural research direction is to optimize the running times of our algorithms or to prove that this is not possible. Theorem 2.2.29 suggests that no improvement of our results can be obtained using flashlight search. We leave as an open prob-

lem whether there are polynomial delay algorithms for DOM-ENUM in the cases that we considered.

Finally, we note that the algorithm of Theorem 2.2.13 has been implemented in python/SageMath [Ray19].

2.5.2 Comparability and incomparability graphs

We provided an incremental-polynomial (resp. polynomial-delay) algorithm enumerating the minimal dominating sets in the comparability (resp. incomparability) graphs of bounded dimension posets. As incomparability graphs include co-bipartite graphs, dropping the dimension in Theorem 2.4.4 is one of the most important algorithmic challenges in enumeration. On the other hand, dropping the dimension in Theorem 2.3.10 seems a more tractable, though fascinating challenge.

Other natural parameters for the classes considered in this paper concern the height and width of the poset. The *height* (resp. *width*) of a poset is the size of its maximum chain (resp. antichain). Since those are related to the largest size of a clique, the algorithm in Section 2.2 covers the comparability graphs of posets of bounded height and the incomparability graphs of posets of bounded width. We can show easily that DOM-ENUM is tractable in the comparability graphs of posets of bounded width.

Proposition 2.5.1. *Let G be the comparability graph of a poset $P = (V, \leq)$ of width α, and D be a minimal dominating set of G. Then $|D| \leq 2\alpha$.*

Proof. Consider for a contradiction a minimal dominating set D of G with $|D| > 2\alpha$. Then D contains three elements x, y, z such that $x < y < z$. As a consequence, $N[y] \subseteq N[x] \cup N[z]$, which contradicts $\mathrm{Priv}(D, y) \neq \emptyset$. □

Proposition 2.5.1 guarantees that a brute-force test of all small subsets yields a polynomial-time algorithm enumerating minimal dominating sets in the comparability graph of posets of bounded width.

As for incomparability graphs of posets of height 2, we consider the incompatibilities of a bipartite order and observe that all co-bipartite graphs can be represented as incomparability graphs of a poset of height at most 2. Therefore, bounded height is not a helpful parameter for incomparability graphs. A more interesting question is perhaps whether restricting posets to lattices yields efficient algorithms, both for comparability and incomparability graphs.

Our algorithm for incomparability graphs of bounded dimension relies heavily on their geometric representation. Geometric graphs seem to be understudied in this context, and the smallest open case is presumably that of unit disk graphs.

Chapter 3

Dualization in lattices given by implicational bases

In this chapter we present new tractability and intractability results for the dualization in lattices given by implicational bases. These results appeared in a joint work with Lhouari Nourine [DN19a] and were extended in [DN20].

We saw in Chapter 2 how the monotone dualization problem could be formulated in graphs in term of domination. We show here that the problem can also be formulated in Boolean lattices—a special kind of poset—as deciding whether two antichains $\mathcal{B}^+$ and $\mathcal{B}^-$ of the lattice partition the elements into two parts: elements that are below $\mathcal{B}^+$, and those that are above $\mathcal{B}^-$. The same problem formulated in arbitrary lattices then appears as a natural generalization we consider in this chapter. Not only the lattice dualization problem is of fundamental interest, it is also of practical interest in lattice-oriented machine learning through hypothesis generation [Kuz04, BK17], and in pattern mining [NP12].

The rest of the chapter is organized as follows. In Section 3.1 we introduce necessary concepts and definitions. The intractability of the general problem is proved in Section 3.2, and a quasi-polynomial time algorithm is given for a restricted case in Section 3.3. Future research directions are pointed in Section 3.4.

3.1 Preliminaries

Recall that a *partial order* on a set X (or *poset*) is a binary relation $\leq$ on X which is reflexive, anti-symmetric and transitive, denoted by $P = (X, \leq)$; see Section 2.1.6. If x is an element of X then $\downarrow x = \{y \in X \mid y \leq x\}$ denotes the *ideal of* x, and $\uparrow x = \{y \in X \mid x \leq y\}$ denotes its *filter*. For a subset $S \subseteq X$, we put $\downarrow S = \bigcup_{x \in S} \downarrow x$ and $\uparrow S = \bigcup_{x \in S} \uparrow x$ as the *ideal* and *filter* of S. An *antichain* of P is a subset of elements of X in which no two distinct elements are comparable.

The following notion is central in this chapter.

Definition 3.1.1. *Let $P = (X, \leq)$ be a poset and B^+, B^- be two antichains of P. We say that the antichains B^+ and B^- are dual in P if $\downarrow B^+ \cup \uparrow B^- = X$ and $\downarrow B^+ \cap \uparrow B^- = \emptyset$.*

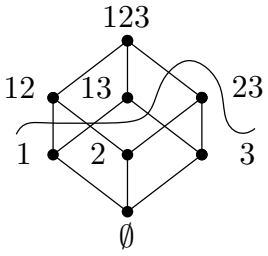


Figure 3.1: The Boolean lattice $\mathcal{L}_X$ of power set of $X = \{1, 2, 3\}$, and the border (curved line) formed by the two dual antichains $\mathcal{B}^+ = \{\{1\}, \{2, 3\}\}$ and $\mathcal{B}^- = \{\{1, 2\}, \{1, 3\}\}$ of $\mathcal{L}_X$. It can be easily verified that the two hypergraphs $\mathcal{H} = \{X \setminus I \mid I \in \mathcal{B}^+\} = \{\{2, 3\}, \{1\}\}$ and $\mathcal{G} = \mathcal{B}^-$ on ground set X are dual. For better readability, closed sets and premises are denoted without braces, i.e., 123 stands for $\{1, 2, 3\}$.

In other words, B^+ and B^- are dual in P if one of $B^+ = \mathrm{Max}_{\leq}\{x \in X \mid x \notin \uparrow B^-\}$ or $B^- = \mathrm{Min}_{\leq}\{x \in X \mid x \notin \downarrow B^+\}$ holds. Hence the problem of deciding whether two antichains B^+ and B^- of P are dual can be solved in polynomial time in the size of P. One has to compute $P - \downarrow B^+$, and to check whether the remaining poset has B^- for minimal elements. The task becomes difficult when the poset is not fully given, but only an implicit coding—of possibly logarithmic size in the size of P—is given: this is usually the case when considering dualization problems in lattices.

3.1.1 Dualization in (implicitly given) Boolean lattices

Somehow conveniently, Boolean lattices can be defined without defining what is a lattice, as follows. A *Boolean lattice* (or *hypercube*) is a poset isomorphic to $\mathcal{L}_X = (2^X, \subseteq)$ for some arbitrary set X. An example of a Boolean lattice is given in Figure 3.1, and should remind Figure 1.1. Note that only X is needed in order to reconstruct the Boolean lattice, and, clearly, the size of $\mathcal{L}_X$ is exponential in that of X. We call X an *implicit coding* of $\mathcal{L}_X$.

Observe that elements of $\mathcal{L}_X$ are subsets of X, and that antichains of $\mathcal{L}_X$ are therefore sets of (inclusion-wise) incomparable subsets of X, i.e., Sperner hypergraphs. Two antichains $\mathcal{B}^+$ and $\mathcal{B}^-$ of $\mathcal{L}_X$ are dual in $\mathcal{L}_X$ if

$$\downarrow \mathcal{B}^+ \cap \uparrow \mathcal{B}^- = \emptyset \text{ and } \downarrow \mathcal{B}^+ \cup \uparrow \mathcal{B}^- = 2^X.$$

The next problem naturally follows.

Dualization in (Implicitly Given) Boolean Lattices (BOOLEAN DUAL)
Input: A set X and two antichains $\mathcal{B}^+, \mathcal{B}^-$ of $\mathcal{L}_X$.
Question: Are $\mathcal{B}^+$ and $\mathcal{B}^-$ dual in $\mathcal{L}_X$?

Note that $\mathcal{L}_X$ is not given as an input of BOOLEAN DUAL, only X is, which is a crucial point. The following equivalence is now folklore and may be found in [NP16]. The intuition here is that if a set T intersects every hyperedge of a hypergraph $\mathcal{H}$, then it is not a subset of the complementary of any hyperedge of $\mathcal{H}$.

Proposition 3.1.2. *Let $\mathcal{H}$ and $\mathcal{G}$ be two Sperner hypergraphs on same ground set X. Then $\mathcal{H}$ and $\mathcal{G}$ are dual if and only if $\mathcal{B}^+ = \{X \setminus E \mid E \in \mathcal{H}\}$ and $\mathcal{B}^- = \mathcal{G}$ are dual in the Boolean lattice $\mathcal{L}_X = (2^X, \subseteq)$.*

Corollary 3.1.3. *The two problems* HYPERGRAPH DUAL *and* BOOLEAN DUAL *are polynomially equivalent.*

Boolean lattices in fact constitute a very low class of lattices. It is then a natural question to consider generalizations of BOOLEAN DUAL to arbitrary lattices. This is the object of this chapter, starting with the next section.

3.1.2 Lattices

What follows is maybe the most iconic definition of a lattice. While these notions will not be used in this chapter—only equivalent characterizations will—, they will be at the core of Chapter 4.

Let $P = (X, \leq)$ be a poset and x, y be two elements of P. If an element u is such that both $x \leq u$ and $y \leq u$ then it is called *upper bound* of x and y; it is called *least upper bound* of x and y if moreover $u \leq v$ for every upper bound v of x and y. Note that two elements x and y of a poset may or may not have a least upper bound. As an example, the elements x_1 and x_3 do not have a least upper bound in Figure 2.1, while x_3 and x_6 do have one. The least upper bound (also known as *supremum* or *join*) of x and y, if it exists, is denoted by $x \vee y$. The greatest lower bound (also known as *infimum* or *meet*) of x and y, if it exists, is denoted by $x \wedge y$ and is defined dually.

A *lattice* is a poset in which every two elements have a least upper bound, and a greatest lower bound. Reference books on the field of lattice theory include the ones of Birkhoff [Bir40], Davey and Priestley [DP02], and Grätzer [Grä11]. A lattice is given in Figure 3.2. It can be verified on this example that every two elements indeed have a least upper bound and a greatest lower bound. In particular, every lattice has a unique minimal element, referred to as the *bot*, and a unique maximal element, the *top*. We say that a lattice is *distributive* if the operations of join and meet distribute over each other, i.e., if for any three elements x, y, z of the lattice,

$$x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z).$$

Lattices, however, can be defined alternatively through their different implicit representations—just as Boolean lattices can be defined as ordered power sets of an arbitrary set. Several such representations exist. One of interest in this chapter is, as the title suggests, the implicational base.

3.1.3 Implicational bases

An implicational base (X, Σ) is a set Σ of implications of the form $A \to B$ where $A \subseteq X$ and $B \subseteq X$; A is called the *premise* of the implication, and B the *conclusion*. Implicational bases have been widely studied in the literature: recent surveys on their role in lattice theory include [Wil17, BDVG18]. In this chapter we only consider implicational bases in their equivalent[1] *unit* form where $|B| = 1$ for every implication, and denote by

[1] In term of their ability to define lattices.

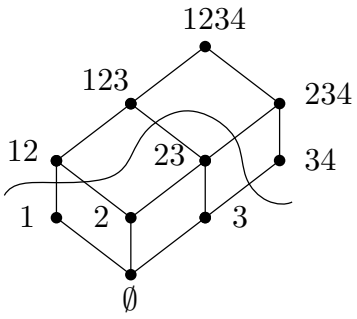


Figure 3.2: The lattice $\mathcal{L}_\Sigma$ of closed sets of the implicational base $\Sigma = \{13 \to 2, 4 \to 3\}$ on ground set $X = \{1, 2, 3, 4\}$, and the border (curved line) formed by the two dual antichains $\mathcal{B}^+ = \{\{1\}, \{2, 3\}\}$ and $\mathcal{B}^- = \{\{1, 2\}, \{3, 4\}\}$ of $\mathcal{L}_\Sigma$. For better readability, closed sets and premises are denoted without braces, i.e., 123 stands for $\{1, 2, 3\}$.

$A \to b$ such implications, where $B = \{b\}$. The *size* of Σ is the number of implications in Σ. It is denoted by $|\Sigma|$. The *dimension* of Σ is the size of a largest premise in Σ.

Let (X, Σ) be an implicational base. We say that a set $C \subseteq X$ is *closed* in Σ if for every implication $A \to b$ of Σ, at least one of $b \in C$ or $A \not\subseteq C$ holds. In other words, a closed set of Σ is a set satisfying all the implications in Σ. Then to Σ we associate the *closure operator* ϕ which maps every $C \subseteq X$ to the smallest closed set of Σ containing C, and that we denote by $\phi(C)$. A closed set $\phi(C)$ for C is obtained by iteratively adding element b to C while there exists $A \to b \in \Sigma$ such that $A \subseteq C$ and $b \notin C$. Efficient closure algorithms may be found in [BO14]. We note $\mathcal{C}_\Sigma$ the set of all closed sets of Σ. The following fact is folklore.

Fact 3.1.4 (Fact π)**.** *Every lattice can be represented as the set of all closed sets of some implicational base, ordered by inclusion.*

To an implicational base (X, Σ) we associate $\mathcal{L}_\Sigma = (\mathcal{C}_\Sigma, \subseteq)$ the lattice of closed sets of Σ. Observe that elements of $\mathcal{L}_\Sigma$ are subsets of X. Then, antichains of $\mathcal{L}_\Sigma$ are families $\mathcal{B} \subseteq \mathcal{C}_\Sigma$ such that $B_1 \not\subseteq B_2$ for any two $B_1, B_2 \in \mathcal{B}$. An example of a lattice of closed sets of an implicational base is given in Figure 3.2. If Σ is empty, then $\mathcal{L}_\Sigma = (2^X, \subseteq)$ is Boolean; see Figure 3.1. If Σ only has premises of size one, then the lattice is distributive, and this is in fact a characterization [Bir37, DP02]. Furthermore in that case, the implicational base can be seen as a poset $P = (X, \leq)$ where $x \leq y$ if and only if $y \to x$, and $\phi(S) = {\downarrow_P} S$ for all $S \subseteq X$, where $\phi(S) = {\downarrow_P} S$ denotes the ideal of S in P (and not in the lattice). We call *underlying poset* of Σ this poset. Note that in general $\mathcal{L}_\Sigma$ may be of exponential size in the size of (X, Σ): this is in particular the case when the implicational base is empty.

3.1.4 Dualization in lattices given by implicational bases

In the remaining of the chapter, we are concerned with the following decision problem and one of its two generation versions.

Dualization in Lattices Given by Implicational Bases (DUAL)
Input: An implicational base (X, Σ) and two antichains $\mathcal{B}^+, \mathcal{B}^-$ of $\mathcal{L}_\Sigma$.
Question: Are $\mathcal{B}^+$ and $\mathcal{B}^-$ dual in $\mathcal{L}_\Sigma$?

Generation version of DUAL (DUAL-ENUM)
Input: An implicational base (X, Σ) and an antichain $\mathcal{B}^+$ of $\mathcal{L}_\Sigma$.
Output: The dual antichain $\mathcal{B}^-$ of $\mathcal{B}^+$ in $\mathcal{L}_\Sigma$.

A positive instance of DUAL is given in Figure 3.2. As for BOOLEAN DUAL, the lattice $\mathcal{L}_\Sigma$ is not given in any of the two problems defined above. Only (X, Σ) is given, which is a crucial point. Recently in [BK17] it was shown that DUAL is coNP-complete, hence that DUAL-ENUM cannot be solved in output-polynomial time unless P=NP. The constructed implicational base, however, has an implication with a premise of unbounded size, leaving open the tractability status of the dualization in the case of implicational bases of bounded dimension. We already saw that when the implicational base is empty—when the lattice is Boolean—the two problems BOOLEAN DUAL and HYPERGRAPH DUAL are equivalent. Then they admit an algorithm running in $N^{o(\log N)}$ time where $N = |\mathcal{B}^+| + |\mathcal{B}^-|$ using the algorithm of Fredman and Khachiyan in [FK96]. In the case of premises of size one—when the lattice is distributive—the best known algorithm is due to Babin and Kuznetsov [BK17] and runs in sub-exponential time $2^{O(n^{0,67}\log^3 N)}$ where $N = |\mathcal{B}^+| + |\mathcal{B}^-|$ and $n = |X|$. Quasi-polynomial time algorithms were given by Elbassioni for subclasses of distributive lattices, including products of chains [Elb09].

It is to be noted that other generalizations of BOOLEAN DUAL exist. One concerns the dualization in lattices given by their meet-irreducible elements[2]. The problem, in this context, was also shown intractable by Babin and Kuznetsov in [BK17]. It is not clear, however, whether this formulation is equivalent to the case where the lattice is given by an implicational base, i.e., DUAL. This kind of issue is raised in Chapter 4.

3.1.5 A few more parameters on implicational bases

We conclude the preliminaries with notions of closure and width that we later consider in this chapter. Let (X, Σ) be an implicational base and ϕ be its associated closure operator. A set $T \subseteq X$ is *independent* w.r.t. ϕ if $x \notin \phi(T \setminus \{x\})$ for any $x \in T$. Given two sets $T, I \subseteq X$ we say that T is a *covering set* of I if $I \subseteq \phi(T)$, and that it is a *generating set* of I if in addition $T \subseteq I$. It is called *minimal* if $I \not\subseteq \phi(T \setminus \{x\})$ for any $x \in T$. Clearly, every minimal covering set of I is independent, and a generating set of I is minimal if and only if it is independent. We point out that these notions only rely on ϕ and not on the implications in Σ. To every $I \subseteq X$ we associate the set $\mathbf{mingen}(I) \subseteq 2^I$ of minimal generating sets of I. Note that several such subsets exist in general. We distinguish a particular one that we denote by $\mathbf{ex}(I)$ and that is obtained from $T = I$ by the following procedure:

> **while there exists** $x \in T$ **such that** $I \subseteq \phi(T \setminus \{x\})$ **do** $T \leftarrow T \setminus \{x\}$
> **return** T **as** $\mathbf{ex}(I)$

[2]These elements cannot be obtained as the greatest lower bound (or meet) of other elements.

In order for such a procedure to be deterministic we chose x of smallest index in T at each step. The next notion now strongly relies on the choice of the implications in Σ. A subset I of implications in Σ is called *independent* if every of its implications has a conclusion that cannot be obtained using the other implications in I, by closure of the premises in I. More formally, a set of k implications $I = \{A_1 \to b_1, \ldots, A_k \to b_k\} \subseteq \Sigma$ with $A = A_1 \cup \cdots \cup A_k$ is independent if $b_i \notin \phi_{I \setminus A_i \to b_i}(A)$ for any $i \in \{1, \ldots, k\}$, where $\phi_{I \setminus A_i \to b_i}$ denote the closure operator of the implicational base $(X, I \setminus A_i \to b_i)$ obtained from I after removing $A_i \to b_i$. In the following, we call *independent-width* of (X, Σ) the size of a maximum independent set of implications in Σ.

3.2 Intractability for implicational bases of dimension two

We show that it is coNP-complete to decide whether two antichains of a lattice given by an implicational base of dimension two are dual. The reduction is based on the one of Kavvadias et al. in [KSS00], pointed in [BK17], except that we manage to hide the Horn clause of unbounded size in one of the two antichains.

Theorem 3.2.1. DUAL *is co*NP*-complete for implicational bases of dimension two.*

Proof. Membership in coNP follows from the fact that checking if $\downarrow \mathcal{B}^+ \cap \uparrow \mathcal{B}^- \neq \emptyset$, or whether a given set $F \subseteq X$, closed in Σ, is such that both $F \notin \downarrow \mathcal{B}^+$ and $F \notin \uparrow \mathcal{B}^-$ can be done in polynomial time in the sizes of (X, Σ), $\mathcal{B}^+$ and $\mathcal{B}^-$; such a set F constitutes a certificate for a 'no' answer.

We show completeness by reducing ONE-IN-THREE 3SAT, restricted to positive literals, to the complement of DUAL. This restricted case of ONE-IN-THREE 3SAT remains NP-complete [KSS00, GJ79]. In this problem, one is given a n-variable, m-clause positive Boolean formula

$$\phi(x_1, \ldots, x_n) = \bigwedge_{j=1}^{m} C_j = \bigwedge_{j=1}^{m} (c_{j,1} \vee c_{j,2} \vee c_{j,3})$$

where $x_1, \ldots, x_n$ and $C_1, \ldots, C_m$ respectively denote the variables and the clauses of ϕ, and where every variable appears in at least one clause ($c_{j,i}$ denotes the variable that appears in clause j at position i). Then the task is of deciding whether there exists an assignment of the variables such that every clause contains exactly one variable to one. We call *one-in-three truth assignment* such an assignment. We construct an instance of DUAL as follows. Let $X = \{x_1, \ldots, x_n, y_1, \ldots, y_m, z\}$ be the ground set made of one element x per variable of ϕ, one element y per clause of ϕ, and an additional special element z. Let Σ be the implicational base defined by

$$\Sigma = \left\{ \begin{array}{rcll} c_{j,1}c_{j,2} & \to & z & (1) \\ c_{j,1}c_{j,3} & \to & z & (2) \\ c_{j,2}c_{j,3} & \to & z & (3) \\ zc_{j,1} & \to & y_j & (4) \\ zc_{j,2} & \to & y_j & (5) \\ zc_{j,3} & \to & y_j & (6) \\ y_j & \to & z & (7) \end{array} \middle| \; j \in \{1, \ldots, m\} \right\}.$$

Then we put

$$\mathcal{B}^+ = \{B_j = X \setminus \{y_j, c_{j,1}, c_{j,2}, c_{j,3}\} \mid j \in \{1, \dots, m\}\},$$
$$\mathcal{B}^- = \{F = \{y_1, \dots, y_m, z\}\}.$$

Clearly, (X, Σ), $\mathcal{B}^+$ and $\mathcal{B}^-$ are constructed in polynomial time in the size of ϕ. Moreover, every $B_j \in \mathcal{B}^+$ is closed in Σ (observe that no literal in $\{c_{j,1}, c_{j,2}, c_{j,3}\}$ is the conclusion of an implication of Σ, and that y_j cannot be implied without any literal in $\{c_{j,1}, c_{j,2}, c_{j,3}\}$). As B_j is the only set of $\mathcal{B}^+$ containing y_j for every $j \in \{1, \dots, m\}$, no two sets in $\mathcal{B}^+$ are inclusion-wise comparable. Hence $\mathcal{B}^+$ is an antichain of $\mathcal{L}_\Sigma$. Also, $\mathcal{B}^-$ is an antichain of $\mathcal{L}_\Sigma$ as it is a singleton and its unique element F is closed in Σ. At last, Σ is of dimension two. Are $\mathcal{B}^+$ and $\mathcal{B}^-$ dual in $\mathcal{L}_\Sigma$? We show that the answer is 'no' if and only if there is a one-in-three truth assignment of ϕ.

We prove the first implication. Let us assume that $\mathcal{B}^+$ and $\mathcal{B}^-$ are not dual in $\mathcal{L}_\Sigma$. Since $\downarrow \mathcal{B}^+ \cap \uparrow \mathcal{B}^- = \emptyset$, there must be some closed set $F' \subseteq X$ such that both $F' \notin \downarrow \mathcal{B}^+$ and $F' \notin \uparrow \mathcal{B}^-$. We consider an inclusion-wise minimal such set F'. Since $F \setminus \{z\}$ is not closed in Σ, and $F \setminus \{y_j\} \subseteq B_j$ for every $j \in \{1, \dots, m\}$, we conclude that $F' \not\subseteq F$. Then $F' \cap \{x_1, \dots, x_n\} \neq \emptyset$. Let $x \in F' \cap \{x_1, \dots, x_n\}$. We show that $z \notin F'$ by contradiction. Suppose that $z \in F'$. Then by Implications (4) to (6), $y_j \in F'$ for all $j \in \{1, \dots, m\}$ such that $x \in C_j$. Hence for every clause C_j containing x, we have that $|F' \cap \{y_j, c_{j,1}, c_{j,2}, c_{j,3}\}| \geq 2$. Hence $F' \setminus \{x\} \not\subseteq B_j$ for any $j \in \{1, \dots, m\}$. Since $F' \setminus \{x\}$ is closed, this contradicts the fact that F' is chosen minimal such that $F' \notin \downarrow \mathcal{B}^+$. Hence F' does not contain z. Clearly $F' \cap \{y_1, \dots, y_m\} = \emptyset$ as otherwise by Implication (7), F' would contain z. As $F' \not\subseteq B_j$ for any $j \in \{1, \dots, m\}$, $|F' \cap C_j| \geq 1$ for every such j. Furthermore $|F' \cap C_j| \leq 1$ for every $j \in \{1, \dots, m\}$ as otherwise by Implications (1) to (3), F' would contain z. Consequently F' is a one-in-three truth assignment of ϕ, concluding the first implication.

We prove the other implication. Let T be a one-in-three truth assignment of ϕ. As $T \subseteq \{x_1, \dots, x_n\}$ and $|T \cap C_j| = 1$ for all $j \in \{1, \dots, m\}$, T is closed in Σ. Furthermore it is not a subset of any $B_j \in \mathcal{B}^+$. Since at last $T \not\supseteq F$, we obtain that both $T \notin \downarrow \mathcal{B}^+$ and $T \notin \uparrow \mathcal{B}^-$. Consequently $\mathcal{B}^+$ and $\mathcal{B}^-$ are not dual in $\mathcal{L}_\Sigma$, concluding the proof. □

Consequently by Corollary 1.2.8, there is no algorithm solving DUAL-ENUM in output-polynomial time unless P=NP, even in the case of implicational bases of dimension two.

3.3 Boolean embedding for implicational bases of bounded independent-width

We show that the dualization in lattices given by implicational bases can be achieved in output quasi-polynomial time whenever the implicational base has bounded independent-width. The approach is similar to the one in [NP14] as we show that the problem can be reduced to hypergraph dualization in that case (the antichains embedded into a Boolean lattice), which allows us to use the quasi-polynomial time algorithm of Fredman and Khachiyan [FK96].

In what follows, let $(X, \Sigma, \mathcal{B}^+)$ be an instance of DUAL-ENUM. Let $\mathcal{B}^-$ be the dual antichain of $\mathcal{B}^+$ that we wish to compute, and $\mathcal{H}$ be the *complementary hypergraph* of $\mathcal{B}^+$ on ground set X defined by

$$\mathcal{H} = \{X \setminus B \mid B \in \mathcal{B}^+\}.$$

Recall that $Tr(\mathcal{H}) = \mathcal{B}^-$ whenever $\mathcal{L}_\Sigma$ is Boolean, that is when the implicational base Σ is empty. We will show how $\mathcal{B}^-$ can be computed from $Tr(\mathcal{H})$ in the general case.

Lemma 3.3.1. *To every transversal T of $\mathcal{H}$ corresponds some $I \in \mathcal{B}^-$ such that $I \subseteq \phi(T)$. This is in particular the case for every minimal transversal of $\mathcal{H}$.*

Proof. Let T be a transversal of $\mathcal{H}$. As $T \cap E \neq \emptyset$ for all $E \in \mathcal{H}$, T satisfies $T \not\subseteq B$ for any $B \in \mathcal{B}^+$. As $T \subseteq \phi(T)$ this is also the case of $\phi(T)$. Hence $\phi(T) \notin\downarrow \mathcal{B}^+$. Now if there is no $I \in \mathcal{B}^-$ such that $I \subseteq \phi(T)$ then $\phi(T) \notin\uparrow \mathcal{B}^-$, contradicting the duality of $\mathcal{B}^+$ and $\mathcal{B}^-$ in $\mathcal{L}_\Sigma$. We conclude that one such I must exist. The last remark follows by inclusion. □

Lemma 3.3.2. *If T is a transversal of $\mathcal{H}$, then every set T^* in $\mathbf{mingen}(T)$ is. In particular, every minimal transversal of $\mathcal{H}$ is independent w.r.t. ϕ.*

Proof. We proceed by contradiction. Let T be a transversal of $\mathcal{H}$ and $T^* \in \mathbf{mingen}(T)$. Suppose that T^* is not a transversal. Then $T^* \subseteq B$ for some $B \in \mathcal{B}^+$. As B is closed, $\phi(T^*) \subseteq \phi(B) = B$. Since T^* is a generating set of T, $T \subseteq \phi(T^*)$. Hence $T \subseteq B$ and thus T is not a transversal of $\mathcal{H}$, a contradiction. Consequently every $T^* \in \mathbf{mingen}(T)$ is a transversal of $\mathcal{H}$. In particular, every minimal transversal T of $\mathcal{H}$ is independent w.r.t. ϕ, as otherwise it can be reduced into an arbitrary independent generating set of T which is smaller, contradicting the minimality of T. □

Lemma 3.3.3. *To every $I \in \mathcal{B}^-$ corresponds $T \in Tr(\mathcal{H})$ such that $T = \mathbf{ex}(I)$.*

Proof. Let $I \in \mathcal{B}^-$. Since $I \not\subseteq B$ for any $B \in \mathcal{B}^+$, I is a transversal of $\mathcal{H}$. Let $T = \mathbf{ex}(I)$. By Lemma 3.3.2 as $\mathbf{ex}(I) \in \mathbf{mingen}(I)$, T is a transversal of $\mathcal{H}$ and since I is closed, $I = \phi(T)$. We show that T is minimal. Let $x \in T$ and $I' = \phi(T \setminus \{x\})$. As T is a minimal generating set of I, $I' \subset I$. By minimality of I it must be that $I' \subseteq B$ for some $B \in \mathcal{B}^+$. Consequently I' does not intersect the hyperedge $E = X \setminus B$ for such a B. As $T \setminus \{x\} \subseteq I'$, $T \setminus \{x\}$ is not a transversal of $\mathcal{H}$. We conclude that $T \in Tr(\mathcal{H})$. □

Algorithm 3.1: An algorithm enumerating the dual antichain $\mathcal{B}^-$ of $\mathcal{B}^+$ in $\mathcal{L}_\Sigma$ given an implicational base (X, Σ) of closure operator ϕ and an antichain $\mathcal{B}^+$ of the lattice $\mathcal{L}_\Sigma$.

```
1 H ← {X \ B | B ∈ B+};
2 for every T ∈ Tr(H) do
3 |   I ← φ(T);
4 |   if I ∈ B− and T = ex(I) then
5 |   |   output I;
6 |   end
7 end
```

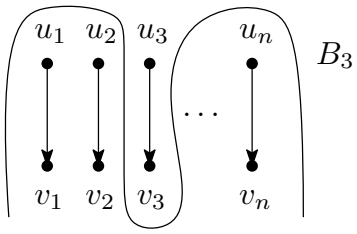


Figure 3.3: An implicational base (X, Σ) on ground set $X = \{u_1, v_1, \ldots, u_n, v_n\}$ where $\Sigma = \{u_i \rightarrow v_i \mid i \in \{1, \ldots, n\}\}$. By taking $\mathcal{B}^+ = \{X \setminus \{u_i, v_i\} \mid i \in \{1, \ldots, n\}\}$, we get $\mathcal{H} = \{\{u_i, v_i\} \mid i \in \{1, \ldots, n\}\}$, $\mathcal{B}^- = \{\{v_1, \ldots, v_n\}\}$ and $Tr(\mathcal{H}) = \{\{z_1, \ldots, z_n\} \mid (z_1, \ldots, z_n) \in \{u_1, v_1\} \times \cdots \times \{u_n, v_n\}\}$.

A consequence of Lemma 3.3.3 is that one can enumerate $\mathcal{B}^-$ from $Tr(\mathcal{H})$ by checking for every $T \in Tr(\mathcal{H})$ whether its closure $I = \phi(T)$ belongs to $\mathcal{B}^-$, whether we have $T = \mathbf{ex}(I)$, and discarding the solution if not. Computing the set $I = \phi(T)$ can be done in $O(|X| \cdot |\Sigma|)$ time. Testing whether I belongs to $\mathcal{B}^-$ can be done in $O(|X|^2 \cdot (|\Sigma| + |\mathcal{B}^+|))$ time by checking for every $x \in I$ whether $I \setminus \{x\}$ is not closed, or whether $I \setminus \{x\} \subseteq B$ for some $B \in \mathcal{B}^+$ otherwise. This holds since I is a transversal and if $I \setminus \{x\} \subseteq B$, then $I \setminus \{x\}$ is not a transversal, which establishes that $I \in B^-$. As for the computation of $\mathbf{ex}(I)$ it can be done in $O(|X|^2 \cdot |\Sigma|)$ time following the definition in Section 3.1. Henceforth, enumerating $\mathcal{B}^-$ can be done in total time

$$M^{o(\log M)} + |Tr(\mathcal{H})| \cdot O(|X|^2 \cdot (|\Sigma| + |\mathcal{B}^+|))$$

where $M = |\mathcal{H}| + |Tr(\mathcal{H})|$, by constructing $\mathcal{H}$ in $O(|X| \cdot |\mathcal{B}^+|)$ time, using the algorithm in [FK96] for the enumeration of $Tr(\mathcal{H})$ in time $M^{o(\log M)}$, and discarding at most $|Tr(\mathcal{H})|$ solutions with a cost of $O(|X|^2 \cdot (|\Sigma| + |\mathcal{B}^+|))$ per solution. Repetitions are avoided by discarding T whenever $T \neq \mathbf{ex}(I)$. This procedure is given in Algorithm 3.1. Its correctness follows from Lemmas 3.3.1 and 3.3.3. The limitation of such a procedure is that the size of $Tr(\mathcal{H})$ may be exponentially larger than that of X, Σ, $\mathcal{B}^+$ and $\mathcal{B}^-$, hence that the described algorithm may run in output-exponential time. An example of one such instance is given in Figure 3.3. However, we will show that it is not the case whenever the implicational base has bounded independent-width.

Our argument relies on the following observation.

Lemma 3.3.4. *Let $I \in \mathcal{B}^-$ and T be a minimal transversal of $\mathcal{H}$ such that $I \subseteq \phi(T)$. Then T is a minimal covering set of I.*

Proof. First recall that by Lemma 3.3.2, T is independent. It may intersect I. Let $x \in T$. By minimality of T, $T \setminus \{x\}$ is not a transversal. By Lemma 3.3.2, neither is $\phi(T \setminus \{x\})$ as otherwise since T is independent then $T \setminus \{x\} \in \mathbf{mingen}(\phi(T \setminus \{x\}))$ is a transversal, which contradicts the hypothesis that T is minimal. Since I is a transversal of $\mathcal{H}$ we have that $I \not\subseteq \phi(T \setminus \{x\})$ for any $x \in T$ and the lemma follows. □

In the following given two subsets $T, I \subseteq X$ such that T is an independent covering set of I, we note $\mathbf{min}(\Sigma, T, I)$ an arbitrary minimal subset of implications of Σ having their premise included in T as a subset and that are needed in Σ in order to derive I from T. In other words, $\mathbf{min}(\Sigma, T, I)$ is obtained from the implications of Σ having

their premise in T by greedily removing an implication of Σ having its premise in T while the inclusion $I \subseteq \phi(T)$ holds. Observe that in consequence no implication in $\mathbf{min}(\Sigma, T, I)$ has a conclusion that is obtained from T by closure of the other implications in $\mathbf{min}(\Sigma, T, I)$, i.e., $\mathbf{min}(\Sigma, T, I)$ is an independent set of implications of Σ.

We now express a bound on the number of minimal covering sets a set admits depending on the number of implications in Σ, and its independent-width. This yields, by Lemma 3.3.4, a bound on the number of minimal transversals of $\mathcal{H}$ depending on the sizes of $\mathcal{B}^-$, Σ, and the independent-width of Σ.

Theorem 3.3.5. *Let I be a subset of X. Then the number of minimal covering sets of I is bounded by $|\Sigma|^k$ where k is the independent-width of Σ.*

Proof. Let $I \subseteq X$ and $T \subseteq X$ be a minimal covering set of I. Consider the implications in $\mathbf{min}(\Sigma, T, I)$. As $\mathbf{min}(\Sigma, T, I)$ is an independent set of implications, we have $|\mathbf{min}(\Sigma, T, I)| \leq k$. Observe in addition that every $x \in T \setminus I$ belongs to at least one premise of an implication in $\mathbf{min}(\Sigma, T, I)$, as otherwise T is not a minimal covering set of I. Furthermore by definition, every x that belongs to the premise of an implication in $\mathbf{min}(\Sigma, T, I)$ is in T. Hence, every such x is either in I or in $T \setminus I$. We conclude that $T \setminus I = \bigcup\{A \mid A \to b \in \mathbf{min}(\Sigma, T, I)\} \setminus I$. Now, observe that T is uniquely characterized by $T \setminus I$ as T is independent: the elements of $T \cap I$ are exactly those of $I \setminus \phi(T \setminus I)$. Since $T \setminus I$ is obtained by union of at most k implications in Σ, the number of minimal covering sets of I is bounded by

$$\sum_{i=1}^{k} \binom{|\Sigma|}{i}$$

hence by $|\Sigma|^k$, as desired. □

A corollary of Lemma 3.3.4 and Theorem 3.3.5 is the following, observing that every solution $I \in \mathcal{B}^-$ admits at most $|\Sigma|^k$ minimal covering sets, hence that at most $|\Sigma|^k$ minimal transversals of $\mathcal{H}$ have their closure containing I.

Corollary 3.3.6. *If Σ is of independent-width k then $|Tr(\mathcal{H})| \leq |\Sigma|^k \cdot |\mathcal{B}^-|$.*

As a consequence, the size of $Tr(\mathcal{H})$ is bounded by a polynomial in $|X| + |\Sigma| + |\mathcal{B}^+| + |\mathcal{B}^-|$ whenever the implicational base is of bounded independent-width. Hence under such a condition, it is still reasonable to test each of the minimal transversals generated by Algorithm 3.1 even though many may not lead to a solution of $\mathcal{B}^-$. We conclude with the following theorem.

Theorem 3.3.7. *There is an algorithm that, for every integer k, given an implicational base (X, Σ) such that Σ is of independent-width k, and an antichain $\mathcal{B}^+$ of $\mathcal{L}_\Sigma$, enumerates the dual antichain $\mathcal{B}^-$ of $\mathcal{B}^+$ in $\mathcal{L}_\Sigma$ in output quasi-polynomial time $N^{o(\log N)}$ where $N = |X| + |\Sigma| + |\mathcal{B}^+| + |\mathcal{B}^-|$.*

Proof. Let k be an integer and (X, Σ) be an implicational base of independent-width k. Let $\mathcal{B}^+$ be an antichain of $\mathcal{L}_\Sigma$, and $\mathcal{H} = \{X \setminus B \mid B \in \mathcal{B}^+\}$ be the complementary hypergraph of $\mathcal{B}^+$. Let $\mathcal{B}^-$ be the dual antichain of $\mathcal{B}^+$ in $\mathcal{L}_\Sigma$ that we wish to compute. By Corollary 3.3.6, the size of $Tr(\mathcal{H})$ is bounded by $|X|^k \cdot |\mathcal{B}^-|$. Let $M = |\mathcal{H}| + |Tr(\mathcal{H})|$ and $N = |X| + |\Sigma| + |\mathcal{B}^+| + |\mathcal{B}^-|$. Since $|\mathcal{H}| \leq |\mathcal{B}^+|$, there exists a constant $c \in \mathbb{N}$

depending in k such that $M \leq N^c$. As a consequence, using the algorithm of Fredman and Khachiyan, the running time of Algorithm 3.1 on instance $(X, \Sigma, \mathcal{B}^+)$ is bounded by

$$M^{o(\log M)} + |Tr(\mathcal{H})| \cdot O(|X|^2 \cdot (|\mathcal{B}^+| + |\Sigma|))$$

hence by

$$N^{c \cdot o(\log N^c)} + \text{poly}(N) = N^{o(\log N)}.$$

□

As a corollary, there is a quasi-polynomial time algorithm solving DUAL in lattices given by implicational bases of bounded independent-width. If the dimension and the independent-width of Σ equal one, Theorem 3.3.7 yields an output quasi-polynomial time algorithm solving DUAL-ENUM in distributive lattices coded by the ideals of an interval order. A poset is an *interval order* if it corresponds to an ordered collection of intervals on the real line such that $[x_1, x_2] < [x_3, x_4]$ if and only if $x_2 < x_3$. Indeed, if Σ is of dimension and independent-width one, then it has no implications $a \to b$ and $c \to d$ such that $d \notin \phi(a)$ and $b \notin \phi(c)$. The underlying poset in that case is 2+2-free, i.e., it does not contain the union of two disjoint 2-elements chains, which is known to characterize interval orders [Fis70].

3.4 Further work and open problems

We state open problems for future research. To an implicational base (X, Σ) we associate its *implication-graph* $G(\Sigma)$ as the directed graph on vertex set X and where there is an arc from x to y if there exists $A \to b \in \Sigma$ such that $x \in A$ and $y = b$. An implicational base (X, Σ) is called *acyclic* if $G(\Sigma)$ has no directed cycle. Acyclic implicational bases have been widely studied in the literature [HK95, BČKK09, Wil17], and are the object of the next chapter. Observe that the negative result of Section 3.2 involves an implicational base which is (highly) cyclic. The next question naturally follows.

Question 3.4.1. *Can* DUAL-ENUM *be solved in output quasi-polynomial time in lattices given by acyclic implicational bases?*

Subclasses of interest include distributive lattices as we recall that the best known algorithm for the dualization in that case is output sub-exponential [BK17]. Superclasses of interest that are not covered by Theorem 3.2.1 include convex geometries that are defined in the next chapter.

Chapter 4

Translating between the representations of a lattice

In this chapter we present new tractability and intractability results on the problem of translating between the representations of a lattice. These results were obtained in a joint work with Lhouari Nourine and Simon Vilmin and may be found in the preprint [DNV19].

We saw in Chapter 3 that any lattice could be considered as the family of closed sets of an implicational base, ordered by inclusion. In fact, finite closure systems arise in various other fields of discrete mathematics and computer science including Horn logic [KKS93, CH11], and relational databases [Mai83, MR92]. They appear in various fields related to lattice theory such as Formal Concept Analysis (FCA) [GW12] and knowledge spaces [DF12]. The study of their representations and how to translate from one to another has gathered increasing attention these last decades, as witnessed by [Wil94, Kha95, BMN17, HN18], the surveys [Wil17, BDVG18], and the Dagstuhl Seminar 14201 [AIBT14].

Among the different ways of representing a closure system, the implicational bases play a central role. We saw that they consist of rules $A \to B$ describing a causality relation within the closure system: a set containing A must contain B. Yet a fundamental aspect that was not raised in the previous chapter is the following observation: several implicational bases can lead to a same closure system (or lattice). Some may be very concise, some huge and redundant. In fact, implicational bases were only considered as an input of the dualization problem in Chapter 3; in that case, being given a very bad implicational base is not a problem as it just increases the size of the input. Moreover, it can always be reduced into a more concise one in polynomial time, if useful, using the algorithms in [Sho86, Wil95]. In consequence to the fact that several implicational bases can code a same closure system, numerous bases were defined and studied in the literature. To cite a few, one can find the Duquenne-Guigues basis having a minimum number of implications [GD86], the unit-minimum having a minimum number of implications among unit implicational bases, or the canonical direct basis having all minimal generators [GW86, BM10]. Refinements of the canonical direct basis include the E-basis and D-basis [FJN95, AN17]. In this chapter, and as in Chapter 3, only unit implicational bases—with conclusions of size one—will be considered, a point that is discussed in the next section.

Another possible representation for a closure system results from a minimum subset of elements from which it can be reconstructed. In Horn logic, these elements are known as the characteristic models [Kha95, HK95]. In lattice theory and closure systems, they are known as the meet-irreducible elements [DP02]. Graphically on the Hasse diagram of a lattice, they correspond to the elements having a unique successor: they cannot be obtained by meet (or intersection) of other elements. We add that these elements are also found in the poset of irreducibles [BM70, Mar75, HN18], or in the reduced context representing the concept lattice in FCA [GW12]. These objects, however, will not be studied in this chapter.

As pointed out by Khardon in [Kha95], the utility of these two representations is not comparable. More notably, there are cases of (unit) minimum implicational bases of exponential size in the number of meet-irreducible elements, and vice versa. Furthermore, different complexities can arise for a same problem when considering one representation, and the other. This is most notably the case for reasoning and abduction [SL90, KKS93], but also when considering dualization in lattices [BK17]. Indeed, we saw in Chapter 3 that the dualization was intractable in lattices given by implicational bases, and pointed that the same result holds when the lattice is given by its meet-irreducible elements [BK17]. The corresponding classes of lattice, however, are not comparable.

Consequently, the problem of translating between implicational bases and meet-irreducible elements is critical in order to reap the benefits of both representations. In Horn logic, the problem of computing the meet-irreducible elements from an implicational base is known as CCM (for Computing Characteristic Models). The problem of computing an implicational base from the meet-irreducible elements is denoted by SID (for Structure Identification). We now give, for completeness, an overview of the progress that has been made on these problems. Their formal definition is postponed to the next section. For the canonical direct and D-basis, output quasi-polynomial time algorithms based on hypergraph dualization were given for both translations in [MR92, AN17]. For the minimum implicational base, output-polynomial time algorithms were obtained in k-meet-semidistributive lattices in [BMN17], as well as in modular lattices in [Wil00]. In [BK13], it is shown that it is coNP-complete in general to decide whether an implication belongs to a minimum implicational base. The existence of an output-polynomial time algorithm constructing a minimum implicational base, however, remains open [BK13, BMN17, Wil17]. In [Kha95], the author is most interested in unit implicational bases as they represent Horn expressions. Indeed, every Horn clause, say $(\neg a_1 \vee \cdots \vee \neg a_k \vee b)$, can be seen as an implication $\{a_1, \ldots, a_k\} \to b$. In that context, the translation problems SID and CCM are shown to be equivalent, and to be at least as hard as hypergraph dualization. This, on its own, justifies the importance of the problem and its place in this manuscript.

The results that are presented in this chapter focus on a particular class of closure systems called *acyclic convex geometries*, defined as closure systems satisfying the anti-exchange property [EJ85], and having an acyclic implication-graph [Wil94], notions that are defined in Section 4.1. We show in Section 4.2 that even when restricted to this class, the problem of translating between one representation and the other is at least as hard as the dualization in distributive lattices, a problem that we already encountered in the previous chapter. Not only this generalizes the observation of Khardon

in [Kha95], but it surprisingly holds in a very low class of closure spaces. In light of this result, we then consider in Section 4.3 a proper subclass of acyclic convex geometries, namely *ranked convex geometries*, as those that admit a ranked implicational base analogous to that of ranked posets. For this class, we provide in Sections 4.4 and 4.5 output quasi-polynomial time algorithms based on hypergraph dualization for translating between the two representations. The first algorithm is based on ordered generation, while the second relies on structural properties of acyclic convex geometries. Future research directions are discussed in Section 4.6.

4.1 Preliminaries

We define general notions related to closure systems. These notions were already encountered in Chapter 3 through implicational bases.

Let X be any set, denoted *ground set* in the following. A map $\phi : 2^X \to 2^X$ is a *closure operator* on X if for all $A, B \subseteq X$:

1. $A \subseteq \phi(A)$;
2. $A \subseteq B$ implies $\phi(A) \subseteq \phi(B)$; and
3. $\phi(A) = \phi(\phi(A))$.

A *closed set* of X w.r.t. ϕ is a set $C \subseteq X$ such that $\phi(C) = C$. The set of all closed sets of X w.r.t. ϕ is denoted by $\mathcal{C}_\phi$. A pair (X, ϕ) where ϕ is a closure operator on X is called *closure space*. It is called *standard* if moreover

1. $\phi(\emptyset) = \emptyset$; and
2. $\phi(x) \setminus \{x\}$ is closed for all $x \in X$.

In the following, all closure spaces are considered standard, a common assumption [AN16]. A *closure system* is a pair $(X, \mathcal{C})$ where $\mathcal{C} \subseteq 2^X$, $X \in \mathcal{C}$ and $C_1 \cap C_2 \in \mathcal{C}$ for all $C_1, C_2 \in \mathcal{C}$. In other words, a closure system is a family of sets, closed under intersection, and containing the ground set as an element. It is well known that to every closure space (X, ϕ) corresponds a closure system $(X, \mathcal{C}_\phi)$, and that to every closure system $(X, \mathcal{C})$ corresponds a closure space (X, ϕ) where $\mathcal{C} = \mathcal{C}_\phi$. We refer to Section 3.1.2 for the definition of a lattice. The following observation is folklore.

Fact 4.1.1. *To any closure space (X, ϕ) corresponds a lattice $\mathcal{L}_\phi = (\mathcal{C}_\phi, \subseteq)$.*

We furthermore have the following equivalences between the meet, join, union, and intersection operators: $C_1 \wedge C_2 = C_1 \cap C_2$ and $C_1 \vee C_2 = \phi(C_1 \cup C_2)$ for every two elements C_1, C_2 in $\mathcal{L}$. An example of a lattice of a closure space is given in Figure 4.1.

4.1.1 Meet-irreducible elements

We define meet-irreducible elements. Let $(X, \mathcal{C} = \mathcal{C}_\phi)$ be a closure system with closure operator ϕ. A set $M \subseteq X$ is a *meet-irreducible element* of $\mathcal{C}$ if $M \in \mathcal{C}$, $M \neq X$, and for all $C_1, C_2 \in \mathcal{C}$ such that $M = C_1 \cap C_2$ it follows that $C_1 = M$ or $C_2 = M$. Similarly,

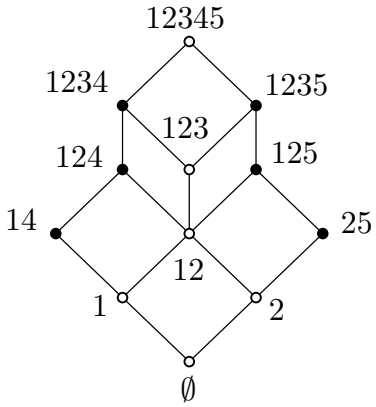


Figure 4.1: The closure lattice $\mathcal{L}_\Sigma = (\mathcal{C}_\Sigma, \subseteq)$ of the implicational base $\Sigma = \{4 \to 1, 5 \to 2, 3 \to 1, 3 \to 2, 45 \to 3\}$ on ground set $X = \{1, 2, 3, 4, 5\}$. Meet-irreducible elements are represented by black vertices. For better readability, closed sets are denoted without braces in the lattice, i.e., 123 stands for $\{1, 2, 3\}$.

a set $J \subseteq X$ is a *join-irreducible element* of $\mathcal{C}$ if $J \in \mathcal{C}$, $J \neq \emptyset$ and for all $C_1, C_2 \in \mathcal{C}$ such that $J = \phi(C_1 \cup C_2)$ it follows that $J = C_1$ or $J = C_2$. We denote by $\mathcal{J}(\mathcal{C})$ and $\mathcal{M}(\mathcal{C})$ the sets of join-irreducible and meet-irreducible elements of $\mathcal{C}$. Put simplier, meet-irreducible (resp. join-irreducible) elements are those that cannot be obtained by intersection (resp. union and closure) of other elements. In the lattice $\mathcal{L} = (\mathcal{C}, \subseteq)$, these elements respectively correspond to those that have a unique predecessor, and a unique successor; see Figure 4.1 for an example. In particular, every atom (i.e., successor of the bottom element) of a lattice is join-irreducible, and every co-atom (i.e., predecessor of the top element) is meet-irreducible. In the following, we will interchangeably note $\mathcal{M}(\mathcal{L})$ and $\mathcal{M}(\mathcal{C})$ whenever $\mathcal{L} = (\mathcal{C}, \subseteq)$.

Let $J, M \in \mathcal{C}$. We define useful notations from [GW12] for closure systems using underlying lattice structure. We note $J \nearrow M$ whenever M is maximal in $\mathcal{L} - \uparrow J$, and $J \swarrow M$ whenever J is minimal in $\mathcal{L} - \downarrow M$. If $J \nearrow M$ and $J \swarrow M$ then we note $J \nearrow\!\!\swarrow M$. We point out here that M and J are element of $\mathcal{L}$, and subsets of X. On the example of Figure 4.1, observe that $J \nearrow M$ for $J = \{2\}$ and $M = \{1, 4\}$. Also, $J \swarrow M$ for $M = \{1, 4\}$ and $J = \{2\}$. Hence, $J \nearrow\!\!\swarrow M$ in that case. In the following, we extend the $\nearrow$ notation to any subset $B \subseteq X$ by defining $B^\nearrow = \mathrm{Max}_\subseteq\{M \in \mathcal{C} \mid B \not\subseteq M\}$. Then $M \in J^\nearrow$ whenever $J \nearrow M$ for any two $M, J \in \mathcal{C}$. We put $j^\nearrow = \{j\}^\nearrow$ for any $j \in X$. The following fact is folklore; see for instance [MR92, Wil95, Wil17].

Fact 4.1.2. *The family $\{J^\nearrow \mid J \in \mathcal{J}(\mathcal{C})\}$ is a (possibly overlapping) set covering of $\mathcal{M}(\mathcal{C})$.*

Since $\mathcal{C}$ is considered standard, another well-known property is that to every set $J \in \mathcal{J}(\mathcal{C})$ corresponds a unique element $j \in X$ such that $J = \phi(j)$. Accordingly, $\mathcal{J}(\mathcal{C})$ coincides with $\{\phi(j) \mid j \in X\}$, and $J^\nearrow$ with $j^\nearrow$ for j such that $\phi(j) = J$.

4.1.2 Links with the implicational bases

Recall that an implicational base (X, Σ) is a set Σ of implications of the form $A \to B$ where $A, B \subseteq X$, and it is called unit if $|B| = 1$ for every such implication. A set S is

closed in Σ if for every implication $A \to B$ of Σ, at least one of $B \subseteq S$ and $A \not\subseteq S$ holds. You might remember as well from Chapter 3 that to every implicational base (X, Σ) corresponds a *closure operator* ϕ which maps every subset $S \subseteq X$ to the smallest closed set $\phi(S)$ of Σ containing S. We say that S implies (or generates) $x \in X$ if $x \in \phi(S)$. A set $S \subseteq X$ is a *minimal generator* of b if $b \in \phi(S)$ and $b \notin \phi(S \setminus \{x\})$ for any $x \in S$. In the following, we will interchangeably denote by $\mathcal{C}_\Sigma$ and $\mathcal{C}_\phi$ the set of all closed sets of Σ. Observe then that (X, Σ) defines a closure space, hence that $(X, \mathcal{C}_\Sigma)$ defines a closure system and $\mathcal{L}_\Sigma = (\mathcal{C}_\Sigma, \subseteq)$ defines a lattice.

As pointed in the introduction of this chapter, to a single closure system can correspond several implicational bases. We say that two implicational bases Σ and Σ' are equivalent, denoted by $\Sigma \equiv \Sigma'$, if $\mathcal{C}_\Sigma = \mathcal{C}_{\Sigma'}$. An implicational base Σ is called *irredundant* if $\Sigma \setminus \{A \to B\} \not\equiv \Sigma$ for all $A \to B \in \Sigma$. We say that an implicational base is *minimum* if it is of minimum size among all equivalent implicational bases. It is called *unit-minimum* if it is of minimum size among all equivalent unit implicational bases.

4.1.3 Implication-graph

The *implication-graph* of (X, Σ) is the directed graph $G(\Sigma)$ defined on vertex set X and where there is an arc between x and y if there exists $A \to B \in \Sigma$ such that $x \in A$ and $y \in B$. An implicational base is called *acyclic* if its implication-graph has no directed cycle. A closure space is called *acyclic* if it admits an acyclic implicational base. We recall from Chapter 3 that if moreover the premises in Σ are of size one, then the closure lattice $\mathcal{L}_\Sigma = (\mathcal{C}_\Sigma, \subseteq)$ is distributive. For a vertex j of $G(\Sigma)$ we shall note j^- the set of all predecessors of j in $G(\Sigma)$, and j^+ all its successors.

4.1.4 Translating between the representations

As mentioned earlier, a closure system $(X, \mathcal{C}_\phi)$ can always be represented and reconstructed from a well-chosen implicational base, or from its meet-irreducible elements. For the latter case, one has to close $\mathcal{M}(\mathcal{C}_\phi)$ under intersection to recover the whole family $\mathcal{C}_\phi$. For the former one, we have to consider Σ such that $\mathcal{C}_\Sigma = \mathcal{C}_\phi$. Such a Σ always exists: $\Sigma = \{A \to \phi(A) \mid A \subseteq X\}$ is an expensive but sufficient representation of ϕ. It can furthermore be turned into a unit implicational base. An example of a closure system represented by its corresponding lattice, a unit-minimum implicational base, and the set of its meet-irreducible elements is given in Figure 4.1.

We are now ready to include the definitions of the problems we consider in this chapter. The names CMI, CCM and SID below originally come from the problems of translating between Horn representations and their characteristic models [Kha95]. Their equivalence with the problems of translating between implicational bases and meet-irreducible elements is well established; see for instance [Wil17]. We point out that the last problem, as put by Khardon, calls for constructing a unit-minimum implicational base. This problem is at least as hard as the one of computing a minimum implicational base, as one can compute a minimum implicational base in polynomial time from a unit-minimum one using the algorithms in [Sho86, Wil95], and that the size of these two implicational bases only differ by a factor $|X|$ being part of the input. It is not clear whether the opposite direction holds, as it was shown in [HK93] that

deciding whether a unit implicational base can be reduced is NP-hard. The problem we consider in this chapter is the hardest one, as in [Kha95].

Meet-irreducible Elements Identification (CMI)
Input: An implicational base (X, Σ) and a family of sets $\mathcal{M} \subseteq 2^X$.
Question: Is $\mathcal{M} = \mathcal{M}(\mathcal{C}_\Sigma)$?

Meet-irreducible Elements Enumeration (CCM)
Input: An implicational base (X, Σ).
Output: The set $\mathcal{M}(\mathcal{C}_\Sigma)$.

Implicational Base Identification (SID)
Input: Two sets X and $\mathcal{M} \subseteq 2^X$.
Output: A unit-minimum implicational base (X, Σ) such that $\mathcal{M} = \mathcal{M}(\mathcal{C}_\Sigma)$.

In [Kha95], the author shows that CCM and SID are equivalent, and that they are at least as hard as TRANS-ENUM. The same result will be observed from a lattice point of view in Section 4.2.

Recall by Fact 4.1.2 that $\{j^{\nearrow} \mid j \in X\}$ is a set covering of $\mathcal{M}(\mathcal{C})$. It follows that the existence of an output-polynomial time algorithm for CCM amounts to the existence of one enumerating $j^{\nearrow}$ for all $j \in X$; see [Wil95, Wil17]. Repetitions are either avoided using exponential memory, or by running the algorithm again on each output to check whether the solution has already been outputted before, as in Lemma 2.2.10, and at the cost of an increasing complexity. In the general case, it can be seen using a result of Babin and Kuznetsov, and of Kavvadias et al. in [KSS00, BK17] on the intractability of generating the co-atoms of a lattice that the computation of $j^{\nearrow}$ is impossible in output-polynomial time unless P=NP. It is however a long-standing open problem whether such a result can be inferred for CCM [Kha95, BMN17, Wil17].

4.1.5 Convex geometries

Let us end the preliminaries by presenting the particular closure system that we consider in the following. A closure space (X, ϕ) satisfies the *anti-exchange property* if for all $x \neq y$ and all closed sets $A \subseteq X$,

$$x \in \phi(A \cup \{y\}) \text{ and } x \notin A \text{ imply } y \notin \phi(A \cup \{x\}).$$

A standard closure space that satisfies the anti-exchange property is called a *convex geometry*. Convex geometries are known to include acyclic closure spaces [Wil94, Wil17]. Sometimes during the paper, we will refer to acyclic closure spaces as *acyclic convex geometries*. It is known from [AGT03] that lattices of convex geometries are both join-semidistributive and lower-semimodular. Whatever that means, it has for consequence that if (X, ϕ) is a convex geometry and $(X, \mathcal{C}_\phi)$ is its associated closure system, then $J \nearrow M$ implies $J \swarrow M$ for all $J \in \mathcal{J}(\mathcal{C}_\phi)$ and $M \in \mathcal{M}(\mathcal{C}_\phi)$ [Ste99, Corollary 4.6.3]. More importantly—and that is the only property that will be used in the following—this has for consequence that the set $\{j^{\nearrow} \mid j \in X\}$ defines a partition (and not only a set covering) of $\mathcal{M}(\mathcal{C}_\phi)$. Consequently, meet-irreducible elements can be enumerated from join-irreducible elements with no need of handling repetitions in that case. This yields the next proposition (the factor two comes from the delay between the output

of the last element in $j^{\nearrow}$, $j \in X$ and the first element in $j'^{\nearrow}$ for a consecutive element $j' \in X$, $j' \neq j$).

Proposition 4.1.3. *Let $f : \mathbb{N} \to \mathbb{N}$ be a function and $(X, \mathcal{C}_\Sigma)$ be the closure system of a given acyclic implicational base (X, Σ). Then there is an algorithm enumerating the set $\mathcal{M}(\mathcal{C}_\Sigma)$ of meet-irreducible elements of $\mathcal{C}_\Sigma$ with delay at most $2 \cdot f(|X| + |\Sigma|)$ whenever there is one enumerating $j^{\nearrow}$ with delay $f(|X| + |\Sigma|)$ given any $j \in X$.*

With all these notions from closure systems we are now armed to show new results on the tractability and intractability of CMI.

4.2 Intractability in acyclic convex geometries

We show that translating between acyclic implicational bases and meet-irreducible elements is at least as hard as the dualization in distributive lattices. For this problem, we recall that the existence of a quasi-polynomial time algorithm is open [BK17].

We briefly recall the dualization problem in lattices given by implicational bases, and refer to Chapter 3 for a more details. Let $\mathcal{L}_\Sigma = (\mathcal{C}_\Sigma, \subseteq)$ be the lattice of closed sets of an implicational base (X, Σ), and $\mathcal{B}^+$ and $\mathcal{B}^-$ be two antichains of $\mathcal{L}_\Sigma$. Then $\mathcal{B}^+$ and $\mathcal{B}^-$ are called dual in $\mathcal{L}_\Sigma$ if

$$\downarrow \mathcal{B}^+ \cap \uparrow \mathcal{B}^- = \emptyset \text{ and } \downarrow \mathcal{B}^+ \cup \uparrow \mathcal{B}^- = \mathcal{C}_\Sigma. \tag{4.1}$$

Then, the dualization problem in lattices given by implicational bases is defined as follows.

Dualization in Lattices given by Implicational Bases (DUAL)
Input: An implicational base (X, Σ) and two antichains $\mathcal{B}^+, \mathcal{B}^-$ of $\mathcal{L}_\Sigma$.
Question: Are $\mathcal{B}^+$ and $\mathcal{B}^-$ dual in $\mathcal{L}_\Sigma$?

The next theorem suggests that translating between implicational bases and meet-irreducible elements is a tough problem, even when restricted to acyclic convex geometries.

Theorem 4.2.1. *There is a polynomial-time algorithm solving* DUAL *in distributive lattices if there is one solving* CMI *in acyclic convex geometries.*

Proof. Let $\mathcal{I}_1 = (X, \Sigma, \mathcal{B}^+, \mathcal{B}^-)$ be an instance of DUAL where the lattice $\mathcal{L}_\Sigma = (\mathcal{C}_\Sigma, \subseteq)$ is assumed to be distributive. Without loss of generality, Σ can be considered acyclic in that case; see [BK17]. We construct an instance $\mathcal{I}_2 = (X \cup \{z\}, \Omega, \mathcal{M})$ of CMI as follows: Ω is the implicational base on ground set $X \cup \{z\}$ constructed from Σ by adding an implication $A \to z$ for every $A \in \mathcal{B}^-$. Clearly the obtained implicational base is acyclic. Then, we put

$$\mathcal{M} = \{M \cup \{z\} \mid M \in \mathcal{M}(\mathcal{L}_\Sigma)\} \cup \mathcal{B}^+. \tag{4.2}$$

Observe that the left and right sides of the union in Equality (4.2) are disjoint. A representation of the lattice $\mathcal{L}_\Omega = (\mathcal{C}_\Omega, \subseteq)$ is given in Figure 4.2. We shall show that $\mathcal{I}_1$ is a positive instance of DUAL if and only if $\mathcal{I}_2$ is one of CMI, and that it can be decided in polynomial time in the size of $\mathcal{I}_1$ given a polynomial-time algorithm for CMI.

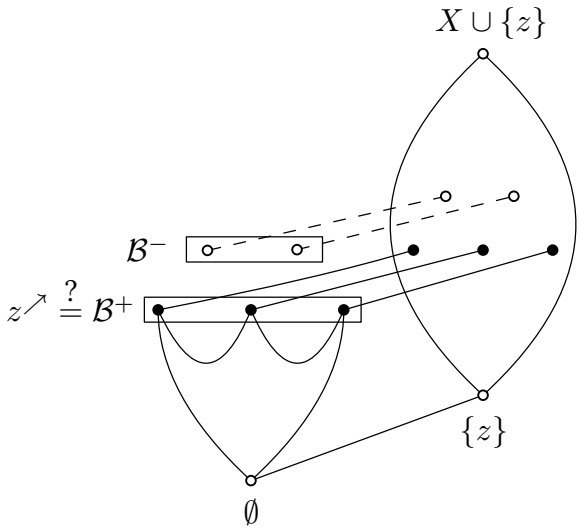


Figure 4.2: The construction of Theorem 4.2.1.

Consider the elements of $\mathcal{L}_\Omega[\uparrow \{z\}]$, i.e., the right part of Figure 4.2. Observe that $\mathcal{L}_\Sigma$ and $\mathcal{L}_\Omega[\uparrow \{z\}]$ are isomorphic. Consequently, there is a bijection from $\mathcal{M}(\mathcal{L}_\Sigma)$ to $\mathcal{M}(\mathcal{L}_\Omega[\uparrow \{z\}])$ given by $f : M \mapsto M \cup \{z\}$ and $f^{-1} : M' \mapsto M' \setminus \{z\}$. Hence

$$\{M \cup \{z\} \mid M \in \mathcal{M}(\mathcal{L}_\Sigma)\} = \mathcal{M}(\mathcal{L}_\Omega[\uparrow \{z\}]). \tag{4.3}$$

Consider now the elements of $\mathcal{L}_\Omega - \uparrow \{z\}$, i.e., the left part of Figure 4.2. Observe that the only way for an element N in such a part to be meet-irreducible is to belong to $z^{\nearrow}$, as otherwise N has at least one successor in $\mathcal{L}_\Omega - \uparrow \{z\}$, and another one in $\mathcal{L}_\Omega[\uparrow \{z\}]$. Hence $\mathcal{M}(\mathcal{L}_\Omega) \setminus \uparrow \{z\} = z^{\nearrow}$. Consequently by Equalities (4.2) and (4.3), $\mathcal{M} = \mathcal{M}(\mathcal{L}_\Omega)$ if and only if $\mathcal{B}^+ = z^{\nearrow}$. As by construction,

$$z^{\nearrow} = \mathrm{Max}_\subseteq \{F \in \mathcal{C}_\Sigma \mid A \not\subseteq F \text{ for any } A \in \mathcal{B}^-\}, \tag{4.4}$$

we conclude that $\mathcal{B}^+ = z^{\nearrow}$ if and only if $\mathcal{B}^+$ and $\mathcal{B}^-$ are dual in $\mathcal{L}_\Sigma$. Hence that $\mathcal{I}_1$ is a positive instance of DUAL if and only if $\mathcal{I}_2$ is one of CMI. Concerning the observation that $\mathcal{I}_1$ can be decided in polynomial time in $|\mathcal{I}_1|$ using a polynomial-time algorithm for CMI, it is a consequence of the fact that $\mathcal{L}_\Sigma$ being distributive, the size of $\mathcal{M}(\mathcal{L}_\Sigma)$ is bounded by $|X|$, hence that $|X \cup \{z\}| + |\Omega| + |\mathcal{M}|$ is bounded by a polynomial in $|\mathcal{I}_1|$. This concludes the proof. □

As a consequence, there is no output quasi-polynomial time algorithm for CCM, nor SID, unless there is one solving the dualization in distributive lattices in quasi-polynomial time, a long-standing open problem [BK17]. This generalizes the observation of Khardon in [Kha95] (which is obtained replacing "distributive lattice" by "Boolean lattice" in Theorem 4.2.1).

As for the computation of $j^{\nearrow}$ given an acyclic implicational base (X, Σ) and some element $j \in X$, a corollary of Equality (4.4) in the proof of Theorem 4.2.1 is that the task is even harder than the dualization in lattices given by acyclic implicational bases (a proper superclass of distributive lattices), whenever Σ is acyclic. To the best of our knowledge, no better algorithms than exponential naive ones are known on such

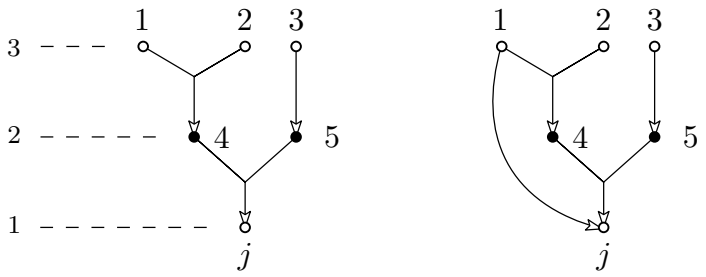


Figure 4.3: A ranked implicational base (left) and an acyclic implicational base that does not admit a rank function (right). Ranks are depicted by dashed lines.

class. This suggests that the technique employed in [BMN17] and pointed by Proposition 4.1.3 for the enumeration of meet-irreducible elements from join-irreducible elements may not be efficient in acyclic convex geometries. However, we will show in Section 4.4 that it can work in a subclass of acyclic convex geometries.

4.3 Ranked convex geometries

Let (X, Σ) be an implicational base. A *rank function* on (X, Σ) is a function $\rho : X \to \mathbb{N}$ such that if $A \to b \in \Sigma$ and $a \in A$, then $\rho(a) = \rho(b) + 1$. We say that (X, Σ) is *ranked* if it admits a rank function. See Figure 4.3 for an illustration. It is not hard to see that ranked implicational bases are acyclic, hence that they define a subclass of acyclic convex geometries. We say that a convex geometry is *ranked* if it admits a ranked implicational base. Observe that the construction of Theorem 4.2.1 is not ranked, as not all distributive lattices admit a ranked implicational base, and in addition the premises of the constructed implications may not contain elements of a same rank.

In this section, we prove structural properties on acyclic and ranked convex geometries. Let us first recall a result from [HK95] (see also [Wil17]) which plays a central role in the remaining of the section. Remind that a set $A \subseteq X$ is a minimal generator of b if it is minimal satisfying $b \in \phi(A)$.

Proposition 4.3.1 ([HK95]). *Let (X, ϕ) be an acyclic convex geometry. Then (X, ϕ) admits a unique irredundant implicational base of minimal generators which is unit-minimum. Furthermore, it can be computed in quadratic time from any unit implicational base.*

In the following, we call *critical base* the irredundant implicational base of minimal generators of an acyclic convex geometry given by Proposition 4.3.1. Accordingly, we call *critical*[1] a minimal generator that belongs to this base, and *redundant* one that does not. The next two propositions can be inferred from other results in matroid and antimatroid theory [Die87, KLS12, NK13]. We nevertheless reprove them here for self-containment.

[1]The terminology coincides with the notion of *critical* from [Die87] (the second part of Proposition 4.3.2) and *essential* from [HK95] (minimal generators that belong to every implicational bases of minimal generators) in acyclic convex geometries. See also [Wil17].

Proposition 4.3.2. *Let (X, ϕ) be an acyclic convex geometry and $A \subseteq X$ be a minimal generator of $b \in X$. Then A is redundant if and only if its closure $\phi(A)$ contains another minimal generator of b.*

Proof. Let A be a redundant minimal generator of b and Σ be the critical base of (X, ϕ) given by Proposition 4.3.1. Since $A \to b$ does not belong to Σ and $b \in \phi(A)$, there exists an implication $C \to b \in \Sigma$, $A \neq C$ such that $C \subseteq \phi(A)$.

Let A be a minimal generator of b and suppose that there exists another minimal generator C of b such that $C \subseteq \phi(A)$. Consider the critical base Σ of (X, ϕ) given by Proposition 4.3.1. Since A and C are minimal generators, $C \setminus A$ is not empty. Since $C \subseteq \phi(A)$ for each $c \in C \setminus A$ there are implications in Σ leading to c from A. Since C is a minimal generator of b, there are implications in Σ leading to b from C. Consequently, there exists a sequence of implications in Σ that does not contain $A \to b$ and that nevertheless produces b from A. We conclude that $A \to b$, if in Σ, is redundant. Hence A is redundant. □

Remark 4.3.3. By the choice of Σ in the proof of Proposition 4.3.2, the minimal generator of b can be required to be critical.

A corollary of this proposition is the next characterization.

Corollary 4.3.4. *Let (X, ϕ) be an acyclic convex geometry and $A \subseteq X$ be a minimal generator of $b \in X$. Then A is redundant if and only if there exists an element $a \in A$ such that the set $\phi(A) \setminus \{a, b\}$ implies b.*

Proposition 4.3.5. *Let (X, ϕ) be an acyclic convex geometry, $A \subseteq X$ be a critical minimal generator of $b \in X$, and Σ be any ranked implicational base of (X, ϕ). Then there exists $C \to b \in \Sigma$ such that $A \subseteq C$.*

Proof. Let A be a critical minimal generator of b and assume for contradiction that there is no implication $C \to b \in \Sigma$ such that $A \subseteq C$. In particular since $b \notin A$ and $b \in \phi(A)$ there exists at least one implication $D \to b$ in Σ such that $D \subseteq \phi(A)$. By hypothesis $A \not\subseteq D$. Hence by acyclicity of (X, ϕ) we deduce $D \subset \phi(A)$. Then there exists $D' \subseteq D \subset \phi(A)$ such that D' is a minimal generator of b. By Proposition 4.3.2, it contradicts A being critical. □

We are now ready to state the main result of this section, which is of interest as far as the computation of a unit-minimum ranked implicational base is concerned (SID).

Theorem 4.3.6. *Let (X, ϕ) be an acyclic convex geometry. Then (X, ϕ) is ranked if and only if its critical base is ranked.*

Proof. The if part follows from the definition. We prove the other direction. Let us assume that (X, ϕ) is ranked and consider its critical base Σ^* given by Proposition 4.3.1. Consider now any other implicational base Σ of (X, ϕ). Then by Proposition 4.3.5 for every implication $A \to b \in \Sigma^*$ there exists $A' \to b \in \Sigma$ such that $A \subseteq A'$. Hence if Σ^* is not ranked it must be that the elements in such A's together with their conclusion b do not admit a rank function in Σ^*. As all such elements appear as is in Σ, they cannot admit a rank function in Σ neither. Consequently Σ^* must be ranked. □

We now consider the problem of deciding whether an acyclic convex geometry is ranked, from an implicational base and from its meet-irreducible elements. We show that the first problem lies in P while the second is in coNP. Whether the second problem is in P or is coNP-complete is left as an open problem. As a preliminary remark, observe that the implicational base obtained by disjoint union of ranked implicational bases is ranked. A corollary is that every closure system obtained by product of chains is ranked, a class of interest in [Elb09].

Proposition 4.3.7. *Checking whether an implicational base (X, Σ) admits a rank function ρ, and computing ρ if it does, can be done in polynomial time in the size of (X, Σ).*

Proof. Let $G(\Sigma)$ be the implication-graph of Σ that can be computed in polynomial time in the size of (X, Σ). Observe that we can restrict ourselves to the case where $G(\Sigma)$ is connected, as otherwise we handle each connected component independently. The algorithm proceeds as follows. At first it picks a vertex x of $G(\Sigma)$ and set $\rho(x) = n$ (n is chosen to ensure that we cannot attribute a negative rank to a vertex in the next steps, hence that $\rho : X \to \mathbb{N}$). Then, for every unmarked vertex x with a rank, the algorithm marks x and extend ρ to every $y \in x^- \cup x^+$, until no such vertex exists. Note that extending ρ is deterministic by definition. Hence if a conflict is detected during the procedure, then we can certificate that Σ is not ranked. Otherwise, a rank function is computed. □

Proposition 4.3.8. *Deciding whether an acyclic convex geometry is ranked from its meet-irreducible elements belongs in co*NP.

Proof. We describe a polynomial-size certificate that can be checked in polynomial time in order to answer negatively. Such a certificate is a set $\{A_1 \to b_1, \ldots, A_k \to b_k\}$ of critical implications that do not admit a rank function. Recall that by Theorem 4.3.6 the convex geometry is not ranked if and only if these implications exist. Furthermore, they must induce an undirected cycle in $G(\Sigma)$, and only one such cycle is needed to answer negatively. Hence k is bounded by $|X|$. One has to check first that each of these implications is indeed a critical minimal generator, following the next three steps. To decide whether $A_i \to b_i$ is valid, we compute the closure of A_i using meet-irreducible elements, as described in the preliminaries. To check that A_i is a minimal generator of b_i, we test whether $b_i \notin \phi(A_i \setminus \{a\})$ for all $a \in A_i$. As for the critical property, we check according to Corollary 4.3.4 whether there exists $a \in A_i$ such that $\phi(A_i) \setminus \{a, b_i\}$ implies b_i. Then we launch the polynomial-time procedure given in Proposition 4.3.7 to check whether these implications admit a rank function. □

4.4 Ordered generation in ranked convex geometries

We give an output quasi-polynomial time algorithm, based on ordered generation and hypergraph dualization, for the enumeration of meet-irreducible elements (CMI) in closure systems given by a ranked implicational base. If in addition the implicational base is of bounded dimension, then the algorithm is shown to perform in output-polynomial time.

Let $\mathcal{C}_\Sigma$ be a closure system given by a ranked implicational base (X, Σ) with rank function ρ and closure operator ϕ. Note that by Proposition 4.3.1 and Theorem 4.3.6 we can assume Σ to be the critical base of (X, ϕ). In the following and for every $B \subseteq X$ we put

$$B^{\nearrow} = \mathrm{Max}_{\subseteq}\{C \in \mathcal{C}_\Sigma \mid C \cap B = \emptyset\}.$$

Observe that this definition generalizes the $\nearrow$ relation for singletons, as the following remark states.

Remark 4.4.1. Let $B \subseteq X$. Then $B^{\nearrow} = B^{\nearrow}$ if $|B| = 1$. Hence the elements in $B^{\nearrow}$ are meet-irreducible in that case.

In the following, we say that a set $B \subseteq X$ is *ranked* with respect to ρ if every element in B shares the same rank according to ρ, that is, if $\rho(a) = \rho(b)$ for all $a, b \in B$. Then to every ranked set B we associate $\rho(B)$ the rank of one of its elements (or an arbitrary rank if B is empty), and $\mathcal{H}_B$ the hypergraph defined on vertex set $V(\mathcal{H}_B) = \{x \in X \mid \rho(x) = \rho(B) + 1\}$ and edge set

$$\mathcal{E}(\mathcal{H}_B) = \{A \mid A \to b \in \Sigma \text{ for some } b \in B\}.$$

Note that every vertex of $\mathcal{H}_B$ has the same rank. In general, $V(\mathcal{H}_B)$ may be a proper superset of $\bigcup \mathcal{E}(\mathcal{H}_B)$: the elements that do not belong to any hyperedge of $\mathcal{H}_B$ are of interest in the following. We will show that the maximal independent sets of $\mathcal{H}_B$ allow to define a partition of $B^{\nearrow}$. For any $C \subseteq X$ we denote by C_i the elements of C with rank i.

Proposition 4.4.2. *Let $B \subseteq X$ be a ranked set of maximum rank in Σ. Then $B^{\nearrow} = \{C\}$ where $C = \{x \in X \mid \rho(x) \leq \rho(B),\ x \notin B\}$.*

Proof. As B is of maximal rank, no implication in Σ has $b \in B$ for conclusion. Since Σ is acyclic, no element $x \in X$ such that $\rho(x) \leq \rho(B)$, $x \notin B$ can imply $b \in B$. The proposition follows. □

In the following, let us put $k = \mathrm{Max}\{\rho(x) \mid x \in X\}$ to be the maximum rank of Σ. The aforementioned partition is the following.

Theorem 4.4.3. *Let $B \subseteq X$ be a ranked set of rank $i < k$, and $\mathcal{H}_B$ be its associated hypergraph. Then there is a partition of $B^{\nearrow}$ given by*

$$\big\{\{C \in B^{\nearrow},\ C_{i+1} = S\} \mid S \in MIS(\mathcal{H}_B)\big\}.$$

The proof of Theorem 4.4.3 is decomposed into Lemmas 4.4.4 and 4.4.5. The partition is illustrated in Figure 4.4.

Lemma 4.4.4. *Let $B \subseteq X$ be a ranked set of rank $i < k$, and $\mathcal{H}_B$ be its associated hypergraph. Then to every $C \in B^{\nearrow}$ corresponds $S \in MIS(\mathcal{H}_B)$ such that $C_{i+1} = S$.*

Proof. Let $C \in B^{\nearrow}$ and $S = C_{i+1}$. Observe that since C is closed and as it does not intersect B, S is an independent set of $\mathcal{H}_B$. We show that it is maximal. Let $x \in V(\mathcal{H}_B)$ such that $x \notin S$. Then $x \notin C$ and by maximality of C, $C \cup \{x\}$ implies an element of B. Since $\rho(x) = i + 1$ and as Σ is ranked, every implication $A \to y \in \Sigma$ with x in its

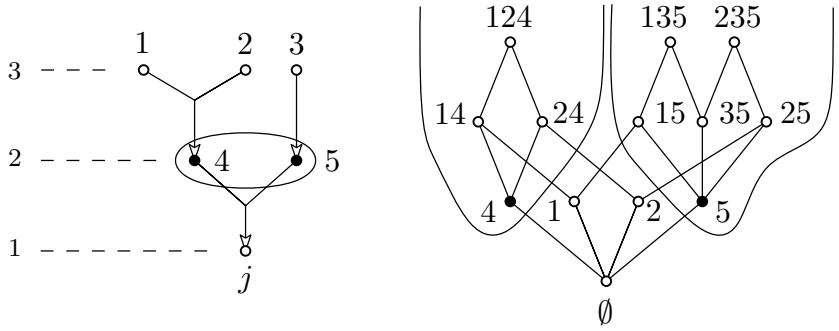


Figure 4.4: The situation of Theorem 4.4.3. On the left a ranked implicational base (X, Σ) and its rank function (dashed). On the right the set of closed sets of Σ not containing j, ordered by inclusion. Maximal independent sets of $\mathcal{E}(\mathcal{H}_j) = \{\{4,5\}\}$ are $\{4\}$ and $\{5\}$. They partition $j^{\nearrow} = j^{\nearrow} = \{\{1,2,4\}\} \uplus \{\{1,3,5\},\{2,3,5\}\}$.

premise has A of rank $\rho(S) = i+1$ and y of rank $\rho(B) = i$. As $\rho(y) = \rho(B)$ no such y belongs to the premise of an implication having $b \in B$ for conclusion. Hence there must be $A \to b \in \Sigma$ such that $x \in A$, $b \in B$ and consequently such that both $A \in \mathcal{E}(\mathcal{H}_B)$ and $A \subseteq S \cup \{x\}$. Therefore $S \cup \{x\}$ is no longer an independent set of $\mathcal{H}_B$. □

Lemma 4.4.5. *Let $B \subseteq X$ be a ranked set of rank $i < k$, and $\mathcal{H}_B$ be its associated hypergraph. Then to every $S \in MIS(\mathcal{H}_B)$ corresponds some $C \in B^{\nearrow}$ such that $C_{i+1} = S$.*

Proof. Consider $S \in MIS(\mathcal{H}_B)$. Recall that every implication having an element $b \in B$ for conclusion has its premise in $\mathcal{E}(\mathcal{H}_B)$. Furthermore since Σ is ranked, the closure of S does not contain any other elements than those of S at rank $i+1$. Consequently as S is an independent set of $\mathcal{H}_B$ it does not imply any element of B. Hence there exists $C \in B^{\nearrow}$ such that $S \subseteq C$. Now since S is maximal, adding any $x \in V(\mathcal{H}_B) \setminus S$ to S results into implying some $b \in B$. We conclude that $C_{i+1} = S$. □

The next lemma offers a recursive characterization of $B^{\nearrow}$ based on the partition defined in Theorem 4.4.3. Recall that since Σ is acyclic, no element $x \in X$ such that $\rho(x) \leq \rho(B)$, $x \notin B$ takes part in a minimal generator of an element of B. In particular, every such x belongs to all $C \in B^{\nearrow}$. For any ranked set S we put $\hat{S}$ to be the elements of rank $\rho(S)$ not in S, i.e., $\hat{S} = \{x \in X \mid \rho(x) = \rho(S),\ x \notin S\}$.

Lemma 4.4.6. *Let $B \subseteq X$ be a ranked set of rank $i < k$, and $\mathcal{H}_B$ be its associated hypergraph. Let $S \in MIS(\mathcal{H}_B)$. Then*

$$\{C \in B^{\nearrow},\ C_{i+1} = S\} = \{I \setminus B \mid I \in \hat{S}^{\nearrow}\}. \tag{4.5}$$

Proof. We show the first inclusion. Let $C \in B^{\nearrow}$ such that $C_{i+1} = S$. Then $\hat{S} \cap C = \emptyset$. Since Σ is acyclic, C contains every $x \in X$ such that $\rho(x) \leq i$ and $x \notin B$. Consequently, $I = C \cup B$ is closed. Since $\rho(B) = i$ and $\rho(\hat{S}) = i+1$, I does not imply any element of $\hat{S}$. We show that I is maximal with this property. Let $x \notin I$. Because $C \in B^{\nearrow}$, there exists $b \in B$ such that $C \cup \{x\}$ implies b. Hence $\phi(C \cup \{x\})$ must contain an edge of $\mathcal{H}_B$ (i.e., the premise of an implication $A \to b$, $b \in B$) and so does $\phi(I \cup \{x\})$. Since $C_{i+1} = S$ and as by hypothesis S is a maximal independent set of $\mathcal{H}_B$, either $x \in \hat{S}$ or $I \cup \{x\}$ implies s for some $s \in \hat{S}$, proving the first inclusion.

We show the other inclusion. Let $I \in \hat{S}^{\nearrow}$. Then $\hat{S} \cap I = \emptyset$. Since Σ is ranked, I contains every $x \in X$ such that $\rho(x) \leq i+1$ and $x \notin \hat{S}$. In particular $B, S \subseteq I$. Since S is an independent set of $\mathcal{H}_B$, $C = I \setminus B$ does not imply b, for any $b \in B$. Hence it is closed and $C_{i+1} = S$. We show that it is maximal with this property. Let $x \notin C$. Either $x \in I$ (and then $x \in B$) or $x \notin I$. Obviously, x cannot be added to C in the first case without intersecting B. Let us assume that $x \notin I$. Then $\rho(x) \geq \rho(\hat{S})$. Since $I \in \hat{S}^{\nearrow}$ and by maximality of I, either $x \in \hat{S}$ or there exists $s \in \hat{S}$ such that $I \cup \{x\}$ implies s, hence such that $C \cup \{x\}$ implies s. As by assumption S is a maximal independent set of $\mathcal{H}_B$, $C \cup \{x\}$ implies some $b \in B$ in both cases, concluding the proof. □

In what follows given $C \subseteq X$ we put $C_{>i} = \{x \in C \mid \rho(x) > i\}$. Since the elements of the left and right parts of Equation (4.5) only differ on B, and as $\rho(B) = i$, a corollary of Lemma 4.4.6 is the following. It is of interest as far as the proof of our algorithm is concerned.

Corollary 4.4.7. *Let $B \subseteq X$ be a ranked set of rank $i < k$, and $\mathcal{H}_B$ be its associated hypergraph. Let $S \in MIS(\mathcal{H}_B)$. Then*

$$\{C_{>i} \mid C \in B^{\nearrow},\ C_{i+1} = S\} = \{I_{>i} \mid I \in \hat{S}^{\nearrow}\}.$$

We now describe an algorithm which enumerates the meet-irreducible elements of $\mathcal{C}_\Sigma$ given (X, Σ) whenever it is ranked. The algorithm proceeds as follows. For every $j \in X$, it computes $j^{\nearrow} = j^{\nearrow}$ recursively rank by rank relying on the partition and the characterizations of Theorem 4.4.3, Lemma 4.4.6 and Corollary 4.4.7. Recall that since $\mathcal{C}_\Sigma$ is a convex geometry, $\{j^{\nearrow} \mid j \in X\}$ is a partition of $\mathcal{M}(\mathcal{C}_\Sigma)$, hence that every solution is obtained by such a procedure, without duplication. The computation of $j^{\nearrow}$ is described in Algorithm 4.1. The correctness of the whole procedure is proved in Theorem 4.4.8.

Algorithm 4.1: A recursive algorithm enumerating the set $B^{\nearrow}$ given a ranked implicational base (X, Σ) and a ranked set $B \subseteq X$.

```
k ← Max{ρ(x) | x ∈ X};
C ← {x ∈ X | ρ(x) ≤ ρ(B), x ∉ B};
RecENUM(Σ, B, C);
Procedure RecENUM(Σ, B, C)
    if ρ(B) = k then output C and return;
    for all S ∈ MIS(H_B) do
        Ŝ ← {x ∈ X | ρ(x) = ρ(S), x ∉ S};
        RecENUM(Σ, Ŝ, C ∪ S);
    end
```

Theorem 4.4.8. *Let $f, g : \mathbb{N} \to \mathbb{N}$ be two functions and $\mathcal{C}_\Sigma$ be the closure system of a given ranked implicational base (X, Σ). Then there is an algorithm enumerating the set $\mathcal{M}(\mathcal{C}_\Sigma)$ of meet-irreducible elements of $\mathcal{C}_\Sigma$ with delay*

$$O(|X| \cdot f(|X| + |\Sigma|)),$$

$O(|X| \cdot g(|X| + |\Sigma|))$ space, and after $O(|X| \cdot |\Sigma|)$ preprocessing time whenever there is one enumerating $MIS(\mathcal{H})$ with delay $f(|\mathcal{H}|)$ and space $g(|\mathcal{H}|)$ given any hypergraph $\mathcal{H}$.

Proof. Recall that by Proposition 4.1.3, there is an algorithm enumerating $\mathcal{M}(\mathcal{C}_\Sigma)$ with delay at most $2 \cdot f(|X|+|\Sigma|)$ whenever there is one enumerating $j^\nearrow$ with delay $f(|X|+|\Sigma|)$ for any $j \in X$. Hence, it is sufficient to show that there is an algorithm enumerating $j^{\nearrow\!\!\nearrow} = j^\nearrow$ with $O(|X| \cdot f(|X| + |\Sigma|))$ delay, $O(|X| \cdot g(|X| + |\Sigma|))$ space and after $O(|X| \cdot |\Sigma|)$ preprocessing time to prove the theorem. We shall show that Algorithm 4.1 correctly outputs this set and that it performs within these time and space bounds.

We show correctness. Assume without loss of generality that $\mathrm{Min}\{\rho(x) \mid x \in X\} = 0$ and let us put $k = \mathrm{Max}\{\rho(x) \mid x \in X\}$. Let $T[B, C]$ denote the recursive execution-tree of $\texttt{RecENUM}(\Sigma, B, C)$ for Σ, a ranked set B and $C \subseteq X$. We show by induction on the rank i of B and for every $C \subseteq X$ that the set $\{C \cup C^*_{>i} \mid C^* \in B^{\nearrow\!\!\nearrow}\}$ is output at the leaves of $T[B, C]$. Note that we aim here to prove both implications at the same time. Let us first consider the case $i = k$. By Proposition 4.4.2, $B^{\nearrow\!\!\nearrow} = \{C^*\}$ where $C^* = \{x \in X \mid \rho(x) \leq \rho(B),\ x \notin B\}$ and $C^*_{>i} = \emptyset$ in that case. Also C is output Line 5 of the algorithm. Hence the property holds for $i = k$, the tree $T[B, C]$ being reduced to a leaf. Let us assume that the property holds for every rank greater than i. We show that it holds for i. Consider B of rank i. Recall that by Theorem 4.4.3, $\{\{C^* \in B^{\nearrow\!\!\nearrow},\ C^*_{i+1} = S\} \mid S \in MIS(\mathcal{H}_B)\}$ is a partition of $B^{\nearrow\!\!\nearrow}$, and furthermore by Corollary 4.4.7,

$$\{C^*_{>i} \mid C^* \in B^{\nearrow\!\!\nearrow},\ C^*_{i+1} = S\} = \{\ I_{>i}\ \mid I \in \hat{S}^{\nearrow\!\!\nearrow}\}.$$

By inductive hypothesis since $\rho(\hat{S}) = i + 1$, the set $\{C \cup I_{>i+1} \mid I \in \hat{S}^{\nearrow\!\!\nearrow}\}$ is output by a call to $\texttt{RecENUM}(\Sigma, \hat{S}, C)$ for every $C \subseteq X$. Since $I_{i+1} = S$ whenever $I \in \hat{S}^{\nearrow\!\!\nearrow}$,

$$\begin{aligned}\{C \cup C^*_{>i} \mid C^* \in B^{\nearrow\!\!\nearrow},\ C^*_{i+1} = S\} &= \{C \cup S \cup I_{>i+1} \mid I \in \hat{S}^{\nearrow\!\!\nearrow}\} \\ &= \{\quad C \cup I_{>i} \quad \mid I \in \hat{S}^{\nearrow\!\!\nearrow}\}\end{aligned}$$

for every $C \subseteq X$. Hence by Theorem 4.4.3 the set $\{C \cup C^*_{>i} \mid C^* \in B^{\nearrow\!\!\nearrow}\}$ is output by a call to $\texttt{RecENUM}(\Sigma, \hat{S}, C \cup S)$ for every $S \in MIS(\mathcal{H}_B)$ Lines 6, 7 and 8. Consequently it is obtained at the leaves of $T[B, C]$. This concludes the induction. Now since every $C^* \in B^{\nearrow\!\!\nearrow}$ intersects the $\rho(B)$ first ranks on $C = \{x \in X \mid \rho(x) \leq \rho(B),\ x \notin B\}$, the set $B^{\nearrow\!\!\nearrow}$ is output by an initial call to $\texttt{RecENUM}(\Sigma, B, C)$.

Let us now consider the complexity of the algorithm. Observe that a rank function of (X, Σ) and an ordered sequence of the elements of X according to their rank can be computed in $O(|X| \cdot |\Sigma|)$ preprocessing time and $O(|X|^2)$ space, so that the computations of C and $\hat{S}$ Lines 2 and 7 can be achieved in $O(|X|)$ time. As for the computation of $\mathcal{H}_B$, it can be achieved in $O(|X| + |\Sigma|)$ time and space by indexing implications of Σ. By assumption since $|\mathcal{H}| \leq |X| + |\Sigma|$, every $S \in MIS(\mathcal{H}_B)$ can be enumerated with $f(|X|+|\Sigma|)$ delay and using $g(|X|+|\Sigma|)$ total space. Assuming that $f, g \in \Omega(|X|+|\Sigma|)$, the computation of $\mathcal{H}_B$ Line 6, $\hat{S}$ Line 7 is meaningless in term of time and space compared to that of $S \in MIS(\mathcal{H}_B)$. This also holds for the space needed by the preprocessing steps. Now, since the depth of the recursive tree is bounded by $|X|$, the time and space complexities follow. □

A trivial bound of $f(|\mathcal{H}| + |MIS(\mathcal{H})|)$ on the delay of an output-polynomial time algorithm enumerating $MIS(\mathcal{H})$ within the same time yields an output quasi-polynomial time algorithm enumerating the meet-irreducible elements in closure systems given by ranked implicational bases, using the algorithm of Fredman and Khachiyan [FK96] as a subroutine for the enumeration of $MIS(\mathcal{H})$. If moreover Σ is of bounded dimension then the described algorithm runs in output-polynomial time using the algorithm of Eiter and Gottlob [EG95]. It performs with polynomial delay whenever Σ is of dimension two using the algorithm of Tsukiyama [TIAS77] on the enumeration of maximal independent sets in graphs.

4.5 Constructing a ranked implicational base

In this section, we give an output quasi-polynomial time algorithm constructing the critical base of a ranked convex geometry given by the set of its meet-irreducible elements (SID). Recall by Theorem 4.3.6 that such an implicational base is unit-minimum, hence that it corresponds to the output expected by SID.

In what follows, let (X, ϕ) be a ranked convex geometry, $(X, \mathcal{C}_\phi)$ be its closure system and $\mathcal{M} = \mathcal{M}(\mathcal{C}_\phi)$ be the set of its meet-irreducible elements. Let (X, Σ) denote the critical base we are willing to compute. Then for every $j \in X$ we put

$$\mathrm{pred}(j) = \{a \in X \mid \exists A \to j \in \Sigma,\ a \in A\}.$$

In other words, $\mathrm{pred}(j)$ is the set of elements of X belonging to some critical minimal generator of j. Recall that since Σ is ranked, there is no minimal generator A of b such that $a \in A$ and $a, b \in \mathrm{pred}(j)$. Then, to $j \in X$ we associate the hypergraph $\mathcal{H}_j$ defined on vertex set $V(\mathcal{H}_j) = X \setminus \bigcap_{M_j \in j^\nearrow} M_j$ and edge set

$$\mathcal{E}(\mathcal{H}_j) = \{X \setminus M_j \mid M_j \in j^\nearrow\}.$$

It is well known that the minimal transversals of this hypergraph are exactly the minimal generators of j. See for instance [MR92, Wil94, BDVG18, HN18]. In the following we show that we can restrict this hypergraph so that its minimal transversals are exactly the critical minimal generators of j. The next lemma provides a characterization of $\mathrm{pred}(j)$ that depends on $\mathcal{M}$.

Lemma 4.5.1. *Let $j \in X$, $M_j \in j^\nearrow$ and $a \notin M_j$. Then $a \in pred(j)$ if and only if the set $M_j \cup \{a, j\}$ is closed.*

Proof. We prove the first implication. Consider $j \in X$, $M_j \in j^\nearrow$ and $a \notin M_j$. Note that one such M_j exists as by previous remarks a belongs to at least one minimal transversal of $\mathcal{H}_j$. Let us assume that $a \in \mathrm{pred}(j)$. Since Σ is ranked, for any x such that $\rho(x) \leq \rho(j)$, $x \neq j$ we have $x \in M_j$. This holds in particular for every element $y \in X$, $y \neq j$ such that there is a minimal generator A of y with $a \in A$ as $\rho(y) < \rho(a)$ and $\rho(a) = \rho(j) + 1$ in that case. Consequently, $M_j \cup \{a, j\}$ is closed.

We prove the other implication. Let $j \in X$, $M_j \in j^\nearrow$ and $a \notin M_j$ such that $M_j \cup \{a, j\}$ is closed. By definition M_j is closed, it does not contain j, and it is maximal with this property. In particular $M_j \cup \{a\}$ implies j, that is, $j \in \phi(M_j \cup \{a\})$. By Proposition 4.3.2

and Remark 4.3.3, there exists a critical minimal generator E of j such that $E \not\subseteq M_j$ and $E \subseteq \phi(M_j \cup \{a\}) = M_j \cup \{a, j\}$ as $M_j \cup \{a, j\}$ is closed by assumption. Then $a \in E$ and consequently $a \in \mathrm{pred}(j)$. □

Observe that the second part of the proof holds in the more general context of acyclic convex geometries. We are now ready to characterize the critical base of a ranked convex geometry. Given a hypergraph $\mathcal{H}$ and $S \subseteq V(\mathcal{H})$ we note $\mathcal{H}[S]$ the *hypergraph induced* by S defined by $V(\mathcal{H}[S]) = S$ and $\mathcal{E}(\mathcal{H}[S]) = \{E \cap S \mid E \in \mathcal{H}\}$.

Theorem 4.5.2. *Let (X, Σ) be the critical base of a ranked convex geometry. Let $j \in X$. Then A is a critical minimal generator of j if and only if $A \in Tr(\mathcal{H}_j[\mathrm{pred}(j)])$.*

Proof. We prove the first implication. Since A is a critical minimal generator of j, it is a minimal transversal of $\mathcal{H}_j$, hence of $\mathcal{H}_j[\mathrm{pred}(j)]$ as $A \subseteq \mathrm{pred}(j)$ by definition.

We prove the other implication. Let A be a minimal transversal of $\mathcal{H}_j[\mathrm{pred}(j)]$. Note that $E \cap \mathrm{pred}(j) \neq \emptyset$ for all $E \in \mathcal{E}(\mathcal{H}_j)$, as $\mathrm{pred}(j)$ is a transversal of $\mathcal{H}_j$. Hence A is a minimal transversal of $\mathcal{H}_j$. Assume for contradiction that A is a redundant minimal generator of j. Then by Corollary 4.3.4 there exists $a \in A$ such that $\phi(A) \setminus \{a, j\}$ implies j. That is, there is at least one critical minimal generator E of j, $a \notin E$ and $E \subseteq \phi(A) \setminus \{a\} \subseteq \phi(A)$. In particular $\rho(E) = \rho(j) + 1$. Furthermore since A is minimal generator of j, $E \not\subseteq \phi(A \setminus \{a\})$. Consequently, there is an element $e \in E$ which is implied by some $A' \subset A$ such that $a \in A'$. Hence a and e must have two different ranks. However since a belongs to $\mathrm{pred}(j)$, it belongs to a critical minimal generator of j and has rank $\rho(j) + 1$. This contradicts $\rho(a) \neq \rho(e)$, and the theorem follows. □

We are now ready to describe an algorithm which enumerates the critical base of a ranked convex geometry given by the set of its meet-irreducible elements. For every $j \in X$, it computes the set $\mathrm{pred}(j)$ of elements belonging to some critical minimal generator of j, and the complementary hypergraph $\mathcal{H}_j$ of meet-irreducible elements in $\nearrow$ relation with j. Then according to Theorem 4.5.2 it enumerates every minimal transversal of $\mathcal{H}_j[\mathrm{pred}(j)]$. A trace of this algorithm is given in Example 4.5.3. Its complexity analysis is given in Theorem 4.5.4.

Example 4.5.3. Consider the implicational base Σ of Figure 4.4. Then $j^{\nearrow} = \{\{1, 2, 4\}, \{1, 3, 5\}, \{2, 3, 5\}\}$. We aim to compute the critical minimal generators of j. Here, $\mathcal{E}(\mathcal{H}_j) = \{\{3, 5\}, \{2, 4\}, \{1, 4\}\}$. Minimal generators of j are $\{1, 2, 3\}, \{1, 2, 5\}$, $\{3, 4\}$ and $\{3, 5\}$. Observe that for 1, $\{2, 3, 5\} \cup \{1, j\}$ is not closed. The same goes for 2 with $\{1, 3, 5\}$, and 3 with $\{1, 2, 4\}$. On the other hand, $\{1, 2, 4\} \cup \{5, j\}$ is closed and $\{2, 3, 5\} \cup \{4, j\}$ too. Hence by Lemma 4.5.1, one has $\mathrm{pred}(j) = \{4, 5\}$, and $\mathcal{E}(\mathcal{H}_j[\mathrm{pred}(j)]) = \{\{4\}, \{5\}\}$. It has a unique minimal transversal which is $\{4, 5\}$. By Theorem 4.5.2, $\{4, 5\}$ is a critical minimal generator of j. Hence $45 \to j$ belongs to Σ. This indeed coincides with Σ.

Theorem 4.5.4. *Let $f, g : \mathbb{N} \to \mathbb{N}$ be two functions and (X, ϕ) be a ranked convex geometry of closure system $\mathcal{C}_\phi$ given by the set $\mathcal{M} = \mathcal{M}(\mathcal{C}_\phi)$ of its meet-irreducible elements. Then there is an algorithm enumerating the ranked implicational base (X, Σ) of critical minimal generators of (X, ϕ) with delay*

$$O(f(|X| + |\mathcal{M}|)),$$

$O(|X| \cdot |\mathcal{M}| + g(|X| + |\mathcal{M}|))$ *space, and after* $O(|X|^3 \cdot |\mathcal{M}|^2)$ *preprocessing time if there is one enumerating* $Tr(\mathcal{H})$ *with delay* $f(|\mathcal{H}|)$ *and space* $g(|\mathcal{H}|)$ *given any hypergraph* $\mathcal{H}$.

Proof. By Theorem 4.5.2, $A \subseteq X$ is a critical minimal generator of $j \in X$ if and only if it is a minimal transversal of the hypergraph $\mathcal{H}_j[\mathrm{pred}(j)]$. Note that the closure $\phi(S)$ of a set $S \subseteq X$ can be computed in $O(|X| \cdot |\mathcal{M}|)$ time by intersecting every $M \in \mathcal{M}$ such that $S \subseteq M$. Given $M \in \mathcal{M}$, computing $j \in X$ such that $M \in j^{\nearrow}$ can be done in $O(|\mathcal{M}| \cdot |X|^2)$ time by checking whether $M \cup \{j\}$ is closed for every $j \in X \setminus M$. Hence, computing the partition $\{j^{\nearrow} \mid j \in X\}$ of $\mathcal{M}$ can be done in $O(|\mathcal{M}|^2 \cdot |X|^2)$ time and $O(|X| \cdot |\mathcal{M}|)$ space. Using Lemma 4.5.1, the sets $\mathrm{pred}(j)$, $j \in X$ can be computed in $O(|X|^3 \cdot |\mathcal{M}|^2)$ time and $O(|X|^2)$ space by checking for every $j \in X$ and $a \in X$ whether there exists $M_j \in j^{\nearrow}$ such that $a \notin M_j$ and $M_j \cup \{a, j\}$ is closed. Note that $O(|X|^2)$ is bounded by $O(|X| \cdot |\mathcal{M}|)$ as $\{j^{\nearrow} \mid j \in X\}$ is a partition of $\mathcal{M}$. In addition, the hypergraphs $\mathcal{H}_j[\mathrm{pred}(j)]$, $j \in X$ can be computed in $O(|X|^2 \cdot |\mathcal{M}|)$ time and $O(|X| \cdot |\mathcal{M}|)$ space as

$$\sum_{j \in X} |\mathcal{E}(\mathcal{H}_j)| = |\mathcal{M}|.$$

Ultimately, this computation can be achieved in $O(|X|^3 \cdot |\mathcal{M}|^2)$ preprocessing time and using $O(|X| \cdot |\mathcal{M}|)$ space.

Now by assumption and since $|\mathcal{H}_j[\mathrm{pred}(j)]| \leq |X| + |\mathcal{M}|$, the critical minimal generators of $j \in X$ can be enumerated with $f(|X| + |\mathcal{M}|)$ delay and using $g(|X| + |\mathcal{M}|)$ space. The theorem follows. □

A trivial bound of $f(|\mathcal{H}| + |Tr(\mathcal{H})|)$ on the delay of an output-polynomial time algorithm enumerating $Tr(\mathcal{H})$ within the same time yields an output quasi-polynomial time algorithm constructing the ranked implicational base of a ranked convex geometry given by the set of its meet-irreducible elements, using the algorithm of Fredman and Khachiyan [FK96] as a subroutine for the enumeration of $Tr(\mathcal{H})$. If furthermore every meet-irreducible element $M \in \mathcal{M}(\mathcal{C}_\phi)$ is such that $|M| \geq |X| - k$ for some fixed integer $k \in \mathbb{N}$, then the described algorithm performs in output-polynomial time using the algorithm of Eiter and Gottlob in [EG95].

4.6 Further work and open problems

Several questions arise for future research. First, we would like to point that the partition given by Theorem 4.4.3 holds for more general closure spaces than ranked convex geometries. One such convex geometry is given in Example 4.6.1 and leads to the question that follows.

Example 4.6.1. We consider the implicational base $\Sigma = \{1 \to 2, 1 \to 3, 2 \to 4, 34 \to 5\}$. Despite the fact that Σ is not ranked, Theorem 4.4.3 holds. Hence, an algorithm in the fashion of Algorithm 4.1 correctly outputs the meet-irreducible elements in that case.

Question 4.6.2. *For which closure spaces does Theorem 4.4.3 hold?*

We recall however that by Theorem 4.2.1 and Equality (4.1) the enumeration of $j^{\nearrow}$ for some $j \in X$ in acyclic convex geometries is harder than the dualization in lat-

tices given by acyclic implicational bases. Hence the existence of an output quasi-polynomial time algorithm for acyclic convex geometries using the techniques presented in this chapter is a challenging question.

The same question holds concerning the characterization of Lemma 4.5.1. Indeed, it is easily seen that the algorithm of Theorem 4.5.4 will correctly output the critical base of the convex geometry given in Example 4.6.1. This yields the next question.

Question 4.6.3. *For which closure spaces does Theorem 4.5.2 hold?*

At last, another intriguing question, raised by Section 4.3, is the following.

Question 4.6.4. *Can ranked convex geometries be identified in polynomial time from their meet-irreducible elements?*

As shown in Proposition 4.3.8, this problem lies in coNP. We believe that the answer to this question is positive for acyclic convex geometries, using an elimination scheme in the *colored poset* introduced by Habib and Nourine in [Nou00, HN18] as a variant of the poset of irreducibles. Identifying the ranks in such a poset, however, seems to be more intricate.

Conclusion

In this thesis, we investigated the complexity of the dualization of monotone Boolean functions, and its generalizations, through the many shapes the problem takes on graphs, hypergraphs, and lattice: minimal dominating sets enumeration, minimal transversals enumeration, lattice dualization, and meet-irreducible enumeration. Several techniques—among them the flipping method, flashlight search, and ordered generation—were confronted to the problem, and new output polynomial-time algorithms were obtained under various restrictions (among them, most notably, Theorems 2.2.13, 2.3.10, 2.4.4, Theorem 3.3.7, and Theorems 4.5.4 and 4.4.8). In addition to these positive results, we exhibited some intractability results when the problem is generalized to the dualization of any lattice (Theorem 3.2.1), or to the translation between the representations of an acyclic convex geometry (Theorem 4.2.1).

Several open problems were stated in the conclusions of Chapters 2, 3 and 4. We reformulate here one question that seems to be of greatest importance, for each of these three chapters.

Question 4.6.5. *Can* DOM-ENUM *be solved in output polynomial-time in C_4-free, comparability, and unit-disk graphs?*

Question 4.6.6. *Can* DUAL-ENUM *be solved in output quasi-polynomial time in distributive lattices*[2]*, and more generally, in acyclic convex geometries?*

Question 4.6.7. *Can* CCM *and* SID *be solved in output quasi-polynomial time in acyclic convex geometries, and more generally, in any convex geometry?*

In addition, we recall that Question 4.6.5 is part of a more general, and much harder question, which is now open for more that forty years [EG95, EMG08].

Question 4.6.8. *Can* HYPERGRAPH DUAL *(or equivalently,* DUALIZATION*) be solved in polynomial time?*

As for Question 4.6.7, we point that while proved for DUAL-ENUM, it is not known whether the general case of CCM (or SID) is intractable, i.e., cannot be solved in output-polynomial time unless P=NP. This, also, constitutes a long-standing open question in the field [Kha95, Wil17].

[2]We would like to mention that the special case of distributive lattices has lately been positively answered by Elbassioni during the rewiewing process of this manuscript [Elb20].

Publications

The results presented in this thesis led to several submissions, and to publications in international journals and conferences. The algorithms for K_t-free graphs and other related graph classes, given in the first part of Chapter 2, were presented at the STACS 2019 conference [BDHR19], and appeared in the journal ACM Transactions on Algorithms [BDH+20]. The algorithms for comparability and incomparability graphs presented in the second part of Chapter 2 were recently submitted, and are available—as every result presented in this manuscript—on arXiv [BDMN20]. The results of Chapter 3 on the dualization in lattices given by implicational bases were presented at the ICFCA 2019 conference [DN19a], and appeared in the journal Theoretical Computer Science [DN20]. Finally, the results presented in Chapter 3 on translating between the representations of a ranked convex geometry were recently submitted, and may be accessed in [DNV19].

In addition to these publications, two contributions related to the dualization are to be indexed:

- In a first contribution, we investigated how known techniques based on neighborhood inclusions for DOM-ENUM (such as used in [KLMN14] for split graphs) could be pushed for more general graphs. We in particular obtained linear and polynomial-delay algorithms in P_7-free and P_8-free chordal graphs, and showed the technique to be inefficient in P_k-free graphs, $k \geq 9$. These results were presented at the ISAAC 2019 conference [DN19b].

- In a second contribution, we considered the dualization problem in distributive lattices, and gave equivalent formulations of the problem in term of graphs, hypergraphs, and posets. In the new framework, a poset on vertices is given together with the input (hyper)graph, and minimal "ideal solutions" are to be generated. These formulations allowed us to study the complexity of the problem under various combined restrictions on graph classes and poset types, including bipartite, split, and co-bipartite graphs, and variants of neighborhood inclusion posets. We for example show that while the enumeration of minimal dominating sets is possible with linear delay in split graphs, the problem, within the same class, gets as hard as for general graphs when generalized to this framework. Same behaviors are observed for bipartite and co-bipartite graphs, even under high restrictions—related to neighborhood inclusions—on the poset. These results may be found in [DNU19] and are to appear in the journal Discrete Applied Mathematics.

At last, two other results were obtained, thanks to different collaborations. These results fall in combinatorics and do not concern enumeration.

- In a first contribution, we revisited a theorem by Folkman on graph coloring, stating that the chromatic number of any graph is at most 2 plus the maximum over all subgraphs of the difference between the number of vertices, and twice the independence number. This appeared in the Electronic Journal of Combinatorics [BCD+20].

- In a second contribution, we considered generalizations of simplicial vertices in graphs, known as avoidable vertices and paths. We answered a question of Beisegel et al. on the existence of an avoidable path of size k in every graph containing a path on k elements, implying a result of Chvátal et al. from 2002. This may be accessed in [BDHT19] and is to appear in the Electronic Journal of Combinatorics.

Bibliography

[AF96] David Avis and Komei Fukuda. Reverse search for enumeration. *Discrete Applied Mathematics*, 65(1-3):21–46, 1996. Introduction

[AGT03] Kira V. Adaricheva, Viktor A. Gorbunov, and V. I. Tumanov. Join-semidistributive lattices and convex geometries. *Advances in Mathematics*, 173(1):1–49, 2003. 4.1.5

[AIBT14] Kira V. Adaricheva, Giuseppe F. Italiano, Hans Kleine Büning, and György Turán. Horn formulas, directed hypergraphs, lattices and closure systems: related formalisms and applications (Dagstuhl Seminar 14201). *Dagstuhl Reports*, 4(5):1–26, 2014. 4

[AKH16] Faisal N. Abu-Khzam and Pinar Heggernes. Enumerating minimal dominating sets in chordal graphs. *Information Processing Letters*, 116(12):739–743, 2016. 1.2.1

[Akk73] Eralp Abdurrahim Akkoyunlu. The enumeration of maximal cliques of large graphs. *SIAM Journal on Computing*, 2(1):1–6, 1973. Introduction

[AN16] Kira V. Adaricheva and James B. Nation. *Bases of Closure Systems*, pages 181–213. Springer International Publishing, Cham, 2016. 4.1

[AN17] Kira V. Adaricheva and James B. Nation. Discovery of the D-basis in binary tables based on hypergraph dualization. *Theoretical Computer Science*, 658:307–315, 2017. 1.3.1, 4, 4

[AN19] Kira V. Adaricheva and Taylor Ninesling. Direct and binary direct bases for one-set updates of a closure system. In *International Conference on Formal Concept Analysis*, pages 55–72. Springer, 2019. 1.3.1

[Bag09] Guillaume Bagan. *Algorithms and complexity of enumeration problems for the evaluation of logical queries*. PhD thesis, Université de Caen Normandie, 2009. Introduction, 1, 1.1.3

[BCD+20] Marthe Bonamy, Pierre Charbit, Oscar Defrain, Gwénaël Joret, Aurélie Lagoutte, Vincent Limouzy, Lucas Pastor, and Jean-Sébastien Sereni. Revisiting a theorem by Folkman on graph colouring. *The Electronic Journal of Combinatorics*, 27(P1.56), 2020. Conclusion

[BČKK09] Endre Boros, Ondřej Čepek, Alexander Kogan, and Petr Kučera. A subclass of Horn CNFs optimally compressible in polynomial time. *Annals of Mathematics and Artificial Intelligence*, 57(3-4):249–291, 2009. 3.4

[BDH+20] Marthe Bonamy, Oscar Defrain, Marc Heinrich, Michał Pilipczuk, and Jean-Florent Raymond. Enumerating minimal dominating sets in K_t-free and variants. *ACM Transactions on Algorithms*, 16(3):1–23, 2020. 2, Conclusion

[BDHR19] Marthe Bonamy, Oscar Defrain, Marc Heinrich, and Jean-Florent Raymond. Enumerating minimal dominating sets in triangle-free graphs. In *36th International Symposium on Theoretical Aspects of Computer Science*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019. 2, Conclusion

[BDHT19] Marthe Bonamy, Oscar Defrain, Meike Hatzel, and Jocelyn Thiebaut. Avoidable paths in graphs. *arXiv preprint arXiv:1908.03788*, 2019. Conclusion

[BDMN20] Marthe Bonamy, Oscar Defrain, Piotr Micek, and Lhouari Nourine. Enumerating minimal dominating sets in the (in)comparability graphs of bounded dimension posets. *arXiv preprint arXiv:2004.07214*, 2020. 2, Conclusion

[BDVG18] Karell Bertet, Christophe Demko, Jean-François Viaud, and Clément Guérin. Lattices, closures systems and implication bases: a survey of structural aspects and algorithms. *Theoretical Computer Science*, 743:93–109, 2018. 3.1.3, 4, 4.5

[BEGK04] Endre Boros, Khaled Elbassioni, Vladimir Gurvich, and Leonid Khachiyan. Generating maximal independent sets for hypergraphs with bounded edge-intersections. In *Latin American Symposium on Theoretical Informatics*, pages 488–498. Springer, 2004. 1.3.2, 2

[Ber84] Claude Berge. *Hypergraphs: combinatorics of finite sets*, volume 45. Elsevier, 1984. 1.3.2, 2.3.3

[BI95] Jan C. Bioch and Toshihide Ibaraki. Complexity of identification and dualization of positive Boolean functions. *Information and Computation*, 123(1):50–63, 1995. 1.3.1

[Bir37] Garrett Birkhoff. Rings of sets. *Duke Mathematical Journal*, 3(3):443–454, 1937. 3.1.3

[Bir40] Garrett Birkhoff. *Lattice theory*, volume 25. American Mathematical Soc., 1940. 3.1.2

[BK13] Mikhail A. Babin and Sergei O. Kuznetsov. Computing premises of a minimal cover of functional dependencies is intractable. *Discrete Applied Mathematics*, 161(6):742–749, 2013. 4, 4

[BK17] Mikhail A. Babin and Sergei O. Kuznetsov. Dualization in lattices given by ordered sets of irreducibles. *Theoretical Computer Science*, 658:316–326, 2017. 3, 3.1.4, 3.1.4, 3.1.4, 3.2, 3.4, 4, 4, 4.1.4, 4.2, 4.2, 4.2

[BKPS19] Endre Boros, Benny Kimelfeld, Reinhard Pichler, and Nicole Schweikardt. Enumeration in Data Management (Dagstuhl Seminar 19211). *Dagstuhl Reports*, 9(5):89–109, 2019. Introduction

[BM70] Marc Barbut and Bernard Monjardet. *Ordre et classification, Algèbre et combinatoire*, volume 1 & 2. Hachette, Paris, 1970. 4

[BM10] Karell Bertet and Bernard Monjardet. The multiple facets of the canonical direct unit implicational basis. *Theoretical Computer Science*, 411(22-24):2155–2166, 2010. 4

[BMN17] Laurent Beaudou, Arnaud Mary, and Lhouari Nourine. Algorithms for k-meet-semidistributive lattices. *Theoretical Computer Science*, 658:391–398, 2017. Introduction, 1.3.1, 4, 4, 4, 4.1.4, 4.2

[BO14] Konstantin Bazhanov and Sergei Obiedkov. Optimizations in computing the Duquenne–Guigues basis of implications. *Annals of mathematics and artificial intelligence*, 70(1-2):5–24, 2014. 3.1.3

[CGK+19] Alessio Conte, Roberto Grossi, Mamadou M. Kanté, Andrea Marino, Takeaki Uno, and Kunihiro Wasa. Listing induced steiner subgraphs as a compact way to discover steiner trees in graphs. In *44th International Symposium on Mathematical Foundations of Computer Science*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019. 1.3.1

[CH11] Yves Crama and Peter L. Hammer. *Boolean functions: Theory, algorithms, and applications*. Cambridge University Press, 2011. 4

[CHvHK13] Jean-François Couturier, Pinar Heggernes, Pim van't Hof, and Dieter Kratsch. Minimal dominating sets in graph classes: combinatorial bounds and enumeration. *Theoretical Computer Science*, 487:82–94, 2013. 1.2.1

[CKMU19] Alessio Conte, Mamadou M. Kanté, Andrea Marino, and Takeaki Uno. Maximal irredundant set enumeration in bounded-degeneracy and bounded-degree hypergraphs. In *International Workshop on Combinatorial Algorithms*, pages 148–159. Springer, 2019. 1.3.1

[CKP+19] Nadia Creignou, Markus Kröll, Reinhard Pichler, Sebastian Skritek, and Heribert Vollmer. A complexity theory for hard enumeration problems. *Discret. Appl. Math.*, 268:191–209, 2019. Introduction, Introduction, 1.2.3

[CMG+19] Alessio Conte, Andrea Marino, Roberto Grossi, Takeaki Uno, and Luca Versari. Proximity search for maximal subgraph enumeration. *arXiv preprint arXiv:1912.13446*, 2019. Introduction

[Cou09] Bruno Courcelle. Linear delay enumeration and monadic second-order logic. *Discrete Applied Mathematics*, 157(12):2675–2700, 2009. 2

[CS18] Florent Capelli and Yann Strozecki. Enumerating models of DNF faster: breaking the dependency on the formula size. *arXiv preprint arXiv:1810.04006*, 2018. 2.4.2

[CU19] Alessio Conte and Takeaki Uno. New polynomial delay bounds for maximal subgraph enumeration by proximity search. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2019, pages 1179—-1190, New York, NY, USA, 2019. Association for Computing Machinery. Introduction, 1.1.3

[Dam06] Peter Damaschke. Parameterized enumeration, transversals, and imperfect phylogeny reconstruction. *Theoretical Computer Science*, 351(3):337–350, 2006. Introduction

[DF12] Jean-Paul Doignon and Jean-Claude Falmagne. *Knowledge spaces*. Springer Science & Business Media, 2012. 4

[Die87] Brenda L. Dietrich. A circuit set characterization of antimatroids. *Journal of Combinatorial Theory, Series B*, 43(3):314–321, 1987. 4.3, 4.3

[DM41] Ben Dushnik and Edwin W. Miller. Partially ordered sets. *American journal of mathematics*, 63(3):600–610, 1941. 2.1.6

[DMP99] Carlos Domingo, Nina Mishra, and Leonard Pitt. Efficient read-restricted monotone CNF/DNF dualization by learning with membership queries. *Machine learning*, 37(1):89–110, 1999. 1.3.1

[DN19a] Oscar Defrain and Lhouari Nourine. Dualization in lattices given by implicational bases. In *International Conference on Formal Concept Analysis*, pages 89–98. Springer, 2019. 3, Conclusion

[DN19b] Oscar Defrain and Lhouari Nourine. Neighborhood inclusions for minimal dominating sets enumeration: Linear and polynomial delay algorithms in P_7-free and P_8-free chordal graphs. In *30th International Symposium on Algorithms and Computation*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2019. Conclusion

[DN20] Oscar Defrain and Lhouari Nourine. Dualization in lattices given by implicational bases. *Theoretical Computer Science*, 814:169–176, 2020. 3, Conclusion

[DNU19] Oscar Defrain, Lhouari Nourine, and Takeaki Uno. On the dualization in distributive lattices and related problems. *arXiv preprint arXiv:1902.07004, to appear in Discrete Applied Mathematics*, 2019. Conclusion

[DNV19] Oscar Defrain, Lhouari Nourine, and Simon Vilmin. Translating between the representations of a ranked convex geometry. *arXiv preprint arXiv:1907.09433*, 2019. 4, Conclusion

[DP02] Brian A. Davey and Hilary A. Priestley. *Introduction to lattices and order*. Cambridge university press, 2002. 3.1.2, 3.1.3, 4

[EG95] Thomas Eiter and Georg Gottlob. Identifying the minimal transversals of a hypergraph and related problems. *SIAM Journal on Computing*, 24(6):1278–1304, 1995. Introduction, Introduction, Introduction, 1.3.1, 1.3.2, 4.4, 4.5, Conclusion

[EGM03] Thomas Eiter, Georg Gottlob, and Kazuhisa Makino. New results on monotone dualization and generating hypergraph transversals. *SIAM Journal on Computing*, 32(2):514–537, 2003. 1.3.1, 1.3.1, 1.3.2, 1.3.2, 2, 2, 2.2.1

[EJ85] Paul H. Edelman and Robert E. Jamison. The theory of convex geometries. *Geometriae dedicata*, 19(3):247–270, 1985. 4

[Elb09] Khaled M. Elbassioni. Algorithms for dualization over products of partially ordered sets. *SIAM Journal on Discrete Mathematics*, 23(1):487–510, 2009. 3.1.4, 4.3

[Elb20] Khaled Elbassioni. On dualization over distributive lattices. *arXiv preprint arXiv:2006.15337*, 2020. 2

[EMG08] Thomas Eiter, Kazuhisa Makino, and Georg Gottlob. Computational aspects of monotone dualization: A brief survey. *Discrete Applied Mathematics*, 156(11):2035–2049, 2008. Introduction, Introduction, 1.3.1, 1.3.2, Conclusion

[FGPS08] Fedor V. Fomin, Fabrizio Grandoni, Artem V. Pyatkin, and Alexey A. Stepanov. Combinatorial bounds via measure and conquer: Bounding minimal dominating sets and applications. *ACM Transactions on Algorithms*, 5(1):9, 2008. 1.2.1

[FGS19] Henning Fernau, Petr A. Golovach, and Marie-France Sagot. Algorithmic Enumeration: Output-sensitive, Input-Sensitive, Parameterized, Approximative (Dagstuhl Seminar 18421). *Dagstuhl Reports*, 8(10):63–86, 2019. Introduction

[Fis70] Peter C. Fishburn. Intransitive indifference with unequal indifference intervals. *Journal of Mathematical Psychology*, 7(1):144–149, 1970. 3.3

[FJN95] Ralph Freese, Jaroslav Ježek, and James B. Nation. *Free lattices*, volume 42. American Mathematical Soc., 1995. 4

[FK96] Michael L. Fredman and Leonid Khachiyan. On the complexity of dualization of monotone disjunctive normal forms. *Journal of Algorithms*, 21(3):618–628, 1996. Introduction, 1.3.1, 1.3.1, 1.3.1, 3.1.4, 3.3, 3.3, 4.4, 4.5

[FLM97] Komei Fukuda, Thomas M. Liebling, and François Margot. Analysis of backtrack algorithms for listing all vertices and all faces of a convex polyhedron. *Computational Geometry*, 8(1):1–12, 1997. 2.2.1

[GD86] Jean-Louis Guigues and Vincent Duquenne. Familles minimales d'implications informatives résultant d'un tableau de données binaires. *Mathématiques et Sciences humaines*, 95:5–18, 1986. 4

[GHK+16] Petr A. Golovach, Pinar Heggernes, Mamadou M. Kanté, Dieter Kratsch, and Yngve Villanger. Enumerating minimal dominating sets in chordal bipartite graphs. *Discrete Applied Mathematics*, 199:30–36, 2016. 2, 2.3, 2.3.1, 2.3.1, 2.3.2, 2.3.2, 2.5.1

[GHK+18] Petr A. Golovach, Pinar Heggernes, Mamadou M. Kanté, Dieter Kratsch, Sigve H. Sæther, and Yngve Villanger. Output-polynomial enumeration on graphs of bounded (local) linear MIM-width. *Algorithmica*, 80(2):714–741, 2018. 2, 2, 2.3, 2.3.1, 2.3.1, 2.3.2, 2.3.3

[GHKV15] Petr A. Golovach, Pinar Heggernes, Dieter Kratsch, and Yngve Villanger. An incremental polynomial time algorithm to enumerate all minimal edge dominating sets. *Algorithmica*, 72(3):836–859, 2015. 1.1.3, 2.1.3, 2.3, 2.3.1, 2.3.1, 2.3.1, 2.3.2, 2.3.1, 2.3.2

[GJ79] Michael R Garey and David S Johnson. *Computers and intractability*, volume 174. freeman San Francisco, 1979. 1.2, 3.2

[GK07] Joshua A. Grochow and Manolis Kellis. Network motif discovery using subgraph enumeration and symmetry-breaking. In *Annual International Conference on Research in Computational Molecular Biology*, pages 92–106. Springer, 2007. Introduction

[GMKT97] Dimitrios Gunopulos, Heikki Mannila, Roni Khardon, and Hannu Toivonen. Data mining, hypergraph transversals, and machine learning. In *PODS*, pages 209–216. ACM, 1997. Introduction, Introduction, 1.3.1

[Grä11] George Grätzer. *Lattice theory: foundation*. Springer Science & Business Media, 2011. 3.1.2

[GRU83] Martin Charles Golumbic, Doron Rotem, and Jorge Urrutia. Comparability graphs and intersection graphs. *Discrete Mathematics*, 43(1):37–46, 1983. 2, 2.4.1

[GW86] Bernhard Ganter and Rudolf Wille. Implikationen und Abhängigkeiten zwischen Merkmalen. *Die Klassifikation und ihr Umfeld, Indeks-Verlag, Frankfurt*, 1986. 4

[GW12] Bernhard Ganter and Rudolf Wille. *Formal concept analysis: mathematical foundations*. Springer Science & Business Media, 2012. 4, 4, 4.1.1

[HK93] Peter L. Hammer and Alexander Kogan. Optimal compression of propositional Horn knowledge bases: complexity and approximation. *Artificial Intelligence*, 64(1):131–145, 1993. 4.1.4

[HK95] Peter L. Hammer and Alexander Kogan. Quasi-acyclic propositional Horn knowledge bases: Optimal compression. *IEEE Transactions on knowledge and data engineering*, 7(5):751–762, 1995. 3.4, 4, 4.3, 4.3.1, 4.3

[HMNS01] Michel Habib, Raoul Medina, Lhouari Nourine, and George Steiner. Efficient algorithms on distributive lattices. *Discrete Applied Mathematics*, 110(2-3):169–187, 2001. Introduction

[HN18] Michel Habib and Lhouari Nourine. Representation of lattices via set-colored posets. *Discrete Applied Mathematics*, 249:64–73, 2018. Introduction, 4, 4, 4.5, 4.6

[JYP88] David S. Johnson, Mihalis Yannakakis, and Christos H. Papadimitriou. On generating all maximal independent sets. *Information Processing Letters*, 27(3):119–123, 1988. Introduction, Introduction, 1.2.2, 1.2.3, 2.1.3

[KBEG07] Leonid Khachiyan, Endre Boros, Khaled Elbassioni, and Vladimir Gurvich. On the dualization of hypergraphs with bounded edge-intersections and other related classes of hypergraphs. *Theoretical Computer Science*, 382(2):139–150, 2007. 1.3.2, 2.3.3, 2.3.7, 2.3.3

[Kha95] Roni Khardon. Translating between Horn representations and their characteristic models. *Journal of Artificial Intelligence Research*, 3:349–372, 1995. Introduction, 4, 4, 4, 4, 4, 4.1.4, 4.1.4, 4.1.4, 4.1.4, 4.2, Conclusion

[KKP18] Mamadou M. Kanté, Kaveh Khoshkhah, and Mozhgan Pourmoradnasseri. Enumerating minimal transversals of hypergraphs without small holes. In *43rd International Symposium on Mathematical Foundations of Computer Science*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018. 1.3.2, 2, 2.3

[KKS93] Henry A. Kautz, Michael J. Kearns, and Bart Selman. Reasoning with characteristic models. In *AAAI*, volume 93, pages 34–39. Citeseer, 1993. 4, 4

[KLM+13] Mamadou M. Kanté, Vincent Limouzy, Arnaud Mary, Lhouari Nourine, and Takeaki Uno. On the enumeration and counting of minimal dominating sets in interval and permutation graphs. In *International Symposium on Algorithms and Computation*, pages 339–349. Springer, 2013. 2

[KLM+15a] Mamadou M. Kanté, Vincent Limouzy, Arnaud Mary, Lhouari Nourine, and Takeaki Uno. A polynomial delay algorithm for enumerating minimal dominating sets in chordal graphs. In *International Workshop on Graph-Theoretic Concepts in Computer Science*, pages 138–153. Springer, 2015. 2, 2.2.5

[KLM+15b] Mamadou M. Kanté, Vincent Limouzy, Arnaud Mary, Lhouari Nourine, and Takeaki Uno. Polynomial delay algorithm for listing minimal edge dominating sets in graphs. In *Workshop on Algorithms and Data Structures*, pages 446–457. Springer, 2015. 2

[KLMN11] Mamadou M. Kanté, Vincent Limouzy, Arnaud Mary, and Lhouari Nourine. Enumeration of minimal dominating sets and variants. In *International Symposium on Fundamentals of Computation Theory*, pages 298–309. Springer, 2011. 2.2.5

[KLMN12] Mamadou M. Kanté, Vincent Limouzy, Arnaud Mary, and Lhouari Nourine. On the neighbourhood helly of some graph classes and applications to the enumeration of minimal dominating sets. In *International Symposium on Algorithms and Computation*, pages 289–298. Springer, 2012. 2

[KLMN14] Mamadou M. Kanté, Vincent Limouzy, Arnaud Mary, and Lhouari Nourine. On the enumeration of minimal dominating sets and related notions. *SIAM Journal on Discrete Mathematics*, 28(4):1916–1929, 2014. Introduction, 1.3.1, 2, 2, 2, 2, 2, 2.1.2, 2.1.1, 2.2.6, Conclusion

[KLS12] Bernhard Korte, László Lovász, and Rainer Schrader. *Greedoids*, volume 4. Springer Science & Business Media, 2012. 4.3

[KN14] Mamadou M. Kanté and Lhouari Nourine. Minimal dominating set enumeration. In Ming-Yang Kao, editor, *Encyclopedia of Algorithms*, pages 1–5. Springer US, Boston, MA, 2014. 2, 2.5.1

[KSS00] Dimitris J. Kavvadias, Martha Sideri, and Elias C. Stavropoulos. Generating all maximal models of a Boolean expression. *Information Processing Letters*, 74(3-4):157–162, 2000. 3.2, 3.2, 4.1.4

[Kuz04] Sergei O. Kuznetsov. Complexity of learning in concept lattices from positive and negative examples. *Discrete Applied Mathematics*, 142(1-3):111–125, 2004. 3

[Mai83] David Maier. *Theory of relational databases*. Computer Science Pr, 1983. 4

[Mar64] Mitchell P. Marcus. Derivation of maximal compatibles using Boolean algebra. *IBM Journal of Research and Development*, 8(5):537–538, 1964. Introduction

[Mar75] George Markowsky. The factorization and representation of lattices. *Transactions of the American Mathematical Society*, 203:185–200, 1975. 4

[Mar13] Arnaud Mary. *Énumération des dominants minimaux d'un graphe*. PhD thesis, Université Blaise Pascal, 2013. Introduction, 1, 1.2.7

[Mar15] Andrea Marino. *Analysis and enumeration: algorithms for biological graphs*, volume 6. Springer, 2015. Introduction, 1

[MM65] John W. Moon and Leo Moser. On cliques in graphs. *Israel journal of Mathematics*, 3(1):23–28, 1965. 1.2.1

[MR92] Heikki Mannila and Kari-Jouko Räihä. *The design of relational databases*. Addison-Wesley Longman Publishing Co., Inc., 1992. 1.3.1, 4, 4, 4.1.1, 4.5

[MS19] Arnaud Mary and Yann Strozecki. Efficient enumeration of solutions produced by closure operations. *Discret. Math. Theor. Comput. Sci.*, 21(3), 2019. Introduction, 1.3.1, 2.2.1, 2.4.2

[MU04] Kazuhisa Makino and Takeaki Uno. New algorithms for enumerating all maximal cliques. In *Scandinavian workshop on algorithm theory*, pages 260–272. Springer, 2004. 1.2.2, 2.1.3

[NK13] Masataka Nakamura and Kenji Kashiwabara. The prime stems of rooted circuits of closure spaces and minimum implicational bases. *The Electronic Journal of Combinatorics*, 20(1):P22, 2013. 4.3

[Nou00] Lhouari Nourine. Colored posets and upper locally distributive lattices. Technical report, Technical Report 00060, LIRMM, 2000. 4.6

[NP12] Lhouari Nourine and Jean-Marc Petit. Extending set-based dualization: Application to pattern mining. In *Proceedings of the 20th European Conference on Artificial Intelligence*, pages 630–635. IOS Press, 2012. Introduction, 3

[NP14] Lhouari Nourine and Jean-Marc Petit. Dualization on partially ordered sets: Preliminary results. In *International Workshop on Information Search, Integration, and Personalization*, pages 23–34. Springer, 2014. 1.3.1, 3.3

[NP16] Lhouari Nourine and Jean-Marc Petit. *Beyond Hypergraph Dualization*, pages 189–192. Springer New York, New York, NY, 2016. 3.1.1

[NR99] Lhouari Nourine and Olivier Raynaud. A fast algorithm for building lattices. *Information processing letters*, 71(5-6):199–204, 1999. Introduction

[PU59] Marvin C. Paull and Stephen H. Unger. Minimizing the number of states in incompletely specified sequential switching functions. *IRE Transactions on Electronic Computers*, (3):356–367, 1959. Introduction

[Ray19] Jean-Florent Raymond. `minimal_dominating_sets`, an implementation of Bonamy et al.'s algorithm for enumerating minimal dominating sets in K_t-free graphs. https://git.sagemath.org/sage.git/commit?id=906cf147fe64ceed73d30fabf61155f65393bd67, 2019. Accessed: 2020-March-02. To be included in SageMath 9.1 http://www.sagemath.org. 2.5.1

[RT75] Ronald C. Read and Robert E. Tarjan. Bounds on backtrack algorithms for listing cycles, paths, and spanning trees. *Networks*, 5(3):237–252, 1975. Introduction, 2.2.1

[Sho86] Robert C. Shock. Computing the minimum cover of functional dependencies. *Information Processing Letters*, 22(3):157–159, 1986. 4, 4.1.4

[SL90] Bart Selman and Hector J. Levesque. Abductive and default reasoning: A computational core. 1990. 4

[Ste99] Manfred Stern. *Semimodular lattices: theory and applications*, volume 73. Cambridge University Press, 1999. 4.1.5

[Str10] Yann Strozecki. *Enumeration complexity and matroid decomposition*. PhD thesis, Université Paris 7, 2010. Introduction, 1, 1.1.3

[Str19] Yann Strozecki. Enumeration complexity. *Bulletin of EATCS*, 1(129), 2019. Introduction, Introduction, 1, 1.1.2, 1.1.2, 1.1.3, 1.2.3

[Tar73] Robert E. Tarjan. Enumeration of the elementary circuits of a directed graph. *SIAM Journal on Computing*, 2(3):211–216, 1973. Introduction

[TIAS77] Shuji Tsukiyama, Mikio Ide, Hiromu Ariyoshi, and Isao Shirakawa. A new algorithm for generating all the maximal independent sets. *SIAM Journal on Computing*, 6(3):505–517, 1977. Introduction, 1.2.2, 2.1.3, 2.2.4, 2.3.1, 2.3.1, 2.3.1, 4.4

[Tie70] James C. Tiernan. An efficient search algorithm to find the elementary circuits of a graph. *Communications of the ACM*, 13(12):722–726, 1970. Introduction

[Tro92] William T. Trotter. *Combinatorics and partially ordered sets: Dimension theory*, volume 6. JHU Press, 1992. 2.1.6

[Uno] Takeaki Uno. Hypergraph Dualization Repository. `http://research.nii.ac.jp/~uno/dualization.html`. Accessed: 2020-March-02. Introduction

[WIK] Enumeration Algorithm Wikipedia's page. `https://en.wikipedia.org/wiki/Enumeration_algorithm`. Accessed: 2020-March-02. Introduction

[Wil94] Marcel Wild. A theory of finite closure spaces based on implications. *Advances in Mathematics*, 108(1):118–139, 1994. Introduction, 4, 4, 4.1.5, 4.5

[Wil95] Marcel Wild. Computations with finite closure systems and implications. In *International Computing and Combinatorics Conference*, pages 111–120. Springer, 1995. 4, 4.1.1, 4.1.4, 4.1.4

[Wil00] Marcel Wild. Optimal implicational bases for finite modular lattices. *Quaestiones Mathematicae*, 23(2):153–161, 2000. 1.3.1, 4

[Wil17] Marcel Wild. The joy of implications, aka pure Horn formulas: mainly a survey. *Theoretical Computer Science*, 658:264–292, 2017. 3.1.3, 3.4, 4, 4, 4.1.1, 4.1.4, 4.1.4, 4.1.4, 4.1.5, 4.3, 4.3, Conclusion

[YYH05] Xifeng Yan, Philip S. Yu, and Jiawei Han. Substructure similarity search in graph databases. In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 766–777. ACM, 2005. Introduction

Abstract

This thesis focuses on graphs, hypergraphs, and lattices. We study the complexity of the dualization of monotone Boolean functions, and its generalizations, through the many shapes it takes on these structures: minimal dominating sets enumeration, minimal transversals enumeration, lattice dualization, and meet-irreducible enumeration. Both tractable and intractable results are obtained, and future research directions are proposed. The thesis is organized as follows. A first part is devoted to the enumeration of minimal dominating sets in graphs. We obtain new output-polynomial time algorithms in graph classes related to K_t-free graphs and to posets of bounded dimension. A second part is devoted to generalizations of this problem in lattices. One generalization concerns the dualization in lattices given by implicational bases, the other deals with the enumeration of meet-irreducible elements. Both tractability and intractability results are obtained under various restrictions concerning width, acyclicity, and premises' size in the implicational base. The two parts are sprinkled with hypergraph transversals enumeration and related notions.

Keywords: *algorithmic enumeration, lattice dualization, minimal dominating sets, minimal transversals, maximal independent sets, meet-irreducibles, implicational bases.*

Résumé

Cette thèse porte sur la théorie des graphes, des hypergraphes, et des treillis. Nous nous intéressons à la complexité du problème de dualisation des fonctions monotones Booléennes, ainsi qu'à ses généralisations, à travers les différentes formes qu'il prend dans ces structures: énumération des dominants minimaux, des transversaux minimaux, dualisation dans les treillis, et énumération des éléments meet-irréductibles. De nouveaux résultats positifs et négatifs sont obtenus, et des directions de recherche futures sont proposées. La thèse se découpe comme suit. Dans une première partie, nous nous intéressons à l'énumération des dominants minimaux dans les graphes. Nous obtenons de nouveaux algorithmes output-polynomiaux dans les graphes sans grande clique, et dans d'autres classes de graphes liées aux ordres partiels de dimension bornée. Dans une seconde partie, nous nous intéressons aux généralisations de ce problème dans les treillis. Une première généralisation concerne la dualisation dans les treillis donnés par une base d'implications, l'autre concerne l'énumération des éléments meet-irréductibles. Des résultats positifs et négatifs sont obtenus sous plusieurs contraintes concernant la largeur, l'acyclicité, et la taille des prémisses dans la base d'implication. Les deux parties de la thèse sont parsemées d'énumération des transversaux minimaux d'un hypergraphe, et de notions liées.

Mots-clés: *énumération algorithmique, dualisation dans les treillis, dominants minimaux, transversaux minimaux, stables maximaux, meet-irréductibles, bases d'implications.*