UNIVERSITÉ CÔTE D'AZUR

ÉCOLE DOCTORALE STIC
SCIENCES ET TECHNOLOGIES DE L'INFORMATION ET DE LA COMMUNICATION

T H È S E

pour obtenir le titre de

Docteur en Sciences

de l'Université Côte d'Azur

Mention : Informatique

Présentée et soutenue par

Guillaume Ducoffe

Propriétés métriques des grands graphes

Thèse dirigée par David Coudert

préparée à l'Inria Sophia Antipolis, Projet COATI

soutenue le 9 décembre 2016

Jury :

Rapporteurs :	Victor Chepoi	-	Aix-Marseille Université
	Laurent Viennot	-	Inria (GANG)
Directeur :	David Coudert	-	Université Côte d'Azur, Inria, CNRS, I3S, France
Examinateurs :	Michele Flammini	-	Università degli Studi dell'Aquila
	Cyril Gavoille	-	LaBRI
	Nicolas Nisse	-	Université Côte d'Azur, Inria, CNRS, I3S, France
	Robert Tarjan	-	Princeton University and Intertrust Technologies
Invité :	Igor Litovsky	-	Université Côte d'Azur, CNRS, I3S, France

Acknowledgements

Je remercie d'abord les membres de mon jury de thèse. Merci en particulier à Victor Chepoi et Laurent Viennot pour avoir eu la gentillesse de rapporter ma thèse. J'attends avec impatience la version roumaine de son rapport, promise par Victor. Je remercie également Robert Tarjan pour m'avoir fait l'honneur d'accepter d'être membre de mon jury de thèse et d'avoir fait un long déplacement depuis les États-Unis à cette occasion. Enfin, merci à Igor Litovsky, dont je n'oublie pas qu'il a été le premier à me pousser vers la recherche, et que je remercie chaleureusement ainsi que Paul Franchi, Ioan Bond et Christophe Papazian.

Mes remerciements vont de même à Jean-Claude Bermond, pour m'avoir encadré lors de mon tout premier stage de recherche, et depuis pour m'avoir guidé et sagement conseillé dans mes choix de carrière... et battu systématiquement au trash ping-pong! Un très grand merci du fond du coeur à mon encadrant, David Coudert, tant pour sa (très) grande patience que sa disponibilité, ses conseils scientifiques et humains précieux, et les efforts de qualité qu'il a mis afin que cette thèse se déroule dans les meilleures conditions possibles. Je tiens bien sûr à remercier tous les autres membres du projet Mascotte/Coati et apparentés: de Patricia, l'âme de cette équipe, à mon co-bureau Nicolas – alias Jésus – en passant par Nicolas (le grand), Christelle l'excellente (non, elle ne m'a pas mis le couteau sous la gorge...), Minus le bon, Majus la brute et Michel le truand, l'énigmatique Stéphane, Julien, Joanna, Steven, William "Hochet", et les anciens (et revenus) dont Fatima, Julio, Dorian et Magali, Nadège, Aurélien, Nathann, Alvinice et Khoa. Ces remerciements s'étendent, outre-Atlantique, à Augustin, Roxana, Mathias, Juba, Chris, Avner, Arthi et Max.

Des remerciements très spéciaux à mes deux soeurs, Anaïs et Mélanie, qui me soutiennent depuis toujours dans mes choix de vie et de carrière. J'ai une pensée particulière pour Mélanie qui, malheureusement, n'a pas pu se libérer pour la soutenance. À vous deux, j'espère vous rendre ne serait-ce que le dixième de ce que vous m'avez donné. Merci, bien sûr, à mes deux parents, à qui ira toujours toute mon affection. J'ai une petite pensée pour Lolita, trop tôt disparue, et pour Pelote, Mistoufle et Catarina. Enfin, je fais un merci général à ma grand-mère ainsi qu'à tout le reste de ma nombreuse petite famille!

J'offre mon dernier remerciement, du fond de mon coeur et de mon âme, à ma chérie Adriana. Il y a eu un avant et un après t'avoir rencontré. Je te dois ma bonne humeur, mon inspiration, et l'éternité ne me suffira pas à te rembourser ma dette (même si je compte bien m'y employer). *Eu te iubesc la nebunie, jumatatea mea*!

Abstract

Large scale *communication networks* are everywhere, ranging from data centers with millions of servers to social networks with billions of users. This thesis is devoted to the fine-grained complexity analysis of combinatorial problems on these networks.

In the first part, we focus on the embeddability of communication networks to tree topologies. This property has been shown to be crucial in the understanding of some aspects of network traffic (such as congestion). More precisely, we study the computational complexity of Gromov hyperbolicity and of tree decomposition parameters in graphs – including treelength and treebreadth. On the way, we give new bounds on these parameters in several graph classes of interest, some of them being used in the design of data center interconnection networks. The main result in this part is a relationship between treelength and *treewidth*: another well-studied graph parameter, that gives a unifying view of treelikeness in graphs and has algorithmic applications. This part borrows from graph theory and recent techniques in complexity theory.

The second part of the thesis is on the modeling of two privacy concerns with social networking services. We aim at analysing information flows in these networks, represented as dynamical processes on graphs. First, a coloring game on graphs is studied as a solution concept for the dynamic of online communities. We give a fine-grained complexity analysis for computing Nash and strong Nash equilibria in this game, thereby answering open questions from the literature. On the way, we propose new directions in algorithmic game theory and parallel complexity, using coloring games as a case example. Finally, we introduce a new learning problem that is motivated by the need for users to uncover any misuse of their personal data online. We give positive and negative results on the tractability of this problem.

Keywords: Graph; Algorithms; Complexity in P; Gromov Hyperbolicity; Treelength; Treebreadth; Treewidth; Coloring games; Nash equilibrium; Boolean function learning.

Résumé

Les grands *réseaux de communication* sont partout, des centres de données avec des millions de serveurs jusqu'aux réseaux sociaux avec plusieurs milliards d'utilisateurs. Cette thèse est dédiée à l'étude fine de la complexité de différents problèmes combinatoires sur ces réseaux.

Dans la première partie, nous nous intéressons aux propriétés des plongements des réseaux de communication dans les arbres. Ces propriétés aident à mieux comprendre divers aspects du trafic dans les réseaux (tels que la congestion). Plus précisément, nous étudions la complexité du calcul de l'hyperbolicité au sens de Gromov et de paramètres des décompositions arborescentes dans les graphes. Ces paramètres incluent la longueur arborescente (treelength) et l'épaisseur arborescente (treebreadth). Au passage, nous démontrons de nouvelles bornes sur ces paramètres dans de nombreuses classes de graphes, certaines d'entre elles ayant été utilisées dans la conception de réseaux d'interconnexion des centres de données. Le résultat principal dans cette partie est une relation entre longueur et largeur arborescentes (*treewidth*), qui est un autre paramètre très étudié des graphes. De ce résultat, nous obtenons une vision unifiée de la ressemblance des graphes avec un arbre, ainsi que différentes applications algorithmiques. Nous utilisons dans cette partie divers outils de la théorie des graphes et des techniques récentes de la théorie de la complexité.

La seconde partie de cette thèse est consacrée à la modélisation de deux problèmes motivés par le respect de la vie privée sur les réseaux sociaux. Notre objectif est d'analyser les flux d'information dans ces réseaux, représentés par des processus dynamiques sur des graphes. Tout d'abord, nous étudions un jeu de coloration sur les graphes, en tant que concept de solution pour la dynamique des communautés en ligne. Nous donnons une analyse fine de la complexité du calcul d'équilibres de Nash dans ce jeu, ce qui nous permet de répondre à des questions ouvertes de la littérature. De plus, nous proposons de nouvelles directions en théorie algorithmique des jeux et en théorie de la complexité parallèle, que nous illustrons à l'aide des jeux de coloration. Finalement, nous proposons un tout nouveau problème d'apprentissage, motivé par le besoin des utilisateurs en ligne d'identifier les mauvais usages de leurs données personnelles. Nous donnons des résultats, positifs comme négatifs, sur la faisabilité de ce problème.

Mots clés: Graphe; Algorithmes; Complexité dans P; Hyperbolicité; Treelength; Treebreadth; Treewidth; Jeux de coloration; Équilibre de Nash; Apprentissage de fonction Booléenne.

Contents

Chapter 1

Introduction

Contents

1.1 Context

Information sharing online has been gaining momentum over the last decades. As examples, as of 2015 there have been 205 billion emails sent on a daily basis [Ema]; Twitter reports on 500 million messages exchanged a day on its social platform [Twi]; more generally, the global Internet traffic has been observed to grow from 100 GB per day in 1992 to 20,235 GBps in 2015 [Cisa]. Accordingly, the volume of data stored also has increased, and it is now expected to exceed 40 zettabytes by 2020 [IDC].

As we now enter into this "zettabyte era" [Cisb], information technologists are confronted to several issues that are regularly covered by the media. Two of them are addressed in this thesis.

- **Scalability** – is defined in [Ten16] as the requirement for the algorithms to run in quasi-linear time in the size of the network. Put in less restrictive terms, there is a need for efficient algorithms in order to process the communication networks. Higher demands for such algorithms emerge from numerous domains, including telecommunications, social networks, bio informatics, computer vision, and economics. However, the rapid expansion of information sharing and data collection has lead these networks to scale up, with now millions of servers in some data centers [DCM], billions of users in social networks [FBN], etc. Textbook methods do not scale well with networks of these sizes, thereby increasing the gap between what we aim at computing and what can be achieved in practice. Hence, there is a need for revisiting what efficient/scalable computation means in this context.

  We will propose advances in this direction based on tools from (algorithmic) graph theory and complexity theory.

- **Privacy** – is defined in [EDP] as "*a right which prevents public authorities [or any other organization or individual] from measures which are [invasive for the respect of private life], unless certain conditions have been met.*" In particular, the agressive collection of data by online companies has started raising alarms as now reports on potential abuses are surfacing on a regular basis [Gou14, Mat12, VDSVS12, The14]. Therefore, there is a need for predictive models in order to detect, on an individual level, when these violations occur, or even better to identify them.

  Our main tools in this task will be computational learning theory and algorithmic game theory.

Before summarizing our contributions in Section 1.2, let us sketch our approach for the thesis. Roughly, this work concentrates on a collection of combinatorial problems on graphs, whose study is motivated by these above two issues in information technology. Since the proposed solutions are aimed at scaling up with large networks, we are particularly interested in obtaining a *fine-grained* complexity analysis for these problems.

In particular, our study in Part I puts the focus on some graph invariants which have been shown in previous works [NS11] to be related with these above two issues in information technology. Studying properties of the "complex networks" and their applications is not new, and this area has been proved successful in finding relevant parameters and properties to study, such as: clustering [LLDM09], power-law degree distribution [BAJ00], navigability [BKC09], (ultra) small world phenomenon [WS98], structural decomposition into a core and peripheries [DGM06], etc. In this work, we emphasize on the *metric tree-likeness* in graphs: a topic that has been receiving growing attention over the last decades and that summarizes at measuring how close the distance distribution of a graph is to a tree metric [Gro87].

We argue that studying the properties of the distance distribution is a natural choice when considering information propagation in the graph. Furthermore, we will detail more in Part I how the advantages and disadvantages of trees (with nice algorithmic applications on the one hand, but vulnerabilities on the other hand) can be translated to the graphs that are (metrically) "tree-like".

This main line of study will be completed with the complexity analysis of two dynamical processes on graphs in Part II, that both cover some aspects of privacy in communication networks. Simply put, the aim of this side line of the thesis is to design scalable tools in order to enforce privacy in these networks.

1.2 Contributions

Our work is presented in two separate parts which can be read independently. We present their content in Sections 1.2.1 and 1.2.2, respectively.

Full papers can be found in the appendix. Indeed, we made the choice not to include all proofs in the body of the chapters, partly for ease of readability as some of them are very long (dozens of pages). We will only give the proofs that, in our

opinion, are the best illustrations of our techniques. Sketches of the longest proofs will be also provided.

1.2.1 Part I: Metric tree-likeness in graphs

A main objective of Part I is to obtain a finer-grained analysis for the complexity of computing (metric) tree-likeness parameters and decompositions of graphs. Especially, can these properties be computed on large-scale graphs, with sometimes millions of nodes and billions of edges ? On the way, our analysis will conduce to study the relationships between metric tree-likeness in graphs and other graph properties (structural, topological, algebraic, etc.).

1.2.1.1 Chapter 2: A survey on graph hyperbolicity

This chapter introduces the notion of graph hyperbolicity, that gives lower and upper bounds on the best possible distortion of the distances in a graph when it is embedded into a tree.

First, we show positive and negative results on the complexity of computing this parameter. In particular, on the positive side we propose a preprocessing method for decreasing the size of the input graph by using the well-known clique-decomposition [BPS10], of which we give a fine-grained analysis. However, on a more negative side, we prove that the recognition of graphs with small hyperbolicity (at most 1/2) is computationally equivalent to the detection of induced squares in a graph. The latter result implies a conditional *cubic* lower-bound on the complexity of computing graph hyperbolicity. This is joint work with Nathann Cohen, David Coudert and Aurélien Lancin [CD14, CCDL17].

Then, we establish new bounds on this parameter in some graph classes that are used in the design of data center interconnection networks. In practice, these bounds can be used in order to sharply estimate the hyperbolicity in these classes of graphs. We complement these results with a fine-grained analysis of the variations of hyperbolicity that may be caused by various graph operations such as line graph, clique graph, etc. This analysis is particularly interesting in some cases where the operation can be efficiently reversed (*e.g.*, the root of a line graph can be computed in linear time [Whi92]), as then it leads to new preprocessing methods for the computation of graph hyperbolicity. This is joint work with David Coudert [CD16a, CD16b].

1.2.1.2 Chapter 3: Tree decompositions with metric constraints on the bags

New results are presented on the complexity of computing *tree decompositions* (decompositions of a graph in a tree-like manner) with metric constraints on their bags (*a.k.a.*, subgraphs resulting from the decomposition).

A finer-grained analysis of the complexity of computing the clique-decomposition is first presented. This problem is proved to be computationally equivalent, under

standard complexity assumptions, to the detection of triangles in graphs and the multiplication of two square matrices. On a more positive side, we show that it can be solved in quasi-linear time on some classes of graph where the maximum size of a clique is bounded. This is joint work with David Coudert [DC17].

Second, we answer open questions in the literature on the complexity of computing treebreadth, pathbreadth and pathlength: that are tree-likeness parameters all related to the notion of graph hyperbolicity. Namely, computing any of these parameters is an NP-hard problem. In particular, recognizing the graphs with treebreadth at most one is NP-complete. However, we prove that the latter problem can be solved in polynomial-time for bipartite graphs and planar graphs. This is joint work with Sylvain Legay and Nicolas Nisse [DLN16a].

Finally, we investigate the relationships between another metric tree-likeness parameter, called *treelength*, and a well-known *structural* tree-likeness parameter that is called *treewidth*. Roughly, we establish upper and lower bounds on the treewidth with linear dependency on the treelength in the classes of graph with bounded-length isometric cycle (*i.e.*, with no shortcut) and bounded genus (*i.e.*, that can be drawn with no edge-crossing in a surface of bounded Euler genus). On the scalability point of view, algorithmic applications of these results will be further discussed. This is joint work with David Coudert and Nicolas Nisse [CDN16].

1.2.2 Part II: Privacy at large scale in social graphs

Two problems on privacy are discussed and studied in this part. Our objective is to obtain a finer-grained analysis for the complexity of these two problems.

1.2.2.1 Chapter 4: The computation of equilibria in coloring games

We consider a coloring game played on a graph. This game has been proposed in [KL13] as a solution concept for the dynamics of communities' formation in social networks. Earlier applications of the game have been suggested in [CKPS10] for securing group communications.

We present some new results on the complexity for computing equilibria in this game. More precisely, better-response dynamics can be used in order to compute a stronger notion of Nash equilibrium: that is robust to every coalition of agents of size at most a *fixed* k. On the positive side, we establish the exact convergence time of the dynamic for coalitions of size at most two. However, on the negative side, we prove that this convergence time is *superpolynomial* for coalitions of size at least four, thereby answering negatively to open questions from [EGM12, KL13]. This is joint work with Dorian Mazauric and Augustin Chaintreau [DMC13a, DMC17].

The latter results are complemented with a refined analysis for the complexity of computing a Nash equilibrium in this game (robust to coalitions of size one). This problem will be shown to be PTIME-hard under parallel reductions (and in particular, to logspace reductions), which is strong evidence that it is inherently sequential [Duc16].

Then, the remaining of the chapter is devoted to a natural generalization of coloring games on edge-weighted graphs. We give sufficient conditions for the existence of equilibria in these games depending on the structure of the underlying graph. We also propose surprising constructions of games that do not admit such equilibria. Last, we prove that the recognition of generalized coloring games that admit such equilibria is NP-complete. Extensions of all these results to broader classes of games will be discussed. This is joint work with Dorian Mazauric and Augustin Chaintreau [DMC12, DMC13a, DMC17].

1.2.2.2 Chapter 5: Learning formulas in a noisy model

We next focus on a learning problem whose context can be roughly described as follows. Suppose we are given a fixed ground-set $\mathcal{D}$ (representing keywords) and a graph where each node is labeled with a subset of $\mathcal{D}$ (*i.e.*, a collection of keywords). The nodes are assigned a Boolean under some (black-box) random process, that is correlated with an unknown Boolean function over the labels. Then, the objective is to learn this function. We aim at modeling with this problem the detection of any (mis)use of individual data by online advertisers.

First, we propose an algorithm for learning the function in the simpler case where it depends on at most one input. The latter algorithm will be the cornerstone of more sophisticated methods in order to learn any function – but under more restrictive hypotheses. Additional constraints are proved to be necessary in the general case, as otherwise the function cannot be learnt already if it depends on two inputs. This is joint work with Mathias Lécuyer, Francis Lan, Max Tucker, Riley Sphan, Andrei Papancea, Theofilos Petsios, Augustin Chaintreau and Roxana Geambasu [LDL+14, DLCG15, DTC17, CD17].

1.3 Preliminaries and notations

We borrow from the graph terminology of [BM08, Die10]. All graphs considered will be finite, undirected, unweighted, simple (hence, with neither loops nor multiple edges) and connected. In this situation, for every graph $G = (V, E)$ we can define the distance between every two vertices $u, v \in V$ as the minimum number of edges onto a uv-path of G. This distance is denoted by $\mathrm{d}_G(u, v)$ in what follows, or simply $\mathrm{d}(u, v)$ when there is no ambiguity on the graph G. Our proofs will make use of the notions of subgraphs, induced subgraphs and isometric subgraphs, the latter denoting a subgraph H of a graph G such that the distance between every two vertices in H is the same in H as in G.

Let us introduce additional distance notations. The *eccentricity* of a vertex $v \in V$, denoted by $ecc(v) = \max_{u \in V} \mathrm{d}_G(u, v)$ is the maximum distance in G between v and another vertex. The *diameter* of G, denoted by $diam(G) = \max_{v \in V} ecc(v)$, is the maximum eccentricity of a vertex of G. Furthermore, let $B_G(v, r) = \{u \in V \mid \mathrm{d}(u, v) \leq r\}$ be the ball of radius r centered on vertex v. The *radius* of G, denoted by $rad(G) = \min_{v \in V} ecc(v)$, is the least r such that $B_G(v, r) = V$ for some vertex

v. Finally, let $N_G[v] = B_G(v, 1)$ be the closed neighbourhood of a vertex. The open neighbourhood of v is defined as $N_G(v) = N_G[v] \setminus v$. By extension, let us define for every subset $S \subseteq V$ its open neighbourhood $N_G(S) = \left(\bigcup_{v \in S} N_G(v)\right) \setminus S$ and its closed neighbourhood $N_G[S] = N_G(S) \cup S$. We will remove the subscript when no ambiguity occurs.

1.4 List of publications

Journal papers

[ABD14] J. Araujo, J-C. Bermond, and G. Ducoffe. Eulerian and hamiltonian dicyles in directed hypergraphs. *Discrete Mathematics, Algorithms and Applications*, 6(1):1450012–1–1450012–29, 2014. (Uncited.)

[CD14] D. Coudert and G. Ducoffe. Recognition of c_4-free and 1/2-hyperbolic graphs. *SIAM Journal of Discrete Mathematics*, 28(3):1601–1617, 2014. (Cited in pages 3, 13, 18, 57, 65 and 221.)

[CD16a] D. Coudert and G. Ducoffe. Data center interconnection networks are not hyperbolic. *Journal of Theoretical Computer Science*, 639(1):72–90, 2016. (Cited in pages 3, 13, 17, 48, 50, 51, 53, 54, 60 and 221.)

[CD16b] D. Coudert and G. Ducoffe. On the hyperbolicity of bipartite graphs and intersection graphs. *Discrete Applied Mathematics*, 214:187–195, 2016. (Cited in pages 3, 13, 17, 42, 43 and 221.)

[CDN16] D. Coudert, G. Ducoffe, and N. Nisse. To approximate treewidth, use treelength! *SIAM Journal of Discrete Mathematics*, 30(3):1424–1436, 2016. (Cited in pages 4, 40, 75, 79, 104, 105, 107, 108, 109 and 222.)

[Duc13] G. Ducoffe. Hamiltonicity of large generalized de bruijn cycles. *Discrete Applied Mathematics*, 161:2200–2204, 2013. (Uncited.)

International conference papers

[DLCG15] G. Ducoffe, M. Lécuyer, A. Chaintreau, and R. Geambasu. Web's transparency for complex targeting: Algorithms, limits and tradeoffs. In *SIGMETRICS'15 Proceedings of the 2015 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, pages 465–466, 2015. (Cited in pages 5, 159, 165, 168, 177 and 223.)

[DLN16] G. Ducoffe, N. Legay, and N. Nisse. On the complexity of computing treebreadth. In *IWOCA 2016 – 27th International Workshop on Combinatorial Algorithms*, pages 3–15, 2016. (Cited in pages 4, 75, 78, 91, 92, 93, 94, 98, 111 and 222.)

[DMC13] G. Ducoffe, D. Mazauric, and A. Chaintreau. Can selfish groups be self-enforcing? In *Workshop on Social Computing and User Generated*

Content at EC'13, pages 1–47, 2013. (Cited in pages 4, 5, 117, 123, 222 and 223.)

[Duc16] G. Ducoffe. The parallel complexity of coloring games. In *SAGT 2016 – 9th International Symposium on Algorithmic Game Theory*, pages 27–39, 2016. (Cited in pages 4, 117, 123, 143 and 223.)

[LDL+14] M. Lécuyer, G. Ducoffe, F. Lan, A. Papancea, T. Petsios, R. Spahn, A. Chaintreau, and R. Geambasu. Xray: Enhancing the web's transparency with differential correlation. In *USENIX Security Symposium*, pages 49–64, 2014. (Cited in pages 5, 159, 160, 161, 162, 163, 165, 166, 168, 169, 170, 176, 188, 189 and 223.)

National conference papers

[CD16] D. Coudert and G. Ducoffe. Liens entre symétries et étirements de routages dans les réseaux d'interconnexion de centres de données. In *ALGOTEL 2016 – 18èmes Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications*, pages 1–4, 2016. (Uncited.)

[CDGL15] A. Chaintreau, G. Ducoffe, R. Geambasu, and M. Lécuyer. Vers une plus grande transparence du web. In *ALGOTEL 2015 – 17èmes Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications*, pages 1–4, 2015. (Uncited.)

[CDN15] D. Coudert, G. Ducoffe, and N. Nisse. Structure vs. métrique dans les graphes. In *ALGOTEL 2015 – 17èmes Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications*, pages 1–4, 2015. (Uncited.)

[DMC13] G. Ducoffe, D. Mazauric, and A. Chaintreau. De la difficulté de garder ses amis (quand on a des ennemis)! In *ALGOTEL 2013 – 15èmes Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications*, pages 1–4, 2013. (Cited in page 123.)

Unpublished papers

[CCDL17] N. Cohen, D. Coudert, G. Ducoffe, and A. Lancin. Applying clique-decomposition for computing gromov hyperbolicity. Submitted (Research Report on HAL, hal-00989024), 2017. (Cited in pages 3, 13, 18, 61, 62, 63 and 221.)

[CD17] A. Chaintreau and G. Ducoffe. A theory for ad targeting identification. In preparation, 2017. (Cited in pages 5, 159, 163, 169, 184 and 223.)

[DC17] G. Ducoffe and D. Coudert. Clique-decomposition revisited. In revision (Research Report on HAL, hal-01266147), 2017. (Cited in pages 4, 75, 78, 86, 87, 88, 90, 111 and 222.)

[DMC17] G. Ducoffe, D. Mazauric, and A. Chaintreau. The complexity of hedonic coalitions under bounded cooperation. Submitted (Research Report on ArXiv, arXiv:1212.3782), 2017. (Cited in pages 4, 5, 117, 123, 131, 132, 133, 135, 137, 150, 222 and 223.)

[DTC17] G. Ducoffe, M. Tucker, and A. Chaintreau. Can web's transparency tools cope with complex targeting? In preparation, 2017. (Cited in pages 5, 159, 165, 168, 171, 175 and 223.)

Theses

[Duc13] G. Ducoffe. Outils théoriques pour le calcul pratique de l'hyperbolicité dans les grands graphes. Master's thesis, MPRI – ENS Cachan, 2013. (Uncited.)

Part I

Metric tree-likeness in graphs

The purpose of the next two chapters is to study geometric and topological properties of graphs. They have been shown to be directly related to some important aspects of communications in large-scale data networks, such as *e.g.*, their performances, reliability and security [NS11]. Hence the need for better understanding and computing these graph properties, in order to better analyse and improve upon these aspects of network communications.

- Chapter 2 is a survey on **graph hyperbolicity**: a parameter that somewhat represents the "curvature" of the network. We are particularly interested in characterizing the graph classes where this parameter is either bounded or unbounded (respectively called hyperbolic and non hyperbolic graph classes), and to improve upon its computation in large-scale graphs.
- Chapter 3 presents new results on **tree decompositions** in graphs. Namely, positive and negative results are obtained on the complexity for computing tree decompositions that are defined via metric constraints on their bags. On the way, a finer-grained study of the relationships between structural and metric graph properties is proposed, that culminates with new relationships between the two graph parameters called *treewidth* and *treelength*.

Chapter 2

A survey on graph hyperbolicity

Summary

This chapter summarizes my work on graph hyperbolicity. It will be presented as a survey. The initial motivation for this work was to improve the practical computation of hyperbolicity on large graphs. In particular, I focused on the following general question: among the graph transformations that can be efficiently computed, which ones do not affect the value of hyperbolicity by more than a moderate term (multiplicative or, preferably, additive) ? I proved it was the case for clique-decomposition (Section 2.6.2.2) and the line graph operation (Section 2.4.3). Furthermore, my work on clique-decomposition has been successfully applied on large co-authorship graphs in order to compute their hyperbolicity [CCDL17].

I also proved new lower-bounds on graph hyperbolicity (using graph endomorphisms) that may further help reducing the complexity for computing the hyperbolicity in some graph classes (Section 2.5.2.4). By doing so, I answered an open question from researchers at the University of Girona (private communication) who aimed at sharply estimating the hyperbolicity of very large underlying topologies that are used for data center interconnection networks. Indeed, these graphs have more than one million nodes each, that overrule the current limitations of the existing algorithms for computing this parameter. By using my lower-bound techniques, I was able to give the exact value of the hyperbolicity for most topologies, and to prove close lower and upper bounds for the hyperbolicity of many other ones [CD16a].

I complemented these results with a conditional lower-bound on the complexity of recognizing graphs with hyperbolicity at most 1/2 (Section 2.6.3). It suggests that there does not exist any truly subcubic *combinatorial* algorithm for computing hyperbolicity on general graphs.

All my papers on graph hyperbolicity [CCDL17, CD16a, CD16b, CD14] are collected in the appendix.

Contents

2.1 Introduction

In this chapter we survey the study on Gromov hyperbolicity in graphs [Gro87, Ben13]. Roughly, it is a parameter which measures how close a given metric space is to a *metric tree* [Ban90, Bun74] (formal definitions are postponed to Section 2.3). In particular, it gives sharp bounds on the least distortion of the distances in a (finite) metric space when its elements are mapped to the nodes of an edge-weighted tree. Trees and bounded diameter graphs (embeddable into any shortest-path tree with constant distortion of their distances) will be shown to be trivially hyperbolic.

Gromov hyperbolicity is a broad concept that can be defined for any metric space. In fact, it has been first investigated for word metric spaces on groups [Gro87]. This notion of hyperbolicity in groups is now regarded as a powerful tool that can be

used in order to capture broad classes of groups with precise and important structural properties [GdLH90]. In particular, it has applications in the study of *automatic groups* [Gro87, EPC+92], where informally speaking, elements of the groups are the vertices of some (Cayley) graph and it can be checked with finite-state automata whether two words represent either a same vertex or two adjacent vertices. Automatic groups have nice algorithmic applications. For instance the word problem can be solved in quadratic time for these groups [EPC+92]. These applications transpose to groups with finite hyperbolicity, that are a particular case of automatic groups.

There is now a rich literature on the hyperbolicity of groups as metric spaces [ABC+91, BH11, GdLH90]. In this chapter, built as a survey, we emphasize on some results that are more specific, and relevant, to graph theory.

I will present my contributions on this topic in this chapter. They will be highlighted at various places in what follows. I hope that the organization of this chapter will help the reader to have a good overview of the positioning of my work in the growing literature on graph hyperbolicity.

Foreword

Let us start motivating the study of graph hyperbolicity in computer science. These aspects will be further developed in Sections 2.2 and 2.7.

In what follows, hyperbolicity should be understood as a graph parameter which gives bounds on the least distortion of the distances in a graph when its vertices are mapped to points in some "tree-like" metric space. Namely, such spaces comprise (weighted) trees, Hyperbolic spaces, and more generally speaking spaces with negative curvature. In general, embedding a graph into one of those spaces with minimum distortion is NP-hard [ABF+98]. As we shall see in this chapter, one interest of hyperbolicity is that it provides sharp bounds on this distortion in polynomial time (we will come back to this aspect in Section 2.7.1).

A rough description of hyperbolicity in graphs can be found at the beginning of Section 2.2. It should be noted, however, that there exists a bewildering zoo of "equivalent" definitions for this concept, whose formal presentation is postponed to Section 2.3.

Why studying hyperbolicity ? Depending on its order of magnitude, the value of hyperbolicity has some implications on network properties which, to my mind, motivate the study of this parameter in graphs. Indeed, studies on it have found applications in the analysis of congestion [CDV16], routing schemes [AGCFV, CDE+12, GL05], network security [JL04], bioinformatics [DMT96, MS99] and even in advertising allocation in social networks [MGHB15] — to name a few. I shall detail more about the above in Section 2.2. Most of these applications follow from, and can be better explained by, the close relationship between hyperbolicity and the best possible stretch (or distortion) of the distances in a graph when it is embedded into a Hyperbolic space (see [BS11, VS14]).

Hyperbolicity in graphs has strong geometric interpretations. It allows to extend the mathematical concept of *curvature* to discrete combinatorial structures such as graphs. Further, it can be used to characterize the so-called "underlying hidden geometry" of complex networks [KPK+10]. In this aspect, graph hyperbolicity adds up to other classification critera for networks such as (ultra) small world phenomenon [WS98], power law degree distribution [BAJ00], navigability [BKC09], high clustering coefficients [LLDM09], existence of a core [DGM06], etc. Relationships between hyperbolicity and these more classical features have been investigated, *e.g.*, in [CFHM13, JLB08, DX09, ASM13].

On the algorithmic side, another interest of hyperbolicity is that it helps analyzing, and designing, some graph heuristics on large-scale networks. For instance, the *2-sweep* heuristic for computing the diameter is well-known to provide very good results in practice [MLH08], and such good results can be explained assuming a bounded hyperbolicity [CDE+08]. I shall come back to the algorithmic applications of hyperbolicity in Section 2.7.

We next introduce two general objectives in the study of graph hyperbolicity, that will be the backbone of the main technical sections of this chapter. On the way, the personal contributions in this chapter are summarized and classified with respect to these two general objectives.

Namely, what we aim at obtaining through this study on hyperbolicity is: a better characterization of hyperbolic and non hyperbolic graph classes (Section 2.1.1), and a finer-grained analysis of the complexity of computing this parameter (Section 2.1.2). The outline of the chapter will be detailed in Section 2.1.3.

2.1.1 First objective: characterizing "hyperbolic" and "non hyperbolic" graph classes

The first objective is to derive lower and upper bound techniques for graph hyperbolicity. Indeed, it has become a growing line of research to characterize the classes of "hyperbolic" graphs, *a.k.a.* graphs with "small" hyperbolicity. – We shall make more precise what a hyperbolic graph is in Section 2.3.3 –. Partial results on that topic have been obtained in [BRSV13, HPR14] and the papers cited therein. They often derive from upper and lower bounds on the hyperbolicity of a given graph w.r.t. some other graph parameters and properties.

In Sections 2.4 and 2.5, I shall revisit the known bounds on the hyperbolicity of a given graph. Equipped with these bounds, I shall detail their application to some graph classes.

My main contributions in this area, found in collaboration with David Coudert, are twofold.

2.1.1.1 New lower bounds on the hyperbolicity of graphs

First, based on a game-theoretic definition of hyperbolicity, we provide some new lower-bound techniques on the hyperbolicity of graphs. Altogether combined with

the existence of certain type of *symmetries* (graph endomorphisms), these techniques are used in order to estimate the correct order of magnitude for the hyperbolicity in various graph classes. In particular, it follows directly from this work that many classical topologies that are used for the design of the data center interconnection networks [AK89] have their hyperbolicity that is proportional to their diameter.

This part of the contributions has been published in [CD16a]. I will describe these lower-bound techniques in Section 2.5.2, with some new results that are yet to be published.

2.1.1.2 A framework to bound the variations of hyperbolicity

Second, I present a simple framework in order to lower and upper bound the variations of hyperbolicity that may be caused by various graph operations. This framework applies to the line graph [Whi92], clique graph [Ham68] and biclique graph [GS10] operations, among some others, and the bounds so obtained are either new or improving upon the existing ones. Furthermore, the framework is mainly based on a new property of the hyperbolicity of bipartite graphs, that is of independent interest.

This part of the results has been published in [CD16b]. I will expand on it in Section 2.4.

2.1.2 Second objective: computing the hyperbolicity of large graphs

Then, as the second main technical part of this chapter, we will consider the complexity of computing the hyperbolicity of a given graph. That is, we will review the best-known algorithms for computing this parameter (exact and approximate), heuristics, and conditional lower-bounds on the best possible complexity for doing so. We note that an efficient computation of hyperbolicity can help characterizing which graph classes are hyperbolic. Furthermore, computing the hyperbolicity is a prerequesite for some of the above-mentioned applications to network problems [KL06, VS14] (see also Section 2.7).

There is a trivial algorithm to compute the hyperbolicity of a given n-vertex graph in $\mathcal{O}(n^4)$-time and $\mathcal{O}(n^2)$-space. Therefore, the problem is polynomial-time solvable (complexity class P). The latter is often regarded as a synonym for "tractable" [Reu16]. However, with the growing size of real-life networks, ranging from thousands to millions of nodes and billions of edges, we need to revisit the time and space complexity of polynomial problems. This finer-grained complexity of polynomial problems has become a boiling topic of research [Wil16]. In this aspect, we note that it is also of independent interest to study on the complexity of computing the hyperbolicity so as to obtain a better understanding of the hardness in P.

I will present in Section 2.6 the state-of-the-art algorithms for computing the hyperbolicity. I will also present some conditional lower-bounds on the time com-

plexity for this problem.

My main contributions in the area can be summarized as follows:

2.1.2.1 A preprocessing method for the computation of hyperbolicity

On the positive side, relationships between hyperbolicity and *clique-minimal decomposition* [BPS10] are proved and exploited for algorithmic purposes. This is joint work with Nathann Cohen, David Coudert and Aurélien Lancin. See also the PHD thesis of Aurélien Lancin [Lan14] for complementary information on this work.

Precisely, we prove that the hyperbolicity of a graph is at most one unit off from the maximum hyperbolicity from its *atoms* – *a.k.a.* the subgraphs resulting from its decomposition by clique minimal separators. Then, we base on this result in order to design a preprocessing algorithm for the computation of hyperbolicity. It substitutes to a given graph a collection of supergraphs of its atoms.

As a byproduct, we obtain a linear-time algorithm for computing the hyperbolicity of a given *outerplanar graph*.

These results [CCDL17] are to be submitted for publication in a journal. They will be detailed in Section 2.6.2.

2.1.2.2 Conditional lower-bound on the recognition of graphs with small hyperbolicity

Finally, a computational equivalence is proven between the recognition of graphs with hyperbolicity at most 1/2 and the detection of induced cycles of length at most four in graphs. It can be derived from this result a conditional lower-bound on the complexity of computing hyperbolicity, as well as a theoretically better algorithm for the recognition of 1/2-hyperbolic graphs.

These results, found in collaboration with David Coudert, have been published in [CD14]. I shall come back to them in Section 2.6.3.

2.1.3 Outline of the chapter

We start providing concrete applications of hyperbolicity in different fields of computer science (Section 2.2). In our opinion, these applications should better motivate the study of this parameter in graphs, and especially in network analysis. A rough definition of hyperbolicity is also given in Section 2.2, whose only role is to make the applications of this parameter more intuitive.

Then, formal definitions and preliminary results will be given in Section 2.3 (restating properly the informal definition of Section 2.2 with details). This section is the most technical one of the chapter, as it goes deeper in the relationships between hyperbolicity and many other "equivalent" graph properties.

Sections 2.4 and 2.5 are devoted to our first main objective: to find upper and lower bounds on graph hyperbolicity, with the two sections being devoted respectively to upper and lower bound techniques.

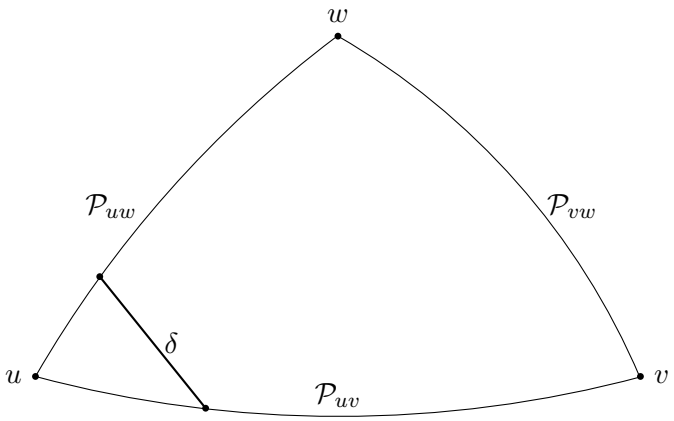


Figure 2.1: a geodesic triangle $\Delta(u, v, w)$.

Finally, the two last technical Sections 2.6 and 2.7 cover the algorithmic aspects of this parameter. In particular, the computational aspects of hyperbolicity are covered in Section 2.6, that is the second main objective in our study.

In Section 2.7 we detail algorithmic applications of hyperbolicity to various graph problems, that can be seen as a technical prolongation of Section 2.2. This section is placed on purpose after all the other sections, so as to give the reader a better overview of the ("hyperbolic") graph classes to which these algorithmic results can be applied. On the way, we mention several interesting open problems that are left for future work.

We finally conclude the chapter in Section 2.8.

2.2 Motivation

In this section, we will outline fields in computer science where the study of graph hyperbolicity plays a role. Our goal in doing so is to motivate the study of this parameter for computer scientists. Before introducing these applications of hyperbolicity, though, we will need to sketch a few graph properties that are related to this notion. They will be used in what follows in order to better intuit the role played by graph hyperbolicity in some applications.

Let us start giving an intuitive definition of hyperbolicity, that is sometimes named Rips condition in the literature [Gro87].

Consider any three vertices u, v, w in a given connected graph $G = (V, E)$. By the triangular inequality, we have $\mathrm{d}_G(u, v) \leq \mathrm{d}_G(u, w) + \mathrm{d}_G(w, v)$, with $\mathrm{d}_G(u, v)$ being the distance (minimum number of edges onto a uv-path) between u and v in G. We can represent this situation with a *geodesic triangle* $\Delta(u, v, w) = \mathcal{P}_{uv} \cup \mathcal{P}_{uw} \cup \mathcal{P}_{vw}$ with its three respective sides being a fixed shortest uv-path $\mathcal{P}_{uv}$, a fixed shortest uw-path $\mathcal{P}_{uw}$ and a fixed shortest vw-path $\mathcal{P}_{vw}$ (cf. Figure 2.1).

Then, one may wonder how far a detour by vertex w can make us go from the

shortest uv-path. The graph G is said to have δ-slim triangles if for every geodesic triangle $\Delta(u, v, w)$, any vertex onto the shortest uv-path $\mathcal{P}_{uv}$ is at distance at most δ from $\mathcal{P}_{uw} \cup \mathcal{P}_{vw}$. The hyperbolicity of G is – up to a constant-factor – the smallest δ such that it has δ-slim triangles.

As an example, if G is a tree then since there exists a unique uv-path, any vertex of $\Delta(u, v, w)$ must lie on two sides of the triangle, and so, the triangles in G are 0-slim. We shall come back to formal definitions of hyperbolicity in Section 2.3. For now, let us describe informally a few properties of graphs with δ-slim triangles.

Property 1: Almost shortest-paths stay close from each other. We first sketch a relationship between the value of hyperbolicity and the distance between (almost) shortest-paths in a graph. Let $\lambda \geq 1$ and $\varepsilon \in \mathbb{R}$ be fixed constants. An (λ, ε)-almost shortest-path between u and v is any uv-path with length at most $\lambda \cdot \mathrm{d}_G(u, v) + \varepsilon$. The length of this path thus differs by at most a fixed constant (multiplicatively or additively) from the length of a shortest uv-path. In particular, a shortest-path is an $(1, 0)$-almost shortest-path. In Figure 2.2, the path drawn with thicker edges is an $(2, 1)$-almost shortest-path.

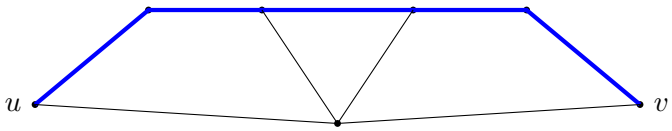


Figure 2.2: a $(2, 1)$-almost shortest uv-path.

Graph hyperbolicity measures the closeness of almost shortest-paths, in the following sense. Two paths $\mathcal{P}, \mathcal{Q}$ are at Haussdorf distance [RW09] at most d if every $x \in \mathcal{P}$ is at distance $\mathrm{d}_G(x, \mathcal{Q}) \leq d$ from the path $\mathcal{Q}$, and in the same way every vertex $y \in \mathcal{Q}$ is at distance $\mathrm{d}_G(y, \mathcal{P}) \leq d$ from the path $\mathcal{P}$. A key property of graphs with bounded hyperbolicity is that *any two almost shortest-paths with same endpoints stay close from each other*. That is, their Haussdorf distance is upper-bounded by a linear function of the hyperbolicity of G [Shc13a]

As an instructive example, consider the particular case of two shortest uv-paths. They can be seen as a "flat triangle" $\Delta(u, v, u)$. In particular, in a graph with δ-slim triangles, any two vertices on these shortest-paths that are at same distance from u (or equivalently, to v) in the graph are at distance at most 2δ (*e.g.*, see Figure 2.3). This property is sometimes called the k-fellow traveler property (here, for $k = 2\delta$) [NS95]. The more general Property 1 that almost shortest-paths stay close to each other is sometimes called geodesic stability [Fin15].

Property 2: Existence of a core. The second property that I want to point out can be summarized as a property of concentration for the almost shortest-paths in a graph. Let us fix two arbitrary constants λ and ε. We call a subset S of vertices an α-core if for some fraction α of all possible pairs of vertices in the graph, every

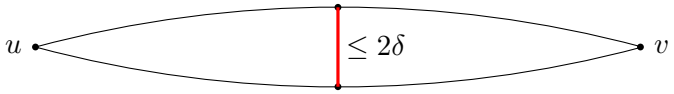


Figure 2.3: Shortest paths stay close in a δ-hyperbolic graph.

(λ, ε)-almost shortest-path with its two ends among these pairs is intersected by S. As an example, the whole vertex-set is trivially a 1-core, and the neighbourhood of a single vertex is an $\frac{n-1}{\binom{n}{2}} = 2/n$-core (it intersects all paths between this vertex and the other $n-1$ vertices). As shown with Figure 2.4, the root of a complete binary rooted tree is an 1/2-core. More generally, every tree has a vertex being an 1/2-core, that is sometimes called a *centroid* [Gol71].

Recall that the hyperbolicity measures the closeness of a graph from a metric tree. The second key property of graphs with bounded hyperbolicity that we focus on in this section is that *there exists a ball of small radius that is an* $\frac{1}{2}$*-core*. Precisely, the radius of the ball is upper-bounded by a linear function of the hyperbolicity of the graph G [CDV16].

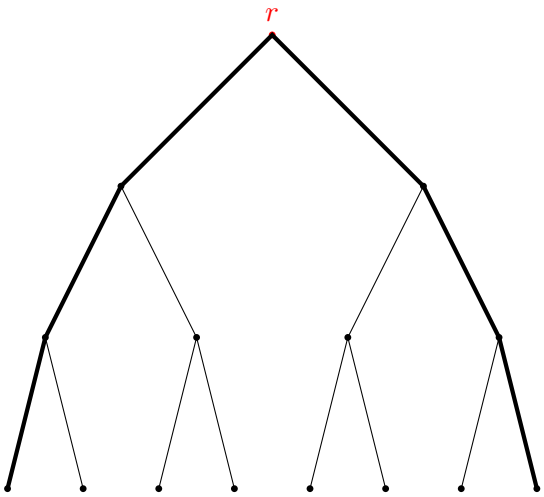


Figure 2.4: all shortest-paths between a vertex in the left subtree and a vertex in the right subtree go through the root.

Altogether, in any graph with bounded hyperbolicity, almost shortest-paths between any pair of vertices stay close to each other and there exists a ball with small radius intersecting almost all of these paths.

Equipped with these two intuitive properties, we will motivate the study of graph hyperbolicity next.

2.2.1 Implications/applications of hyperbolicity

We now list applications and implications of graph hyperbolicity in different fields of computer science. They encompass most of the work on the hyperbolicity in real-life graphs over the last decades. In what follows, these applications are more or less presented from the earliest ones to the newest ones.

Biology

One of the earliest applications of graph hyperbolicity that we are aware of is in biology, where there is a need to obtain some trees reflecting the similarity between a collection of species, *a.k.a.* phylogenetic trees [DMT96, MS99]. Known similarities between the species can be encoded as a graph, whose vertices are the species and whose edge-set corresponds to the pairs of species that are closely similar. Then, the problem summarizes as embedding the species into the leaves of some rooted tree so that the distance between any two species in the tree corresponds to their similarity. However, the available data is biased, and so, such a tree may not always exist. Since, hyperbolicity is a measure of the closeness of a graph to a metric tree, it has been proposed as a natural estimate for the bias of the data. Thus, standard results on graphs with small hyperbolicity (summarized in the later sections) can be applied on the data in order to find an approximate distance-preserving phylogenetic tree [DHH+05].

Geometric routings

Hyperbolicity comes into play in the study of certain *geometric routing schemes*. More precisely, we recall that the hyperbolicity is a measure of the closeness of a graph to a tree. As we shall explain, graph hyperbolicity was shown to provide (lower and upper) bounds on the stretch of the paths obtained with geometric routing schemes in some "tree-like" spaces [AGCFV, VS14].

Roughly, a *routing scheme* is a mapping of each pair of vertices u, v to a uv-path, that is to be followed in order to transit a message between u and v. Usually, we evaluate the quality of a routing scheme on the amount of information that needs to be stored locally at each node in order to retrieve the paths, and on the length of the paths that are used for the transit. That is, on the distributed computing point of view, the aim of *compact routing* schemes is to achieve a good compromise between minimizing the local information to be stored and keeping close to optimal the length of the paths that are used for the mapping.

A geometric routing scheme is one that embeds a given graph into a "simpler" metric space. Then, the paths of the routings are constructed greedily, starting from the source, with each vertex choosing as its successor on the path any of its neighbours that is strictly closer – w.r.t. their coordinates in the metric space – to the destination. In general, routing this way may not allow to reach all possible destinations. For instance, it may lead to infinite loops, and so, additional features are required in order to prevent loss of packet [PR05].

However, in his seminal paper [Kle07], Kleinberg has proved that for *every* graph, there exist embeddings into the *Hyperbolic space* (*i.e.*, canonical space with negative curvature, where the classical Euclidean geometry is replaced by hyperbolic geometry) such that greedy routing is always successful ! This paves the way to an in-depth study of greedy routings in the Hyperbolic space [BPK10, ST08], as well as in other "tree-like" metric spaces such as the word metric space of the free group [CPFV14]. In particular, in some classes of graphs with bounded hyperbolicity, we obtain compact routings with this greedy approach. We also refer to [DDGY07, GL05, KLNS15] for more examples of compact routing schemes in some classes of graphs with bounded hyperbolicity.

Furthermore, it is worth pointing out that embeddings with coordinates of polylogarithmic size in the number of vertices can be computed for those above spaces. In contrast to this positive result, there are graphs for which *greedy routing is always successful* in a given space but that cannot be embedded into the space with coordinates of sublinear size [BL05].

Network congestion

Of importance is also the implications of hyperbolicity on congestion in networks for all-to-all communications. Precisely, consider a unit traffic between each pair of vertices in a network, with the unit flow between any two vertices u, v being equally split among the shortest uv-paths. The *load* of a given vertex is the amount of flow which transits by this vertex. In more graph-theoretic terms, it corresponds to the *betweenness centrality* of the vertex [Bra01]. It is well-known and easy to observe that in trees, there is a a vertex with *quadratic* load $\Theta(n^2)$. What has been observed experimentally in [NS11] is that, more generally, for every graph with small hyperbolicity there is a ball of small radius such that the sum of the loads of the vertices in the ball is also quadratic.

Basing on the above observations, the authors in [JLBB11] have conjectured the existence in hyperbolic graphs of a ball of small radius through which it transits a *constant* proportion of traffic paths. The existence of a 1/2-core with small radius in graphs with bounded hyperbolicity (*i.e.*, Property 2) was shown in order to prove the above conjecture [CDV16]. See also [BT12, LT15, Yan15] for more implications of hyperbolicity on network congestion that take into account different traffic rates on the communications.

Network security

In their survey [JL04] and the papers cited therein, Jonckheere and Lohsoonthorn also have demonstrated the implication of "geometric" graph properties on some aspects of network security. On the way, they classified these geometric properties according to three levels of granularity (small, medium and large scale). At large scale, when considering graphs with a growing diameter, going to infinity (topologies in expansion such as the Internet Service Provider graph), the authors claim the

hyperbolicity to be the relevant parameter to study for a better understanding of the geometric aspects of network security.

They support their claim through a case-study of various security attacks. For instance, consider an attempt of "eavesdropping" or "packet sniffing" on the network — unauthorized packet interception along a given link. Due to the limited abilities to reorder the packets with TCP, they are often sent along near-optimal routes, *i.e.*, almost shortest-paths. Hence, since almost shortest-paths stay close to each other in hyperbolic graphs (Property 1), a small hyperbolicity might be detrimental in Information Warfare, causing the routes of the packets to be too close by security standards.

Other attacks and defense strategies where the value of hyperbolicity plays a role are Distributed Denial of Service (DDoS) attacks, and Worm Propagation, to name a few [JL04].

Democracy in complex networks

More recently, a new implication of hyperbolicity was suggested in [BCC15], as a measure of democracy in complex networks, on which we now emphasize. The latter is usually measured through *assortativity*, *i.e.*, the likeliness of vertices that are "similar" in some ways to be adjacent [New02] (see also [ALPT16, Lot15] for other recent approaches). In contrast with this more classical approach, the authors in [BCC15] (see also [ADM14]) consider a set of vertices to be "influencial" if it intersects the (almost shortest) paths between a large number of vertices. With respect to their interpretation, the graph is all the more democratic that it has no influential set of small size.

From this classification, it follows that graphs with small hyperbolicity are "aristocratic" (non democratic). Indeed, we recall that a small hyperbolicity implies the existence of a core with small radius (Property 2), which combined with some properties of real-life graphs (sparse, power-law, etc.) can be shown to be an influential set of small size. Let us point out that it has been experimentally shown that social networks have small hyperbolicity [AAD16]. Therefore, I think that this new notion of "influential set" and its relationships with hyperbolicity could and should be used in the study of *elites* in these networks — *i.e.*, relatively small subsets of vertices that are well-connected and highly connected to the other vertices [ALNP15, ALP11].

The above listing, which of course may be not exhaustive, shows the implications and applications of graph hyperbolicity in various areas. We expect more applications of hyperbolicity to be found.

2.3 Definitions of hyperbolicity

The purpose of this section is to present the formal definitions of graph hyperbolicity and related concepts. The standard definitions for this parameter will be introduced in Section 2.3.1. Then, the focus of Section 2.3.2 will be on "reformulations" of

hyperbolicity , *i.e.*, other geometric graph parameters than can be lower and upper bounded by functions of the hyperbolicity. We will end discussing on what should be understood as a "hyperbolic" graph in the remaining of this chapter (Section 2.3.3).

2.3.1 δ-hyperbolic graphs

Let us start introducing the standard definition for graph hyperbolicity. It can be written in two equivalent ways, that will be presented and explained next.

2.3.1.1 Four-point Condition

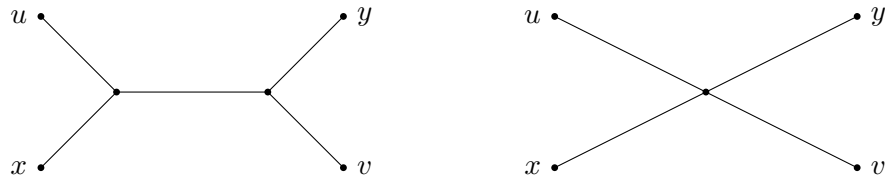
In what follows, the classical definition of hyperbolicity and its interpretation in relation to tree embeddings are given. In the line of many papers [BKM01, BC03, KM02], we define hyperbolicity via the following, rather abstract, four-point condition.

Definition 1 (**4-points Condition, [AJ13, Gro87]**). Let $G = (V, E)$ be a connected graph.

For every 4-tuple u, v, x, y of V, let $\delta(u, v, x, y)$ be defined as half of the difference between the two largest sums amongst:

$$S_1 = \mathrm{d}_G(u, v) + \mathrm{d}_G(x, y),\ S_2 = \mathrm{d}_G(u, x) + \mathrm{d}_G(v, y), \text{ and } S_3 = \mathrm{d}_G(u, y) + \mathrm{d}_G(v, x).$$

The graph hyperbolicity, denoted by $\delta(G)$, is equal to $\max_{u,v,x,y \in V} \delta(u, v, x, y)$. Moreover, we say that G is *δ-hyperbolic* for every $\delta \geq \delta(G)$.



(a) Every vertex on the central path is a centroid of the 4-tuple.

(b) The central vertex is the unique centroid of the 4-tuple.

Figure 2.5: Possible 4-tuples in a tree. Each edge represents a path in the tree.

Definition 1 generalizes a well-known four-point characterization of *metric trees*. Indeed, a discrete metric space (and in particular, a graph), can be isometrically embedded into the nodes of an edge-weighted tree if and only if it is 0-hyperbolic [Bun74]. We show one part of this equivalence with Figure 2.5. Indeed, for every 4-tuple u, v, x, y in a tree, it can always be found a centroid such that there is no more than two nodes among u, v, x, y in each branch. Then, it can be checked by the calculation that any such 4-tuple has null hyperbolicity.

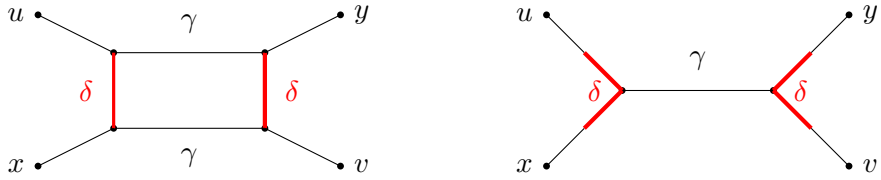
Furthermore, for general graphs G (not necessarily metric trees), hyperbolicity can also be interpreted in terms of tree embedding. In order to show that, let us fix

any four vertices u, v, x, y of G. Suppose we aim at embedding u, v, x, y in a tree T such that $\mathrm{d}_G(s,t) \leq \mathrm{d}_T(s,t)$ for every $s, t \in \{u, v, x, y\}$ (non contractive embedding) and the additive distortion $\alpha(u, v, x, y) = \min_T \max_{s,t\in\{u,v,x,y\}} \mathrm{d}_T(s,t) - \mathrm{d}_G(s,t)$ is minimized. We claim that $\alpha(u, v, x, y) = \delta(u, v, x, y)$, *i.e.*, the least possible distortion is given by the hyperbolicity of the 4-tuple.

On the one direction, let us fix T minimizing the distortion, and let us write:

$$S'_1 = \mathrm{d}_T(u,v) + \mathrm{d}_T(x,y),\ S'_2 = \mathrm{d}_T(u,x) + \mathrm{d}_T(v,y), \text{ and } S'_3 = \mathrm{d}_T(u,y) + \mathrm{d}_T(v,x).$$

In this situation, for every i we have $S_i \leq S'_i \leq S_i + 2\alpha(u, v, x, y)$, since by the hypothesis $\mathrm{d}_G(s,t) \leq \mathrm{d}_T(s,t) \leq \mathrm{d}_G(s,t) + \alpha(u, v, x, y)$ for every $s, t \in \{u, v, x, y\}$. Two cases need to be distinguished. If $S'_1 < \max\{S'_2, S'_3\}$ then we have $S_1 \leq S'_1 < \max\{S'_2, S'_3\} \leq S_2 + 2\alpha(u, v, x, y)$. In this situation, since $S_1 = S_2 + 2\delta(u, v, x, y)$, we get $\delta(u, v, x, y) < \alpha(u, v, x, y)$. Otherwise, $S'_1 \geq \max\{S'_2, S'_3\}$. In particular, the two largest sums amongst S'_1, S'_2, S'_3 must differ by at least $2(\delta(u, v, x, y) - \alpha(u, v, x, y))$. Since T is a tree, and so, it is 0-hyperbolic, it follows that $\alpha(u, v, x, y) \geq \delta(u, v, x, y)$ also in this case.



(a) Canonical realization of the 4-tuple. Distances in the realization are exactly the distances in the graph.

(b) Non contractive tree embedding with distortion δ that is obtained from the realization.

Figure 2.6: A 4-tuple so that $S_1 = \mathrm{d}(u,v) + \mathrm{d}(x,y) \geq S_2 = \mathrm{d}(u,y) + \mathrm{d}(v,x) \geq S_3 = \mathrm{d}(u,x) + \mathrm{d}(v,y)$. We denote by $\delta = (S_1 - S_2)/2$ and $\gamma = (S_1 - S_3)/2$.

On the other direction, consider in Figure 2.6a the so-called "canonical realization" of the metric space $(\{u, v, x, y\}, \mathrm{d}_G)$ with four elements. Using this representation, it is not difficult to see that u, v, x, y can be mapped to the four leaves of an edge-weighted tree with 6 nodes so that the embedding is non contractive and with distortion $\delta(u, v, x, y)$. Altogether combined, $\alpha(u, v, x, y) = \delta(u, v, x, y)$, and so, the hyperbolicity $\delta(G)$ is the least value δ such that for *every* 4-tuple of G, there exists a non contractive embedding into a tree with distortion at most δ.

In particular, we point out that since distances in an unweighted graph are integer-valued, the hyperbolicity is always a half-integer. This observation is sometimes useful in order to refine the bounds on the hyperbolicity, and in order to simplify some arguments in the proofs.

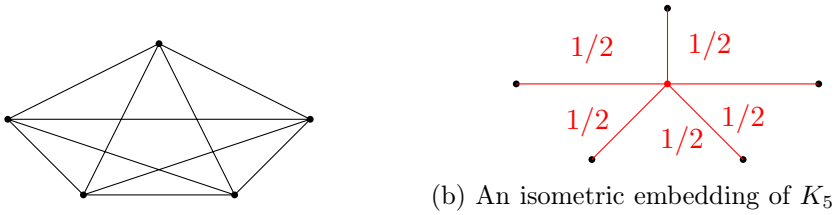
2.3.1.2 Toy examples

In order to give a better intuition of what this parameter represents, let us give the hyperbolicity of a few simple graphs.

Trees. In a tree, it is trivial that every 4-tuple can be embedded into a tree with null distortion. Therefore, every tree is 0-hyperbolic.

Intuitively, similar arguments should apply to the graphs that are "metrically" tree-like, *i.e.*, embeddable into a tree with constant distortion of their distances. This will be further discussed in Section 2.4.1 (upper-bounds on graph hyperbolicity).

Complete graphs. Perhaps more surprisingly, complete graphs are another example of 0-hyperbolic graphs. Indeed, as shown with Figure 2.7, a complete graph K_n with n vertices can be isometrically embedded into a star with $n+1$ nodes and all its edges weighted $1/2$.



(a) A complete graph K_5 with five vertices.

(b) An isometric embedding of K_5 to the leaves of an edge-weighted star.

Figure 2.7: Complete graphs are 0-hyperbolic.

Cycles. In spite of their simple structure, the cycles are the classical examples of graphs with large hyperbolicity. For instance, let $C_{4n} = (v_0, v_1, \ldots, v_{4n-1}, v_0)$ be a cycle with $4n$ vertices. Then, it follows from the four-point condition that $\delta(v_0, v_n, v_{2n}, v_{3n}) = n$ (see also Figure 2.8). Therefore, the hyperbolicity of a cycle grows linearly with its length. More generally, for every $n \geq 1$ and $\varepsilon \in \{0, 1, 2, 3\}$, we have $\delta(C_{4n+\varepsilon}) = n - 1/2$ if $\varepsilon = 1$ and $\delta(C_{4n+\varepsilon}) = n$ otherwise [WZ11].

Grids. Last, consider a rectangular grid with n columns and m rows. By taking the four corners of the grid, it comes from the 4-point Condition that the hyperbolicity of the grid is at least $\min\{n, m\} - 1$, that turns out to be its exact value [WZ11]. We refer to Figure 2.9 for an illustration.

It might help to observe that for grids and cycles, the shortest paths between the two vertices of any diametral pair do not stay close from each other. In contrast, we mentioned in previous Section 2.2 that in every graph with constant hyperbolicity, almost shortest paths stay close from each other (Property 2).

Furthermore, let us call a subgraph H of a graph G *isometric* if for every two vertices in H, their distance in this subgraph is exactly their distance in G. Since cycles and grids have unbounded hyperbolicity, any graph that contains a large cycle or a large grid as an isometric subgraph also has a large hyperbolicity, that directly follows from Definition 1. This point will be further discussed in Section 2.5.2.1 (lower-bounds on graph hyperbolicity).

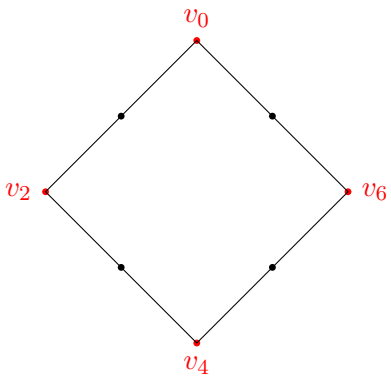


Figure 2.8: A cycle with eight vertices.

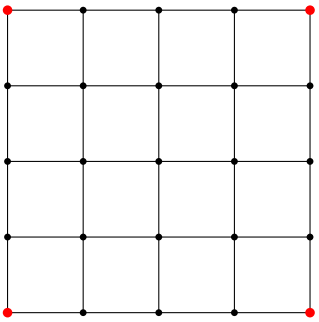


Figure 2.9: A square grid with side length four.

2.3.1.3 Gromov product, Farris transform and ultrametrics

In his seminal paper [Gro87], Gromov defines hyperbolicity via a different (but equivalent) formulation than Definition 1. In what follows, this formulation and its interpretation in terms of ultrametric embedding are stated. Before this, we need to introduce additional notions and terminology that are of independent interest.

Definition 2. Let $G = (V, E)$ be a graph. For every $u, v, w \in V$ the *Gromov product* of u and v with base vertex w is defined as $\langle u, v\rangle_w = (\mathrm{d}_G(u, w) + \mathrm{d}_G(w, v) - \mathrm{d}_G(u, v))/2$.

This notion of Gromov product naturally arises in the above canonical realization of 4-tuples (Figure 2.6a). Indeed, by the calculation we have that the length of the edge between vertex u and the central rectangle in the realization is exactly $\langle x, y\rangle_u$.

Note that $\langle u, v\rangle_w \geq 0$ by the triangular inequality. In particular, $\langle u, v\rangle_w = 0$ if and only if w lies onto a shortest uv-path. Thus, the Gromov product $\langle u, v\rangle_w$ can be seen as a measure of how close w is from a shortest uv-path.

In order to have a better insight of what this product represents, let us consider the particular case where G is a tree rooted at w. Let r be the lowest common

ancestor of u and v. In this situation, $\langle u,v\rangle_w = (\mathrm{d}(u,w)+\mathrm{d}(v,w)-\mathrm{d}(u,v))/2 = ((\mathrm{d}(u,r)+\mathrm{d}(r,w))+(\mathrm{d}(v,r)+\mathrm{d}(r,w))-(\mathrm{d}(u,r)+\mathrm{d}(r,v)))/2 = \mathrm{d}(r,w)$. Therefore, in a tree rooted at w, the Gromov product $\langle u,v\rangle_w$ is equal to the depth of the lowest common ancestor of u and v.

Let us also point out that $\langle u,w\rangle_v + \langle w,v\rangle_u = \mathrm{d}_G(u,v)$. In order to exemplify this equality, let us again consider the particular case where G is a tree. Then, $\langle u,w\rangle_v = \mathrm{d}(r,v)$ and $\langle v,w\rangle_u = \mathrm{d}(r,u)$, with r being the lowest common ancestor of u and v when the tree is rooted at w. As a result, $\langle u,w\rangle_v + \langle w,v\rangle_u = \mathrm{d}(r,v)+\mathrm{d}(r,u) = \mathrm{d}(u,v)$, as desired.

Finally, let $D \geq diam(G)$ be any upper-bound on the distances in G. We fix any base vertex x and define:

$$\mathrm{d}^{(x)}(u,v) = \begin{cases} 2D - \langle u,v\rangle_x \text{ if } u \neq v \\ 0 \text{ otherwise.} \end{cases}$$

Then, it can be checked that $\mathrm{d}^{(x)}$ is a distance function, that is sometimes called a *Farris transform* [Far72]. Furthermore, an interesting property of the Farris transform is that for a 0-hyperbolic G, the distance function $\mathrm{d}^{(x)}$ is an *ultrametric*. That is, $\mathrm{d}^{(x)}(u,v) \leq \max\{\mathrm{d}^{(x)}(u,y), \mathrm{d}^{(x)}(y,v)\}$ for every three vertices u,v,y [Ban90]. Put in simpler terms, the above property just says that in a tree rooted at x, if we denote by $r_{s,t}$ the lowest common ancestor between s and t, then for every u,v,y we have that $\mathrm{d}(x,r_{uv}) \geq \min\{\mathrm{d}(x,r_{uy}), \mathrm{d}(x,r_{vy})\}$.

Hyperbolicity of a graph can be seen as a measure of the closeness of its Farris transform to an ultrametric. We can formalize it as follows.

Definition 3 ([Gro87]). A connected graph $G=(V,E)$ is δ-hyperbolic if and only if for every 4-tuple $u,v,x,y \in V$, we have $\langle u,v\rangle_x \geq \min\{\langle u,y\rangle_x, \langle v,y\rangle_x\} - \delta$.

A proof of the equivalence between Definitions 1 and 3 can be found, *e.g.*, in [AJ13]. The two of them use a characterization of metric trees, and they define δ-hyperbolic graphs by relaxing these characterizations. The same can be done with other characterizations of metric trees, but then the corresponding values so obtained may not equal the hyperbolicity of the graph. Nonetheless, as seen in the following Section 2.3.2, they can only differ from the hyperbolicity by a constant-factor.

2.3.2 Reformulation of hyperbolicity

In what follows, we will complete the picture by presenting some of the alternative definitions for graph hyperbolicity. They are useful in order to prove some properties of δ-hyperbolic graphs. On the way, we will report on known relationships between these definitions (Table 2.1). We deem it as an important task. Indeed, the use of multiple definitions quickly lead to large constant-factors in the proofs, with negative consequences on the analysis of some graph algorithms [CCPP14].

Note that except for Section 2.3.2.2, we will not use these alternative definitions in what follows. Therefore, this part can be read independently from the remaining of the chapter. In what follows, some of the reformulations of hyperbolicity will be grouped together when they can be defined in a similar fashion.

2.3.2.1 Definitions with triangles

Let us start from the definition given in previous Section 2.2. First, we recall that a geodesic triangle $\Delta(u, v, w)$ is the union of three shortest-paths $\mathcal{P}_{uv}, \mathcal{P}_{vw}, \mathcal{P}_{wu}$ with respective ends u and v, v and w, w and u. The above shortest-paths are called the sides of the triangle.

Definition 4 (Rips condition, [Gro87, BH11])**.** A connected graph $G = (V, E)$ has δ_0-slim triangles if and only if for every geodesic triangle $\Delta(u, v, w)$, for every vertex $x \in \mathcal{P}_{uv}$, we have that $\mathrm{d}_G(x, \mathcal{P}_{vw} \cup \mathcal{P}_{wu}) \leq \delta_0$.

In order to see the relationship between Definitions 1 and 4, the following construction was proposed in [SG11].

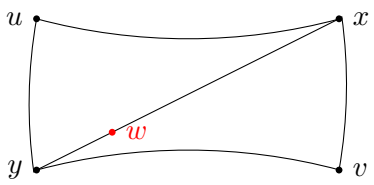


Figure 2.10: Split of a 4-tuple in two triangles. The vertex w is chosen so that $\mathrm{d}(x, w) = \mathrm{d}(v, x) - \lfloor \langle x, y \rangle_v \rfloor$, and so, $\mathrm{d}(y, w) = \mathrm{d}(v, y) - \lceil \langle x, y \rangle_v \rceil$.

Let u, v, x, y be any 4-tuple satisfying $\mathrm{d}(u, v) + \mathrm{d}(x, y) \geq \mathrm{d}(u, x) + \mathrm{d}(v, y) \geq \mathrm{d}(u, y) + \mathrm{d}(v, x)$. We fix a shortest path between every two pairs of vertices in the 4-tuple, and then we use these paths in order to construct the two geodesic triangles $\Delta(u, x, y)$ and $\Delta(v, x, y)$. The gist of the construction is to show that the hyperbolicity of the 4-tuple depends linearly on the slimness of these two triangles. To show that, we choose a vertex $w \in \mathcal{P}_{xy}$ such that $\delta(u, v, x, y) \leq \delta(u, w, x, y) + \delta(w, v, x, y) + 1/2$ (see Figure 2.10 for an illustration). Finally, a clever analysis from [SG11] shows that when the triangles $\Delta(u, x, y)$ and $\Delta(v, x, y)$ are δ-slim, it implies $\delta(u, w, x, y) \leq \delta$, and in the same way $\delta(w, v, x, y) \leq \delta$. Therefore, if G has δ-slim triangles then it is $(2\delta + 1/2)$-hyperbolic and the bound is sharp, as shown in [SG11].

We refer to [BH11] for a proof that conversely, every δ-hyperbolic graph has 3δ-slim triangles.

Other definitions of hyperbolicity than Definition 4 can be stated in terms of geodesic triangles. We summarize some of them below.

In order to get a better intuition of the following definitions, we recall that hyperbolicity measures the closeness of a graph to a metric tree. Let us fix any

geodesic triangle $\Delta(u, v, w)$. The three vertices u, v, w can be isometrically embedded into a tree as follows. We map them to the three leaves u', v', w' of a star with center node $s \notin V$ so that the edges $\{s, u'\}, \{s, v'\}, \{s, w'\}$ have respective length $\langle v, w\rangle_u, \langle u, w\rangle_v, \langle u, v\rangle_w$. We refer to Figure 2.11 for an illustration.

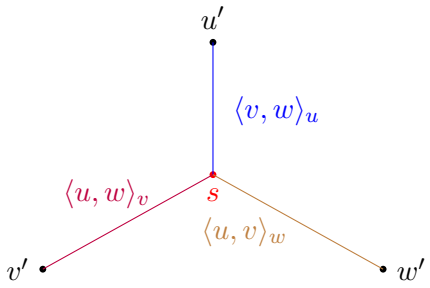


Figure 2.11: Isometric embedding of a 3-tuple to the leaves of a star (*a.k.a.*, tripod). We recall that $\langle v, w\rangle_u + \langle u, w\rangle_v = \mathrm{d}(u, v)$.

Then, by an appropriate subdivision of the three edges of the star, it can be obtained a tree T so that the shortest path $\mathcal{P}_{uv}$ (resp., $\mathcal{P}_{vw}$, resp., $\mathcal{P}_{wu}$) can be isometrically embedded to the unique $u'v'$-path in T (resp., $v'w'$-path, resp., $w'u'$-path). However, by doing so, some vertices in different sides of the triangle are mapped to the same node of T, and so, we aim at keeping small the distance in G between any two such vertices.

Definition 5 ([ABC+91, BH11, Gro87, GdLH90]). For every graph $G = (V, E)$ (with hyperbolicity $\delta(G)$), the following properties hold true:

- There exists $\delta_1(G) = \Theta(\delta(G))$ such that G has $\delta_1(G)$-*thin triangles*: for every triangle $\Delta(u, v, w)$ and for every $x \in \mathcal{P}_{uv},\ y \in \mathcal{P}_{uw}$ such that $\mathrm{d}(u, x) = \mathrm{d}(u, y) \leq \langle v, w\rangle_u$, we have that $\mathrm{d}(x, y) \leq \delta_1(G)$.
- There exists $\delta_2(G) = \Theta(\delta(G))$ such that G has triangles with *insize* at most $\delta_2(G)$: for every triangle $\Delta(u, v, w)$ and for every $x \in \mathcal{P}_{uv},\ y \in \mathcal{P}_{uw}$ such that $\mathrm{d}(u, x) = \mathrm{d}(u, y) = \lfloor \langle v, w\rangle_u \rfloor$[1], we have that $\mathrm{d}(x, y) \leq \delta_2(G)$.
- There exists $\delta_3(G) = \Theta(\delta(G))$ such that G has triangles with *girth* at most $\delta_3(G)$: for every triangle $\Delta(u, v, w)$, there exist $x \in \mathcal{P}_{uv}, y \in \mathcal{P}_{uw}, z \in \mathcal{P}_{vw}$ such that $\max\{\mathrm{d}(x, y), \mathrm{d}(x, z), \mathrm{d}(y, z)\} \leq \delta_3(G)$.
- There exists $\delta_4(G) = \Theta(\delta(G))$ such that: for every triangle $\Delta(u, v, w)$, there is some vertex $m \in V$ such that $\max\{\mathrm{d}(m, \mathcal{P}_{uv}), \mathrm{d}(m, \mathcal{P}_{uw}), \mathrm{d}(m, \mathcal{P}_{vw})\} \leq \delta_4(G)$.

Further geometric interpretation of the above definitions of hyperbolicity can be found, *e.g.*, in [BH11]. Interestingly, not all geodesic triangles need to be considered. In fact, we can constrain ourselves to "flat" triangles, *a.k.a.* bigons, and define hyperbolicity as follows:

[1]The ceiling ensures the distances to be integer values.

Definition 6. A graph $G = (V, E)$ has ε-thin bigons if for every $u, v, x, y \in V$ such that all of the following hold:

$$\mathrm{d}(u, v) = \mathrm{d}(u, x) + \mathrm{d}(x, v) = \mathrm{d}(u, y) + \mathrm{d}(y, v) \text{ and } \mathrm{d}(u, x) = \mathrm{d}(u, y)$$

we have $\mathrm{d}(x, y) \leq \varepsilon$.

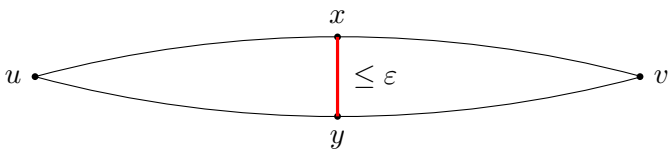


Figure 2.12: An ε-thin bigon.

We refer to Figure 2.12 for an illustration. Notice that when we take u, v, x, y as in the above Definition 6 then we obtain by the calculation $\delta(u, v, x, y) = \mathrm{d}(x, y)/2 \leq \varepsilon/2$. Therefore, a δ-hyperbolic graph has 2δ-thin bigons (see also Figure 2.3 and Property 1 in Section 2.2). Surprisingly, a converse relationship holds: if we subdivide once every edge in a graph G and the subdivided graph has ε-thin bigons, then G is $f(\varepsilon)$-hyperbolic for some (doubly exponential) function f [Pap95]. It is open whether f can be chosen as a linear function.

2.3.2.2 Cop and Robber games with different speeds

More recently, a game-theoretic characterization of hyperbolicity was proved.

A *Cop and Robber game* is a well-known two-player game that is played on a graph $G = (V, E)$. Classically, the two players are named the Cop and the Robber. At first, the Cop chooses any vertex $v_0 \in V$ as her position in the graph, then the Robber also chooses her initial position $u_0 \in V$. Then, the two players move sequentially, with the Cop playing first. At each turn $t \geq 1$, a player can either stay on her current position or move on an adjacent vertex.

The graph G is called *Cop-win* if whatever the Robber does, the Cop can end up on the same position as the Robber within a finite number of moves. Cop-win graphs have been characterized early in [NW83, Qui83]. Since then, several extensions of Cop and Robber games have been studied [Nis14]. One of them has a relationship with hyperbolicity.

Precisely, in this variant the Cop and the Robber move at different speeds s' (for the Cop) and s (for the Robber), with $s' \leq s$, where the speed of a player denotes the maximum distance in the graph between any two of its consecutive positions [CCNV11]. The graph G is called (s, s')-Cop-win if it is Cop-win in this variant. In particular, Cop-win graphs in the classical Cop and Robber game are exactly the $(1, 1)$-Cop-win graphs. Perhaps surprisingly, the values of s and s' for which a given graph G is (s, s')-Cop-win are related with its hyperbolicity. We first need to introduce the following dismantling orderings. We recall that throughout

this thesis, we will denote by $B_G(v, r)$ the ball of radius r centered on the vertex v in a given graph G.

Definition 7. An $(s, s')^*$-dismantling ordering of $G = (V, E)$ is a total ordering $(v_1, v_2, \ldots, v_n)$ of V such that for every $i < n$, we have $B_G(v_i, s) \cap \{v_i, v_{i+1}, \ldots, v_n\} \subseteq B_G(v_j, s')$ for some $j > i$.

It can be shown that every graph with an $(s, s')^*$-dismantling ordering is (s, s')-Cop-win. Conversely, if a graph is (s, s')-Cop-win, for some $s' < s$, then it has an $(s, s-1)^*$-dismantling ordering [CCPP14].

Lemma 8 ([CCPP14]). *Let $G = (V, E)$ be a graph.*

If G is δ-hyperbolic then it has a $(2r, r + 2\delta)^*$-dismantling ordering for every positive integer $r \geq 2\delta$.

Conversely, if G has a $(s, s')^*$-dismantling ordering, for some $s' < s$, then it has hyperbolicity at most $16(s + s') \left\lceil \frac{s+s'}{s-s'} \right\rceil + 1/2$.

One important byproduct of Lemma 8 is that every δ-hyperbolic graph G admits a $(4\delta, 4\delta)^*$-dismantling ordering, that is a classical dismantling ordering for its *graph power* $G^{4\delta}$ — obtained from G by adding an edge between every two distinct vertices that are at distance no more than 4δ in G. Simply put, if G is δ-hyperbolic then $G^{4\delta}$ is Cop-win. As we will show in Section 2.5.2, this original characterization of hyperbolicity is helpful in order to obtain new lower-bounds on this parameter.

2.3.2.3 Other definitions

In an attempt to make this part as exhaustive as possible, some other reformulations for graph hyperbolicity are now mentioned. These alternative definitions are not detailed, as it would require to introduce new technical notions that I feel to be unnecessary for the understanding of what follows. Below, the interested reader will be referred to some papers that are related with these alternative definitions.

Definition 9. The hyperbolicity of a graph G can be defined via the smallest parameters defining:

- its asymptotic upper curvature, denoted by κ (curvature) and c (an adjustment variable) [BF06];
- or a divergence function on its shortest-paths that is superlinear, denoted by $e(0)$ (initial value) and α (rate of divergence) [BH11] ;
- or a linear isoperimetric inequality, denoted by N (filling) and K [CCPP14].

Reformulations of hyperbolicity and their relationships with the standard definition are summarized in Table 2.1. In what follows, we name δ the hyperbolicity of the graph (w.r.t. Definition 1). The symbols that are used for each reformulation correspond to the ones that are given in the above definitions.

δ_0-slim triangles	$\delta_0 \leq 3\delta$ [AJ13]	$\delta \leq 2\delta_0 + 1/2$	[SG11]
δ_1-thin triangles	$\delta_1 \leq 4\delta$ [ABC+91]	$\delta \leq \delta_1$	[BH11]
insize δ_2	$\delta_2 \leq 12\delta$ [BH11]	$\delta \leq \delta_2$	[BH11]
δ_3	$\delta_3 \leq 12\delta$ [BH11]	$\delta \leq 3\delta_3$	[BH11]
δ_4	$\delta_4 \leq 12\delta + 1$ [BH11]	$\delta \leq 6\delta_4$	[BH11]
ε-thin bigons	$\varepsilon \leq 2\delta$ [Gro87]	$\delta = 2^{2^{\mathcal{O}(\varepsilon)}}$	[CN04]
$(s, s')^*$-dismantlable	$s \leq 2s' - 4\delta$ [CCNV11]	$\delta \leq 16(s + s') \left\lceil \frac{s+s'}{s-s'} \right\rceil + 1/2$	[CCPP14]
asymptotic upper curvature "κ, c"	$\kappa \leq -1/(4\delta^2)$ [BF06]	$\delta \leq \log 2/\sqrt{-\kappa} + c$	[BF06]
divergence function e	$e(0) \leq 4\delta,\ e(r) \geq 2^{\frac{r-4\delta-1}{12\delta}}$ [BH11]	$\delta \leq 6e(0) + 9\alpha$	[BH11]
(N, K)-filling	$N \leq 16\delta$ [ABC+91]	$\delta \leq 32KN^2 + 1/2$	[CCPP14]

Table 2.1: Comparison between the definitions of hyperbolicity. The first column is for the upper-bounds that are implied by δ for each reformulation. Conversely, the second column is for the upper-bounds on δ that are implied by each reformulation.

2.3.3 What is a "hyperbolic" graph ?

In the seminal work of Gromov [Gro87], hyperbolic graphs simply refer to the graphs with finite hyperbolicity. This definition makes sense since he studies on the hyperbolicity of Cayley graphs of finitely generated groups, that may and will be infinite. However according to the above definition, finite graphs are trivially hyperbolic in the sense that for every graph G, there exists a finite δ such that G is δ-hyperbolic. Thus, we shoud call the cycle C_n "hyperbolic" whereas it has hyperbolicity $\Omega(n)$!

In order to override this limitation, we can transpose the notion of hyperbolicity to *graph classes*. As a first attempt, let us define the hyperbolicity of a given graph class $\mathcal{G}$ as $\delta(\mathcal{G}) = \sup_{G\in\mathcal{G}} \delta(G)$. Then, we call $\mathcal{G}$ hyperbolic if $\delta(\mathcal{G}) < +\infty$. As expected, we have that the class of trees is hyperbolic, but the class of cycles is non hyperbolic. By abuse of notation, we refer by "hyperbolic graphs" for the graphs in a hyperbolic graph class.

In the literature [Ben98], a broader concept of hyperbolic graph class is preferred. It is based on the property that the hyperbolicity of a given graph is upper-bounded by its diameter (we shall come back to this relationship later on in Section 2.4) [WZ11]. The latter means that any graph G with diameter D_G is trivially D_G-hyperbolic, that does not really look satisfying. Indeed, we would prefer to call it hyperbolic only if $\delta(G) \ll D_G$.

Formally, let $\mathcal{G}$ be any class of graphs and let $\mathcal{G}_n = \{G_n \in \mathcal{G} \mid diam(G_n) = n\}$. Since graphs in $\mathcal{G}_n$ are trivially n-hyperbolic, the hyperbolicity $\delta(\mathcal{G}_n)$ is finite (by convention, $\delta(\emptyset) = 0$). Then, the graph class $\mathcal{G}$ is called hyperbolic if and only if $\lim_{n\to+\infty} \frac{\delta(\mathcal{G}_n)}{n} = 0$.

Further refinements of the concept have been suggested, *e.g.*, in [CFHM13]. They are listed in what follows.

Definition 10 ([CFHM13])**.** A given graph class $\mathcal{G}$ is called:

- *constantly hyperbolic* if $\delta(\mathcal{G}_n) = \mathcal{O}(1)$ (that corresponds to the case where $\delta(\mathcal{G})$ is finite);
- *(poly)logarithmically hyperbolic* if $\delta(\mathcal{G}_n) = \mathcal{O}(\log n)$ or $\delta(\mathcal{G}_n) = \log^{\mathcal{O}(1)} n$;
- *weakly hyperbolic* if $\delta(\mathcal{G}_n) = o(n)$;
- and *non hyperbolic* otherwise.

A shorter classification is adopted in [AD15]. Namely, a graph class is called hyperbolic in [AD15] only if it is logarithmically hyperbolic (w.r.t. Definition 10), and non hyperbolic otherwise. Furthermore, a graph class is called *strongly hyperbolic* in [AD15] if $\delta(\mathcal{G}_n) = \mathcal{O}(\log\log n)$.

Finally, we note that in [DKMY15], the authors consider a graph class to be hyperbolic only if it has the additional requirement that the graphs in the class have their maximum degree Δ that is constantly upper-bounded. By doing so, since the diameter of an n-vertex graph must be $\Omega(\log n/\log\Delta)$, there can be no constant upper-bound on the diameter in an infinite graph class, and so, we can dismiss all

the classes of bounded diameter graphs (that are trivially hyperbolic). As we will discuss next in Section 2.7, this choice presents algorithmic advantages.

2.4 Hyperbolic graph classes

The next two sections are devoted to the first objective in this study of hyperbolicity, *i.e.*, the characterization of hyperbolic and non hyperbolic graph classes. In particular, this section covers known upper-bound techniques on graph hyperbolicity. We list *sufficient* conditions for a graph class to be constantly hyperbolic. Examples of (hyperbolic) graph classes for which these conditions hold are given. We also provide examples of hyperbolic graphs that do *not* satisfy these conditions. The latter will show the limitations of these upper-bound techniques.

Outline of the section. In Section 2.4.1, we present upper-bounds depending on the best distortion of the distances in a graph when it is embedded in a tree. We also discuss about relationships between hyperbolicity and tree decompositions. Then in Section 2.4.2, we present two more upper-bounds on the hyperbolicity depending on the diameter and the chordality properties of the graph. We end up in Section 2.4.3 on personal contributions, showing upper and lower bounds on the variations of hyperbolicity that may be caused by various graph operations. The latter result is joint work with David Coudert.

2.4.1 Tree-likeness in graphs and hyperbolicity

We start presenting upper-bounds on the hyperbolicity that depend on the best possible distortion of the distances in a graph when it is embedded into a tree.

Indeed, we recall that hyperbolicity measures how close a given graph is to a metric tree. Unsurprisingly, there exists a strong relationship between this parameter and the (NP-hard) problem of embedding a given graph into a tree with minimum distortion (additive or multiplicative). In particular, as we showed in Section 2.3 the hyperbolicity $\delta(G)$ of a given graph G is the minimum possible δ such that every 4-tuple of vertices in G can be (non contractively) embedded into a tree with additive distortion at most δ. Therefore, $\delta(G)$ is a *lower bound* on the parameters:

- *tree-distortion* (minimum multiplicative distortion in a tree embedding);
- and *tree-stretch* (minimum t such that G admits a *tree t-spanner*, *i.e.*, a spanning tree with multiplicative distortion at most t).

These above relationships are described in the survey [AAD16] and the papers cited therein. Summarizing, we get the following upper-bounds on hyperbolicity:

Theorem 11 ([AAD16])**.** *Every graph with tree-distortion at most d is d-hyperbolic. Similarly, for every $t \geq 1$, every graph with a tree t-spanner is t-hyperbolic.*

2.4.1.1 Application: hyperbolic graph classes

Below, we give examples of graph classes that are (metrically) "tree-like", and so, hyperbolic.

Graphs with a tree t-spanner. By Theorem 11, for any fixed $t \geq 1$, the class of graphs with a tree t-spanner is constantly hyperbolic. The latter includes well-known classes such as: trees (trivially), interval graphs [LB62], split graphs [FH76], convex bipartite graphs [Glo67] and chordal bipartite graphs (*a.k.a.*, bipartite graphs with no induced cycle of length at least six) [GG78], etc.

Graphs with bounded tree distortion. Similarly, by Theorem 11 any class of graphs with bounded tree distortion is constantly hyperbolic. In particular, the classes of chordal graphs (graphs with no induced cycles of length at least four) and dually chordal graphs (*a.k.a.*, $(2,1)$-Cop win graphs, see Section 2.3.2.2) are constantly hyperbolic [Dir61, BDCV98]. It can be intuited (and, with slightly more work, formally proved) from the existence of their respective tree-representations, sometimes called the clique-tree (for chordal graphs) [Gav74] and the compatible tree (for dually chordal graphs) [DCG14].

2.4.1.2 Examples of hyperbolic graphs that are not "tree-like"

However, a converse of Theorem 11 does not hold : not all hyperbolic graphs have a constant tree-distortion or tree-stretch. In fact, these two parameters can differ from $\delta(G)$ by at most a logarithmic or polylogarithmic factor [AAD16], and this is sharp. We illustrate this fact with the following construction in Figure 2.13, sometimes called a *ringed tree* [CFHM13].

The ringed tree $RT(k)$ is obtained from a rooted complete binary tree with k levels by connecting the vertices at the same level with a circle, that is constructed under rules that we now detail. Formally, we start from a complete binary tree, then we label the vertices as follows. The root is labeled 0, and the two children of a vertex labeled i are labeled $2i+1$ and $2i+2$. Finally, at each level $l \geq 0$, nodes are labeled from $2^l - 1$ to $2^{l+1} - 2$, and we add edges in order to obtain the cycle $(2^l - 1, 2^l + 1, \ldots, 2^l + i, \ldots, 2^{l+1} - 2)$.

As a side contribution of this thesis (not published elsewhere), we improve upon the best-known upper-bound on the hyperbolicity of ringed trees:

Lemma 12. $\delta(RT(k)) \leq 3$.

Proof. For every vertex v, let $\ell(v)$ be its level in the underlying rooted tree (its distance to the root). Suppose for the sake of contradiction that $\delta(RT(k)) > 3$. Let u, v, x, y be such that $\delta(u, v, x, y) > 3$ and $\ell(u) + \ell(v) + \ell(x) + \ell(y)$ is minimized. W.l.o.g., vertex u is on the lowest level, *i.e.*, $\ell(u) \geq \max\{\ell(v), \ell(x), \ell(y)\}$. As proved in [CFHM13], it implies that for every vertex w in a upper level $\ell(w) \leq \ell(u)$, there exists a shortest uw-path which first goes up for some time, then stays on the same

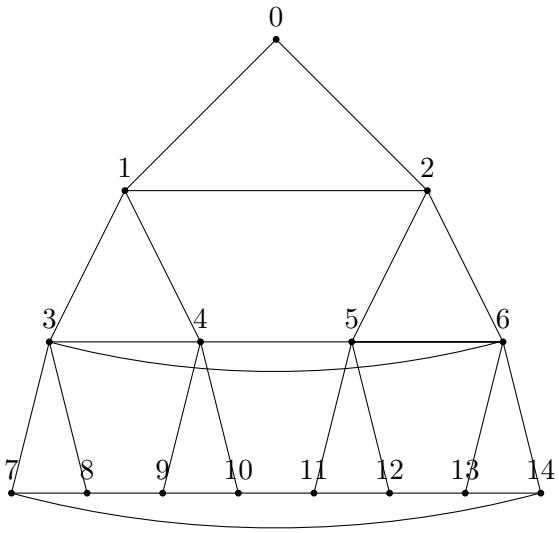


Figure 2.13: a ringed tree $RT(3)$.

level for at most three hops, and finally goes down. Indeed, this construction can be intuited by noticing that the two ends s and t of a "horizontal" st-path of length $p \geq 4$, staying on the same level $\ell(s) = \ell(t)$, can be connected via a path of length $\leq 2 + \lceil p/2 \rceil \leq p$ which first goes up for one hop, then stays at the same level $\ell(s) - 1$ and finally goes down for one hop. We call it a canonical shortest path.

Let us use the above property in order to prove the existence of some vertex of v, x, y that is at distance at most three from u. Indeed, let u' be the parent node of u in the underlying rooted tree. Since $\ell(u') = \ell(u) - 1$, we have by the minimality of $\ell(u) + \ell(v) + \ell(x) + \ell(y)$ that $\delta(u', v, x, y) \leq 3$. In this situation, we note that if it were the case that for any of v, x, y, there is a shortest path between this vertex and u passing by u', then it would follow from the 4-point Condition (Definition 1) that $\delta(u, v, x, y) = \delta(u', v, x, y) \leq 3$, that is a contradiction. So, let us assume w.l.o.g. that u' does not lie on any shortest uv-path. In particular, the canonical shortest uv-path does not go up, and so, $\ell(v) = \ell(u)$. Furthermore, since this path stays at most three hops on the same level, we get $\mathrm{d}(u, v) \leq 3$.

However, in this situation $\delta(u, v, x, y) \leq \mathrm{d}(u, v) \leq 3$ [SG11], that is a contradiction. Indeed, the upper-bound $\delta(u, v, x, y) \leq \mathrm{d}(u, v)$ can be seen as follows. As we observed earlier (Figure 2.11), the three vertices u, x, y can be embedded to the three leaves u', x', y' of an edge-weighted star S with null distortion. If we add a new leaf node v' that we make adjacent to u' in S, then by weighting $\mathrm{d}(u, v)$ the edge $\{u', v'\}$, one obtains a tree embedding of the 4-tuple with distortion at most $\mathrm{d}(u, v)$, and so, $\delta(u, v, x, y) \leq \mathrm{d}(u, v)$.

Altogether, $\delta(RT(k)) \leq 3$. □

Lemma 12 improves on [CFHM13], where they proved that $\delta(RT(k)) \leq 40$. It proves that we have a constant upper-bound on the hyperbolicity of any ringed tree.

In contrast, the following lemma shows that the tree distortion of a ringed tree can be arbitrarily large.

Lemma 13 ([Yan15]). *Any tree embedding of $RT(k)$ has distortion $\Omega(k)$.*

To have a better intuition of Lemma 13, we first observe that the underlying rooted tree of $RT(k)$ is a shortest-path tree. In a rooted tree T, the path between two vertices at same distance from the root r must pass by their lowest common ancestor, that is strictly closer from r. In contrast, all vertices at the same layer ℓ in $RT(k)$ can be connected via a circle, with only vertices at same distance ℓ from the root. Intuitively, it implies that in a (non expansive) tree embedding of $RT(k)$, the circles in each layer should be contracted to a single node [2]. Hence, the distortion of any tree embedding of $RT(k)$ should be at least the maximum distance between any two vertices at the same level, that is $\Omega(k)$ for the lowest level.

2.4.1.3 Relationship with tree decompositions

We complement Section 2.4.1 with relationships between hyperbolicity and tree decompositions [RS86], that are a more common way to measure tree-likeness in graphs. Formally, a *tree decomposition* $(T, \mathcal{X})$ of G is a pair consisting of a tree T and of a family $\mathcal{X} = (X_t)_{t \in V(T)}$ of subsets of V indexed by the nodes of T and satisfying:

- $\bigcup_{t \in V(T)} X_t = V$;
- for any edge $e = \{u, v\} \in E$, there exists $t \in V(T)$ such that $u, v \in X_t$;
- for any $v \in V$, $\{t \in V(T) \mid v \in X_t\}$ induces a subtree, denoted by T_v, of T.

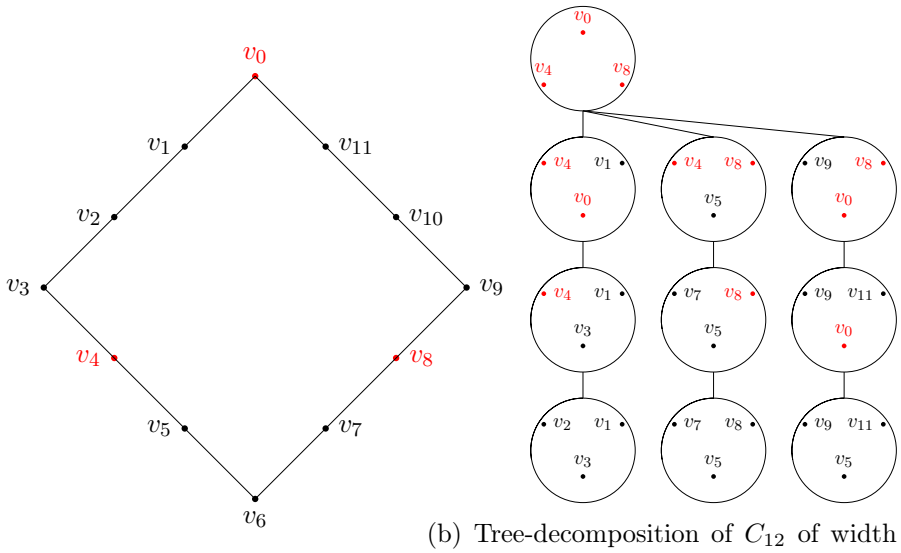
The sets X_t are called *the bags* of the decomposition. As an example, we give a tree decomposition of a cycle in Figure 2.14b.

A graph has *treewidth* at most k if it has a tree decompositions with bags of size at most $k + 1$. As an example, trees are exactly the graphs with treewidth 1.

Treewidth is a well-studied parameter [Bod06], and is generally accepted as a good measure of the structural tree-likeness in graph. In contrast, hyperbolicity is a measure of the metric tree-likeness in graphs, and as such it is uncomparable with treewidth. Indeed, as shown with Figure 2.14b, cycles have treewidth at most 2, whereas we proved in Section 2.3.1.2 that the hyperbolicity of cycles grows linearly with their size. Conversely, it is well-known that the complete graph K_n with n vertices has treewidth $n - 1$, whereas we proved in Section 2.3.1.2 that it has null hyperbolicity.

On the other hand, we can compare graph hyperbolicity with *treelength* [DG07] and *treebreadth* [DK14], that can also be defined in terms of tree decompositions. A graph has treelength at most l if it has a tree decomposition where the distance in the graph between any two vertices in a same bag is at most l. It has treebreadth

[2] This intuition can be formalized through the notion of *layering tree* [CD00], that will be further discussed in the next Section 2.7.



(a) Cycle C_{12} with twelve vertices.

(b) Tree-decomposition of C_{12} of width two and length four.

Figure 2.14: Cycles have treewidth two and treelength $\lceil n/3 \rceil$.

at most r if it has a tree decomposition whose every bag is contained in a ball of radius at most r (the center of the ball may not be in the bag). Treelength and treebreadth differ from tree distortion by at most a constant-factor, and so, they can be compared with hyperbolicity the same way [AAD16].

I will expand more on treelength and treebreadth in the next chapter on tree decompositions. In particular, I will show that in some cases where there is no large clique-minor and no long isometric cycle in the graph, treewidth can be compared with treelength (and so, with hyperbolicity) [CDN16].

2.4.2 Classical upper-bounds on hyperbolicity

In this subsection, we now survey two classical techniques in order to upper-bound graph hyperbolicity. Section 2.4.2.1 is devoted to the relationship between diameter and hyperbolicity. In Section 2.4.2.2, relationships between hyperbolicity and chordality properties of the graph are presented.

2.4.2.1 Diameter

As stated earlier, there is a standard upper-bound of graph hyperbolicity using the diameter of the graph.

Lemma 14 ([KM02, MP14, WZ11]). *For every graph $G = (V, E)$, we have $\delta(G) \leq \lfloor diam(G)/2 \rfloor$.*

A simple proof of Lemma 14 can be easily derived from the 4-point condition (Definition 1). Furthermore, we point out that since any graph G can be embedded

in a shortest-path tree with distortion $\mathcal{O}(diam(G))$, Lemma 14 is not that surprising. Of course, the converse of the lemma holds false, as easily seen with any path.

It follows that any class of graphs with constant upper-bound on the diameter is (trivially) constantly hyperbolic. Since the domination number and other domination-like parameters are themselves upper-bounds on the diameter, the authors in [HPR14] notice that the class of graphs with bounded domination number is also constantly hyperbolic.

We note that in [BCCM15, CCL15], it can be found variations of Lemma 14 (some of them using the eccentricity of the vertices, *i.e.*, the maximum distance in the graph between a given vertex and any other vertex).

2.4.2.2 Chordality

Much stronger upper-bounds on the hyperbolicity can be derived from the *chordality* of the graph. Namely, a k-chordal graph is a graph with no induced cycle of length at least $k+1$ [Ueh99]. In particular, 3-chordal graphs are exactly the usual chordal graphs. We recall that the class of chordal graphs is constantly hyperbolic [BKM01]. The result extends to the class of k-chordal graphs:

Theorem 15 ([CD00, WZ11]). *For every $k \geq 4$, every k-chordal graph G is $\lfloor k/2 \rfloor /2$-hyperbolic, and the bound is sharp.*

The converse of Theorem 15 holds false. As an example, consider a wheel W_n (obtained from the cycle C_n with n vertices by adding a universal vertex). On the one hand, it has diameter at most two and so, it has hyperbolicity at most 1 by Lemma 14. On the other hand, it is n-chordal.

Application: even more hyperbolic graph classes. By Theorem 15, the class of k-chordal graphs is constantly hyperbolic for every fixed $k \geq 4$. The latter encompass well-studied graph classes such as: chordal graphs (trivially), with well-known subclasses such as strongly chordal graphs [Far83]; weakly chordal graphs [Hay85]; AT-free graphs [COS97], and so, cocomparability graphs [GMT84] and permutation graphs [EPL72]; distance-hereditary graphs [BM86] and cographs [Sei74].

More recently, a result of the same flavour as Theorem 15 was proved in [MP15] with a different (and more technical) notion of chordality. Given $G = (V, E)$ and a cycle C in G, a *bridge* (or shortcut) of C is any shortest uv-path between two vertices $u, v \in C$ such that $\mathrm{d}_C(u, v) > \mathrm{d}_G(u, v)$. The bridge is called strict when it intersects the cycle C only in its two endvertices. Let $D_m(C) \subseteq V(C)$ contain the ends of all strict bridges of C of length at most m.

Then, a graph G is called *ε-densely (k, m)-path chordal* if for every cycle C with length at least k, every vertex in C is at distance at most ε from a vertex in $D_m(C)$ (see Figure 2.15 for an example). In particular, k-chordal graphs are $\lfloor k/2 \rfloor$-densely $(k, \lfloor k/2 \rfloor)$-path chordal [MP15].

Theorem 16 ([MP15]). *Every ε-densely (k, m)-path chordal graph has $(\max\{k/4, \varepsilon + m\})$-slim triangles.*

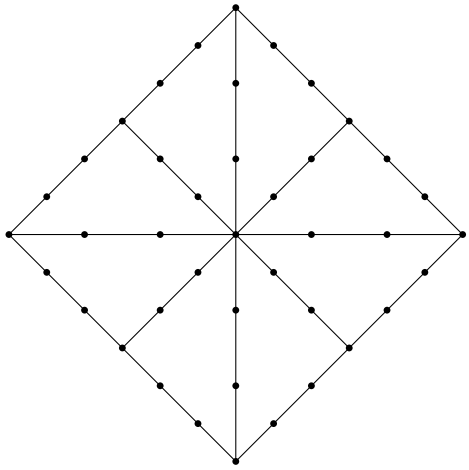


Figure 2.15: The uniform subdivision of the wheel is 3-densely $(9, 3)$-path chordal.

I confess that the impact of this result, compared to Theorem 15, is unclear to me.

2.4.3 Contribution: Graph operations and hyperbolicity

Finally, a generic framework is presented in order to prove that some graph operations preserve the hyperbolicity up to an additive term. In particular, this can be used in order to construct new hyperbolic graph classes from existing ones. Although we concentrate more on how to use this framework in order to prove that some graph classes are hyperbolic, it gives precise information on the variations of hyperbolicity that can be useful in a broader context (*e.g.*, in preprocessing and approximation algorithms for computing this parameter).

More precisely, new classes of hyperbolic graphs can be obtained from classes already known to be hyperbolic, by applying some graph operations such as line graphs [Whi92], clique graphs [Ham68], etc. In [CD16b], we designed a unifying framework in order to prove that these graph operations preserve hyperbolicity up to an additive term. The purpose of this work was to make simpler the computation of the sharp distortion of the hyperbolicity constant under these operations. It is based on two ingredients. The first is that the hyperbolicity of a given *bipartite graph* can be closely approximated (up to an additive term) by considering only one side of its bipartition.

Lemma 17. *Let $B = (V_0 \cup V_1, E)$ be a bipartite graph. For every $i \in \{0, 1\}$, let $G_i = (V_i, \{\{u, v\} \mid \mathrm{d}_B(u, v) = 2\})$.*

Then, $2\delta(G_i) \leq \delta(B) \leq 2\delta(G_i) + 2$ and the bounds are sharp.

It can be observed that since V_i is a dominating set of the bipartite graph G, we can relate every 4-tuple in G with a 4-tuple in G_i by substituting every vertex in V_{1-i} of the 4-tuple with any one of its neighbours. By doing so, we can use

the 4-point Condition directly (Definition 1) in order to prove a weaker version of Lemma 17. This weaker relationship between dominating set and hyperbolicity was already known and used in some algorithms for computing this parameter [CCL15]. In the case of bipartite graphs, the main technical difficulty was to obtain the sharp upper-bound on the distortion of hyperbolicity, which has required us a finer-grained analysis of the 4-tuples with maximum hyperbolicity in G.

The second property used in the framework is that for every $G = (V, E)$, since the distances in its j^{th} graph power are roughly divided by j, the hyperbolicity of this power is roughly $\delta(G)/j$.

Lemma 18. *For every graph $G = (V, E)$ and $j \geq 1$, we have $\frac{\delta(G)+1}{j} - 1 \leq \delta(G^j) \leq \frac{\delta(G)-1}{j} + 1$ and the bounds are sharp.*

Finally, we recall that an intersection graph over a ground-set S has for vertices a family of subsets in S together with an edge between every two intersecting subsets. It can be naturally represented as a bipartite graph, with vertices of the graph on one side and the elements of S on the other side. Combining the two above lemmas, we obtain our main result in [CD16b]:

Theorem 19. *For every graph $G = (V, E)$ and $j \geq 1$, let $\mathcal{S} = \{S_1, S_2, \ldots, S_k\}$ be a clique edge cover of G^j (a collection of cliques of G^j covering all its edges). Then the intersection graph $I_{\mathcal{S}}$, constructed from the subsets in $\mathcal{S}$ satisfies:*

$$\frac{\delta(G)+1}{j} - 2 \leq \delta(I_{\mathcal{S}}) \leq \frac{\delta(G)-1}{j} + 2.$$

Proof. We recall that every $S_i \in \mathcal{S}$ is a subset of V. Let $B_{\mathcal{S}}$ be the bipartite graph with sides V and $\mathcal{S}$, and with edge-set $\{\{v, S_i\} \mid v \in S_i\}$. By construction, two subsets $S_i, S_j \in \mathcal{S}$ are at distance two in $B_{\mathcal{S}}$ if and only if they intersect, that is if and only if $\{S_i, S_j\}$ is an edge of $I_{\mathcal{S}}$. Furthermore, since by the hypothesis $\mathcal{S}$ is a clique edge cover of G^j, two vertices $u, v \in V$ are at distance two in $B_{\mathcal{S}}$ if and only if $\{u, v\}$ is an edge of G^j. It follows by applying twice Lemma 17:

$$2\delta(I_{\mathcal{S}}) \leq \delta(B_{\mathcal{S}}) \leq 2\delta(I_{\mathcal{S}}) + 2,$$

$$2\delta(G^j) \leq \delta(B_{\mathcal{S}}) \leq 2\delta(G^j) + 2.$$

By mixing up the two chains of inequalities, one obtains $\delta(G^j) - 1 \leq \delta(I_{\mathcal{S}}) \leq \delta(G^j) + 1$. Then, by Lemma 18, it implies $\frac{\delta(G)+1}{j} - 2 \leq \delta(I_{\mathcal{S}}) \leq \frac{\delta(G)-1}{j} + 2$, as desired. □

The line graph and the clique graph of $G = (V, E)$, respectively denoted by $L(G)$ and $K(G)$, are respectively the intersection graph of its edges and of its maximal cliques. Therefore, Theorem 19 applies to these two typical graph operations by taking $j = 1$, which gives $\delta(G) - 1 \leq \delta(L(G)) \leq \delta(G) + 1$ and $\delta(G) - 1 \leq \delta(K(G)) \leq \delta(G) + 1$ for every graph G. These bounds are proved to be sharp in [CD16b]. In fact, we show in [CD16b] that for every possible $i \in \{-1, -1/2, 0, 1/2, 1\}$, there are graphs G_i and H_i such that $\delta(L(G_i)) - \delta(G_i) = i$ and similarly, $\delta(K(H_i)) - \delta(H_i) = i$.

Other graph operations to which the theorem applies are: the k-edge graph (intersection graph of the cliques of size k and the maximal cliques of size at most $k-1$ [Pri94]) with $j = 1$, the middle graph (intersection graph of the cliques of size at most two [Pri95]) with $j = 1$, the biclique graph (intersection graph of the maximal induced complete bipartite subgraphs [GS10]) with $j = 2$, etc. Furthermore, for all these above operations (except for line graph) these are the first bounds proved on the variations for hyperbolicity.

2.4.3.1 New classes of hyperbolic graphs

Finally, some new graph classes are proved to be constantly hyperbolic by using Theorem 19. A *clique-chordal graph* is a graph whose clique graph is chordal [BDCV98]. Since chordal graphs are 1-hyperbolic [BKM01], by Theorem 19 clique-chordal graphs are 2-hyperbolic.

Another example is the class of n-*convergent* graphs: $G = (V, E)$ is n-convergent if its n^{th} iterated clique-graph is a complete graph [LdMS98]. By iterating Theorem 19, we obtain that if G is n-convergent then $\delta(G) \leq \delta(K_{|V|}) + n = n$. Therefore, every n-convergent graph is n-hyperbolic.

2.4.4 Conclusion and open perspectives

Some classical graph parameters are shown to give upper-bounds on hyperbolicity in Sections 2.4.1 and 2.4.2. It would be very interesting to enrich this list. Similarly, it is now a growing topic to provide bounds on the variations for hyperbolicity that may be caused by various graph operations [MRSV10, CRS15]. In this aspect, it would be interesting to prove some new results in the spirit of Theorem 19.

2.5 Obstructions to hyperbolicity

In the continuity of Section 2.4, we now cover some known lower-bound techniques on graph hyperbolicity. The latter results will complete our first objective in the study of this parameter by giving characterizations for non hyperbolic graph classes, or equivalently *necessary* conditions for a graph to be hyperbolic. Like we did in Section 2.4, we will also provide examples of non hyperbolic graph classes that do *not* satisfy these characterizations, thereby showing the limitations of the lower-bound techniques.

Outline of the section. The rest of the section is divided as follows. First, I survey some results on the hyperbolicity of random graphs in Section 2.5.1. They show that, in some sense, most graphs are non hyperbolic. Then I present in Section 2.5.2 the typical obstructions that are used to show that a given graph class is non hyperbolic. These tools comprise: forbidden isometric subgraphs (Section 2.5.2.1), quasi-cycles (Section 2.5.2.2) and graph powers with some given properties (Section 2.5.2.4). Finally, some open problems are mentioned in Section 2.5.3.

My personal contributions: two new techniques using graph powers in order to lower-bound hyperbolicity, are presented in Section 2.5.2.4. This is joint work with David Coudert. Furthermore, as a side contribution of this thesis, I answer an open question from [VS14] on the relationship between hyperbolicity and quasi-cycles (Section 2.5.2.2).

2.5.1 Related work: random graphs are non hyperbolic

It is natural to ask for hyperbolicity, as for any graph parameter, what its typical value is on graphs. Put in other terms, the question is whether classes of random graphs are hyperbolic. The tendency is that, for a large spectrum of random graph models [CFHM13, NST15, Sha11, Sha12, Sha13, FGL+15, Tuc13, MP14, BHO+11], the graphs so obtained are non hyperbolic. The following results could be used in probabilistic methods in order to give lower-bounds on graph hyperbolicity.

In Sections 2.5.1.1 and 2.5.1.2, we emphasize on the results obtained on the hyperbolicity of the (classical) Erdös-Rény random graphs and the random regular graphs. We briefly mention the techniques used in the proofs of these results, that will be further detailed in Section 2.5.2. Then, Section 2.5.1.3 covers the known results on the hyperbolicity for other types of random graphs, and some open questions.

2.5.1.1 Erdös-Rényi random graphs

In particular, the most common model of random graphs is the Erdös-Rényi model $\mathcal{G}_{n,p}$, sometimes called the binomial random graph model. In a binomial random graph $G_n \in \mathcal{G}_{n,p}$, each possible edge exists with probability p. Note that p may, and usually does, depend on the number n of vertices in the graph.

It turns out that, for most regimes of p, the binomial random graphs are *non hyperbolic* with high probability. Precisely, the authors in [NST15] proved that in the sparse case $p = \mathcal{O}(1/n)$, binomial random graphs are non constantly hyperbolic. The latter result follows from the existence of arbitrarily long isometric cycles with positive probability (see Section 2.5.2.1). In a denser case where $p = 1 - \omega(1/n^2)$, Mitsche and Hell proved in [MP14] that binomial random graphs are non hyperbolic in the strong sense, *i.e.*, diameter-hyperbolic.

2.5.1.2 Random d-regular graphs

Similar results are obtained in [BHO+11, Tuc13] for the class $\mathcal{G}_{n,d}$ of random d-regular graphs with the uniform probability distribution, that are proved to be non hyperbolic in the strong sense (diameter-hyperbolic). In order to prove that these random graphs are non hyperbolic, the authors in [BHO+11] show the existence with high probability of large *quasi-cycles*. I shall come back in details on the notion of quasi-cycles when I present the known lower-bounds on graph hyperbolicity in Section 2.5.2.2.

2.5.1.3 Other random models of complex networks

Finally, since the above-mentioned models do not reflect well the structure of real-life graphs [BAJ00], it is interesting to ask whether random models of complex networks exhibit the same behaviour. Unfortunately, that seems to be the case.

In particular, it is proved in [CFHM13] that in most regimes, the random graphs that are obtained with the small-world model of Kleinberg are either non hyperbolic or non polylogarithmically hyperbolic. Some range of random graphs that are obtained with the Chung-Lu model are proved to be non constantly hyperbolic in [Sha13].

Perspectives. Surprisingly, we are not aware of any lower-bound on the hyperbolicity of Barabási-Albert random graphs (this problem has been studied only through experimentations [JLB08]). Furthermore, to find a pertinent class of random graphs that is hyperbolic – reflecting the properties of real-life networks such as the graph of the Autonomous of the Internet, that has a small hyperbolicity [CCL15, dMSV11] – is to my mind an important open question. In particular, the HOT model [FKP02] may be worth studying since it has been first defined to generate random trees.

2.5.2 Lower-bounds on the hyperbolicity

The remaining of the section will be devoted to a detailed presentation of the known lower-bound techniques on graph hyperbolicity, some of them have been briefly mentioned in our survey on the hyperbolicity of random graphs in Section 2.5.1. In Section 2.5.2.1, we present a basic technique in order to lower-bound hyperbolicity using isometric subgraphs. Next, we introduce quasi-cyclicity in Section 2.5.2.2, and as a side contribution of this thesis, we answer an open question from [VS14] on its relationship with graph hyperbolicity. Other personal lower-bound techniques, that are based on a game-theoretic characterization of hyperbolicity in [CCNV11], are finally presented in Section 2.5.2.4. The results in this last section are joint work with David Coudert.

2.5.2.1 Forbidden isometric subgraphs

We say that a graph parameter Π is closed under taking subgraphs if for every graph G and for every subgraph H of G, $\Pi(H) \leq \Pi(G)$. We now discuss on the stability of hyperbolicity under taking subgraphs.

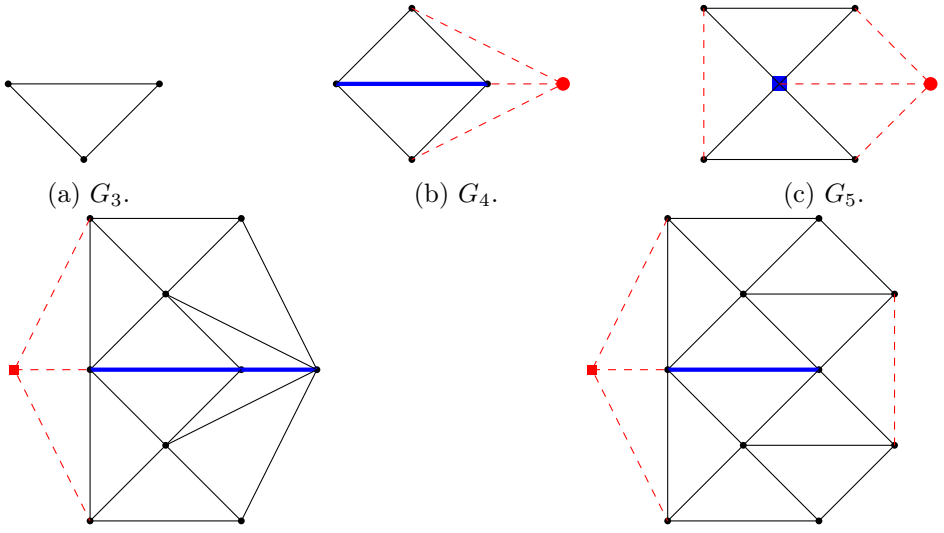
Unlike many graph properties, hyperbolicity is not closed under taking subgraphs. That can be easily seen with the complete graph K_n, that is 0-hyperbolic and contains all possible n-vertex graphs as a subgraph. It is not closed under taking *induced* subgraphs either. Indeed, every graph G is the induced subgraph of a 1-hyperbolic graph G' with diameter two, obtained from G by adding a universal vertex u (the shortest-path tree of G' rooted at u is a star with additive distortion of the distances in G' at most one). However, we recall that a subgraph H

of $G = (V, E)$ is called *isometric* if it is distance-preserving, *i.e.*, the distance between every two vertices in H is the same in H as in G. By the 4-point Condition (Definition 1), it implies that $\delta(H) \leq \delta(G)$ for any isometric subgraph H of G. Hence, a classical technique in order to lower-bound the hyperbolicity is to exhibit an isometric subgraph from a well-known non hyperbolic graph class, such as *e.g.*, cycles and grids.

As an example, recall that the *girth* of a given graph G, denoted by $g(G)$ in what follows, is a well-known parameter that is the minimum length of a cycle in G. By minimality of its length, any cycle with length $g(G)$ is isometric, and so, the hyperbolicity can be lower-bounded using the girth:

Lemma 20 ([WZ11]). *For every $G = (V, E)$, we have $\delta(G) \geq \lfloor g(G)/4 \rfloor - 1/2$ if $g(G) \equiv 1 \mod 4$, and $\delta(G) \geq \lfloor g(G)/4 \rfloor$ otherwise.*

It follows that in order for a graph class to be constantly hyperbolic, the graphs must have a girth that is constantly upper-bounded. Actually, the length of *any* isometric cycle in the graphs must be constantly upper-bounded. This is a strictly stronger condition since there are graphs with bounded girth and arbitrarily large isometric cycles. I illustrate this fact with the construction of Figure 2.16, that is a side contribution of this thesis. Namely, the construction shows examples of planar graphs G_ℓ that are $(1, 1)$-*dismantlable* (see Section 2.3.2.2), and so, with girth tree, but with an isometric cycle of length ℓ.



(a) G_3. (b) G_4. (c) G_5.

(d) G_6. (e) G_7.

Figure 2.16: Examples of plane cop-win graphs G_ℓ such that their outerface is an isometric cycle of length ℓ. The graph G_ℓ is obtained from two copies of $G_{2\lfloor \ell/2 \rfloor - 1}$ by identifying a path on their respective outerface (drawn in thick blue), then adding a new dominated vertex on its outerface and additional edges (drawn in dashed red).

The graph G_ℓ of the construction satisfies a stronger property, that is, it admits

a planar embedding where the outerface is an isometric cycle of length ℓ. For every $i \geq 2$, G_{2i} and G_{2i+1} are obtained from two copies of G_{2i-1} as follows. We start identifying a path $\mathcal{P}_i$ on their outerface with length $i-1$ (for the even case $\ell = 2i$) or $i-2$ (for the odd case $\ell = 2i+1$). Then, let us fix one end v_i of $\mathcal{P}_i$. In each of the two copies of G_{2i-1}, v_i has one neighbour on the outerface that is not part of $\mathcal{P}_i$. We add a new vertex of degree three that is made adjacent to v_i and to its two neighbours $u_i, u_i' \notin \mathcal{P}_i$ on the outerface in each copy. Note that the closed neighbourhood of this new vertex is dominated by v_i by construction. Furthermore, in doing so, we obtain in the even case $\ell = 2i$ an outerface which is an isometric cycle of length $2(2i-1) - 2|\mathcal{P}_i| + 1 = 2i = \ell$. Finally, in order to complete the construction in the odd case $\ell = 2i+1$, we consider the second end of $\mathcal{P}_i$ and we make adjacent its two neighbours $x_i, x_i' \notin \mathcal{P}_i$ on the outerface in each copy.

Note that on the other hand, not every graph with bounded-length isometric cycle has small hyperbolicity. For instance, the hexagonal grid with n columns and m rows (cf. Figure 2.17) is a *bridged graph* – *i.e.*, with no isometric cycle of length at least four – yet it is $(\min\{n,m\}-1)/2$-hyperbolic [CD16a].

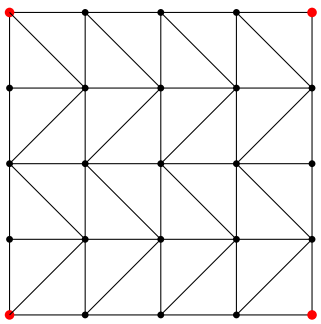


Figure 2.17: Hexagonal grid.

2.5.2.2 Quasi-cycles

We now describe quasi-cyclicity and its relationship with hyperbolicity. A lower bound technique is derived from the relationship, that is successful in some cases where we fail exhibiting an isometric cycle (e.g., grid-like graphs). Namely, in [VS14], Verbeek and Suri relax the notion of isometric cycles to the one of (weak) quasi-cycles. Given $G = (V, E)$, a cycle C of length n is an (α, β)-quasi-cycle if for every $u, v \in C$ such that $\mathrm{d}_C(u, v) \geq \beta n$ we have that $\mathrm{d}_G(u, v) \geq \alpha\, \mathrm{d}_C(u, v)$. Verbeek and Suri have proved in [VS14] that every graph G has an $(\alpha, 1/3)$-quasi-cycle of length $\Omega(\delta(G))$, for some constant α independent from $\delta(G)$. Therefore, the existence of large quasi-cycles is a necessary condition for a graph to have a large hyperbolicity.

They proved the condition to be sufficient when $\alpha > 1/2$. Indeed, an easy application of the 4-point Condition (Definition 1) shows that in this situation, the graph has hyperbolicity at least $\Omega((\alpha - 1/2)n)$ [VS14].

Answering an open question from [VS14], we now prove more cases where the existence of large quasi-cycles implies a large hyperbolicity. The latter result is a side contribution of this thesis that has not been published elsewhere.

Lemma 21. *For every $\alpha \leq 1, \beta \leq 1/3$, if $G = (V, E)$ has an (α, β)-quasi-cycle of length n then $\delta(G) = \Omega\left(\alpha^2 n\right)$.*

Proof. We give an illustration of the proof with Figure 2.18. For simplicity, we will ignore the ceilings in the proof.

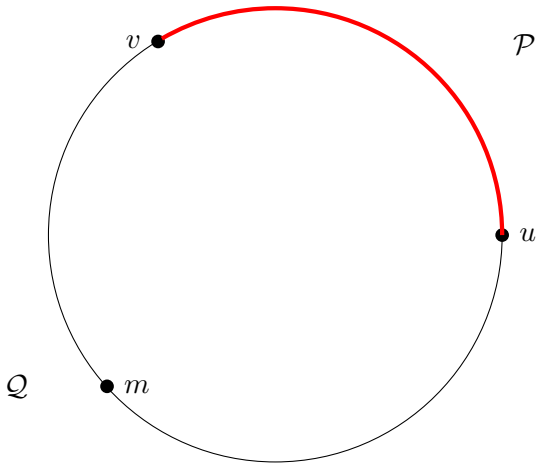


Figure 2.18: Proof of Lemma 21.

Let C be an (α, β)-quasi-cycle of length n, which exists by the hypothesis. Let us pick $u, v \in C$ such that $\mathrm{d}_C(u, v) = n/3$. We can partition the cycle C into two uv-paths $\mathcal{P}, \mathcal{Q}$ of respective length $n/3$ and $2n/3$. In this situation, since C is assumed to be an (α, β)-quasi-cycle and $\beta \leq 1/3$, we have $\mathrm{d}_G(u, v) \geq \alpha n/3$, and so, $\mathcal{P}$ and $\mathcal{Q}$ are $(\frac{2}{\alpha}, 0)$-almost shortest uv-paths. Then, let $m \in \mathcal{Q}$ be a middle-vertex, *i.e.*, chosen such that $\mathrm{d}_C(m, u) = |\mathcal{Q}|/2$. By the choice of m, $\mathrm{d}_C(m, \mathcal{P}) = \mathrm{d}_C(m, u) = n/3$. Furthermore, since $\beta \leq 1/3$, it implies $\mathrm{d}_G(m, \mathcal{P}) \geq \alpha n/3$. However, recall that in a hyperbolic graph, almost shortest-paths stay close to each other. Precisely, the Hausdorff distance between $\mathcal{P}$ and $\mathcal{Q}$ is an $\mathcal{O}\left(\delta(G)/\alpha\right)$ [Shc13b, GdLH90]. In particular, we have $\alpha n/3 \leq \mathrm{d}_C(m, \mathcal{P}) = \mathcal{O}\left(\delta(G)/\alpha\right)$. Altogether, $\delta(G) = \Omega\left(\alpha^2 n\right)$. □

2.5.2.3 Graph expansion

Other lower-bounds can be deduced from the existence of a core in graphs with small hyperbolicity[3]. Namely, we now present lower-bound techniques for hyperbolicity that are based on graph expansion (defined below). Lower-bounds are more complex to derive with this technique than with isometric subgraphs and quasi-cycles.

[3]The following result can also be intuited with another property of hyperbolic graphs, that is called the exponential divergence of shortest-paths [BH11].

The expansion of $G = (V, E)$, sometimes called the Cheeger constant, is the largest h such that for every subset S with at most $|V|/2$ vertices, there are at least $h|S|$ edges of G with one end in S and the other end in $V \setminus S$. The graphs in a class $\mathcal{G}$ are *expander* if there exist constants h, Δ such that every $G \in \mathcal{G}$ has maximum degree at most Δ and expansion at least h [HLW06]. The authors in [Ben98, Mal15] proved that expander graphs are non hyperbolic.

Theorem 22 ([Mal15])**.** *For every h, Δ, there exists a constant $C_{\Delta,h}$ such that every $G = (V, E)$ with maximum degree at most Δ and expansion at least h has hyperbolicity at least $C_{\Delta,h} \cdot \log(|V|)$.*

Intuitively, Theorem 22 can be explained as follows. In an expander graph with diameter D, since the number of vertices is exponential in D, removing a ball of radius $\Theta(D)$ will only remove a sublinear number of vertices, that does not affect too much the expansion. In particular, the order of magnitude of the diameter stays $\Theta(D)$, and so, the removal of the ball can only increase the distances by at most a constant-factor. In contrast, in a δ-hyperbolic graph there must be a core, *i.e.*, a ball of radius $\mathcal{O}(\delta)$ intersecting the (almost) shortest-paths between half of the pairs of vertices [CDV16]. By removing a core, one could increase the distances by more than any fixed constant-factor. This forces the core to have radius $\Omega(D)$, and so, the hyperbolicity of a given expander graph must scale with its diameter.

2.5.2.4 Contribution: Using dismantlable graph powers

Finally, we show how to use the game-theoretic characterization for hyperbolicity that has been proved in [CCPP14] in order to obtain new non-trivial lower-bounds on this parameter. New examples of non hyperbolic graph classes will be derived from these techniques. The results in what follows are joint work with David Coudert.

We refer to Section 2.3.2.2 for the game-theoretic characterization of hyperbolicity. Recall that for every $j \geq 1$, the j^{th} power of $G = (V, E)$ is the graph G^j that is obtained from G by adding an edge between every two distinct vertices u, v such that $\mathrm{d}_G(u, v) \leq j$. If G is δ-hyperbolic for some $\delta > 0$, then by Lemma 8 G has a $(4\delta, 4\delta)$-dismantlable ordering [CCNV11]. The latter is a (classical) dismantling ordering for its power $G^{4\delta}$, hence $G^{4\delta}$ is a Cop-win graph. Conversely, disproving that G^j is Cop-win, for some range of j, will give lower-bounds on $\delta(G)$. This approach is used in [CD16a] in order to prove that most underlying graphs of the data center interconnection networks are non hyperbolic.

We start this section with additional properties of Cop-win graphs. They will be used in what follows.

Required background. Let us recall that an endomorphism of $G = (V, E)$ is an edge-preserving mapping $\sigma : V \to V$.

Lemma 23 ([AF84])**.** *If $G = (V, E)$ is a connected dismantlable graph that is regular then G is a complete graph.*

Lemma 24 ([BCF94]). *If $G = (V, E)$ is a connected dismantlable graph then it has the clique invariant property: for every endomorphism σ of G, there is a nonempty clique C of G such that $\sigma(C) = C$.*

Next, we present our lower-bound techniques.

New lower-bound techniques. Our contributions are summarized in Propositions 26 and 25. Given an endomorphism σ of $G = (V, E)$, let the mobility of σ be defined as $\min_v \mathrm{d}_G(v, \sigma(v))$. Then, generalizing the terminology of [DRB99], the *weak mobility* of G is the largest l such that G has an endomorphism with mobility l. Note that by Lemma 24, any tree (and more generally, any Cop-win graph) satisfies the clique-invariant property. Since a clique has diameter one, it follows that any tree (and more generally, any Cop-win graph) has weak mobility at most one. Based on this observation, we prove in [CD16a] that a large weak mobility implies a large hyperbolicity. Indeed, a weak mobility at least l can be shown to imply that no graph power $G^{l'}$, for $l' = \mathcal{O}(l)$, can satisfy the clique-invariant property. As a result, no such power can be a Cop-win graph by Lemma 24, and so, since $G^{4\delta(G)}$ must be Cop-win by Lemma 8, the latter implies that G must have hyperbolicity $\delta(G) = \Omega(l)$. Below, we formalize this intuition.

Proposition 25. *If $G = (V, E)$ has weak mobility $l \geq 2$ then $\delta(G) \geq \lceil l/2 \rceil /2$.*

Proof. We prove that $G^{l'}$ is not dismantlable for every $1 \leq l' \leq l - 1$. It implies by Lemma 8 that G is not δ-hyperbolic for any $\delta < l/4$, and so, since $\delta(G)$ is a half-integer, $\delta(G) \geq \lceil l/2 \rceil /2$. Indeed, since G has weak mobility l and every endomorphism of G is also an endomorphism of $G^{l'}$, the graph power $G^{l'}$ has weak mobility at least $\lceil l/l' \rceil \geq 2$. Therefore, $G^{l'}$ falsifies the clique invariant property, hence it is not dismantlable by Lemma 24. □

Then, we recall that in a tree, there exists a leaf ℓ, *i.e.*, a vertex of degree one. In this situation, let p be its unique neighbour. Clearly, every node at distance $d > 1$ from ℓ is at distance $d - 1$ from p. The latter means that for every tree T with diameter $D > 1$, its powers T^j are not regular for every $j < D$ (because for any ℓ on a diametral path, its parent p has at least one more neighbour than ℓ). Following this intuition, if a given graph G with diameter D has small hyperbolicity δ then there should exist a small constant $j_0 = \mathcal{O}(\delta)$ such that: for every $j_0 \leq j \leq D-1$, its graph power G^j is not regular. We formalize this intuition below, using Lemma 23.

Proposition 26. *Let $G = (V, E)$ and $2 \leq r \leq diam(G)$ be such that G^{r-1} is a regular graph. Then, $\delta(G) \geq \lceil r/2 \rceil /2$.*

Proof. Suppose for the sake of contradiction that $4\delta(G) < r$. In particular, G is $\lfloor (r-1)/2 \rfloor /2$-hyperbolic, and so, by Lemma 8, it has a $(2\lceil (r-1)/2 \rceil, r-1)^*$-dismantling ordering. The latter ordering is also a $(r-1, r-1)^*$-dismantling ordering, hence G^{r-1} is Cop-win. However, since G^{r-1} is assumed to be regular, it must be a complete graph by Lemma 23. The latter contradicts that $r - 1 < diam(G)$. As a result, $4\delta(G) \geq r$, as desired. □

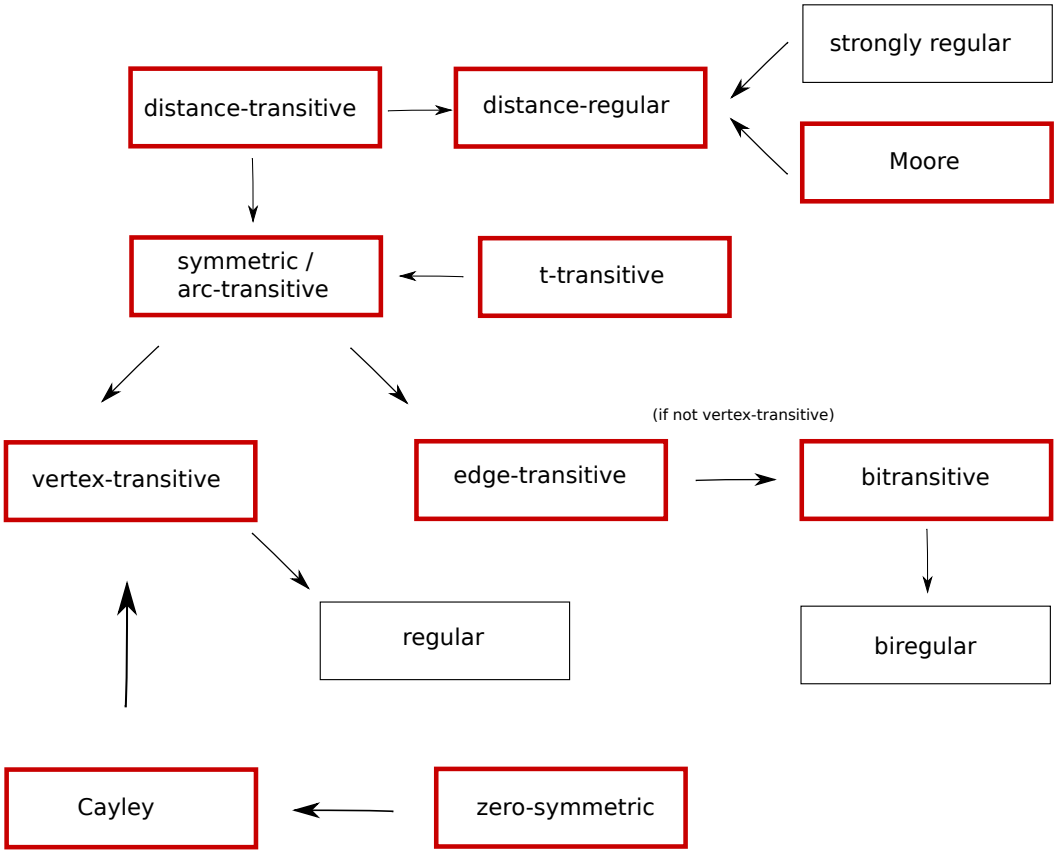


Figure 2.19: Relationships of inclusion between some graph classes. The rectangles for non hyperbolic graph classes (in red) are drawn thicker.

Application: non hyperbolic graph classes. We finally present some graph classes that can be proved to be non hyperbolic by using Propositions 26 and 25. To the best of our knowledge, these results are new, except for vertex-transitive graphs (defined below), of which we give a simpler proof they are non hyperbolic than in [BS12]. Furthermore, relationships of inclusion between the following graph classes are presented in Figure 2.19.

- We recall that an automorphism is a one-to-one endomorphism, and $G = (V, E)$ is *vertex-transitive* if for every $u, v \in V$, there is an automorphism mapping u to v.

  Note that most underlying graphs of data center interconnection networks that are proposed in the literature are vertex-transitive [AK89].
- A graph G is said to be *distance-regular* if it is a regular graph such that for every $i, j, k \geq 0$, there is some constant $c_{i,j,k}$ with the property that for every two vertices u and v at distance i in G, the number of vertices that are simultaneously at distance j from u and distance k from v in G is exactly $c_{i,j,k}$ [BH12].
- *Moore graphs* [Dam73] are a particular case of distance-regular graphs: namely,

an n-vertex d-regular graph is a Moore graph if $n = 1 + d \cdot \sum_{k=0}^{D-1}(d-1)^k$, with D being the diameter of the graph.

Theorem 27. *If a graph is vertex-transitive, distance-regular or Moore then it is non hyperbolic.*

Proof. Let G be a vertex-transitive graph. Since an endomorphism of G is also an endomorphism for every of its powers, it implies that if G is vertex-transitive then so are all its powers. Hence all the powers of G are regular graphs. Altogether, by Proposition 26 the hyperbolicity of G is constantly proportional to its diameter.

Similar arguments apply to distance-regular graphs and Moore graphs. Indeed, if a graph belongs to these classes then all its powers are regular [BH12]. Therefore, its hyperbolicity is constantly proportional to its diameter. □

A *bitransitive graph* is a bipartite graph such that for every two vertices u, v that are in the same side of the bipartition, there exists an automorphism mapping u to v. In the spirit of what is done for the framework presented in Section 2.4.3 (Lemma 17), let us pick one side of the bipartition and add an edge between every two vertices in this side that are at distance two. Then, the graph so obtained is vertex-transitive. This observation allows to prove that the class of bitransitive graphs, and so, the related classes of edge-transitive and nonedge-transitive graphs [GR13] are also non hyperbolic.

Refinements of Proposition 26 can lead to sharper lower-bounds on the hyperbolicity (but under stronger assomptions). In Table 2.2, we report on some results obtained with our lower-bound techniques (detailed in [CD16a]). For every graph in the table, the values of the diameter and the hyperbolicity are compared, with the two values only differing by at most a constant-factor in most cases. All these results are mainly obtained with Propositions 25 and 26, or some of their variations that are proved in [CD16a]. However, we also report on the hyperbolicity of grid-like graphs, on which these lower-bound techniques do not apply. We managed to obtain the *exact* value for the hyperbolicity of these graphs through a deeper analysis of their shortest-path distribution.

2.5.3 Open problems

So far, there are few reported lower-bounds on graph hyperbolicity. Finding new lower-bounds is an important open problem, that would improve our understanding of this parameter and could also help improving its computation. A related open problem is to prove some new lower-bounds on the hyperbolicity of random graph classes, such as Barabási-Albert random graphs and random geometric graphs in the Hyperbolic plane [KPK+10][4].

[4] Note that there exist duality results between these two random models [FCM14].

Name	Degree max.	Diameter	Order	δ
de Bruijn graph, $UB(d,D)$	$2d$	D	d^D	$\frac{1}{2}\left\lfloor\frac{D}{2}\right\rfloor \leq \delta \leq \left\lfloor\frac{D}{2}\right\rfloor$
Kautz graph, $UK(d,D)$	$2d$	D	$d^D(d+1)$	$\left\lfloor\frac{D}{4}\right\rfloor + \varepsilon \leq \delta \leq \left\lfloor\frac{D}{2}\right\rfloor$, $\varepsilon \in \{0,1\}$
Shuffle exchange, $SE(n)$	3	$2n-1$	2^n	$\frac{1}{2}\left\lfloor\frac{n}{2}\right\rfloor \leq \delta \leq n-1$
(n,m)-grid	4	$n+m-2$	nm	$\min\{n,m\}-1$
d-dimensional grid of size s	$2d$	$d(s-1)$	s^d	$(s-1)\left\lfloor\frac{d}{2}\right\rfloor$
Triangular (n,m)-grid	6	$n+m-2$	nm	$\frac{\min\{n,m\}-1}{2}$
Hexagonal (n,m)-grid	6	$\begin{cases} n-1+\left\lceil\frac{m-1}{2}\right\rceil & \text{when } m \leq 2n-1 \\ m-1 & \text{otherwise} \end{cases}$	nm	$\frac{\min\{n,m\}-1}{2}$
Torus (n,m)-grid	4	$\left\lfloor\frac{n}{2}\right\rfloor + \left\lfloor\frac{m}{2}\right\rfloor$	nm	$\left\lfloor\frac{1}{2}\left(\left\lfloor\frac{n}{2}\right\rfloor + \left\lfloor\frac{m}{2}\right\rfloor\right)\right\rfloor - 1 \leq \delta \leq \left\lfloor\frac{1}{2}\left(\left\lfloor\frac{n}{2}\right\rfloor + \left\lfloor\frac{m}{2}\right\rfloor\right)\right\rfloor$
Gen. hypercube, $G(m_1,\ldots,m_r)$	$\sum_{i=1}^{r} m_i - r$	r	$\prod_{i=1}^{r} m_i$	$\left\lfloor\frac{r}{2}\right\rfloor$
Cube Connected Cycle, $CCC(n)$	3	$2n-2+\max\left\{2,\left\lfloor\frac{n}{2}\right\rfloor\right\}$	$n2^n$	$n \leq \delta \leq n-1+\left\lfloor\frac{\max\left\{2,\left\lfloor\frac{n}{2}\right\rfloor\right\}}{2}\right\rfloor$
$\mathrm{BCube}_k(n)$	$\max\{n,k+1\}$	$2(k+1)$	$n^k(n+k+1)$	$k+1$
Fat-Tree_k	k	6	$\frac{k^2}{4}(k+5)$	2
Butterfly graph, $BF(n)$	4	$2n$	$2^n(n+1)$	n
k-ary n-fly	$2k$	$2n$	$k^n(n+1)$	n
k-ary n-tree	$3k$	$2n$	$k^{n-1}(n+k)$	$n-1$
Bubble-sort graph, $BS(n)$	$n-1$	$\binom{n}{2}$	$n!$	$\left\lfloor\frac{n(n-1)}{4}\right\rfloor$
Transposition graph, $T(n)$	$\binom{n}{2}$	$n-1$	$n!$	$\frac{1}{2}\left\lfloor\frac{n-1}{2}\right\rfloor \leq \delta \leq \left\lfloor\frac{n-1}{2}\right\rfloor$
Star graph, $S(n)$	$n-1$	$\left\lfloor\frac{3(n-1)}{2}\right\rfloor$	$n!$	$\left\lfloor\frac{1}{2}\left\lfloor\frac{3(n-1)}{2}\right\rfloor - \frac{1}{2}\right\rfloor \leq \delta \leq \left\lfloor\frac{1}{2}\left\lfloor\frac{3(n-1)}{2}\right\rfloor\right\rfloor$

Table 2.2: Bounds and exact value of the hyperbolicity of some graph classes [CD16a].

2.6 On computing the hyperbolicity of graphs

The remaining of this chapter is devoted to algorithmic and complexity problems. In particular, computational aspects of hyperbolicity will be covered in this section, thereby fulfilling our second main objective in the study of this parameter.

Motivations for an efficient computation of hyperbolicity are: to help characterizing the hyperbolic graph classes, or to measure the quality of approximations obtained with some graph heuristics (the latter will be further dicussed in Section 2.7) [VS14, CDE+08, CE07, EKS16, KL06, DKMY15].

By using the 4-point Condition (Definition 1), it is easy to see that the hyperbolicity of a given n-vertex graph can be computed in $\Theta(n^4)$-time. However, this too simple approach is prohibitive on large graphs, even when we use massively parallelization [ASHM13]. In what follows, improved algorithms for computing or approximating graph hyperbolicity will be sketched, with an emphasis on my personal contribution in this topic.

Note that we will only consider *finite* graphs in this section. Computing the hyperbolicity of *infinite* graphs is highly nontrivial. However, surprisingly, there exists a simple (approximation) partial algorithm for computing the hyperbolicity of the graph representations of finitely generated groups [Pap96].

Outline of the section. The best known algorithms for computing graph hyperbolicity are collected in Section 2.6.1. We sketch their basic principles and their limitations. Then, the next two Sections 2.6.2 and 2.6.3 are mostly centered on the contributions of this thesis.

In particular, the design and the analysis of some preprocessing methods for the computation of hyperbolicity are presented in Section 2.6.2. This part is largely devoted to personal contributions on the study of the relationships between the hyperbolicity of a graph and the maximum hyperbolicity from its *atoms* — *a.k.a.*, the subgraphs resulting from its decomposition by clique-minimal separators [BPS10] (Section 2.6.2.2). As a side contribution, I will also present a short analysis of the heuristic from [KNS13] (Section 2.6.2.1). Finally, conditional lower-bounds on the time complexity for computing graph hyperbolicity will be also mentioned in Section 2.6.3, including one of my own invention.

This is joint work with Nathann Cohen, David Coudert and Aurélien Lancin.

2.6.1 Related work

In this subsection, a state of the art on exact and approximate algorithms for computing the hyperbolicity of a graph is presented. We also comment on the limitations of these algorithms. In what follows, exact algorithms will be presented first (Section 2.6.1.1), then the approximation algorithms will be introduced by increasing approximation factor (Section 2.6.1.2).

2.6.1.1 Exact algorithms

Best known algorithm. The best known algorithm for computing the hyperbolicity runs in $\mathcal{O}(n^{3.69})$-time [FIV15]. It relates the computation of graph hyperbolicity with a variation of matrix multiplication.

Indeed, recall (Definition 3) that $G = (V, E)$ is δ-hyperbolic if and only if we have for every u, v, x, y that $\langle u, v\rangle_x \geq \min\{\langle u, y\rangle_x, \langle y, v\rangle_x\} - \delta$, where $\langle \cdot, \cdot \rangle_x$ denotes the Gromov product with base vertex x. In particular, let M_x be the $n \times n$ matrix such that $M_x[u, v] = \langle u, v\rangle_x$ for every $u, v \in V$. The $(\max, \min)$-product of M_x with itself is an $n \times n$ matrix denoted by $M_x \otimes M_x$ such that for every $u, v \in V$,

$$(M_x \otimes M_x)[u, v] = \max_{y \in V} \min\{M_x[u, y], M_x[y, v]\} = \max_{y \in V} \min\{\langle u, y\rangle_x, \langle y, v\rangle_x\}.$$

By Definition 3, G is δ-hyperbolic if and only if for every $x \in V$, all entries in $M_x \otimes M_x - M_x$ are lower than or equal to δ. Therefore, $\delta(G)$ can be computed with n computations of $(\max, \min)$-products.

Combinatorial algorithms. One drawback of the above algorithm is that it uses as a subroutine the best known algorithm for computing the (classical) matrix multiplication [DP09]. This algorithm requires quadratic-space and its time complexity $\mathcal{O}(n^{2.3729})$ hides a large constant-factor [LG14]. So, in order to compute hyperbolicity in practice on real-life graphs, *combinatorial algorithms* should be preferred.

In [CCL15], Cohen et al. base on the following simple, but elegant observation.

Lemma 28 ([CCL15]). *Let $G = (V, E)$ and $u, v, x, y \in V$ be such that $\mathrm{d}(u, v) + \mathrm{d}(x, y) \geq \max\{\mathrm{d}(u, x) + \mathrm{d}(v, y), \mathrm{d}(u, y) + \mathrm{d}(v, x)\}$. Then, $\delta(u, v, x, y) \leq \min\{\mathrm{d}(u, v), \mathrm{d}(x, y)\}/2$.*

The latter lemma gives a simple "cut-rule" in order to avoid considering all possible 4-tuples. Indeed, let us consider the 4-tuples u, v, x, y of $G = (V, E)$ by non increasing value of $\mathrm{d}(u, v) + \mathrm{d}(x, y)$. A lower-bound δ^* on the hyperbolicity $\delta(G)$ is maintained. By Lemma 28, every time the lower-bound improves, all 4-tuples such that $\min\{\mathrm{d}(u, v), \mathrm{d}(x, y)\} \leq 2\delta^*$ can be discarded. While this algorithm still runs in $\mathcal{O}(n^4)$-time, experiments have shown that it is much faster in practice.

Since then, additional cut-rules have been introduced in [BCCM15], which further speed-up the practical computation of hyperbolicity. So far, the hyperbolicity of graphs with tens of thousands of nodes can be computed within a reasonable amount of time. The true limitation of the algorithm comes from the storage in quadratic space of the distance matrix.

2.6.1.2 Approximation algorithms

Then, we report on the few existing approximation algorithms for computing hyperbolicity. The main message here is that these algorithms either have a large approximation factor (sometimes non constant) or they require the challenging best-known algorithm for computing matrix multiplication as a subroutine.

Using $(\max, \min)$-product. The simplest of these approximation algorithms reduces to the problem Hyperbolicity with fixed Base vertex: given $G = (V, E)$ and $x \in V$, compute $\delta_x(G) = \max_{u,v,y \in V} (\min\{\langle u, y\rangle_x, \langle y, v\rangle_x\} - \langle u, v\rangle_x)$. Note that $\delta(G) = \max_{x \in V} \delta_x(G)$. Furthermore, it can be proved using the triangular inequality that for every fixed $x \in V$, we have $\delta_x(G) \geq \delta(G)/2$ [Gro87]. As a result, solving the problem Hyperbolicity with fixed Base vertex gives a 2-approximation for computing hyperbolicity, and it can be done in $\mathcal{O}(n^{2.69})$-time by using the above-mentioned relationship with $(\max, \min)$-product [FIV15].

More recently, Duan has proved that the $(\max, \min)$-product can be computed faster when all entries in the matrices are bounded. Based on this result, he has described $(1 + \varepsilon)$-approximation algorithms for computing graph hyperbolicity, for every $\varepsilon \geq 0$ [Dua14].

Using Cop and Robber games. Another constant-factor approximation algorithm for computing this parameter was proposed in [CCPP14]. Roughly, given the distance-matrix of the graph (it can be precomputed in $\mathcal{O}(\min\{nm, n^{2.3729}\})$-time) this algorithm computes in $\mathcal{O}(n^2)$-time the smallest r such that the input graph has a $(4r, 3r)^*$-dismantling ordering. Altogether combined with the game-theoretic definition of hyperbolicity (Definition 7), the value gotten for r differs from the hyperbolicity by at most an (unfortunately large) constant-factor.

Using Tree embeddings. Finally, another approach for approximating the hyperbolicity is based on the relationships between this parameter and tree embeddings. Precisely, every δ-hyperbolic graph can be embedded into a tree with additive distortion of the distances at most $2\delta \log n$ [Gro87] (that will be further discussed in Section 2.7). In [FIV15], Fournier et al. notice that computing this tree embedding does not require the knowledge of the hyperbolicity. Therefore, an $\mathcal{O}(\log n)$-approximation algorithm for computing the hyperbolicity of a graph can be obtained in $\tilde{O}(n^2)$-time by computing this tree embedding, and then the resulting distortion of the distances in the tree[5].

2.6.2 Contribution of this thesis: Preprocessing

In order to overcome the current limitations for computing graph hyperbolicity (sketched above), it looks natural to seek for *preprocessing methods*, that aim at decreasing the size of the input and, possibly, at simplifying its structure. My main contribution in the field is the design and the analysis of some of these methods. I will first sketch a short analysis of the heuristic from [KNS13], before presenting my work on graph decompositions.

[5] The time complexity of this algorithm was proved in [FIV15]. However, the authors in [FIV15] assume that the distance matrix is given as input. We explain in [CD14] how to obtain the same time complexity for graphs encoded as adjacency lists.

2.6.2.1 Reducing the size of the graph by contracting matchings

In order to make tractable the approximate computation of hyperbolicity on large graphs, the authors in [KNS13] present a simple renormalization process. Put in more graph-theoretic terms, their process pick a maximal matching of the graph and then contract its edges. By doing so, the number of vertices is decreased by half. They repeat the process until the size of the graph is judged small enough in order to compute its hyperbolicity.

In what follows, we analyze the quality of this above heuristic for computing hyperbolicity. In order to do so, the hyperbolicity of a given graph G is compared with the hyperbolicity of its contraction minors (graphs obtained by contracting some edges of G), that is a study of independent interest.

Contraction minors and hyperbolicity. Although the distances in a graph cannot increase when we contract an edge, it turns out that, surprisingly, the hyperbolicity can do so. For instance, a cycle C_5 of length five is 1/2-hyperbolic, but contracting any one of its edges results in a cycle C_4 of length four, that is an 1-hyperbolic graph.

More generally, the following result is a side contribution of this thesis.

Lemma 29. *For every δ-hyperbolic n-vertex graph G, every contraction minor of G is $\mathcal{O}(\delta \log n)$-hyperbolic and this upper-bound is sharp.*

Proof. The upper-bound can be established by using the relationships between hyperbolicity and another tree-likeness parameter called *treelength* (see Section 2.4.1). Indeed, if G is a δ-hyperbolic n-vertex graph then it has treelength at least δ and at most $2\delta \log n + 1$ [AAD16]. The treelength is a contraction closed parameter. Therefore, every contraction minor of G must have hyperbolicity $\mathcal{O}(\delta \log n)$. The main difficulty is to prove the sharpness of the upper-bound.

In Figure 2.20, we illustrate this worst-case scenario with a ringed tree $RT(k)$ (previously introduced in Section 2.4.1). Note that this graph has $n = 2^{\mathcal{O}(k)}$ vertices, and in addition we have $\delta(RT(k)) \leq 3$ by Lemma 12[6]. So, every contraction minor of this ringed tree is $\mathcal{O}(k)$-hyperbolic. We aim at proving the existence of a contraction minor of $RT(k)$ with hyperbolicity $\Omega(k)$. the gist of the construction is to show that $RT(k)$ has a contraction minor H with a large *induced* (cylindrical) grid of dimensions $\Omega(k) \times \Omega(k)$. It can be constructed by fixing some level $\ell = \Theta(k)$ and then contracting on the cycles in each lower level the consecutive nodes with a common ancestor at level ℓ (*i.e.*, see Figure 2.20). Furthermore, since the graph is planar, it can be obtained an *isometric* (square) grid of comparable dimensions $\Omega(k) \times \Omega(k)$ by removing one third of the rows and one third of the columns on the borders. We recall that a grid of dimensions $\Omega(k) \times \Omega(k)$ has hyperbolicity $\Omega(k)$ [WZ11]. Altogether combined, this contraction minor H has hyperbolicity $\Omega(k)$, as desired. □

[6] This value of the hyperbolicity can be increased to some constant $\Theta(\delta)$ for every $\delta > 0$ by taking a uniform subdivision of $RT(k)$.

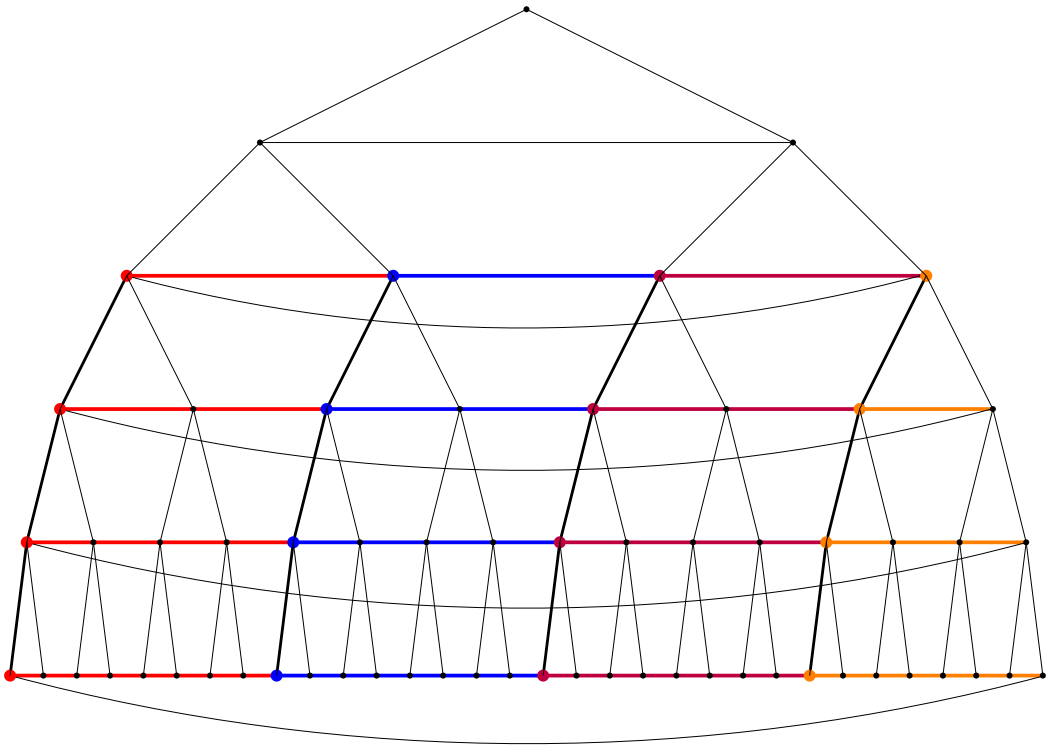


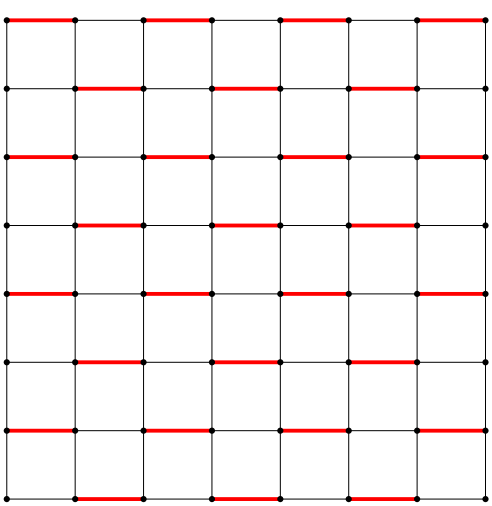
Figure 2.20: Construction of a cylindrical grid in $RT(k)$. We fix some level $\ell = \Theta(k)$ and then we contract on each lower level the nodes with a common ancestor at level ℓ. Paths contracted to a single node are delimited with thicker nodes.

Variations of hyperbolicity under one renormalization. The edge contractions in the renormalization process of [KNS13] are more controlled. Indeed, they must induce a matching. In this situation, let $\varphi : V(G) \to V(\hat{G})$ map every vertex of G to the corresponding vertex to which it has been contracted in the renormalized graph $\hat{G}$. We have that $\lfloor \mathrm{d}_G(u,v)/2 \rfloor \leq \mathrm{d}_{\hat{G}}(\varphi(u),\varphi(v)) \leq \mathrm{d}_G(u,v)$ for every $u, v \in V(G)$. So, it follows from the preservation of hyperbolicity under quasi-isometry [Shc13b, GdLH90] that $\delta(\hat{G}) = \Theta(\delta(G))$. The above Θ notation hides a large constant-factor that may be improved with a more in-depth analysis. Nonetheless, what can be shown is that there exist infinitely many graphs G such that $\delta(G) \geq 4\delta(\hat{G})$. We illustrate this fact with Figure 2.21.

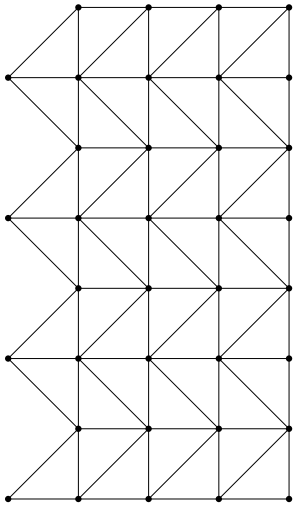
To summarize, it is my opinion that the confidence interval that is provided by the renormalization process is too large to give good estimates of graph hyperbolicity.

2.6.2.2 Relationship between clique-decomposition and hyperbolicity

Contrary to Section 2.6.2.1, the approach in this part rather consists in bounding the hyperbolicity of a given graph from the computation of the hyperbolicity of some of its subgraphs. Equivalently, given a decomposition of $G = (V, E)$ into some of its subgraphs, it is studied whether we can upper and lower bound $\delta(G)$ by using



(a) The square grid with side length n is $(n-1)$-hyperbolic.



(b) The renormalized square grid is $(n-1)/4$-hyperbolic.

Figure 2.21: Renormalization process on a Square grid. The edges contracted are drawn in thick red. Roughly, it gives a Hexagonal grid with twice less columns. Since the hyperbolicity of a Rectangular grid is twice larger than the hyperbolicity of a Hexagonal grid with same dimensions [CD16a], it shows that the renormalization process divides the hyperbolicity of a square grid by four.

the maximum hyperbolicity from the subgraphs. Let us motivate this approach and present existing results.

On the one hand, when G is "prime" (undecomposable w.r.t. the decomposition process), the input cannot be split, and so, we don't decrease the size of the input either. On the other hand, it happens that many interesting classes of real-life graphs are *not* prime. Furthermore, in all cases we gain more insights on the structure of the input.

Let us outline interesting byproducts of this decomposition approach:

- when every graph in a given class can be decomposed in "trivial" subgraphs, the class is proved to be constantly hyperbolic;
- for some other graph classes, the decomposition is a first step toward an efficient computation of the hyperbolicity in this class of graphs.

Related work. Of course, we need some structure on the graph decomposition in order to be able to prove something. Soto [SG11] has proved that two well-known graph decompositions can be used as a preprocessing step for computing graph hyperbolicity. Namely, these are the modular decomposition [Gal67] and its generalization the split-decomposition [Cun82] where informally, the graph is disconnected by using some edge-cutsets inducing a complete bipartite subgraph. More precisely, the hyperbolicity of a given graph is equal to the maximum hyperbolicity taken from the subgraphs output by these decompositions.

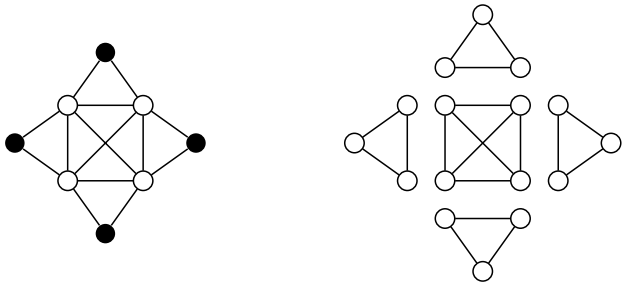


Figure 2.22: Clique-decomposition of a graph in five atoms. A 4-tuple with hyperbolicity 1 is drawn in bold.

Our main result. In a joint work with Nathann Cohen, David Coudert and Aurélien Lancin [CCDL17], we have proved similar results for the clique-decomposition [BPS10]. Given $G = (V, E)$, an atom of G is any subset $A \subseteq V$ such that there is no clique-separator in $G[A]$ and A is inclusion wise maximal w.r.t. this property. The clique-decomposition of G is the collection of its atoms. See Figure 2.22 for an example. It can be computed in $\mathcal{O}(|V||E|)$-time.

Theorem 30. *Given $G = (V, E)$, let $A_1, \ldots, A_k$ be its atoms. Then, $\max_i \delta(G[A_i]) \leq \delta(G) \leq \max_i \delta(G[A_i]) + 1$ and the bounds are sharp.*

Below, we detail further the proof of Theorem 30. It is based on two ingredients. The first is that disconnecting the graph with a separator of small diameter D can change the value of the hyperbolicity by at most an additive term $D/2$. This part requires a tedious analysis of the different types of 4-tuples in the graph in order to be proved.

Lemma 31 ([SG11])**.** *Given $G = (V, E)$, let $X \subseteq V$ be such that $G[X]$ is isometric and has diameter at most D. Then, let $C_1, \ldots, C_k$ be the connected components of $G \setminus X$, we have:*

$$\max_{1 \leq i \leq k} \delta(G[C_i \cup X]) \leq \delta(G) \leq \max\{D/2, \max_{1 \leq i \leq k} \delta(G[C_i \cup X])\} + D/2.$$

In [CCDL17], we give a proof of this result in the case of clique-separator ($D \leq 1$). Note that $G[X]$ must be isometric in order to ensure that the resulting subgraphs $G[C_i \cup X]$ are also isometric. Indeed, we recall that the hyperbolicity is not stable under taking induced subgraphs. However, we observe that when X is a clique-separator, the requirement for $G[X]$ to be isometric is always satisfied.

By Lemma 31, if we disconnect the graph with a small diameter separator then we can approximate the hyperbolicity up to an additive term. Unfortunately, these additive errors can add up when we further decompose the graph. We prove it is the case even for separators of diameter at most two [CCDL17]. However, in the special case of clique-separators, we can bound the final additive error with the following lemma.

Lemma 32. *Given $G = (V, E)$, let $u, v, x, y \in V$ satisfy $\delta(u, v, x, y) \geq 3/2$. There exists an atom A_0 intersecting all the paths between any two vertices of the 4-tuple.*

Proof. Let $(T, \mathcal{X})$ be a tree decomposition of G whose bags are the atoms of G. Such a tree decomposition was proved to exist in [BPS14]. In order to prove the lemma, it suffices to find an atom A_0 such that there is no more than one vertex of the 4-tuple u, v, x, y in each component of $G \setminus A_0$. We shall find an atom A_0 with the weaker property that no more than two vertices among $\{u, v, x, y\} \setminus A_0$ are in the same connected component of $G \setminus A_0$. Then, we will prove that in fact, there is no more than one vertex of the 4-tuple in each component, by elaborating on the property that $\delta(u, v, x, y) \geq 3/2$. First, in order to find the desired atom, we will weight the bags of $\mathcal{X}$ (we will then choose the atom A_0 in the *weighted centroid* of T).

Precisely, for every of u, v, x, y we pick an atom which contains it and we define the weight of an atom as the number of times it has been picked. In particular, an atom has weight between 0 and 4, and the sum of weight of the atoms is equal to $\mathcal{W} = 4$. It is well-known that for any node-weighted tree with sum of weights $\mathcal{W}$, there is a node whose removal splits the tree into connected components where the sum of weight of the nodes is at most $\mathcal{W}/2$ [Gol71]. So, let A_0 be an atom of G such that no component of $T \setminus \{A_0\}$ has the sum of weight of its bags greater than 2. We claim that $\forall s \in \{u, v, x, y\} \setminus A_0$, there is a clique-separator $X_s \subseteq A_0$ which separates s from $\{u, v, x, y\} \setminus \{s\}$, that will prove the lemma.

Indeed, let $s \in \{u, v, x, y\} \setminus A_0$ be arbitrary. By the properties of a tree decomposition, T_s (induced by the atoms containing s) is the subtree of a component C_s of $T \setminus \{A_0\}$. Let $V_s \subseteq V$ be the subset of vertices that are contained in an atom in C_s, and let $A_s \in C_s$ be the atom that is adjacent to A_0 in T. Since A_s and A_0 are atoms of G, their intersection, denoted by $X_s = A_s \cap A_0$, is a clique [BPS10]. Furthermore, by the properties of a tree decomposition, X_s is a is a separator of G that disconnects V_s from $V \setminus V_s$. Therefore, we are left to prove that no vertex of $\{u, v, x, y\} \setminus \{s\}$ is in V_s, for the latter will prove that X_s is a clique-separator which separates s from $\{u, v, x, y\} \setminus \{s\}$. Assume for the sake of contradiction the existence of a vertex $t \in \{u, v, x, y\} \setminus \{s\}$ that is contained in V_s. We distinguish between two cases.

- Suppose that $t \notin X_s$. In this situation, T_s, T_t are subtrees of C_s. It implies that the sum of weight of the atoms in C_s is at least 2, and so, by the choice of atom A_0, it is equal to 2. In particular, s and t are the only two vertices of the 4-tuple that are in $V_s \setminus X_s$ (else, the sum of weight of the atoms in C_s should be at least 3). However, we prove in [CCDL17] that in this situation, $\delta(u, v, x, y) \leq 1$, that contradicts the hypothesis that $\delta(u, v, x, y) \geq 3/2$. This part of the analysis makes use of our proof of Lemma 31 for the case of clique-separators.
- Else, $t \in X_s$ and we can assume w.l.o.g. that no vertex of $\{u, v, x, y\} \setminus \{s\}$ is in $V_s \setminus X_s$ (else, we go back to the previous case). However, we prove in [CCDL17], as before, that in this situation, $\delta(u, v, x, y) \leq 1$, that again contradicts the hypothesis that $\delta(u, v, x, y) \geq 3/2$.

As a result, no vertex of $\{u, v, x, y\} \setminus \{s\}$ is in V_s, and so, X_s is a clique-separator which separates s from $\{u, v, x, y\} \setminus \{s\}$. Since $X_s \subseteq A_0$, the latter proves the claim on A_0, hence the lemma. □

The gist of Lemma 32 is that the atoms of $G = (V, E)$ are the bags of a tree decomposition of G (this will be further discussed in the next chapter on tree decompositions). We use it in [CCDL17] in order to prove that the hyperbolicity of any 4-tuple with large hyperbolicity is at most one unit off from the hyperbolicity of a given atom, and so, Theorem 30 holds.

Further applications of clique-decompositions. On the way to prove Theorem 30, we were able to (partly) characterize the cases where the hyperbolicity of a graph cannot be deduced from its clique-decomposition directly. We leverage from this characterization the following result:

Theorem 33. *Given $G = (V, E)$, let $A_1, A_2, \dots A_k$ be its atoms. In $\mathcal{O}(|V||E|)$-time, we can compute $G_1^*, \dots, G_k^*$ such that:*

- *each G_i^* is obtained from $G[A_i]$ by adding simplicial vertices;*
- *and if $\delta(G) \geq 1$ then $\delta(G) = \max\{1\} \cup \{\delta(G_i^*) \mid 1 \leq i \leq k\}$.*

The above preprocessing method has been successfully applied on large co-authorship graph in order to compute their hyperbolicity. On a more theoretical side, we have used it in order to improve the computation of hyperbolicity for outerplanar graphs, *a.k.a.* the graphs whose atoms are cycles [Sys79]:

Theorem 34. *If $G = (V, E)$ is outerplanar then $\delta(G)$ can be computed in $\mathcal{O}(|V|)$-time.*

In order to prove Theorem 34, we have established a simple characterization of outerplanar graphs with hyperbolicity strictly less than one. More precisely, this characterization is based on the property that every induced cycle in an outerplanar graph is isometric [Sys79]. In particular, since every cycle of length at least six has hyperbolicity at least one [WZ11], every outerplanar 1/2-hyperbolic graph is 5-chordal. So, we obtain our characterization of outerplanar 1/2-hyperbolic graphs as a particular case of the characterization in [WZ11] of 1/2-hyperbolic 5-chordal graphs.

Then, for outerplanar graphs with hyperbolicity at least one, we have refined the results of Theorem 33. In particular, since the atoms of outerplanar graphs are cycles, the graphs $G_1^*, \dots, G_k^*$ output by the preprocessing method have a very simple structure (they are obtained from a cycle by adding, for every edge e in the cycle, at most one simplicial vertex that is adjacent to the two ends of e). So, their hyperbolicity can be derived from the hyperbolicity of cycles and additional parity conditions. Details can be found in our report [CCDL17].

Final remark: combining many decompositions. It may be the case that the atoms can be further split or reduced, using another graph decomposition. For instance, a graph is EPT if it is the edge intersection graph of paths in a tree [GJ85]. The atoms of an EPT graph are line graphs [Tar85]. So, we can replace each atom with its root (the graph of which it is the line graph), and we have by Theorem 19 that it does not affect their hyperbolicity by more than an additive term. Furthermore, computing the root of each atom can be done in linear time [Leh74].

Then, the roots of the atoms may be further decomposable using modular, split or clique decomposition, etc. If the root is prime under all these decompositions but it is a bipartite graph, we may still decrease its size by half as follows. We take the smaller side of its bipartition and we add an edge between every two vertices at distance two in the root. By Lemma 17, the hyperbolicity of the gotten graph is roughly half of the hyperbolicity of the root.

2.6.3 Hardness results

In the previous Section 2.6.2, we show that the computation of hyperbolicity (exact or approximate) can be sped up on certain graph classes by using graph decompositions. This approach does not extend to general graphs. So, a complementary approach is to prove, or show strong evidence of, lower-bounds on the complexity of computing this parameter. In this section, conditional lower-bounds on this complexity are presented, with an emphasis on a reduction from the Quadrangle Detection problem, that is part of my contributions.

2.6.3.1 Related work

As a warm-up, we recall that the problem Hyperbolicity with fixed Base vertex can be reduced in quadratic-time to the computation of a $(\max, \min)$-product between two matrices. In [FIV15], the authors prove that a converse reduction also holds true: if Hyperbolicity with fixed Base vertex can be solved in $\mathcal{O}(n^\upsilon)$-time on n-vertex graphs then the $(\max, \min)$-product of two $n \times n$ matrices can be computed in $\mathcal{O}(n^{2+\upsilon/3} \log n)$-time. In particular, any $\mathcal{O}(n^{2.05})$-time algorithm for solving Hyperbolicity with fixed Base vertex would immediately improve the best-known algorithms for $(\max, \min)$-product. These relationships suggest a strong equivalence between the computation of hyperbolicity and the $(\max, \min)$-product, that resembles the existing ones between all-pairs-shortest-paths and $(\min, +)$-product [FM71].

SETH-hardness. More recently, several authors have proved conditional lower-bounds on the complexity of polynomial-time problems on graphs under the Strong Exponential Time Hypothesis (SETH) [Wil16]. Roughly, the hypothesis says that SAT cannot be solved in $2^{(1-\varepsilon)n}$-time for any $\varepsilon > 0$ [IPZ98]. Under SETH it has been proved that computing the diameter of a graph cannot be done in truly subquadratic-time, even on sparse graphs; that is, it cannot be computed in $\mathcal{O}(n^{2-\varepsilon})$-time for any

$\varepsilon > 0$ [BCH16]. The authors in [BCH16] have used this result in order to prove conditional lower-bounds on the complexity of computing the hyperbolicity of a graph:

Theorem 35 ([BCH16]). *Under SETH, none of the following problems can be solved in truly subquadratic time, even on sparse graphs:*

- *computing the hyperbolicity of a given graph;*
- *deciding whether a given graph has hyperbolicity at most one.*

A similar but weaker result was proved by Fang in [Fan11].

2.6.3.2 Contribution of this thesis: Truly subcubic reduction to Quadrangle Detection

The concept of q-reduction was introduced by Williams and Vassilevska Williams in [VWW10]. Informally, if there is a q-reduction from a problem A to a problem B, and B can be solved in $\tilde{\mathcal{O}}(n^{q-\eta})$-time[7] for some $\eta > 0$, then problem A can be solved in $\tilde{\mathcal{O}}(n^{q-\varepsilon})$-time for some other $\varepsilon > 0$. More formally, a Turing reduction from a problem A to a problem B is an algorithm to solve A using an oracle to solve B as a soubroutine. It is called a q-reduction if for every $\eta > 0$ there exists ε such that the following holds for every input of size n:

- the reduction runs in $\tilde{\mathcal{O}}(n^{q-\varepsilon})$-time;
- and if the oracle to solve problem B is called on instances with respective sizes $n_1, n_2, \ldots, n_k$ then $\sum_{i=1}^{k} \tilde{\mathcal{O}}(n_i^{q-\eta}) = \tilde{\mathcal{O}}(n^{q-\varepsilon})$.

This concept formalizes prior work from, *e.g.*, [GO95, KS06a].

Two problems are called subcubic equivalent if every of the two problems can be 3-reduced to the other. In this situation, either both problems are solvable in truly subcubic time, or none of them is. My main contribution in [CD14], found with David Coudert, can be stated as follows.

Theorem 36. *The two following problems are subcubic equivalent:*

- *deciding whether a graph has hyperbolicity equal to* $1/2$*;*
- *deciding whether a graph contains an induced cycle of length four.*

Furthermore, both problems can be solved in deterministic $\mathcal{O}(n^{3.26})$*-time and in randomized* $\tilde{\mathcal{O}}(n^{2.3729})$*-time.*

Theorem 36 shows a surprising gap in the complexity of recognizing graphs with small hyperbolicity. Indeed, it has been proved in [How79] that the 0-hyperbolic graphs can be recognized in linear time. In contrast, recognizing 1/2-hyperbolic graphs in (deterministic) truly subcubic time seems to be a much harder task.

A reduction from Quadrangle detection to the recognition of 1/2-hyperbolic graphs has been sketched in earlier papers [KM02, WZ11]. So, the main difficulty was to show the converse reduction. Our proof for Theorem 36 makes use

[7] The $\tilde{\mathcal{O}}$ notation suppresses the polylog factors.

of a (non algorithmic) characterization of 1/2-hyperbolic graphs from Bandelt and Chepoi [BC03]. On the way to prove our result, we have established the following simpler characterization for these graphs. We recall that for every $G = (V, E)$ and $j \geq 1$, the graph power G^j is obtained from G by adding an edge between every two distinct vertices that are at distance at most j in G.

Definition 37. For every $G = (V, E)$, the graph $G^{[2]} = (V^{[2]}, E^{[2]})$ is defined as follows:

- $V^{[2]} \simeq V \times \{0, 1\}$;
- $G[V \times \{0\}] \simeq G$;
- $G[V \times \{1\}] \simeq G^3$;
- and for every $u, v \in V$, the vertices $(u, 0)$ and $(v, 1)$ are adjacent in $G^{[2]}$ if and only if $\mathrm{d}_G(u, v) \leq 2$. In particular, for every $u \in V$, there is an edge between $(u, 0)$ and $(u, 1)$ in $G^{[2]}$.

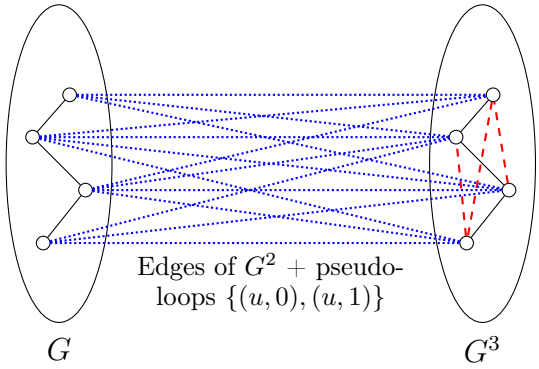


Figure 2.23: The graph $G^{[2]}$.

We refer to Figure 2.23 for an illustration. Intuitively, the graph $G^{[2]}$ can be seen as an intermediate power between the square and the cube of G. Our characterization of 1/2-hyperbolic graphs can now be stated as follows.

Theorem 38. *$G = (V, E)$ is 1/2-hyperbolic if and only if none of the graphs $G^j, j \geq 1$ and $G^{[2]}$ contain an induced cycle of length four.*

By Theorem 38, it can be decided whether $G = (V, E)$ is 1/2-hyperbolic with $diam(G)$ calls to an oracle solving Quadrangle detection – given as inputs $G^{[2]}$ and $G, G^2, G^3, \ldots, G^{diam(G)-1}$. If we precompute, in truly subcubic time, a polylogarithmic-factor approximation for hyperbolicity then this number of calls can be reduced to $\log^{\mathcal{O}(1)}(|V| + |E|)$ (because some powers of G can be discarded), and we so obtain a subcubic reduction from the recognition of 1/2-hyperbolic graphs to Quadrangle detection.

Discussion. As said earlier in this subsection, the authors in [BCH16] show that under SETH, graph hyperbolicity cannot be computed in truly subquadratic time. In contrast, it is proved with Theorem 36 that the weaker task of recognizing 1/2-hyperbolic graphs is equivalent to the Quadrangle Detection problem. The latter problem can be solved in $\mathcal{O}(m^2)$-time on m-edge graphs, and so, in quadratic time on sparse graphs. However, no truly subquadratic *deterministic* algorithm is known to exist, even for sparse graphs. In [VWWWY15], Vassilevska Williams et al. describe an $\mathcal{O}(m^{1.41})$-time *randomized* algorithm for Quadrangle Detection, but it is not combinatorial (*i.e.*, it calls matrix multiplication as a subroutine). In order to reinforce this view, we note that there is a linear time reduction from Triangle Detection to Quadrangle Detection [FKLL15], and so, to the problem of computing graph hyperbolicity. It is conjectured that there does not exist any truly subcubic combinatorial algorithm for Triangle Detection on general graphs [Wil16].

2.7 Algorithmic applications

Finally, this section covers more technical applications of hyperbolicity, in the field of graph algorithms. The previous sections can help the reader to have better insights on the (hyperbolic) graph classes on which these algorithmic results apply, and the (non hyperbolic) graph classes on which they do not apply. Note that this section is *not* part of the contributions of this thesis. However, I will highlight on the way some open questions on which I am interested to work.

The hyperbolicity has been used recently for the analysis of graph algorithms. Indeed, it is the idea that when the hyperbolicity is small, there are some hard problems on graphs that can be efficiently approximated. In what follows, we outline some interesting algorithmic properties that are enjoyed by constantly hyperbolic graphs. Note that in some cases, the algorithms that are presented in this section keep some interest even for more general hyperbolic graph classes (say, polylogarithmically hyperbolic).

Outline of the section. The first parts of this section (Sections 2.7.1 and 2.7.2) cover distance-related problems in graphs. In Section 2.7.1, we survey applications of hyperbolicity in the analysis of approximate distance oracles. These results are mainly based on the relationships between hyperbolicity and the best possible distortion of the distances in a graph when it is embedded into a "tree-like" metric space. Perspectives for improving upon these relationships, and for refining the proposed constructions, will be discussed. Then, in the continuity of Section 2.7.1, we will cover in Section 2.7.2 some applications of hyperbolicity to graph clustering problems. The techniques presented leave space for promising extensions to a broader family of graph problems, that will be further examined. Finally, we will end the section with algorithmic applications of hyperbolicity to some problems in structural graph theory (Sections 2.7.3 and 2.7.4). Section 2.7.3 is devoted to a PTAS for the

Traveling Salesman Problem in hyperbolic graphs with bounded degree. This algorithm is based on new separability results in hyperbolic graphs, that I think could be useful in other graph problems. Last, constructive relationships between hyperbolicity and vertex expansion are presented in Section 2.7.4. I think that these relationships can be helpful in the design of approximation algorithms for computing the treewidth in hyperbolic graphs with bounded degree.

2.7.1 Distance approximations

This section surveys the known results on the relationship between hyperbolicity and the best-possible stretch for the distances in a graph when it is embedded in a "tree-like" space. Indeed, the basic use of hyperbolicity is for the analysis of approximate distance oracles. Computing the all-pairs-shortest-paths in a graph can be done in polynomial time and space, but in practice this is often too costly on large graphs and there is a need for subquadratic approximations. Some of them consist in embedding the graph into a "simpler" combinatorial or geometrical structure. When the structure is a "tree-like" metric space, the hyperbolicity of the graph comes into play in the distortion.

Note that these results have useful applications in compact routing [GL05].

2.7.1.1 Hyperbolic embedding

As an example, Verbeek and Suri proved in [VS14] that for any embedding of $G = (V, E)$ into a hyperbolic space the multiplicative distortion of the distances is $\Omega(\delta(G)/\log \delta(G))$, and if G has bounded degree then there exists a linear-time computable embedding of G in a Hyperbolic space with additive distortion $\mathcal{O}(\delta(G))$.

As noted in [ACHK16], every $G = (V, E)$ with maximum degree Δ can be embedded into a graph G' with maximum degree three, up to a multiplicative distortion of the distances $\mathcal{O}(\log \Delta)$. In this situation, $\delta(G') = \mathcal{O}(\delta(G) \log \Delta)$ (we refer to [Shc13b, GdLH90] for a proof of the preservation of hyperbolicity under quasi-isometry). Therefore, every $G = (V, E)$ can be embedded into a Hyperbolic space in linear-time with multiplicative distortion $\mathcal{O}(\delta(G) \log \Delta)$.

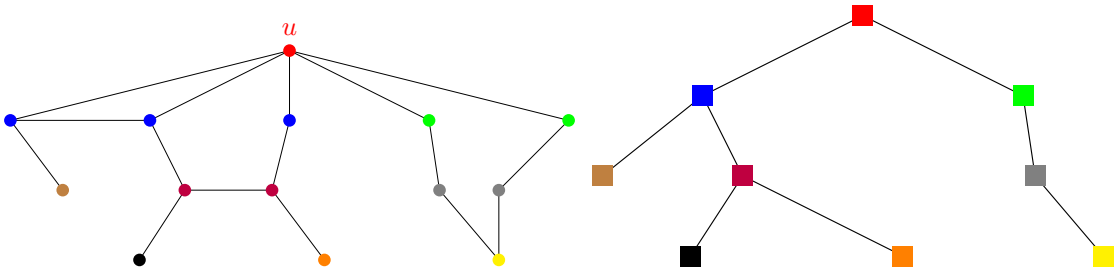
2.7.1.2 Tree embedding

In what follows, we survey the relationships between hyperbolicity and the distortions of the distances in a graph that are obtained with different algorithms for embedding a graph into a tree. Some interesting open questions will be also mentioned. Most notably, Gromov has proved the following result on tree embeddings:

Theorem 39 ([Gro87])**.** *Every $G = (V, E)$ can be embedded into a tree in quadratic-time, up to an additive distortion of the distances at most $2\delta(G) \log |V|$.*

In order to prove Theorem 39, the main contribution of Gromov was to exhibit a pseudo-distance on graphs, and then to upper-bound the additive distortion resulting from the pseudo-distance by $2\delta(G) \log(|V|)$. By construction, every graph

equipped with the Gromov pseudo-distance is 0-hyperbolic, and there exist efficient constructions in order to embed 0-hyperbolic spaces into a tree with null distortion. One of them is due to Buneman, and it can be implemented to run in quadratic-time [Bun74, Gro87].



(a) Graph G (b) Layering tree $\mathcal{LC}(u)$.

Figure 2.24: Example of a layering tree.

Relationship with other constructions. Recently, Yancey [Yan15] has proved a close relationship between the construction of Gromov and the so-called *layering trees* [CD00]. Given $G = (V, E)$ and $u \in V$, the layering tree $\mathcal{LC}_G(u)$ is obtained from the shortest-path tree rooted at u as follows: we merge into one node all vertices v, w such that $\mathrm{d}(u, v) = \mathrm{d}(u, w)$ and there exists a vw-path $\mathcal{P}$ such that $\mathrm{d}(u, x) \geq \mathrm{d}(u, v)$ for every $x \in \mathcal{P}$ (see Figure 2.24 for an illustration).

It was already proved that embedding G into one of its layering trees causes a distortion of the distances $\mathcal{O}(\delta(G) \log(|V|))$ [CDE+08]. However, what Yancey proves is that the Gromov distance approximating tree is essentially a layering tree with Steiner points (additional nodes in the tree such that all the edges incident to that node have weight zero). On the algorithmic side, since a layering tree can be computed in linear time [CD00], it gives a simpler and more efficient construction for Theorem 39.

The Gromov distance approximating tree is also equivalent to another construction in the litterature, that is called an Anchored Buneman tree [BFÖ+03].

Perspectives. There exists a "refined Buneman tree" [BFÖ+03], that has been observed to give a lower distortion of the distances in a graph than an Anchored Buneman tree. It can be computed in cubic time. I think that it would be interesting to analyse the distortion caused by an embedding into this tree (w.r.t. graph hyperbolicity), and to improve on its computation (possibly, by using the relationship between Anchored Buneman trees and layering trees).

Another interesting question on tree embeddings was asked by the authors in [ASM16]. Indeed, they notice that for real-life graphs with diameter $\mathcal{O}(\log(|V|))$, a shortest-path tree is enough in order to approximate the distances up to an additive term $\mathcal{O}(\log(|V|))$. Therefore, the tree embedding of Theorem 39 does not look that appealing in that case. Under which conditions can a δ-hyperbolic graph

with diameter D be embedded into a tree with distortion $\mathcal{O}(\delta \log D)$? Let us point out that by Lemma 13 the ringed tree $RT(k)$ (defined in Section 2.4.1) has diameter $\Theta(k)$ and hyperbolicity 3 but cannot be embedded into a tree width additive distortion $o(k)$.

2.7.1.3 Approximate extremal distances.

Finally, before concluding this subsection, we point out that if we relax our goal and we only want to approximate the *extremal distances* in $G = (V, E)$ (*i.e.*, the eccentricities, where the eccentricity of a vertex is defined as its largest distance to another vertex in G), then it can be done up to a better additive term $\mathcal{O}(\delta(G))$. In particular, there is a simple algorithm in order to approximate the diameter, that is named Two-Sweep in the literature [MLH08]. Suppose that we compute a breadth-first search from any vertex of the graph $G = (V, E)$, and that it ends on some vertex v. Then, we compute a second breadth-first search from v, and it can be proved that v has eccentricity at least $diam(G) - 2\delta$. The latter generalizes an algorithm of Jordan in order to compute the diameter of trees in linear time [Jor69]. The radius of the graph can be approximated in a similar fashion. We refer to [CDE+08] for details.

2.7.2 p-centers

Next, we present a more refined algorithmic application of hyperbolic graphs to graph clustering problems, that was proposed in [CE07]. This application requires prior results on the relationships between hyperbolicity and tree embeddings (Theorem 39). Precisely, the *p-radius* of $G = (V, E)$ is the smallest radius $r_p(G)$ such that $V = \bigcup_{v \in S} B_G(v, r_p(G))$ for some subset $S \subseteq V$ with $|S| \leq p$ vertices. In particular, the 1-radius of G is simply its *radius*, *a.k.a.*, the minimum eccentricity of a vertex in G. A dual invariant is the *p-diameter* of $G = (V, E)$, that is the largest $d_p(G)$ so that there are at least p vertices of G that are pairwise at distance at least $d_p(G)$. In particular, the 2-diameter of G is simply its diameter, *a.k.a.*, the largest distance between two vertices in G. Furthermore, any subset minimizing $r_p(G)$, resp. maximizing $d_p(G)$, is called a *p-center*, resp. a *p-packing*.

Shier has proved that for any tree T, we have $d_{p+1}(T)/2 \leq r_p(T) \leq d_{p+1}(T)/2 + 1$ [Shi77]. In [CE07], Chepoi and Estellon propose the following generalization to δ-hyperbolic graphs:

Lemma 40 ([CE07]). *For every $G = (V, E)$, it holds $d_{p+1}(G)/2 \leq r_p(G) \leq d_{p+1}(G)/2 + 4\delta(G) + 1$.*

From Lemma 40, they obtain an $\mathcal{O}(n^3)$-algorithm for computing an approximate p-center of graphs [CE07]. It gives an approximation algorithm for computing the p-radius of a given δ-hyperbolic graph up to an additive term $\mathcal{O}(\delta)$. This was recently improved in [EKS16], where Edwards et al. detail an algorithm with the same performances as above, running in $\mathcal{O}(p\delta(n + m) \log n)$-time on δ-hyperbolic graphs.

The gist of these algorithms is to compute an approximate $(p+1)$-packing and then to elaborate on it. It can be done by embedding the graph into a tree with additive distortion of the distances $\mathcal{O}(\delta \log n)$, then to compute an optimal packing for this tree.

Perspectives. Proper generalizations of Lemma 40 to the transversal and the packing numbers of given set families in δ-hyperbolic graphs can be found in [CDV16]. These results are obtained from a primal-dual approach using a linear programming formulation of these parameters. Can it be derived from the relationships in [CDV16] efficient (quasi-linear time) approximation algorithms for computing transversals of these set families ? In particular, can the techniques applied in [EKS16] be useful in the design of such algorithms ?

2.7.3 Traveling Salesman Problem

So far, the problems mentioned in Sections 2.7.1 and 2.7.2 were purely metric. The two last applications (Sections 2.7.3 and 2.7.4) combine some metric aspects of graphs (distances) with structural properties. In particular, we present in this part results on "balanced" separators in hyperbolic graphs, with applications to the Traveling Salesman Problem.

In [KL06], Krauthgamer and Lee initiated a more general study of approximate algorithms on negatively curved spaces. Their algorithms apply to constantly hyperbolic graphs with bounded maximum degree. Their main technical tools are separability properties of hyperbolic graphs, that extend those of trees. As an example, in a rooted tree T with maximum degree Δ, there exists a node z whose subtree comprises between $|T|/(2\Delta) - 1$ and $|T|/2$ nodes. It can be extended to hyperbolic graphs as follows:

Lemma 41 ([KL06]). *Let $G = (V, E)$ be a δ-hyperbolic graph with maximum degree Δ and let $w \in V$. For every $v \in V$ and $t \geq 0$, let us define $X_v^t = \{u \in V \mid \langle u, v\rangle_w \geq \mathrm{d}_G(u, w) - t\}$. Then, for every $S \subseteq V$ such that the vertices in S are pairwise at distance at least 20δ, there exists $c \in V$ such that:*

$$|S|/\Delta^{\mathcal{O}(\delta^2)} \leq |S \cap X_c^{\delta}| \leq |S \cap X_c^{3\delta}| \leq |S|/2.$$

Using Lemma 41, Krauthgamer and Lee are able to design a hierarchical data structure for approximate nearest neighbour search [KL06] [KL06].

Their second contribution is a randomized polynomial-time approximation scheme (PTAS) for the well-known Traveling Salesman Problem (TSP). It is based on the existence, for bounded degree hyperbolic graphs, of some *padded probabilistic decompositions*. Roughly, the graph can be decomposed into small diameter subsets in a way that every ball with small radius is contained in one of the subsets with high probability. Assuming the graph has bounded maximum degree, it is the idea that hard problems such as TSP can be solved by brute-force on the subsets (or at least sharply approximated). Then, a global solution for the graph can be computed from the partial solutions by using dynamic programming.

Open questions. Lemma 41 extends a separability property of trees to hyperbolic graphs. What other separability properties of trees can be generalized to hyperbolic graphs in a similar fashion ? Can we use such properties in order to design approximation algorithms on hyperbolic graphs with bounded maximum degree, using dynamic programming, for other problems such as MAXIMUM CLIQUE or MAXIMUM INDEPENDENT SET ?

2.7.4 Cut problems

We end the section with some algorithmic consequences on the relationships between hyperbolicity and graph expansion (Section 2.5.2.3). Unlike the other problems mentioned in the section, the following algorithms also apply to non constantly hyperbolic graph classes. More precisely, although the above algorithmic work on hyperbolicity can sometimes apply to non constantly hyperbolic graph, the authors in [DKMY15] have been the first, to the best of my knowledge, to design algorithms for more general hyperbolic graphs (with non constant hyperbolicity).

We recall the results in [Ben98, Mal15] where they prove that expander graphs are non hyperbolic. In [DKMY15], the authors give constructive proofs on the relationship between graph expansion, maximum degree and hyperbolicity. Precisely, they obtain improved algorithms for the following graph problems. Given an n-vertex graph with maximum degree Δ and hyperbolicity at most δ, the following can be computed in polynomial-time:

- Upper-bounds on the vertex-expansion depending on δ and Δ. The algorithm also outputs a large family of subsets satisfying these bounds, with limited overlap;
- Large st-cuts with $\Delta^{\mathcal{O}(\delta)}$ edges.

The authors also propose an improved algorithm for minimizing the number of bottleneck edges that arises in network design applications. It works in the case where $\delta = o(\log n / \log \Delta)$;

Finally, the authors in [DKMY15] have considered the SMALL-SET EXPANSION problem on hyperbolic graphs, that is a promise problem defined as follows: given a graph $G = (V, E)$ and two constants c and η, distinguish whether `(i)` there exists a subset of V with size $c \cdot |V|$ and vertex-expansion at most η, or `(ii)` every such a subset has vertex-expansion at least $1 - \eta$ [RS10]. It is conjectured that for every fixed η, there exists some constant c such that the corresponding SMALL-SET EXPANSION problem is NP-complete for general graphs [RS10]. In contrast, the authors in [DKMY15] proved that for every constants η and c the SMALL-SET EXPANSION problem can be solved in polynomial time for n-vertex graphs with bounded maximum degree and hyperbolicity $\delta = o(\log n)$.

Conclusion and open perspectives. The SMALL-SET EXPANSION problem implies the UNIQUE GAME conjecture, that is related to the complexity of a label assignment problem on graphs and that has been shown to imply tight inapproximability results for many classic graph problems [Kho02]. Furthermore, the SMALL-SET

EXPANSION problem also implies the nonexistence of constant-factor approximations for treewidth [APW12].

Therefore, the result of [DKMY15] raises the following open problem: can the treewidth of hyperbolic graphs with bounded degree be approximated up to a constant-factor ? Note that computing the treewidth is NP-hard on bounded-degree graphs and on hyperbolic graphs [BT97].

2.8 Conclusion

In Sections 2.4 and 2.5, we presented bounds on graph hyperbolicity. Enriching these results with new lower and upper bound techniques is an important open problem, with potential implications for a faster computation of this parameter in practice. In particular, I believe that new results in the spirit of Section 2.4.3: on the preservation of hyperbolicity under some graph operations, would give a better insight on the structure of hyperbolic graphs. Similarly, new lower-bounds could help the computer scientists in better distinguishing complex networks that are hyperbolic or strongly hyperbolic (*e.g.*, biological and social networks) from those that are non hyperbolic (such as road networks). We refer to [AAD16, AD15, BCCM15, CCL15, ASM13, KNS13] for experiments on the hyperbolicity in complex networks.

On the complexity point of view, it is proved in Section 2.6.3 that the recognition of 1/2-hyperbolic graphs is subcubic equivalent to the detection of induced cycles of length four in graphs, and so, that no truly subcubic *combinatorial* algorithm for computing the hyperbolicity is likely to exist. It is worth pointing out that in practice, hard instances for the above problem are indeed graphs with small hyperbolicity. I thus conjecture that graphs with large hyperbolicity (say, proportional to their size) can be recognized more efficiently. Results of this fashion have been proved recently for the related problem of computing graph diameter [Dam16].

Open perspectives

As pointed out in Section 2.2, it can be inferred interesting network properties when the graph is δ-hyperbolic. Before we finish this chapter, it is worth mentioning that some other geometric graph parameters have been explored with the same goal in mind as above. Most of them are close in spirit from hyperbolicity, and they can often be defined via a suitable variation of the 4-point Condition (Definition 1) or another reformulation of hyperbolicity. We refer, *e.g.*, to [ABK+07, ADM14, JLB08, LT15, Yan15] for partial relationship between these properties and graph hyperbolicity.

Let us put a focus on two of these competitors to graph hyperbolicity. The first one is the average hyperbolicity, defined as $\frac{1}{\binom{n}{4}} \sum_{u,v,x,y \in V} \delta(u,v,x,y)$ [ADM14]. The second one is the notion of (p,δ)-hyperbolic graphs, that are graphs with at least a fraction p of their geodesic triangles that are δ-slim [LT15]. I think that both concepts should deserve more attention in the future, given that the maximum

value for the hyperbolicity is reached by an extremely small fraction of 4-tuples in real-life graphs (*e.g.*, less than 3% in social graphs [AAD16]).

Finally, let us point out that in some cases, complex networks have a meaningful orientation on the edges, *i.e.*, they are directed graph. So far, graph hyperbolicity has been defined and studied only in the undirected case. Thus, it would be very interesting to extend the notion of hyperbolicity (and of Gromov product, see Definition 2) to digraphs. Partial attempts in this direction can be found in [GK14, PRST13]. I let this topic as a future work.

Chapter 3

Tree decompositions with metric constraints on the bags

Summary

We make a complexity study for computing tree decompositions in graphs. The tree decompositions considered are defined via metric constraints on their bags. We aim at obtaining a finer-grained complexity for computing these decompositions in general graphs and in some graph classes with structural properties. To do so, we will prove conditional lower-bounds through reductions.

In Section 3.3, we prove that Triangle Detection reduces in quadratic time to the computation of clique-decomposition. This is a hint that there does not exist any truly subcubic *combinatorial* algorithm for this problem. Furthermore, we prove that computing the clique-decomposition can be reduced to Matrix Multiplication, which combined with the relationships between Matrix Multiplication and Triangle Detection, suggests a computational equivalence between these two problems and computing the clique-decomposition. On the parameterized point of view, we conjecture that clique-decomposition can be computed in quasi-linear time on graphs with bounded *clique-number*, that is formally proved for triangle-free graphs and other special graph classes.

Then, in Section 3.4 we answer open questions of Dragan et al. on the complexity of computing treebreadth, pathlength and pathbreadth in graphs. Namely, we prove that all these problems are NP-hard. More precisely, we prove that the recognition of graphs with treebreadth one is already NP-complete, and the same holds true for the recognition of graphs with pathbreadth one and the recognition of graphs with pathlength at most two. On a more positive side, we prove that deciding whether a bipartite or planar graph has treebreadth one is polynomial-time solvable. The algorithm for planar graphs and its analysis are surprisingly intricate.

Finally, we prove in Section 3.5 new relationships between treelength and treewidth. Precisely, we prove a nontrivial upper-bound on the diameter of minimal separators in a graph by using an algebraic tool called the *cycle basis*. We deduce from this result that the treelength is linearly upper-bounded by the treewidth in the class of graphs with bounded-length isometric cycles. Conversely, we prove that the treewidth is linearly upper-bounded by the treelength in the class of *apex-minor free* graphs, thereby generalizing a result from Dieng and Gavoille on planar graphs [DG09].

All my papers on tree decompositions [CDN16, DLN16a, DC17] are collected in the appendix.

Contents

3.1 Introduction

In the previous chapter, we studied on graph hyperbolicity and its algorithmic applications. Hyperbolicity is a measure of the closeness of a graph metric to a tree metric. Yet, it is not related to a structural decomposition of a graph directly[1]. On the algorithmic point of view, graph decompositions can be useful in order to design divide-and-conquer algorithms on large graphs. In particular, tree decompositions [RS86] aim at decomposing graphs into pieces, called *bags*, organized in a tree-like manner (formal definitions are postponed to Section 3.2). They have been proved to be useful in order to extend some efficient algorithms on trees to larger classes of graphs.

The purpose of this chapter is to describe my work on these decompositions.

3.1.1 Context

The general idea is that when the bags have a "simple enough" structure, there are hard problems on general graphs which can be solved efficiently by using dynamic programming on the tree decomposition. There is now a rich literature on

[1] There does exist a relationship between graph hyperbolicity and some decompositions of graphs with dismantling orderings (Definition 7).

tree decompositions with algorithmic applications, such as *e.g.*, algorithmic meta-theorems (for solving hard problems on graphs with a specified tree decomposition) [Cou90, DH08, FG01], and the well-known biconnected decomposition [Tar72], triconnected decomposition [HT73], clique-decomposition [BPS10], etc.

Furthermore, with the growing size of real-life graphs, tree decompositions have been found useful in order to identify the key aspects of the structure of complex networks, such as *e.g.*, core and periphery [ASM16].

Treewidth is a classical measure for studying tree decompositions. Roughly, the *width* of a tree-decomposition is the maximum size of its bags. The treewidth of a graph is the minimum width among all its tree-decompositions. A lot of work has been dedicated to compute tree-decompositions with small width since such decompositions can be efficiently exploited for algorithmic purposes [Bod06]. However, computing the treewidth of a graph is NP-hard [ACP87] and no constant-approximation algorithm is likely to exist [WAPL14]. Furthermore, real-life networks generally have a large treewidth [dMSV11]. These drawbacks motivated the study of other optimization criteria for tree-decompositions [DG07, KLNS15, Sey16].

Metric tree-likeness in graphs. In this chapter, we mainly focus on optimizing the metric properties of the bags. One first example is an *atom tree* [BPS10], where the bags are maximal subgraphs with no clique-separators. The bags in an atom tree are isometric subgraphs. An atom tree has already nice algorithmic applications, however it may be sometimes more interesting to further decompose the graph. Roughly, the *length* and the *breadth* of a tree-decomposition are the maximum diameter and radius of its bags respectively. The corresponding graph parameters are the *treelength* [DG07] and the *treebreadth* [DK14] respectively. As I mentioned it in Section 2.4.1 (p. 36), these two parameters are closely related to hyperbolicity, and to the best possible distortion of the distances in a graph when it is embedded into a tree. Algorithmic applications of hyperbolic graphs (Section 2.7, p. 67) thus transpose to bounded treelength graphs. See also [DDGY07] for some other applications of treelength in graph algorithms. We point out that recent studies suggest that some classes of real-life networks – including biological networks and social networks – have bounded treelength and treebreadth [AAD16].

3.1.2 General objective: efficient computation of tree decompositions

In the continuity of my work on computing graph hyperbolicity (Section 2.6), I have been interested in computing efficiently tree decompositions with bags of small diameter or radius. To a lesser extent, my results also apply to the computation of other tree-likeness parameters such as, *e.g.*, treewidth.

In what follows, I shall introduce my main contributions to the field.

3.1.2.1 Finer-grained complexity of clique-decomposition

The decomposition of a graph by its clique-separators is sometimes called "clique-decomposition" in the litterature [BPS10]. Its output is an atom tree (mentioned above), that is a tree decomposition whose bags induce subgraphs with no clique-separators, *a.k.a.* atoms. One interest of clique-decomposition is that it can be used for preprocessing the graph in the computation of many other parameters (exact or approximate). In particular, the treewidth of a graph is the maximum treewidth of its atoms, and the same holds true for treelength and treebreadth. In Section 2.6.2, I also detailed a novel application of clique-decomposition for computing the hyperbolicity of large graphs.

My purpose in Section 3.3 is to improve our understanding of the complexity of computing this decomposition. Clique-decomposition can be computed in polynomial-time [Tar85]. However, the best-known algorithms for the problem run in $\mathcal{O}(nm)$-time on n-vertex m-edge graphs, that is prohibitive for large graphs.

In [DC17], we show how to reduce the TRIANGLE DETECTION problem to clique-decomposition, that is strong evidence that the state-of-the-art algorithm for clique-decomposition is essentially optimal. Furthermore, we describe an improved algorithm for computing the clique-separators of a graph, that suggests an interesting relationship between the complexity of computing clique-decomposition and the *clique-number* of a graph (size of a maximum clique).

These results are in revision for *SIAM Journal of Discrete Mathematics*. They are joint work with my supervisor David Coudert. I will detail them in Section 3.3.

3.1.2.2 The (NP-)hardness of computing treebreadth

The remaining of this chapter (Sections 3.4 and 3.5) is devoted to the length and the breadth of tree decompositions. On the complexity point of view, it has been proved by Lokshtanov in [Lok10] that deciding whether a graph has treelength at most k is NP-complete for every fixed $k \geq 2$. However, this was left open for treebreadth [DK14].

We answer to this open problem in [DLN16a]. Precisely, it is proved in the paper that deciding whether a graph has treebreadth at most k is NP-complete for every fixed $k \geq 1$. Similar results are obtained for the "path counterparts" of treelength and treebreadth, that are named pathlength and pathbreadth [DKL14].

On a more positive side, we initiate the study of the complexity of computing treebreadth on certain graph classes. This approach has been well explored for treewidth [BKK95, KK95, Klo96, BKKM98, BM93]. However it has been so far underexplored for treelength and treebreadth. Precisely, it is proved in [DLN16a] that bipartite graphs and planar graphs of treebreadth one can be recognized in polynomial time.

I will expand on this joint work with Nicolas Nisse and Sylvain Legay in Section 3.4.

3.1.2.3 Relationships between treewidth and treelength

Finally, the last Section 3.5 is devoted to new relationships between treelength and treewidth. We obtain this way a unifying view of tree-likeness in graphs. Further motivations to find such relationships are to derive improved algorithms for solving hard problems on certain classes of bounded-treelength graphs, improved approximation algorithms for computing the treewidth on certain graph classes, etc.

In order to better depict the results in this section, found in collaboration with David Coudert and Nicolas Nisse, let it be said that complete graphs are the classical example of graphs with large treewidth but bounded treelength, whereas on the other hand the cycles have bounded treewidth but unbounded treelength [DG07]. These two graph families thus can be used in order to show that treewidth and treelength cannot be compared on general graphs. We prove in [CDN16] that removing these obstructions allows one to upper and lower bound treewidth with functions of the treelength. More formally, what we prove in [CDN16] is that on *apex-minor free* graphs with bounded-length isometric cycles, treelength and treewidth can only differ by at most a constant-factor (full definition for this class of graphs is postponed to Section 3.5).

Definitions and preliminary results are presented in Section 3.2. The technical sections are structured as follows. We start with a short summary of the topic, then, we list our main contributions and we discuss about their implications. We end the sections with sketch proofs of the main results.

3.2 Some basics on tree decompositions

The notion of tree decomposition was briefly introduced in the previous chapter (Section 2.4.1). We restate the definition here for convenience of the reader. A *tree decomposition* $(T, \mathcal{X})$ of $G = (V, E)$ is a pair consisting of a tree T and of a family $\mathcal{X} = (X_t)_{t \in V(T)}$ of subsets of V indexed by the nodes of T and satisfying:

- $\bigcup_{t \in V(T)} X_t = V$;
- for any edge $e = \{u, v\} \in E$, there exists $t \in V(T)$ such that $u, v \in X_t$;
- for any $v \in V$, the set of nodes $\{t \in V(T) \mid v \in X_t\}$ induces a subtree, denoted by T_v, of T.

The sets X_t are called *the bags* of the decomposition. Its *adhesion sets* are the intersections $X_t \cap X_{t'}$ for every edge $\{t, t'\} \in E(T)$. As an example, we show a tree decomposition of the wheel in Figure 3.1. In this case, the tree T is a path, so, we call it a *path decomposition*.

We point out that any graph admits a tree decomposition, resp. a path decomposition. Indeed, the single node tree with bag V satisfies the three above conditions. However, this trivial tree decomposition is not that interesting, so, we aim at imposing additional constraints on the bags or on the adhesion sets.

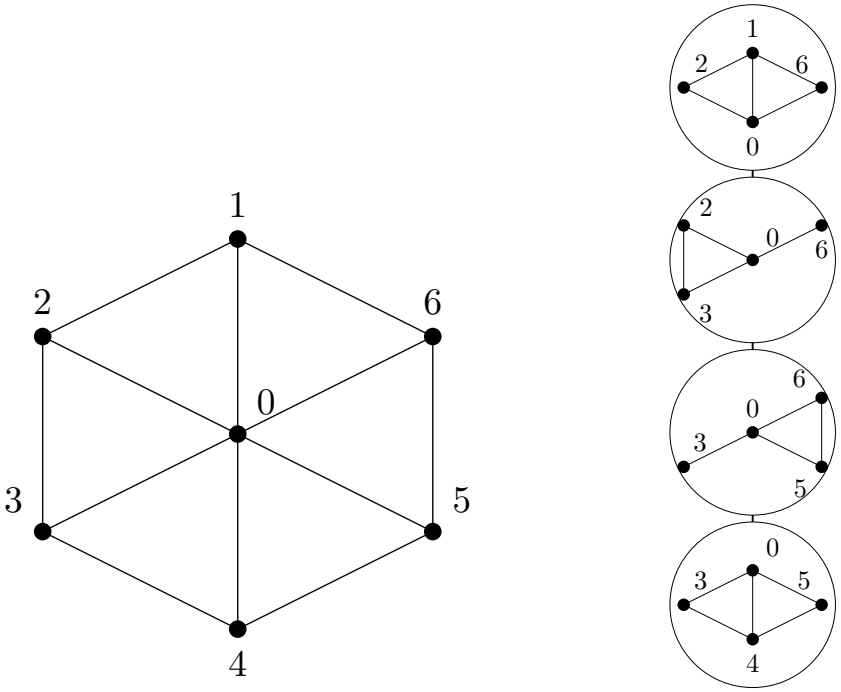


Figure 3.1: A path decomposition of the wheel W_6.

3.2.1 Tree-likeness parameters

Treewidth

The *width* of a tree decomposition is the size of a largest bag minus one. The *treewidth*, resp. the *pathwidth* of a graph G is the least possible width over its tree decompositions, resp. over its path decompositions. In what follows, we denote these two parameters by $tw(G)$ and $pw(G)$, respectively.

Example: graphs with small treewidth. Graphs with treewidth one are exactly the trees (hence, the minus one in the definition).

Furthermore, cycles have treewidth two. It can be shown as follows. When we remove any vertex from a cycle, that will leave a path. This path is a tree, so, it has a tree decomposition of unit width. Then, by adding in every bag the removed vertex, we obtain a tree decomposition of the cycle of width two.

Examples of graphs with large treewidth are the complete graphs and the grids.

Precisely, a complete graph K_n with n vertices has treewidth $n-1$. This well-known result derives from the Helly property: every collection of pairwise intersecting subtrees in a tree have a nonempty intersection. We detail this a bit more below as it is a useful technique in the study of tree decompositions.

Let us fix $(T, \mathcal{X})$ a tree decomposition of K_n. We have for every $u, v \in V(K_n)$

that since u and v are adjacent they must be contained in a common bag. As a result, the subtrees $T_v,\ v \in V(K_n)$ are pairwise intersecting. By the Helly property, it implies that there must be a bag of $(T, \mathcal{X})$ with all the n vertices in K_n, hence $tw(K_n) \geq n - 1$. The bound is reached by the trivial tree decomposition with one node.

Observe that more generally, we have with the same proof as above that for every G, and every tree decomposition $(T, \mathcal{X})$ of G, every clique of G must be fully contained in one bag of $(T, \mathcal{X})$ [Bod06]. Therefore, the treewidth is lower-bounded by the clique-number (size of a largest clique).

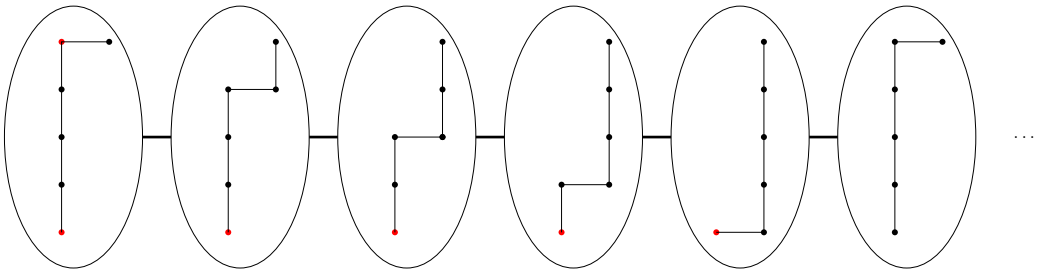


Figure 3.2: Bags in a path decomposition of the grid with side length four (partial view).

Last, given a grid with dimensions m and n, with $n \leq m$, it is not difficult to construct a tree decomposition of width n (see Figure 3.2). This construction is optimal [Die10] but it is technically challenging to prove it.

I will study treewidth in Section 3.5.

3.2.1.1 Treelength and treebreadth

The *length* of a tree decomposition is the maximum distance in the graph between every two vertices in a same bag. The *treelength*, resp. the *pathlength* of a graph G is the least possible length over its tree decompositions, resp. over its path decompositions. In what follows, we denote these two parameters by $tl(G)$ and $pl(G)$, respectively. Note that they are trivially upper-bounded by the diameter $diam(G)$ (that is the length of the trivial tree decomposition with one node).

Close to its length, the *breadth* of a tree decomposition is the minimum r such that every bag is contained in a ball of radius r in the graph (the center of the ball may not be in the bag). The *treebreadth*, resp. the *pathbreadth* of a graph G is the least possible breadth over its tree decompositions, resp. over its path decompositions. In what follows, we denote these two parameters by $tb(G)$ and $pb(G)$, respectively. As an example, the wheel in Figure 3.1 has treebreadth one and treelength two.

Treelength and treebreadth can be seen as a particular case of acyclic (R, D)-clustering, *a.k.a.* tree decompositions with breadth at most R and length at most D [DL07]. The two parameters are closely related. Precisely, $tb(G) \leq tl(G) \leq$

$2 \cdot tb(G)$ and the bounds are sharp [DK14]. The same relationship holds true between pathlength and pathbreadth.

Examples of graphs with small treelength. It turns out that many interesting graph classes with unbounded treewidth have small treelength. As an example, the chordal graphs are a strict generalization of complete graphs. They can be characterized as those graphs admitting a *clique-tree*, that is a tree decompositions whose bags are cliques [Gav74]. Thus, chordal graphs are exactly the graphs with unit treelength. More generally, every k-chordal graph (graph with no induced cycle of length at least $k+1$) has treelength at most $\lfloor k/2 \rfloor$ [DG07].

Related to chordal graphs, the *dually chordal graphs* are the clique-graphs (*i.e.*, intersection graphs of the maximal cliques) of chordal graphs [BDCV98]. We claim that dually chordal graphs have treebreadth one, and so, treelength at most two. Indeed, for every dually chordal graph G, there exists a one-to-one mapping φ from the maximal cliques of some chordal graph H to the vertices of G. Let $(T, \mathcal{X})$ be a clique-tree of H. Since bags of this tree decomposition are maximal cliques of H, we can define, for every node $t \in V(T)$, $Y_t = N_G[\varphi(X_t)]$. Then, it can be checked that $(T', \mathcal{Y}) = (T, (Y_t)_{t \in V(T)})$ is a tree decomposition of G of breadth one. In particular, for every vertex $v \in V(G)$ we have that $T'_v = \bigcup_{u \in \varphi^{-1}(v)} T_u$. It follows, as claimed, that dually chordal graphs have treebreadth one, but this inclusion is proper. To see that, it suffices to notice that every chordal graph also has treebreadth one, while not all chordal graphs are dually chordal [BDCV98].

Another interesting fact is that every graph with diameter at most D also has treelength at most D (trivially). In particular, adding a universal vertex to any graph G with treewidth k will result in a graph G' with $tw(G') = k+1$ and $tl(G') \leq diam(G') \leq 2$. This simple observation will be useful in order to better intuit our results in Section 3.5.

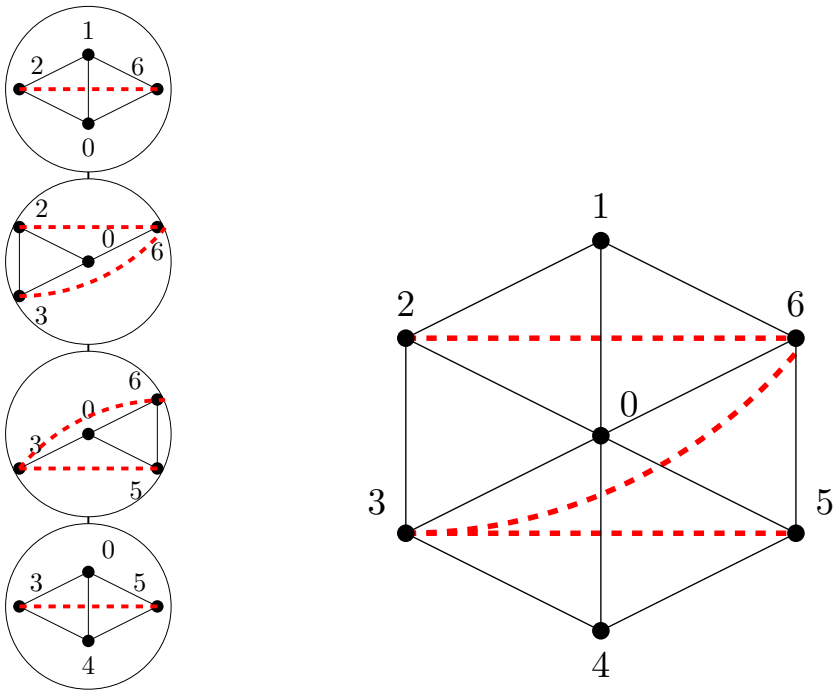
On the other way around, **examples of graphs with large treelength** include cycles and grids [DG07]. Intuitively, this can be explained by the Balanced separation property in tree decompositions: in any tree decomposition $(T, \mathcal{X})$ of G, there must exist a bag $B \in \mathcal{X}$ so that every component of $G \setminus B$ contains no more than $|V|/2$ vertices (it generalizes the existence of a centroid in a tree [Gol71]). It is not hard to see that on a cycle C_n with n vertices, any balanced separator has diameter $\Omega(n)$ (see also Fig. 2.14b). Similar arguments apply to the case of grids.

Finally, it should be noticed that complete graphs have unbounded treewidth and unit treelength, whereas n-vertex cycles have treewidth two and unbounded treelength $\lceil n/3 \rceil$ [DG07]. Altogether combined, it shows that treewidth and treelength are uncomparable on general graphs. We shall discuss when they can be compared in Section 3.5.

3.2.2 Relationship with triangulations

Tree decompositions can be defined equivalently in terms of graph triangulations. As we will show throughout this chapter, this reformulation is very convenient to use in the proofs.

A triangulation of $G = (V, E)$, sometimes called a fill-in of G, is any chordal supergraph $H = (V, E \cup F)$ of G. Recall that chordal graphs are exactly those graphs with a clique tree, *a.k.a.* tree decomposition whose bags are cliques [Gav74]. If H is a triangulation of G, then any of its clique tree is clearly a tree decomposition for G. Conversely, given a tree decomposition $(T, \mathcal{X})$ of G, we can define a triangulation of G by adding an edge between every two vertices that are in a same bag of the decomposition (*e.g.*, see Figure 3.3 for an illustration).



(a) A tree decomposition of W_6 (b) The corresponding triangulation.

Figure 3.3: Triangulation of the wheel W_6.

Altogether combined, the tree decompositions of G can be defined as the clique trees of its triangulations H. In particular:

- $tw(G) \leq k$ if and only if there exists a triangulation H of G with no clique of size greater than $k + 1$ (sometimes called a k-tree) [Bod06];
- $tl(G) \leq l$ if and only if there exists a triangulation H of G so that $E(H) \subseteq E(G^l)$, where $G^l = (V, \{\{u, v\} \mid 0 < \mathrm{d}_G(u, v) \leq l\})$ is the l^{th} power of G [Lok10][2].

[2]I am not aware of any "natural" reformulation of treebreadth in terms of triangulation. It is my opinion that the hypergraph terminology from [BDCV98] would be best suited to reach the

Minimal triangulation and minimal separators. Let $G = (V, E)$ be a graph. A triangulation $H = (V, E \cup F)$ of G is *minimal* if for every strict subset $F' \subset F$, we have that $H' = (V, E \cup F')$ is not chordal. Similarly, a minimal tree decomposition of G is a clique tree of some minimal triangulation of G.

Every triangulation $H = (V, E \cup F)$ of G can be transformed into a minimal one by removing a subset of edges $F' \subseteq F$. Note that it does not make increase the width, length and breadth of the corresponding tree decompositions of G. As a result, it can always be found a minimal tree decomposition of minimum width, resp. of minimum length or of minimum breadth. This observation has motivated an in-depth study of minimal triangulations and their characterizations [Heg06].

In particular, the following characterization is due to Parra and Scheffler [PS97]. Before we can state it properly, we need to introduce standard notions on graph separators.

A *separator* of $G = (V, E)$ is any subset $S \subseteq V$ satisfying that $G \setminus S$ is disconnected. If a, b are two vertices in different components of $G \setminus S$ then we call S an ab-separator. A *minimal separator* is an inclusion wise minimal ab-separator S for some pair of vertices $a, b \in V \setminus S$. Equivalently, a separator S is called minimal if there exist two components A, B of $G \setminus S$ such that $N(A) = N(B) = S$. We note that inclusion wise minimal separators are also minimal separators, but the converse holds false.

Two minimal separators S_1, S_2 of G *cross* if S_1 intersects two connected components of $G \setminus S_2$ (this is an equivalence relation on minimal separators [PS97]). If S_1, S_2 do not cross then they are called *parallel*.

Theorem 42 ([PS97]). *H is a minimal triangulation of a graph G if and only if it is obtained by transforming into cliques all sets in a maximal family of pairwise parallel minimal separators of G.*

3.2.3 Tree decompositions with constrained adhesion sets

The dominant approach in the study of tree decompositions is to try to optimize some properties on the bags. This is the approach presented in Section 3.2.1. Another approach is to impose more structures on the adhesion sets (intersections of adjacent bags). Many graph decompositions can be defined this way. We present some of them below, with an emphasis on clique-decomposition.

First examples. The *biconnected decomposition* of $G = (V, E)$ is the collection of its maximal sets of vertices with no separator of size one (also called cut-vertex). These sets are called biconnected components. It is well-known that the biconnected components are the bags of a tree decomposition of G, sometimes called a *block-cut*

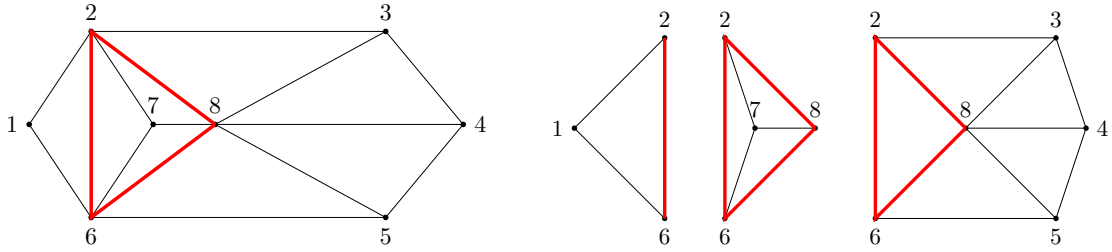
goal. Namely, define for every graph G the hypergraphs $\mathcal{C}(G)$ and $\mathcal{N}(G)$ whose hyperedges are, respectively, the maximal cliques and the closed neighbourhoods in G. Furthermore, given two hypergraphs $\mathcal{H}_1$ and $\mathcal{H}_2$ with same vertex-set, let us write $\mathcal{H}_1 \subseteq \mathcal{H}_2$ if every hyperedge of $\mathcal{H}_1$ is a subhyperedge of $\mathcal{H}_2$. Then, $tb(G) \leq j$ if and only if there exists a chordal supergraph H of G such that $\mathcal{C}(G) \subseteq \mathcal{C}(H) \subseteq \mathcal{N}(G^j)$

tree [Tar72]. In particular, we observe that the adhesion sets of a block-cut-tree are exactly the cut-vertices of G.

Similarly, the so-called *triconnected components* [HT73] are the bags of a tree decomposition of G, sometimes called a *SPQR-tree* [GM00]. The adhesion sets of a SPQR-tree are pairwise parallel minimal separators of size two. Generalizations to tree decompositions with adhesion sets of size at most k are discussed in [CDHH16, Gro16].

3.2.3.1 Clique-decomposition

Instead of bounding the size of the adhesion sets, we can bound their diameter. A clique-minimal separator of $G = (V, E)$ is a minimal separator inducing a clique of G. The atoms of G are the maximal sets of vertices with no clique separator. Finally, the *clique-decomposition* of G is the collection of its atoms (see Figure 3.4 for an illustration).



(a) A graph G. (b) The clique-decomposition of G.

Figure 3.4: Example of clique-decomposition.

In the same way as above, the atoms of G are the bags of a tree decomposition, sometimes called an *atom tree* [BPS14]. The atom trees of G are exactly the clique-trees of some triangulation H^+ of G [BPS10]. In general, H^+ is not a minimal triangulation of G. However, we have that H^+ is a supergraph of *any* minimal triangulation of G. More precisely:

Proposition 43 ([BPS10]). *For every minimal triangulation H of $G = (V, E)$, the clique-minimal separators of G are exactly the minimal separators of H that induce a clique of G.*

What Proposition 43 implies is that in order to compute a minimal triangulation of G, it suffices to do so for each atom separately [Tar85]. In particular, it follows that treewidth, treelength and treebreadth can be computed on each atom separately (we obtain their value for G by taking the maximum value over the atoms). This motivates us to study the complexity of computing clique-decomposition in Section 3.3.

3.3 Computational aspects of clique-decomposition

This section is devoted to my work on the time complexity for computing clique-decomposition. We refer the reader to [DC17] for the full version.

3.3.1 State of the art

The clique-decomposition is well-known to be computable in polynomial $\mathcal{O}(nm)$-time on n-vertex m-edge graphs [Lei93, Tar85]. For dense graphs, it can be improved to $\mathcal{O}(n^{2.69})$ [KS06b], but the algorithm is non combinatorial (*i.e.*, it uses matrix multiplication as a routine). Faster combinatorial algorithms have been proposed on certain graph classes such as subclasses of hole-free graphs and claw-free graphs [BBGM15, BW12]. Still, the best-known combinatorial algorithms have $\mathcal{O}(nm)$-time complexity, that is cubic for dense graphs and quadratic for sparse graphs.

As shown with Proposition 43, clique-decomposition is strongly related with minimal triangulations. However, Kratsch and Spinrad proved in [KS06a] that finding a clique-separator is at least as hard as finding a simplicial vertex, *even if a minimal triangulation is given as part of the input.* The latter result implies that computing a minimal triangulation is not the only complexity bottleneck of clique-decomposition algorithms.

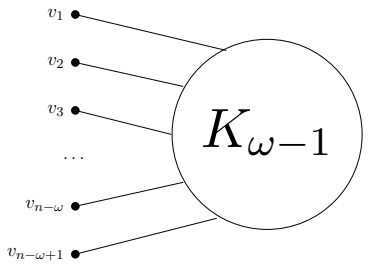


Figure 3.5: An n-vertex split graph with clique-number ω. The vertices are bipartitioned in a clique $K_{\omega-1}$ with $\omega-1$ vertices and an independent set with $n-\omega+1$ vertices. Furthermore, each vertex in the independent set is adjacent to all vertices in the clique. The atoms of the graph are exactly the closed neighbourhoods $N[v_i]$, $1 \leq i \leq n-\omega+1$. Therefore, there are $\omega(\omega-1)(n-\omega+1)/2$ edges in total in the subgraphs induced by the atoms.

Overview. Our results – presented below – suggest that another difficulty comes from the *clique-number* of the graph (size of a largest clique). In order to support our claim, we illustrate with Figure 3.5 that there are n-vertex graphs with clique-number ω such that the total number of edges cumulated on the subgraphs that are induced by their atoms is $\Omega(\omega^2 n)$. It implies that when a clique-decomposition algorithm not only computes the atoms, but also the subgraphs that are induced by them, its time complexity must be $\Omega(\omega^2 n)$.

3.3.2 Contributions

The following is joint work with my supervisor David Coudert.

3.3.2.1 Time complexity lower bound

In the spirit of what has been presented for graph hyperbolicity (Section 2.6.3, p. 64), it is proved in this section a *conditional lower-bound* on the time complexity for computing clique-decomposition. Precisely, computing the clique-decomposition is at least as hard as detecting a triangle in a graph.

We prove the following result in our paper [DC17].

Theorem 44. *The problem of detecting a triangle in an n-vertex graph reduces in quadratic time to the problem of computing the clique-decomposition of a graph with $3n + 2$ vertices.*

It is conjectured that no combinatorial truly subcubic algorithm for TRIANGLE DETECTION exists [Wil16]. So, altogether combined, this is hint that the $\mathcal{O}(nm)$-time state-of-the-art algorithm for computing clique-decomposition is essentially optimal.

3.3.2.2 Matching upper bound

In order to better understand the hardness of computing clique-decomposition, we next turn our attention on the *non combinatorial* algorithms. On a more theoretical side, it is proved in our paper [DC17] that clique-decomposition can be computed in $\mathcal{O}(n^\alpha \log n) = \mathcal{O}(n^{2.3729} \log n)$-time by using fast matrix multiplication.

Theorem 45. *For every n-vertex graph $G = (V, E)$, its clique-decomposition can be computed in $\mathcal{O}(n^{2.3729} \log n)$-time.*

Under well-established complexity hypotheses, the latter result matches the lower-bound obtained with TRIANGLE DETECTION for the non combinatorial algorithms. Indeed, we refer to [VWW10] for computational equivalences between TRIANGLE DETECTION AND MATRIX MULTIPLICATION[3]. Hence, these results are hint that (up to logarithmic factors), the time complexity for computing clique decomposition is in $\tilde{\mathcal{O}}(n^{2.3729})$.

3.3.2.3 The role of clique-number

Finally, we consider the seemingly simpler problem of computing the clique-decomposition when a minimal triangulation is given as part of the input. Let us call it the CLIQUE-DECOMPOSITION WITH MINIMAL TRIANGULATION problem.

[3] More explicitly, if MATRIX MULTIPLICATION can be solved in $\mathcal{O}(M(n))$-time then TRIANGLE DETECTION can be solved in $\mathcal{O}(M(n))$-time, and conversely if TRIANGLE DETECTION can be solved in $\mathcal{O}(T(n))$-time then MATRIX MULTIPLICATION can be solved in $\tilde{\mathcal{O}}(n^2 \cdot T(n^{1/3}))$-time.

We shall seek for efficient *parameterized* algorithms for the problem, where the parameter is the clique-number of the graph.

A new paradigm has emerged in Fixed-Parameter Tractability, sometimes called P-FPT (polynomial FPT), where the dependency in the fixed parameter k is required to be polynomial. There have been recent revisitings of polynomial-time graph problems in this polynomial parameterized setting [AVWW16, FLP+15, GMN15]. Our result, that can be found in our paper [DC17], is that CLIQUE-DECOMPOSITION WITH MINIMAL TRIANGULATION can be solved in linear time when the clique-number of the graph is assumed to be a constant.

Theorem 46. *For every $G = (V, E)$ with clique-number ω, and $H = (V, E \cup F)$ any minimal triangulation of G with $f = |F|$ fill edges, the* CLIQUE-DECOMPOSITION WITH MINIMAL TRIANGULATION *problem can be solved in time $\mathcal{O}(m + f + \omega^2 n)$.*

It is open whether more generally, the clique-decomposition can be computed in quasi-linear time on graphs with bounded clique-number. I conjecture that it is the case and this is left as an interesting open question. Furthermore, in order to support my conjecture, I will prove at the end of this section that it holds true for triangle-free graphs ($\omega = 2$).

3.3.3 Summarizing the proofs

3.3.3.1 Reduction from a counting problem

The proof for the lower bound is based on the following result on counting the number of simplicial vertices in a graph.

Lemma 47 ([KS06a])**.** *Counting the number of simplicial vertices in a graph with $3n + 2$ vertices is at least as hard as detecting a triangle in an n-vertex graph.*

I prove that a vertex is simplicial if and only if it is contained in a unique atom and this atom is a clique [DC17]. Based on this characterization, it can be shown that counting the number of simplicial vertices can be done in linear time if the clique-decomposition is given. Theorem 44 follows from this result directly.

Proof of Theorem 44. Let $G = (V, E)$ be any graph with $3n+2$ vertices. In order to prove the theorem, by Lemma 47 it is sufficient to prove that counting the number of simplicial vertices in G can be done in $\mathcal{O}(n+m)$-time if the clique-decomposition of G is given.

We claim that for every simplicial vertex $v \in V$, its closed neighbourhood $N[v]$ is an atom, and in particular it is the unique atom containing v. Indeed, suppose for the sake of contradiction that there exists $u \notin N[v]$ such that u and v lie on a same atom A. Then, $N(v) \cap A$ is an uv-separator in the subgraph $G[A]$. Since $N(v) \cap A$ is a clique, the latter contradicts that $G[A]$ has no clique-separator. Therefore, every atom containing v is a subset of $N[v]$. Finally, since $G[N[v]]$ is complete, we have that $G[N[v]]$ has no clique-separator, and so, by inclusion wise maximality of the atoms, $N[v]$ is the unique atom containing v, that proves the claim.

In particular, it follows from this above claim that a vertex is simplicial if and only if it is contained in a unique atom and this atom is a clique. Indeed, if a vertex is simplicial then by the above claim it satisfies the desired property. Conversely, if a vertex v is uniquely contained in an atom A and A is a clique then v is trivially simplicial with its neighbourhood being equal to $N[v] = A$.

Let us take advantage of this above characterization of simplicial vertices in order to count them in G. Let $A_1, A_2, \ldots, A_k$ be the atoms of G. We will use in the following analysis that $\sum_{i=1}^{k} |A_i| = \mathcal{O}(n+m)$ [BPS10].

We first compute an atom tree of G. In order to do so, we recall that a *dual hypertree* is a hypergraph whose hyperegdes are the maximal cliques of some chordal graph (obtained by adding an edge between every two vertices that are contained in a same hyperedge). Tarjan et al. prove in [TY84] that dual hypertrees can be recognized in linear-time, and that for every dual hypertree, a clique-tree of its underlying chordal graph can be computed within the same amount of time. Therefore, we can use this algorithm from [TY84] in order to compute an atom tree in $\mathcal{O}(\sum_{i=1}^{k} |A_i|) = \mathcal{O}(n+m)$-time.

Then, let A_i be any leaf-bag in the atom tree (a bag whose corresponding node in the tree has degree at most one). Since the intersection of two atoms is a clique [BPS10], we have that A_i is a clique if and only if every vertex that is uniquely contained in A_i has degree $|A_i| - 1$. Furthermore, by removing the set C_i of vertices that are uniquely contained in A_i then discarding A_i from the atom tree, one obtains an atom tree of $G \setminus C_i$. Therefore, we can repeat the above process in order to list all the atoms of G that are cliques. Overall, it takes time $\mathcal{O}(\sum_{v \in V} |N(v)| + \sum_{i=1}^{k} |A_i|) = \mathcal{O}(n+m)$.

Finally, let $A_{i_1}, \ldots, A_{i_l}$ be the atoms of G that are cliques. We can count all the vertices that are only contained in A_{i_j}, for some $1 \leq j \leq l$, simply by scanning all the atoms in $\mathcal{O}(\sum_{i=1}^{k} |A_i|) = \mathcal{O}(n+m)$-time. Since we proved that these are exactly the simplicial vertices of G, the latter achieves proving that counting the number of simplicial vertices can be done in $\mathcal{O}(n+m)$-time if the atoms are given. □

3.3.3.2 Computing the clique-minimal separators

Berry et al. have proved the following result in [BPS14]. Given an n-vertex m-edge graph $G = (V, E)$, suppose we are given $H = (V, E \cup F)$ a minimal triangulation of G with $f = |F|$ fill edges, *and the collection of the clique-minimal separators* of G. Then, the clique-decomposition of G can be computed in time $\mathcal{O}(m+f)$. So, we focused on the problem of computing the clique-minimal separators, given G and H as inputs.

Outline of the method. The gist of the approach for doing so is to use Proposition 43. Indeed, since H is chordal, its minimal separators can be computed in linear $\mathcal{O}(m+f)$-time [Gav72]. In order to extract from these the clique-minimal separators of G, by Proposition 43 it suffices to decide which are cliques of G.

- We prove in [DC17] that it can be done by using the incidence matrix of G and fast matrix multiplication. More precisely, we compute the clique-matrix of the triangulation H, where the minimal separators of H are listed, and then we multiply this matrix with the incidence matrix of G in order to determine which of those are cliques of G. Since in addition, a minimal triangulation H of G can be computed in $\mathcal{O}(n^\alpha \log n)$-time [HTV05], Theorem 45 follows.
- In order to do the same in a combinatorial way, we propose the following algorithm. Let us consider the vertices in G sequentially. At each step i, and for every minimal separator S of H which contains the current vertex v_i, we check whether v_i is adjacent to all the previous vertices $v_j \in S$ with $j < i$. When that is not the case, S cannot be a clique of G and so, we can discard it from the collection of (potential) clique-minimal separators of G. The central idea of the analysis is that since G has clique-number ω, we shall detect whether a minimal separator S of H is not a clique of G by considering no more than $\omega+1$ vertices in S. — Note that we needn't compute ω for the algorithm. — Theorem 46 now follows.

Discussion. The reason why we don't have an algorithm in time $\omega^{\mathcal{O}(1)}(n+m)$ for computing the clique-decomposition is that we don't know how to compute a minimal triangulation within these time bounds. However, there exist quasi-linear time algorithms for computing a minimal triangulation in some classes such as, *e.g.*, planar graphs [Dah98], bounded degree graphs [Dah02] and bounded-treewidth graphs [FLP+15]. Furthermore, we prove that in the special case of triangle-free graphs ($\omega = 2$), a minimal triangulation is not needed in order to compute the clique-decomposition. The latter result generalizes a remark from [BPS11], where Berry et al. notice without giving too much details that computing the clique-decomposition of a given bipartite graph can be done in linear time.

Lemma 48. *If $G = (V, E)$ is triangle-free then an atom tree of G can be computed in $\mathcal{O}(|V| + |E|)$-time.*

Proof. First, we compute a block-cut-tree of G (*a.k.a.* a tree decomposition whose bags are exactly the biconnected components of G, see Section 3.2). It can be done in linear time [Tar72]. We observe that since a cut-vertex is a clique-separator, the atoms of G are exactly the atoms of its biconnected components. In particular, an atom tree of G can be obtained by substituting each biconnected component G_i, in the block-cut-tree, by an atom tree of G_i. So, we can process the biconnected components separately and we now assume that G is biconnected for the remaining of the proof.

Then, we compute the SPQR-tree of G, that can also be done in linear time [GM00]. In [GM00] Gutwenger and Mutzel prove the following result using a different terminology than Parra and Scheffler. We have that up to further splitting the cycles among the triconnected components (using nonedge separators), the collection $\mathcal{F}_2$ of the adhesion sets in the SPQR-tree is a maximal family of pairwise

parallel minimal 2-separators of G. In this situation, we observe that since the two ends of an edge cannot be disconnected by any separator of G, an edge-separator is trivially parallel with any other minimal 2-separator of G, and so, it must be contained in $\mathcal{F}_2$. In particular, we can compute all the edge-separators of G by computing $\mathcal{F}_2 \cap E$, that can be done in $\mathcal{O}(|E| + |\mathcal{F}_2|) = \mathcal{O}(|V| + |E|)$-time.

Finally, since G is assumed to be biconnected and triangle-free, its edge-separators are exactly its clique-minimal separators. Therefore, we can compute the atoms of G as follows. We compute the maximal subtrees T_i of T so that for every $\{t, t'\} \in E(T_i)$, the minimal 2-separator $X_t \cap X_{t'}$ is not an edge. It can be done in $\mathcal{O}(\sum_{t \in V(T)} |X_t|) = \mathcal{O}(|V| + |E|)$-time. Then, the atoms of G are exactly the unions of bags in the subtrees, *i.e.*, $\bigcup_{t \in V(T_i)} X_t$ for every i. □

Finally, it would be interesting to determine whether more generally, a graph can be decomposed by its clique-minimal separators of size at most k in $k^{\mathcal{O}(1)}(n+m)$-time. By Lemma 48, it is the case if $k \leq 2$. Furthermore, a positive answer for every k would directly imply that computing the clique-decomposition can be done in $\omega^{\mathcal{O}(1)}(n+m)$-time — given the clique-number ω as part of the input.

3.4 On the complexity of computing treebreadth and its relatives

Computing an atom tree is a first step toward computing more interesting tree decompositions, *e.g.* with optimal width, length or breadth. In this section, we now answer open questions from [DK14] and [DKL14] on the complexity of computing treebreadth, pathlength and pathbreadth. Full results are presented in [DLN16a, DLN16b].

3.4.0.3 Motivations and related work

Treelength and treebreadth. The complexity of computing *treelength* on general graphs is now well understood. Graphs with unit treelength are exactly the chordal graphs [DG07], and they can be recognized in linear time. In contrast, recognizing graphs with treelength at most k is NP-complete for every fixed $k \geq 2$ [Lok10]. However on a more positive side, there exist 3-approximation algorithms for computing this parameter [DG07].

In [Lok10], the reduction used for treelength goes through edge-weighted graphs, and then goes back to unweighted graphs using rather elegant gadgets. It is not clear how to adapt this proof for treebreadth. Since the value for this parameter is a 2-approximation for treelength [DK14], any polynomial-time algorithm for computing treebreadth, or even an α-approximation algorithm for some $\alpha < 3/2$, would improve the best-known approximation algorithms for treelength. Our results (presented below) suggest that no such algorithm is likely to exist.

Pathlength and pathbreadth. As for pathlength (resp., pathbreadth), a 2-approximation (resp., a 3-approximation) algorithm is given for computing this parameter but the computational complexity of both problems is left open in [DKL14]. In the same paper, pathlength and pathbreadth have been shown to be useful in the design of approximation algorithms for bandwidth and line-distortion.

We note that recently, the MINIMUM ECCENTRICITY SHORTEST-PATH problem has been proved NP-hard [DL15]. The latter is a minimization problem where given a graph $G = (V, E)$, it is aimed at computing a shortest-path $\mathcal{P}$ with minimum *eccentricity* $\max_{v \in V} \mathrm{d}_G(v, \mathcal{P})$. Furthermore, it has been proved in [DKL14] that the minimum eccentricity of a shortest-path in G is an $\Theta(pl(G))$ with $pl(G)$ being the pathlength of G. Let us point out that for every fixed k, it can be decided in polynomial time whether a graph admits a shortest-path with eccentricity at most k [DL15]. The following results will show that the situation is different for pathlength and pathbreadth.

3.4.1 Summarize of our contributions

The main contributions in this section are to answer the open questions from [DK14, DKL14] on the complexity of computing treebreadth, pathlength and pathbreadth. Namely, the main results in our paper [DLN16a] can be stated as follows.

Theorem 49. *Recognizing the graphs with pathbreadth at most one is NP-complete.*

Theorem 50. *Recognizing the graphs with pathlength at most two is NP-complete.*

Theorem 51. *Recognizing the graphs with treebreadth at most k is NP-complete for every fixed $k \geq 1$.*

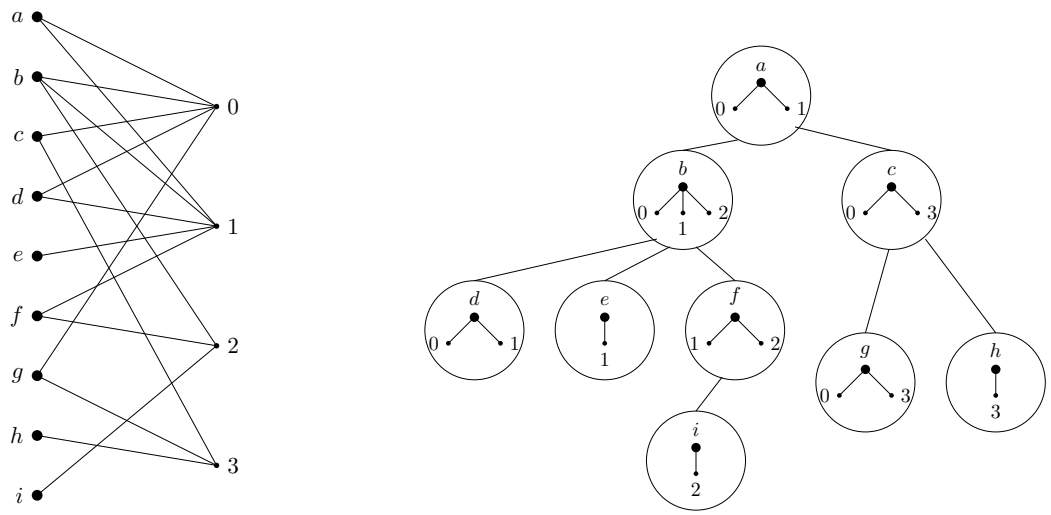
It is likely that recognizing graphs with pathbreadth at most k, resp. pathlength at most $k + 1$, is NP-complete for every fixed $k \geq 1$. This is left open in [DLN16a].

3.4.1.1 Graphs with treebreadth one

We now concentrate on the recognition of graphs with treebreadth at most one. This class of graphs already encompasses well-studied subclasses such as chordal graphs and dually chordal graphs. As it is stated in Theorem 51, recognizing graphs with treebreadth one is NP-complete. However, we prove in [DLN16a] that it can be done in polynomial time for bipartite graphs and planar graphs.

Case of bipartite graphs. Precisely, we obtain in our paper [DLN16a] a simple characterization of bipartite graphs with treebreadth one. Let us call a bipartite graph *tree-convex* if it admits a tree decomposition whose bags are exactly the closed neighbourhoods of the vertices in one side of its bipartition [WLJX12]. We refer to Figure 3.6 for an illustration.

Theorem 52. *A bipartite graph has treebreadth at most one if and only if every of its atoms is tree-convex. It can be verified in linear time.*



(a) A tree-convex graph G.

(b) A star-decomposition of G.

Figure 3.6: Tree-convex graphs have treebreadth one.

In contrast, recognizing bipartite graphs with treebreadth at most two is NP-complete. We observe that bipartite graphs with treebreadth one already encompass well-known graph classes such as convex bipartite graphs and chordal bipartite graphs (*a.k.a.*, bipartite graphs with no induced cycle of length at least six).

Case of planar graphs. We don't have a full characterization of planar graphs with treebreadth one. As proved in [DLN16a], a planar graph has treebreadth one only if it has treewidth at most four (more general relationships between treebreadth and treewidth will be discussed in the next Section 3.5). However, this condition is not sufficient, since any cycle of length at most five has treewidth two but treebreadth greater than one. Nonetheless, we have designed an algorithm in order to recognize planar graphs with treebreadth one in polynomial time.

Theorem 53. *Recognizing planar graphs with treebreadth one can be done in quadratic time. Furthermore, given a planar graph with treebreadth one, a tree decomposition with breadth one can be computed in cubic time.*

The algorithm for planar graphs is rather involved and it will be only sketched in what follows. We refer to our research report [DLN16b] for full details.

This part of my contributions is joint work with Nicolas Nisse and Sylvain Legay.

3.4.2 Approach and the techniques used in the proofs

3.4.2.1 A central lemma for graphs of treebreadth one

We start with a structural lemma that is used throughout all the proofs. We name *star-decomposition* a tree decomposition such that for every node $t \in V(T)$, there

exists a vertex $u \in X_t$ such that $X_t \subseteq N[u]$. That is, star-decompositions are similar to decompositions of breadth one, but the dominator of each bag has to belong to the bag itself. We prove with the following Lemma 54 that a graph has treebreadth one if and only if it has a star-decomposition.

In what follows, a tree decomposition is called *reduced* if no bag is included in another one. Starting from any tree decomposition, a reduced tree decomposition can be obtained in polynomial time by contracting any two adjacent bags with one contained in the other until it is no more possible to do that. Note that such a process does not modify the width, the length nor the breadth of the decomposition.

Lemma 54. *For any graph G with $tb(G) \leq 1$, every* reduced *tree decomposition of G of breadth one is a star-decomposition.*

The proof of Lemma 54 is an application of the Helly property: if B is a bag of a tree decomposition $(T, \mathcal{X})$ of G and there exists a vertex u dominating this bag, then by the properties of a tree decomposition, the subtrees T_u and T_v, $v \in B$, are pairwise intersecting, and so, by the Helly property there must be a bag with $B \cup \{u\}$. If the tree decomposition is reduced then it implies that $u \in B$.

3.4.2.2 Hardness of treebreadth, pathlength and pathbreadth

On the complexity point of view, the main result in [DLN16a] is the NP-completeness of deciding whether $tb(G) \leq k$, for every fixed $k \geq 1$. We first prove that the problem is NP-complete for $k = 1$, that will be our focus in this section. Then, we show that the problem of computing the treebreadth of a graph is polynomially equivalent to the problem of recognizing graphs with treebreadth one. Using similar techniques, we can prove that computing pathlength, resp., pathbreadth, is NP-hard [DLN16b].

Theorem 51 is proved by reducing a variation of the Chordal Sandwich problem to the recognition of graphs with treebreadth one. The Chordal Sandwich problem takes as input two graphs $G_1 = (V, E_1), G_2 = (V, E_2)$ with $E_1 \subseteq E_2$, and it asks whether there exists a chordal graph $H = (V, E)$ such that $E_1 \subseteq E \subseteq E_2$. This problem is NP-complete [GKS95]. In [Lok10], the author also proposed a reduction from Chordal Sandwich in order to prove that computing treelength is NP-hard. However, we need different gadgets than in [Lok10], and the arguments to prove correctness of the reduction are completely different.

Let us give a flavour of our reduction with Figure 3.7. Suppose we are given an instance $\langle G_1, G_2 \rangle$ of Chordal Sandwich. We aim at computing a supergraph G of G_1 such that in any tree decomposition of G of breadth one, there can be no two nonadjacent vertices in G_2 that are in the same bag. This way, any tree decomposition of G of breadth one can be transformed into a clique-tree for a chordal sandwich between G_1 and G_2. In order to reach this goal, for every nonedge $\{u, v\} \notin E(G_2)$ we add a copy of the gadget in Figure 3.7 and we make both u and v adjacent to both s_{uv}, t_{uv}. By construction, the four vertices (u, s_{uv}, v, t_{uv}) induce a cycle of length four. If we were studying treelength, then this would not give us that much information; indeed, in a tree decomposition of length at least two, all four vertices

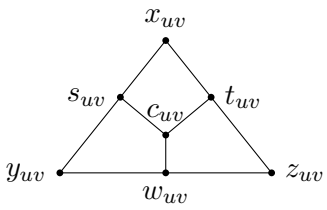


Figure 3.7: Gadget graph F_{uv}. The two vertices x_{uv}, w_{uv} are on disjoint $s_{uv}t_{uv}$-paths. Since they have no common neighbour, it ensures that s_{uv}, t_{uv} must be contained in a same bag in any star-decomposition of F_{uv}.

could be placed in a same bag without violating any constraint. However, this is no more the case for a tree decomposition with unit *breadth*. Indeed, since no vertex dominates the four vertices of the cycle, they cannot be part of a common bag. Hence, the gist of the construction is to ensure that s_{uv}, t_{uv} must be in a common bag in *any* tree decomposition of G of breadth one. Then, one can prove by elaborating on the Helly property that it implies that u and v cannot be in a same bag in any tree decomposition of G of breadth one.

On the technical point of view, the most difficult part of the reduction is to ensure that conversely, if $\langle G_1, G_2 \rangle$ is a yes-instance of Chordal Sandwich then the resulting graph G has treebreadth one. Ideally, we would like to transform some tree decomposition of G_1, with all vertices in a same bag being adjacent in G_2, to a star-decomposition of G. We tried to do so by adapting a technique from Lokshtanov [Lok10] that consists in adding a dominating clique in the graph. However, vertices from the gadgets in Figure 3.7 need to be inserted in the bags as well, thereby complicating the picture. In order to overcome the difficulties that are posed by these gadgets, we aim at better controlling in which bags their vertices need to be inserted, but then we need to impose additional constraints on the tree decomposition of G_1. In general, we are not able to prove that a tree decomposition with the desired constraints always exists. That is why we need to consider a variation of Chordal Sandwich where we impose more structure on the input.

Theorem 55. *The problem of deciding whether a graph has treebreadth one is NP-complete.*

Proof. The problem is in NP. To prove the NP-hardness, we will reduce from a variation of Chordal Sandwich that we name Chordal Sandwich with $\overline{nK_2}$. In this variation, we constrain ourselves to the instances $\langle G_1, G_2 \rangle$ so that the complementary $\bar{G}_2$ of G_2 induces a perfect matching. The problem Chordal Sandwich with $\overline{nK_2}$ is NP-complete [BFW92, GKS95]. Furthermore, perhaps surprisingly, the restriction on the structure of $\bar{G}_2$ will be shown to be a key element in our reduction.

Let $\langle G_1, G_2 \rangle$ be any instance of Chordal Sandwich with $\overline{nK_2}$. Let G' be the graph constructed from G_1 as follows. First, a clique V' of $2n = |V|$ vertices is added to G_1. Vertices $v \in V$ are in one-to-one correspondance with vertices $v' \in V'$.

Then, for every $\{u, v\} \notin E_2$, u and v are respectively made adjacent to all vertices in $V' \setminus v'$ and $V' \setminus u'$. Finally, we add a copy of the gadget F_{uv}, depicted in Figure 3.8a, and the vertices s_{uv} and t_{uv} are made adjacent to the four vertices u, v, u', v'.

We will prove that $tb(G') = 1$ if and only if $\langle G_1, G_2 \rangle$ is a yes-instance of Chordal Sandwich with $\overline{nK_2}$.

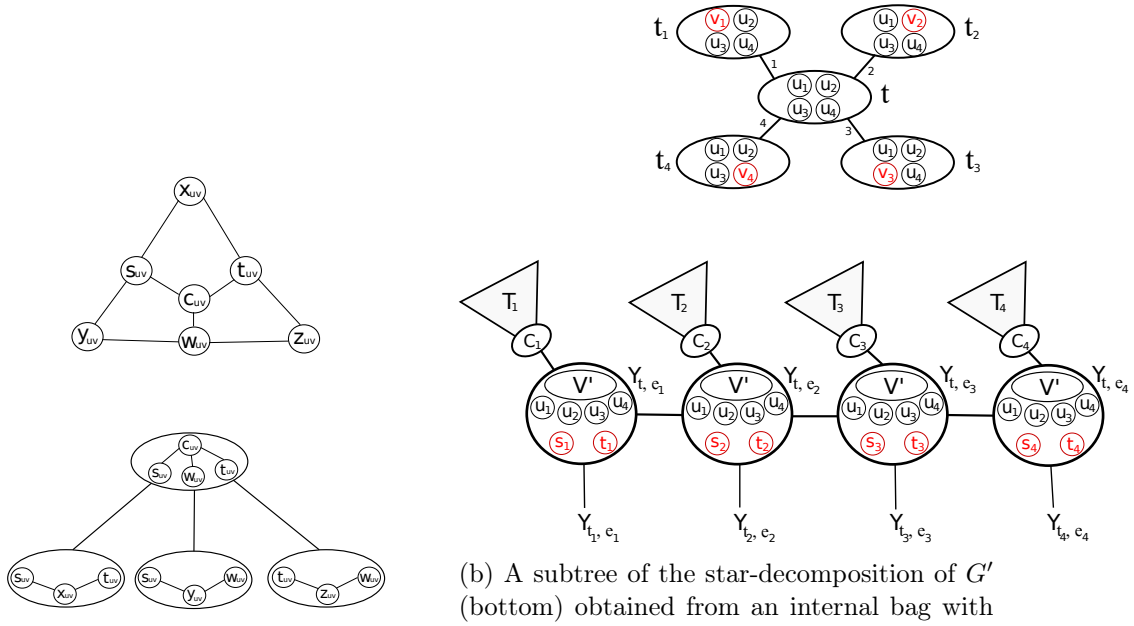
In one direction, assume $tb(G') = 1$, let $(T, \mathcal{X})$ be a star-decomposition of G' (which exists by Lemma 54). We prove that the triangulation of G_1 obtained from this star-decomposition is the desired chordal sandwich. Let $H = (V, \{\{u, v\} \mid T_u \cap T_v \neq \emptyset\})$. H is a chordal graph such that $E_1 \subseteq E(H)$. To prove that $\langle G_1, G_2 \rangle$ is a yes-instance of Chordal Sandwich with $\overline{nK_2}$, it suffices to prove that $T_u \cap T_v = \emptyset$ for every $\{u, v\} \notin E_2$. We claim that it is implied by $T_{s_{uv}} \cap T_{t_{uv}} \neq \emptyset$. Indeed, assume $T_{s_{uv}} \cap T_{t_{uv}} \neq \emptyset$ and $T_u \cap T_v \neq \emptyset$. Since $s_{uv}, t_{uv} \in N(u) \cap N(v)$, $T_u, T_v, T_{s_{uv}}, T_{t_{uv}}$ pairwise intersect, there is a bag with u, v, s_{uv}, t_{uv} by the Helly property. The latter contradicts that $(T, \mathcal{X})$ is a star-decomposition because no vertex dominates the four vertices. Hence the claim is proved. So, let us prove that $T_{s_{uv}} \cap T_{t_{uv}} \neq \emptyset$. By contradiction, if $T_{s_{uv}} \cap T_{t_{uv}} = \emptyset$ then every bag B onto the path between $T_{s_{uv}}$ and $T_{t_{uv}}$ must contain c_{uv}, x_{uv}. Since $N[c_{uv}] \cap N[x_{uv}] = \{s_{uv}, t_{uv}\}$ and $(T, \mathcal{X})$ is a star-decomposition, it implies either $s_{uv} \in B$ and $B \subseteq N[s_{uv}]$ or $t_{uv} \in B$ and $B \subseteq N[t_{uv}]$. So, there are two adjacent bags $B_s \in T_{s_{uv}}, B_t \in T_{t_{uv}}$ such that $B_s \subseteq N[s_{uv}]$ and $B_t \subseteq N[t_{uv}]$. In particular, $B_s \cap B_t$ must intersect the path (y_{uv}, w_{uv}, z_{uv}) because $y_{uv} \in N(s_{uv})$ and $z_{uv} \in N(t_{uv})$. However, $N[s_{uv}] \cap N[t_{uv}] \cap \{y_{uv}, w_{uv}, z_{uv}\} = \emptyset$, that is a contradiction. As a result, $T_{s_{uv}} \cap T_{t_{uv}} \neq \emptyset$ and so, $T_u \cap T_v = \emptyset$ for any $\{u, v\} \notin E_2$.

Conversely, assume that $\langle G_1, G_2 \rangle$ is a yes-instance of Chordal Sandwich with $\overline{nK_2}$. Let H be any chordal supergraph of G_1 such that $E(H) \subseteq E(G_2)$ and H is edge-maximal w.r.t. this property. We prove in [DLN16b] that every clique-tree of H is a tree decomposition $(T, \mathcal{X})$ of G_1 with $|\mathcal{X}| = |V|/2+1$ bags such that for every $\{u, v\} \notin E_2$, $T_u \cap T_v = \emptyset$ and there are two adjacent bags $B_u \in T_u$ and $B_v \in T_v$ such that $B_u \setminus u = B_v \setminus v$. The latter is proved by elaborating on the hypothesis that $\bar{G_2}$ is a perfect matching.

In what follows, we will modify $(T, \mathcal{X})$ in order to obtain a star-decomposition of G'. To do so, we will use the fact that there are $|V|/2 = n$ edges in $E(T)$ and that for every $\{u, v\} \notin E_2$, there are two adjacent bags $B_u \in T_u$ and $B_v \in T_v$ such that $B_u \setminus u = B_v \setminus v$. Indeed, this implies that there is a one-to-one mapping $\alpha : E(T) \to E(\bar{G_2})$ between the edges of T and the non-edges of G_2. Precisely, for any edge $e = \{t, s\} \in E(T)$, let $\alpha(e) = \{u, v\} \in E(\bar{G_2})$ be the non-edge of G_2 such that $u \in X_t, v \in X_s$ and $X_t \setminus u = X_s \setminus v$.

Intuitively, the star-decomposition $(T', \mathcal{X}')$ of G' is obtained as follows. For any $t \in V(T)$ with incident edges $e_1, \cdots, e_d$, we first replace X_t by a path decomposition $(Y_{t,e_1}, \cdots, Y_{t,e_d})$. Then, for any edge $e = \{t, s\} \in E(T)$, an edge is added between $Y_{t,e}$ and $Y_{s,e}$. Finally, the center-bag of some star-decomposition of the gadget $F_{\alpha(e)}$ is made adjacent to $Y_{t,e}$ (see Figure 3.8b for an illustration).

More formally, let $t \in V(T)$ and $e \in E(T)$ incident to t, and let $\{u, v\} = \alpha(e)$.



(a) Gadget F_{uv} (top) with a star-decomposition of F_{uv} (bottom).

(b) A subtree of the star-decomposition of G' (bottom) obtained from an internal bag with degree four of $(T, \mathcal{X})$ (top). Subtrees T_i are star-decompositions of the gadgets $F_{u_i v_i}$.

Figure 3.8

Let $Y_{t,e} = V' \cup X_t \cup \{s_{uv}, t_{uv}\}$ (note that $Y_{t,e}$ is dominated by $u' \in V'$). Let $e_1, \cdots, e_d$ be the edges incident to t in T, in any order. For $1 \leq i < d$, add an edge between Y_{t,e_i} and $Y_{t,e_{i+1}}$. For any edge $e = \{t, s\} \in E(T)$, add an edge between $Y_{t,e}$ and $Y_{s,e}$. Finally, add the star-decomposition $(T^e, \mathcal{X}^e)$ for the gadget $F_{\alpha(e)}$ as depicted in Figure 3.8a and add an edge between its center and $Y_{t,e}$.

The resulting $(T', \mathcal{X}')$ is a star-decomposition of G', so, $tb(G') = 1$. □

3.4.2.3 Polynomial cases

Our polynomial-time algorithms are based on a divide and conquer approach. We recall that a separator S of G is minimal if there exist two connected components A, B of $G \setminus S$ such that $N(A) = N(B) = S$. Furthermore, A and B are called *full components* for S, and a *block* is the union of a minimal separator with one of its full components. A remarkable property of graphs with treebreadth one, whose proof is deferred to our research report [DLN16b], is that they are stable under taking blocks.

Lemma 56. *Let $G = (V, E)$, S be a separator and W be the union of some connected components of $G \setminus S$. If $tb(G) = 1$ and W contains a full component for S, then $tb(G[W \cup S]) = 1$.*

Proof. Let $(T, \mathcal{X})$ be a star-decomposition of G. We remove vertices in $V \setminus (W \cup S)$ from bags in $\mathcal{X}$, that yields a tree decomposition $(T, \mathcal{X}')$ of $G[W \cup S]$. We will prove

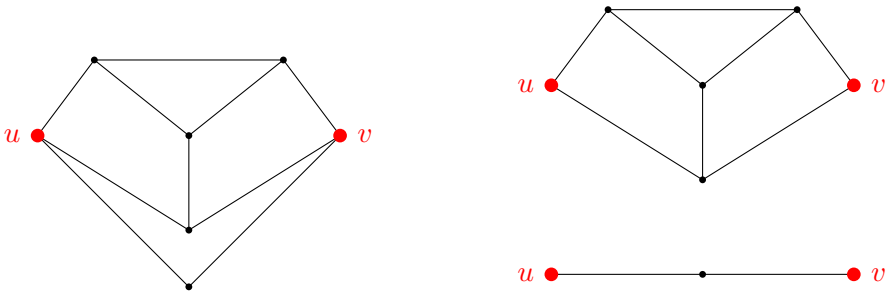


Figure 3.9: The 2-separator $\{u, v\}$ disconnects the graph G (left) in two blocks with treebreadth one (right). However, $tb(G) = 2$.

that $(T, \mathcal{X}')$ has breadth one (but is not necessarily a star-decomposition). Indeed, let $X'_t \in \mathcal{X}'$. By construction, $X'_t \subseteq X_t$ with $X_t \in \mathcal{X}$. Let $v \in X_t$ satisfy $X_t \subseteq N_G[v]$. If $v \in X'_t$, then we are done. Else, since for all $x \notin S \cup W, N(x) \cap (S \cup W) \subseteq S$ (because S is a separator by the hypothesis), we must have that $X_t \subseteq S$. Let $A \subseteq W$ be a full component for S, that exists by the hypothesis, let T_A be induced by the bags intersecting A. Since T_A and the subtrees $T_x, x \in X_t$ pairwise intersect — because for all $x \in X_t$, $x \in S$ and so, x has a neighbour in A —, then by the Helly property there is a bag in $\mathcal{X}$ containing X_t and intersecting A. Furthermore, any $u \in V$ dominating this bag must be either in S or in A, so, in particular there is $u \in A \cup S$ such that $X_t \subseteq N[u]$. □

The converse of Lemma 56 does not hold in general (see Fig. 3.9), yet there are interesting cases where it does. In fact, all our algorithms in what follows are based on particular cases where the converse of Lemma 56 also holds true. One of them is the case where S is a *clique-minimal separator*. In particular, a graph has treebreadth one if and only if every of its atoms have treebreadth one [DLN16a], and so, we may further constrain our studies to graphs without a clique-separator, *a.k.a. prime graphs*.

Case of bipartite graphs. For prime bipartite graphs, it is almost immediate that in any star-decomposition (tree decomposition with a dominator in each bag, see Sec. 3.4.2.1), every two adjacent bags must be dominated by vertices that are on the same side of the bipartition. Indeed, otherwise the adhesion set between these two bags would be either a cut-vertex or an edge-separator. The latter implies that a prime bipartite graph must be tree-convex and so, Theorem 52 follows.

Now, given a bipartite graph G, we can check whether it has treebreadth one as follows. We compute its atoms, that can be done in linear time by Lemma 48. Then, we check whether each of its atoms is tree-convex, that can also be done in linear time[4] [WLJX12]. Finally, by Theorem 52 we output $tb(G) = 1$ if and only if

[4]This problem can be reduced to dual hypertree recognition. See the proof of Theorem 44 for similar techniques.

all its atoms are tree-convex.

Case of planar graphs. Much more work was needed for the recognition of planar graphs with treebreadth one. Perhaps surprisingly, this part was arguably the most difficult one in our work on treebreadth.

The algorithm for planar graphs is recursive. Given $G = (V, E)$, we search for a specific vertex, called a *leaf-vertex*, whose closed neighborhood must be a leaf-bag of a star-decomposition if $tb(G) = 1$ (bag whose corresponding node in the tree has degree at most one). Basing on Lemma 56 and a delicate case-by-case analysis of the structure of star-decompositions, we define three types of leaf-vertices (*e.g.*, see Figure 3.10). A vertex v is a *leaf-vertex* if one of the following conditions hold.

Type 1. $N(v)$ induces an $a_v b_v$-path for some $a_v, b_v \in V \setminus \{v\}$, denoted by Π_v, of length at least 3 and there is $d_v \in V \setminus \{v\}$ such that $N(v) \subseteq N(d_v)$.

Type 2. $N(v)$ induces a path, denoted by $\Pi_v = (a_v, b_v, c_v)$, of length 2.

Type 3. $N(v)$ consists of two non adjacent vertices a_v and c_v, and there is $b_v \in (N(a_v) \cap N(c_v)) \setminus \{v\}$.

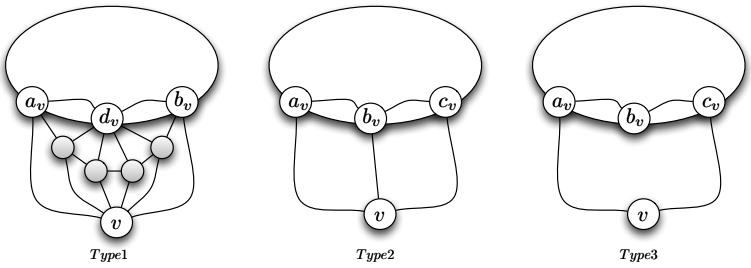


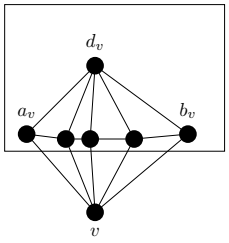
Figure 3.10: The three types of leaf vertices.

Ideally, we would like to remove v from G and apply recursively our algorithm on $G \setminus v$. However, in some case $tb(G \setminus v) = 1$ while $tb(G) > 1$ (see Fig. 3.9). So, we must also add edges between vertices that must be in a common bag of a star-decomposition of G if $tb(G) = 1$[5]. The choice of the edges to add is made more difficult by the need for the resulting graph G' to stay prime and planar in order to apply our algorithm recursively on G'. To show that $tb(G) = 1$ if and only if the resulting graph has treebreadth one also requires tedious lemmas.

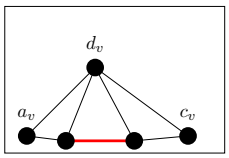
Sketch Proof of Theorem 53. Let $G = (V, E)$ be a prime planar graph. We can assume $|V| \geq 8$ and G has no star-decomposition with two bags (both cases are treated separately by exhaustive search). In such case, $tb(G) = 1$ implies there exists a leaf-vertex v, that can be found in linear time.

We first consider the case where $G \setminus v$ is prime. In this situation, we aim at removing v and applying the algorithm recursively on $G \setminus v$ (*e.g.*, see Figures 3.11a

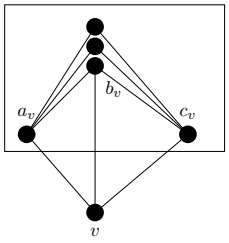
[5]We aim at turning the separator $N(v)$ into a clique. However, we cannot do that directly since it would break the distances in G, and the graph needs to stay planar.



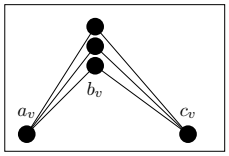
(a) v is of Type 1



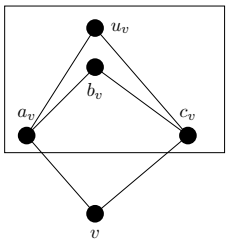
(b) G'



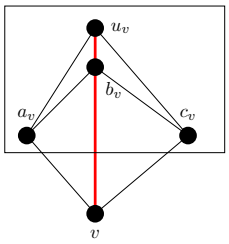
(c) $|(N(a_v) \cap N(c_v)) \setminus v| \geq 3$



(d) $G \setminus v$



(e) $(N(a_v) \cap N(c_v)) \setminus v = \{u_v, b_v\}$



(f) G'

Figure 3.11: Cases where $G \setminus v$ is prime. In every subcase, we apply the algorithm recursively on the graph to the right, that is either smaller or denser than G.

and 3.11b). However, we can do that only if it can be ensured that when $tb(G \setminus v) = 1$, there is a star-decomposition of the subgraph that can be transformed into a star-decomposition of G. Precisely, if v is of Type 1 then we seek for a star-decomposition $(T', \mathcal{X}')$ of $G \setminus v$ such that all the vertices in $N(v)$ are contained into a bag. If v is of Type 2 or 3 then we seek for a star-decomposition $(T', \mathcal{X}')$ of $G \setminus v$ such that either $T'_{a_v} \cap T'_{c_v} \neq \emptyset$, or there are two adjacent bags $B'_{a_v} \in T_{a_v}, B'_{c_v} \in T'_{c_v}$ that are respectively dominated by a_v and c_v. What we prove is that if $tb(G \setminus v) = 1$ and $G \setminus v$ is prime, then a star-decomposition as above always exists, unless we fall in the special case where v is of Type 2 or 3 and $|(N(a_v) \cap N(c_v)) \setminus v| \leq 2$. We do so by proving that it were not the case, there would exist a K_5-minor or a $K_{3,3}$-minor of G. By Kuratowski theorem, it would contradict our assumption that G is planar.

Furthermore, we prove for the latter subcase that a_v, c_v must have two common neighbours u_v, b_v in $G \setminus v$ (else, $tb(G) > 1$). In this situation, the graph G', obtained from G by adding the edges $\{v, u_v\}, \{v, b_v\}$, is planar and prime, and it satisfies $tb(G) = 1$ if and only if $tb(G') = 1$. See Figures 3.11e and 3.11f for an illustration.

So, we call the algorithm either on G' or on $G \setminus v$[6]. We refer to Figure 3.11 for an illustration.

We note that it is conceivable this first part of the analysis could apply to larger classes of H-minor free graphs. This is less clear for what follows.

Indeed, the most difficult situation is when $G \setminus v$ contains a clique-separator. Roughly, in this case we need to test the leaf-vertex v for certain properties. If it satisfies some of them then we can either remove vertices or add new edges in the graph and we call the algorithm recursively on the resulting graph G'. However, in some situations the leaf-vertex v does not satisfy *any* of the desired properties, and then we need to find a better leaf-vertex in its neighbourhood.

First, based on a fine-grained analysis of clique-separators in the subgraph $G \setminus v$, this case is reduced to the one where:

- v is of Type 2;
- there is an edge-separator (b_v, u_v) of $G \setminus v$;
- and $\{a_v, u_v\} \notin E$.

In this situation, our first idea was to add an edge between a_v and c_v in order to force these two vertices to be contained in a common bag in *any* star-decomposition of G', obtained from $G \setminus v$ by adding the edge $\{a_v, c_v\}$. Then, we aim at applying the algorithm recursively on G'. However, $tb(G') = 1$ does not imply $tb(G) = 1$ in general. We prove it is the case if u_v, c_v are nonadjacent or $N(u_v) \cap N(a_v)$ does not disconnect a_v from u_v in $G \setminus (c_v, v)$.

Else, we compute a plane embedding of G, and a vertex $x \in N(a_v) \cap N(u_v)$ such that: v, c_v and all other common neighbours of a_v, u_v are in a same region $\mathcal{R}$, bounded by (a_v, x, u_v, b_v). As illustrated with Figure 3.12, we wish to create an $a_v u_v$-path in $V \setminus \mathcal{R}$ by adding edges in $N(b_v) \cap N(x)$. In doing so, we go back to the previous subcase as now $N(a_v) \cap N(u_v)$ is no more a $a_v u_v$-separator of $G \setminus (c_v, v)$. However, we have to ensure that it is possible to add such a path in $V \setminus \mathcal{R}$, and that its addition does not affect the value of treebreadth for the graph. We prove it is the case unless $V \subseteq \mathcal{R}$ (in which case we apply the algorithm recursively on G', obtained from G by identifying b_v with x), or if there is a leaf-vertex $\ell \in N(b_v) \cap N(x)$. Furthermore, in the latter case we replace v with ℓ in the above analysis, *i.e.*, ℓ becomes the actual leaf-vertex to be considered. It can be shown that $G \setminus \ell$ is prime, so, we can prove that the algorithm always terminates.

Finally, we observe that in the above algorithm, we delete a vertex or add an edge before each recursive call. Moreover, the number of edges removed at each step can be linearly upper-bounded by the number of deleted vertices. Since planar graphs are sparse, we can elaborate on this property in order to upper-bound the number of recursive calls on n-vertex m-edge planar graphs by a linear function $\Theta(n) - m = \Theta(n)$. Each step of the algorithm can be done in linear time, so, altogether combined, it shows that the algorithm runs in quadratic time. □

[6]When v is of Type 1 we call the algorithm on G', obtained from $G \setminus v$ by contracting the internal nodes of Π_v to an edge, in order to obtain a quadratic complexity. We refer to Figures 3.11c and 3.11f for an illustration of that case.

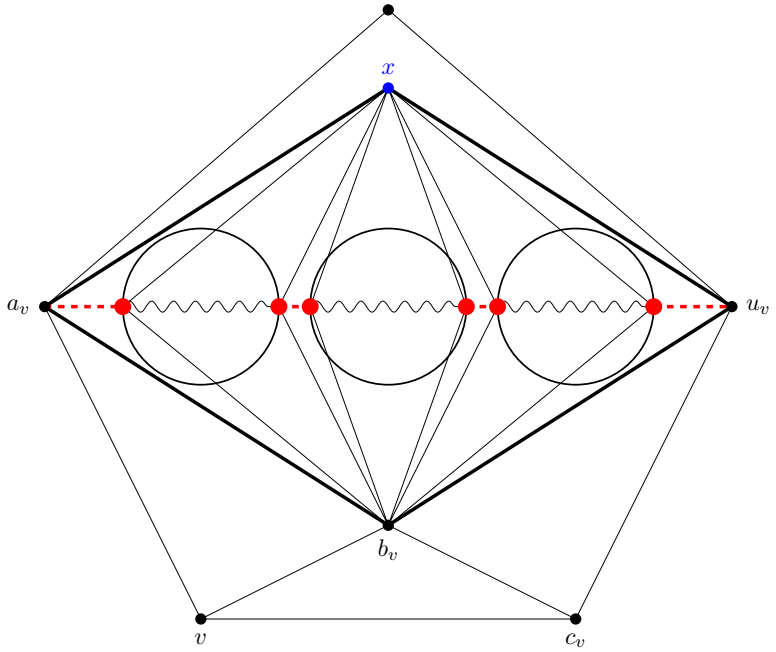


Figure 3.12: Addition of an $a_v u_v$-path in $V \setminus \mathcal{R}$. Each ball is a connected component of $G[V \setminus \mathcal{R}]$. The edges that are added in order to obtain the $a_v u_v$-path are drawn in dashed red.

3.4.3 Open problems and future work

We conclude this complexity study by some questions that remain open. First, it would be interesting to know the complexity of computing the treebreadth and the treelength of planar graphs. We did a first step in this direction with Theorem 53. Note that the complexity of computing the treewidth of planar graphs is still open. Second, all the reductions presented in this paper rely on constructions containing large clique or clique-minor. We left open the problem of recognizing graphs with treebreadth one in the class of graphs with bounded treewidth or bounded clique-number. More generally, is the problem of computing the treebreadth Fixed-Parameter Tractable when it is parameterized by the treewidth or by the size of a largest clique-minor? It is part of my ongoing work to answer these questions.

Last, I point out that in this work on star-decompositions, one important tool has been "breadth-maximal" triangulations. Precisely, for any G with $tb(G) = 1$, we call a triangulation H of G breadth-maximal if it has a clique-tree that is a star-decomposition of G and H is edge-maximal w.r.t. this property. Breadth-maximal triangulations have many nice properties which greatly simplify the analysis for the hardness reduction and the polynomial-time algorithms. So, I think this notion of "maximal" triangulation is worth being more investigated in the future, as well as for treebreadth as for treelength, treewidth, etc. The reader may refer to [BK06, BHV06] for related work, where they give sufficient conditions for edges to be always present in a triangulation of minimum width.

3.5 Treewidth versus treelength!

Finally, I present in this section new relationships between treewidth and treelength, that were obtained in collaboration with David Coudert and Nicolas Nisse. On the algorithmic side, we aim at finding such relationships in order to combine the best of both worlds (structural and metric tree-likeness in graphs).

That is, on the one hand treelength and treewidth are both NP-hard to compute [ACP87, Lok10], however treelength is much easier to approximate than treewidth. In particular, there exists a 3-approximation algorithm for computing treelength that only relies on a few breadth-first search [DG07]. In contrast, under the Small Set Expansion Hypothesis (that implies the Unique Games Conjecture) there does not exist a constant-factor polynomial-time approximation algorithm for treewidth [APW12]. On the other hand, there are more algorithmic applications for treewidth than for treelength [Cou90], which comes from the fact that several hard problems on graphs remain so even on bounded diameter graphs, thereby preventing the design of dynamic programming algorithms on tree decompositions with bounded length. Thus, by using relationships between treelength and treewidth, we wish to extend the algorithmic applications for bounded treewidth graphs to a large class of bounded treelength graphs. Furthermore, we also wish to compute efficiently (and practically) tree decompositions with bounded width on certain graph classes.

3.5.1 State of the art

As said earlier (*e.g.*, Sec. 3.2.1) the two parameters treewidth and treelength are uncomparable on general graphs. This fact prevents us from expecting simple relations between them.

On the one direction, the cycles have bounded treewidth but unbounded treelength. This suggests that having a large treelength relies on the existence of long cycles in the graph. The authors in [DG07] supported this intuition, proving that the treelength of a graph G is upper-bounded by half of the maximum length of a chordless cycle in G (the latter generalizes a similar Theorem 15 on the relationship between chordality and hyperbolicity). However, not all bounded treelength graphs have bounded chordality, as seen with the case of the wheel W_n which contains an induced C_n while it has treelength ≤ 2. Therefore, it is natural to constrain ourselves to the subcase of *isometric* cycles in graphs. We remind that a subgraph H of G is isometric if for any two vertices of H, the distance between them is the same in H as in G. Unfortunately, there are graphs such as grids with bounded-length isometric cycles and arbitrarily large treelength. As shown below, our results imply that in such a case, we always have that $tl(G) = \mathcal{O}(tw(G))$.

On the other direction, the complete graphs have unbounded treewidth but bounded treelength. Another interesting example is the graph H obtained by adding a universal vertex to a square-grid with n^2 vertices, for which it holds $tw(H) = n+1$ and $tl(H) = 2$. We observed in Section 3.2.1 that more generally, adding a universal

vertex to a graph G with arbitrarily large treewidth k will result in a graph G' with large treewidth $k+1$ and treelength at most two. One common trait of these graphs is that they have a large *genus* (they cannot be drawn with no edge-crossings onto a surface with small oriented Euler genus). That is, they are in a sense arbitrarily far from planar graphs. In contrast, it has been proved in [DG09] that $tw(G) < 12 \cdot tl(G)$ for planar graphs. Consequently, it is quite natural to ask whether a treewidth arbitrarily larger than treelength requires a large genus. In what follows, we will prove it is the case, *i.e.*, $tw(G) = \mathcal{O}(tl(G))$ for bounded-genus graphs.

Finally, and independently from this work, Belmonte et al. [BFGR15] proved that $tw(G) = \mathcal{O}(\Delta^{tl(G)})$ for any graph G with *maximum degree* Δ. On the algorithmic point of view, the authors in [BFGR15] built upon their relation in order to design a fixed-parameter-tractable algorithm to compute the metric dimension on bounded treelength graphs.

This upper-bound shows that in a way, our pathological construction which adds a universal vertex in the graph is the only one that prevents from comparing treewidth with treelength. However, it has to be noted that on the converse direction, treelength *cannot* be upper-bounded by any function $f(tw(G), \Delta)$ of the treewidth and the maximum degree, as it can be observed with cycles.

In this section, I will use different techniques in order to upper-bound the treewidth with linear dependency on the treelength.

3.5.2 Contributions: upper and lower bounds for treewidth by using treelength

3.5.2.1 Lower bound

The first result in this section is that treewidth can be lower-bounded by treelength on certain graph classes.

In what follows, a *distance-preserving elimination ordering* of $G = (V, E)$ is a total ordering of its vertex-set V such that every suffix induces an isometric subgraph of G. In particular, it is a *dismantlable ordering* if for every suffix, the closed neighbourhood of the starting vertex is dominated in the subgraph that is induced by this suffix. The latter type of ordering has been introduced in the previous chapter (Definition 7, p. 33). We also refer to the previous chapter for a definition of hyperbolicity, and especially Definition 1 (p. 25).

Theorem 57 ([CDN16]). *For every $G = (V, E)$ we have $tl(G) \leq c \cdot tw(G)$ where:*

- *$c \leq \lfloor \ell(G)/2 \rfloor$ if G has no isometric cycle of length greater than $\ell(G)$;*
- *$c \leq 2\delta(G) + 1$ with $\delta(G)$ being the hyperbolicity of G;*
- *$c \leq 2$ if G admits a distance-preserving elimination ordering.*
- *$c \leq 1$ if G admits a dismantlable ordering.*

Sharper estimates of the constant c will be discussed in what follows. One interesting consequence of this result is that every bounded-treewidth graph G can be embedded into a tree with additive distortion $\Theta(\delta(G))$. Furthermore, it tells

us that the hyperbolicity is upper-bounded by the treewidth on graph classes with a dismantlable ordering. These remarks complement Section 2.4.1 in the previous chapter on graph hyperbolicity.

3.5.2.2 Upper bound

On the other hand, treewidth can be upper-bounded by treelength on certain topological graph classes.

Let us introduce the terminology for those classes. We refer to [MT01] for details. We recall that a planar graph is a graph that can be drawn in the Euclidean plane so that edges may only intersect at their endpoints. More generally, a graph has *genus* at most g if it can be drawn in an oriented surface with Euler genus g so that edges may only intersect at their endpoints. Planar graphs are exactly the null-genus graphs. An *apex graph* is obtained from a planar graph by adding a new vertex with arbitrary neighbourhood. Finally, a class of graphs is *apex-minor free* if there is no graph in the class with a H-minor for some fixed apex graph H. Planar graphs and bounded-genus graphs are apex-minor free.

Theorem 58 ([CDN16]). *Let H be an apex graph. There exists a constant c_H that only depends on H and such that for every H-minor free graph G, we have $tw(G) \leq c_H \cdot tl(G)$.*

In particular if G has genus at most g then $tw(G) \leq 72\sqrt{2}(g+1)^{3/2} \cdot tl(G) + \mathcal{O}(g^2)$.

One unexpected consequence of this above result is that on some cases where the treewidth can be efficiently approximated, nontrivial bounds on the genus of the graph can be computed. The exact and approximate computation of graph genus are notoriously hard problems [Tho89, CKK97, KS15].

Our study paves the way to a better understanding on the relationship between structural and metric tree-likeness in graphs, and on its algorithmic consequences. Unfortunately, similar relationships for *path-likeness* in graphs look more challenging to obtain, even for trees. In particular, there are n-node trees with pathlength $\Omega(n)$ [DG07] whereas the pathwidth of an n-node tree is $\mathcal{O}(\log n)$ [Sch92].

So far, the main drawback of Theorem 58 is that it is non *constructive*. That is, when we compute a tree decomposition with bounded length $\mathcal{O}(tl(G))$, we obtain a bound on the treewidth $tw(G) = \mathcal{O}(tl(G))$, but we do not obtain a tree decomposition with bounded *width* $\mathcal{O}(tl(G))$. It is part of my ongoing work to make Theorem 58 constructive, possibly by using the graph minor decomposition from Robertson and Seymour [GKR13, DH04].

3.5.3 Proving the bounds

3.5.3.1 A detour through the diameter of minimal separators in graphs

We recall that there always exists a minimal tree decomposition (clique-tree of some minimal triangulation) with optimal width. See Section 3.2.2. Our results in what

follows provide a relationship between the width and the length in any minimal tree decomposition.

More precisely, by Theorem 42, a corresponding minimal triangulation results from the completion of all sets in a maximal family of pairwise parallel minimal separators of the graph G. In this situation, we observe that for every edge in the triangulation, either it is an edge of G or its two ends are in a same separator in the family. Note that in the latter case, the distance in G between the two ends is at most the maximum diameter in the graph over the separators in the family. Therefore, we observe that the length of the tree decomposition ($\geq tl(G)$) is exactly the maximum diameter in the graph over the separators in the family. Furthermore, since each minimal separator of the family induces a clique in the triangulation, it has size upper-bounded by the width of the tree decomposition — that is $tw(G)$ for a minimal tree decomposition with optimal width.

As a result, we are left to upper-bound the diameter of minimal separators in graphs as a function of their size.

Connectivity properties of the minimal separators. Before going into the details of the proof, let us describe the main intuition behind it and the difficulties we had to face on. Let us consider a minimal separator S for G. If it is connected, then it has diameter $\mathcal{O}(|S|)$, and so, we are done. Hence, we may assume that S consists of several connected components. The idea is to find a set of isometric cycles, each of length at most $\ell(G)$ (by definition of $\ell(G)$), such that any of these cycles intersects two components and the subgraph induced by S and these cycles is connected.

For this purpose, let us consider a minimum-length cycle crossing two components of S (such a cycle surely exists because there are at least two full components in $G \setminus S$). If this cycle is isometric, then we are done. Otherwise, it means that there is a shortcut between two nodes of the cycle. However, this shortcut could intersect S more than once which does not help our purpose.

The key point is that, using the shortcut, the initial cycle can be viewed as the sum (symmetric difference) of two smaller cycles. This kind of local view can be generalized to a global one using our main tool, namely the cycle basis (defined below). Indeed, the initial cycle is actually the symmetric difference of a set of isometric cycles [Hor87]. Using this set, we can then prove our upper bound on the diameter of minimal separators in graphs.

The set $\mathcal{C}(G)$ of Eulerian subgraphs of G is called the *cycle space* of G. It is well-known that every Eulerian subgraph can be obtained from the symmetric difference (on the edges) of cycles in G. In fact, the set $\mathcal{C}(G)$ with the symmetric difference is a vector space of dimension $m - n + 1$ if G is connected [Die10, Theorem 1.9.6]. We will call the symmetric difference of two subgraphs H_1, H_2, denoted $H_1 \oplus H_2$, the *sum* of H_1 with H_2. A cycle basis is an inclusion wise minimal set of cycles generating the whole cycle space

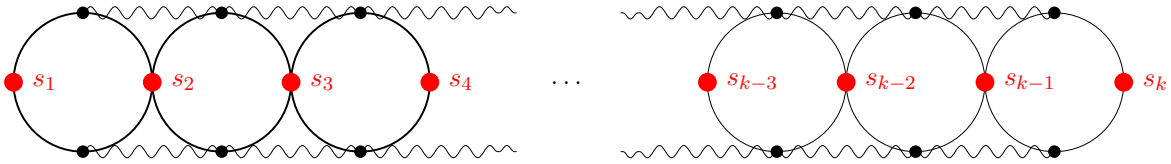


Figure 3.13: A minimal k-separator S for $G \in \mathcal{G}_\ell$ with diameter $\lfloor \ell/2 \rfloor \cdot (k-1)$. Vertices in S are ordered so that any two consecutive vertices s_i and s_{i+1} are diametrically opposed in an isometric cycle of length ℓ. Furthermore, the removal of S disconnects G in two parts, respectively containing the upper and lower sections of these cycles.

The use of the cycle space. For every $\ell \geq 3$, a graph G belongs to the class $\mathcal{G}_\ell$ if any of its cycles can be obtained from the symmetric difference on the edges of cycles of length at most ℓ. More formally, its cycle space admits a cycle basis with only cycles of length at most ℓ. As an example, by Mac Lane's Theorem the inner faces of a plane graph generate its cycle space, and so, a planar graph with inner faces of length at most ℓ is in $\mathcal{G}_\ell$. Furthermore, trees are a trivial example of graphs in $\mathcal{G}_3$ (they have no cycle). Chordal graphs are also in $\mathcal{G}_3$. More generally, every ℓ-chordal graph is in the class $\mathcal{G}_\ell$. Indeed, every chord in a cycle C can be used in order to write C as the sum of two smaller cycles, thereby proving that the induced cycles in a graph can generate its cycle space.

In [CDN16], we prove that $\mathcal{G}_\ell$ is stable under edge-contraction and addition of an edge between two vertices that are at distance at most $\lfloor \ell/2 \rfloor$. The dimension of the cycle space plays an important role in these proofs, as it often provides elegant shortenings of our technical reasonings. In order to illustrate the techniques we used, we prove below the stability of $\mathcal{G}_\ell$ under edge-contractions.

Lemma 59. *Let $\ell \geq 3$, the class $\mathcal{G}_\ell$ is stable under edge-contraction.*

Proof. Let $G \in \mathcal{G}_\ell$ with n vertices and m edges. W.l.o.g., G is connected. The dimension $dim(\mathcal{C}(G))$ of the cycle space $\mathcal{C}(G)$ is $s = m-n+1$ [Die10, Theorem 1.9.6]. Let $e \in E(G)$ such that e lies on $k \geq 0$ triangles in G. By contracting e, we loose one vertex and $k+1$ edges, the edge e and for each triangle which contains e we have to remove one of the resulting multi-edges. Hence, $dim(\mathcal{C}(G/e)) = dim(\mathcal{C}(G)) - k$. Let $\{C_1 \cdots, C_s\}$ be a basis of $\mathcal{C}(G)$ such that each C_i has length at most ℓ. Let $\{C'_1, \cdots, C'_t\}$ be the set of cycles in G/e which are obtained by contracting e on each C_i and by removing triangles that contain e from the list. Then, $t \geq dim(\mathcal{C}(G/e)) = s - k$ (since at most k triangles have been removed) and each C'_i has length at most ℓ. We show that $C'_1, \cdots, C'_t$ are linearly independent in $\mathcal{C}(G/e)$, which proves that they form a basis of $\mathcal{C}(G/e)$. For purpose of contradiction, let us assume that $C'_{i_1} \oplus \cdots \oplus C'_{i_r} = 0_{G/e}$ for $1 \leq i_1 < \cdots < i_r \leq s$ and $r > 0$, with $0_{G/e}$ being the trivial Eulerian subgraph of G/e with no edges (*a.k.a.*, the neutral element of the cycle space). Then $C_{i_1} \oplus \cdots \oplus C_{i_r}$ is either 0_G or e, with 0_G being the trivial Eulerian subgraph of G with no edges. Therefore, the sum equals e since the C_{i_j}'s are linearly independent in $\mathcal{C}(G)$. This is a contradiction as $(V(G), \{e\})$ is not Eulerian. Hence,

since all cycles in the basis $\{C'_1, \cdots, C'_t\}$ have length at most ℓ, it implies that $G/e \in \mathcal{G}_\ell$. □

Furthermore, by combining these two above properties (stability under contraction or addition of some edges), we obtain in our paper [CDN16] the following lemma:

Lemma 60. *For every $G \in \mathcal{G}_\ell$, any minimal separator S for G induces a connected subset in its power $G^{\lfloor \ell/2 \rfloor}$. In particular, the diameter of S in G is at most $\lfloor \ell/2 \rfloor \cdot (|S| - 1)$.*

Proof. By contradiction, let $G \in \mathcal{G}_\ell$, and let S be a minimal separator in G that does not satisfy the property. We first make adjacent every two vertices in S that are at distance at most $\lfloor \ell/2 \rfloor$ in G. We claim that the resulting graph still belongs to $\mathcal{G}_\ell$. Indeed, we proved in [CDN16] that $\mathcal{G}_\ell$ is stable under addition of an edge between two vertices that are at distance at most $\lfloor \ell/2 \rfloor$. Furthermore, adding an edge cannot make the distances increase in the graph, so, we can use this stability result for every edge added by the construction. Consequently, the resulting graph is still in $\mathcal{G}_\ell$. Finally, we contract each connected component of the subgraph induced by S in a single node, thus contracting S to obtain a stable set S', and since $\mathcal{G}_\ell$ is proved to be stable under edge-contractions in Lemma 59, the resulting graph G' still belongs to the class. Furthermore, the stable set S' is a minimal separator in G' by construction. Since S does not satisfy the property of the theorem, we have that all nodes in S' are pairwise at distance at least $\lfloor l/2 \rfloor + 1$ in G'. However, we proved in [CDN16, Lemma 3.3] that for every graph in $\mathcal{G}_\ell$, minimal separators are either cut-vertices or they contain two distinct vertices at distance at most $\lfloor \ell/2 \rfloor$. In particular, since the vertices in S' are pairwise at distance at least $\lfloor l/2 \rfloor + 1$ in G' by construction, it contradicts that $G' \in \mathcal{G}_\ell$. □

The above result improves upon [ASM16] and [DM15]. It is sharp, in the sense that for every size k and for every $\ell \geq 3$, there exists a graph $G \in \mathcal{G}_\ell$ with a minimal separator of size k and diameter $\lfloor \ell/2 \rfloor \cdot (k-1)$ (*e.g.*, see Figure 3.13).

Finally, Theorem 57 follows from our additional proofs in [CDN16] that all graphs with isometric cycles of length at most ℓ belong to the class $\mathcal{G}_\ell$, and in the same way all δ-hyperbolic graphs belong to $\mathcal{G}_{4\delta+3}$, all graphs with a distance-preserving ordering (resp., with a dismantling ordering) belong to $\mathcal{G}_4$ (resp., to $\mathcal{G}_3$).

Discussion. The main idea in this section is that for every G, $tl(G) \leq j \cdot tw(G)$, with j being the minimum index such that all minimal separators for G induce connected subsets in its power G^j. This index satisfies $j \leq \lfloor \ell/2 \rfloor$ for the graphs in $\mathcal{G}_\ell$. In particular, the minimal separators for a graph $G \in \mathcal{G}_3$ induce connected subsets of G, but not all graphs with this property belong to $\mathcal{G}_3$ [DLVM86]. The latter result raises the following open question: does there exist a universal constant ℓ such that the class $\mathcal{G}_\ell$ contains all graphs with connected minimal separators ?

3.5.3.2 Using the bidimensionality theory

For the upper bound, we sketch our approach and its limitations. First we observe that treelength and treewidth are stable under edge-contractions. The *bidimensionality theory* [DH08] offers meta-theorems which, for maximization problems whose solutions cannot increase under edge-contractions[7], are the cornerstone of FPT algorithms with *subexponential dependency* on the treewidth. On the theoretical point of view, these meta-theorems are based on the property that a graph with large treewidth can be edge-contracted to either a large complete graph or a large grid-like minor. The latter result is a refinement of the well-known Excluded Grid Minor Theorem from Robertson and Seymour [RST94].

We will use the same tools for proving our result on the relationship between treelength and treewidth on bounded genus graphs. Precisely, we seek for a subclass of graphs where this large obstruction to treewidth can also be shown to have a large treelength, that will imply the desired upper-bound.

Discarding complete graphs. Complete graphs are the classical examples of graphs with unbounded treewidth but bounded treelength. So, in order to get rid of this first obstruction, it is natural to constrain ourselves to H-minor free graphs, for some fixed graph H. Unfortunately, this is still not enough. Indeed, Fomin et al. proved in [FGT11] that for every fixed H, an H-minor free graph with large treewidth can be contracted either to some canonical partial triangulation of a large square grid[8], or to the same graph augmented with a universal vertex. We illustrate these two cases with Figure 3.14. In the latter case, the obstruction has unbounded treewidth and bounded treelength, which does not help our purposes.

Discarding grid-like obstructions with a universal vertex. The key observation is that this augmented partial triangulation of the grid (Figure 3.14b) is an *apex graph*. We recall that every planar graph is the minor of a grid with large enough dimensions [RS84]. Therefore, in the special case where H is a fixed apex-graph, Fomin et al. were able to refine their results. Precisely, they proved in [FGT11] that every apex-minor free graph with large treewidth can be contracted to the partial planar triangulation of a large grid, that is depicted in Figure 3.14a.

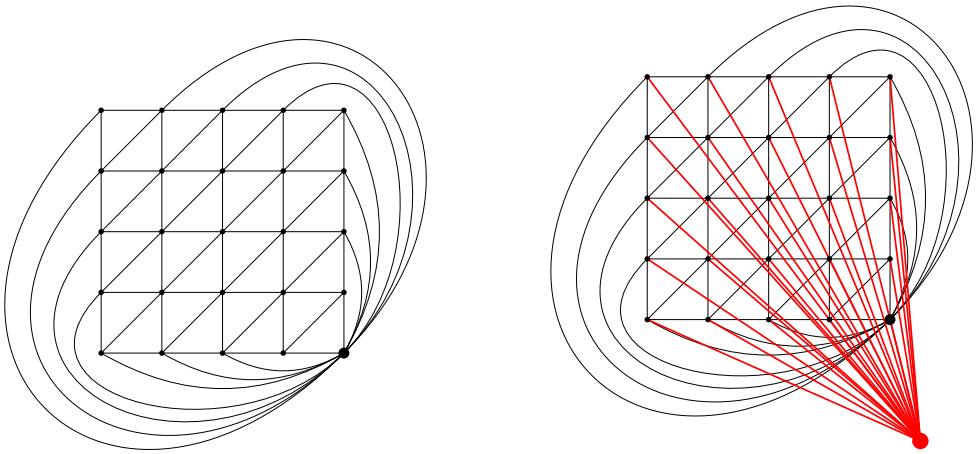
Our contributions in [CDN16] is to prove that any such a partial triangulation must have a large treelength. We do so by adapting some of the lower-bound techniques for the treelength of grids in [DG07].

Lemma 61 ([CDN16]). *Let G be a partially triangulated $(r \times r)$-grid, then $tl(G) \geq \lfloor r/3 \rfloor - 1$.*

Proof. The result holds if $r \leq 3$ because in such a case $tl(G) \geq 1 \geq \lfloor r/3 \rfloor - 1$. Else, let G' be the $(r \times r)$-grid from which G is obtained by planar triangulation.

[7] Some results also have been obtained under different stability assumptions.

[8] A triangulation of a planar graph is a planar supergraph where all the faces are triangles. Despite they share the same terminology, planar triangulations should not be confused with the triangulations from Section 3.2.2 (chordal supergraphs).



(a) Canonical partial triangulation of a grid.

(b) Triangulation augmented with one universal vertex.

Figure 3.14: Contraction obstructions to bounded treewidth.

Let V' be the set of vertices that are at distance at least $\lfloor \frac{r-1}{3} \rfloor$ from the external face of G'. The vertices of V' induce a partially triangulated $(r' \times r')$-grid F in G, $r = 2\lfloor \frac{r-1}{3} \rfloor + r'$, such that the external face has not been triangulated. Moreover, F is isometric in G. Hence, $tl(G) \geq tl(F)$. We show that $tl(F) \geq \lfloor r/3 \rfloor - 1$.

Our proof adapts from the lower-bound techniques in [DG07, Sec. 2.3]. Let $(T, \mathcal{X})$ be any tree-decomposition of F. Consider the two subsets of vertices A, B that contain the first and the last row of F respectively. Since A induces a connected subgraph of F, by the properties of tree decompositions the bags in $\mathcal{X}$ that intersect A form a subtree T_A of T. Similarly, the bags in $\mathcal{X}$ that intersect B form a subtree T_B of T. Furthermore, either $T_A \cap T_B \neq \emptyset$ (in which case, the diameter of every bag in $T_A \cap T_B$ is at least $r' - 1$), or by [DG07, Lemma 5] there exists a bag which intersects all paths between A and B in F. In the latter case, such bag must intersect the first and last column of F, and so, it has diameter at least $r' - 1$. Therefore, $(T, \mathcal{X})$ has length at least $r' - 1$ in both cases, that proves that $tl(F) \geq r' - 1 \geq \lfloor r/3 \rfloor - 1$. □

Theorem 58 now follows.

We note that in [Epp00], Eppstein has proved that the apex-minor free graphs are exactly the minor-closed families of graphs with treewidth upper-bounded by a function of their diameter. Since treelength is upper-bounded by the diameter, our result can be seen as a strict generalization of his.

3.6 Conclusion

I have been mainly interested in characterizing the complexity of computing tree decompositions with metric constraints on their bags. On the parameterized point of view, my results suggest that the hard instances for this family of problems are graphs with a large clique-number or a large Hadwiger number (size of a largest clique-minor). I insisted on this aspect when I discussed on the complexity of computing the clique-decomposition in Section 3.3. Other examples from metric graph properties studied in the literature support this observation. As an example, under the Strong Exponential Time Hypothesis the diameter of a graph cannot be computed in truly subquadratic time (see also Section 2.6.3). Hard instances for the diameter computation problem are *split graphs*, *a.k.a.* graphs who vertex-set can be bipartitioned into a clique and an independent set [BCH16].

Intuitively, the existence of a large clique makes the diameter lower in the graph, with a shortest-path between most pairs of vertices passing by the clique. In a way, it thus forces the distance distribution in the graph to be very simple. But at the same time, it gives a larger degree of freedom on the adjacency relations for the vertices out of the clique, in the sense that the edges incident to these vertices shall not affect too much the distances in the graph. Since tree decompositions must span the edge-set of the graph, it may be the case that complicated adjacency relationships for the peripheral vertices render their computation intractable.

This above intuition has guided the hardness reductions in [DLN16a, DC17]. Hence, all the graphs resulting from the hardness reductions for treebreadth, pathlength and pathbreadth have a large clique-number or Hadwiger number. The graphs resulting from the hardness reduction for treelength also satisfy this property [Lok10]. What remains to explore in more details is whether large cliques and clique-minors represent the only obstructions for an efficient computation of these above parameters. Throughout my work, partial results have been obtained in this direction. In particular, planar graphs and bipartite graphs with treebreadth one can be recognized in polynomial time. I conjecture that more generally, graphs of treebreadth one with bounded *clique-number* can be recognized in polynomial time.

However, the above example of bipartite graphs shows that a similar conjecture does not hold true for the more general problem of computing the treebreadth. Indeed, we prove in [DLN16a] that the NP-complete problem of recognizing general graphs with treebreadth one can be reduced to the problem of recognizing bipartite graphs with treebreadth at most two. The latter result suggests that the existence of a large clique-*minor* suffices to render the problem intractable.

Planar graphs are K_5-minor free, and we are currently exploring whether computing the treelength is fixed-parameter-tractable on this class of graphs. Precisely, we are investigating whether we can adapt the algorithm from Bodlaender and Kloks [BK96] to our needs. This work has been started recently during the internship of Simon Nivelle with Nicolas Nisse. I conjecture that computing the treelength of a graph G is FPT when it is parameterized by $tl(G) + tw(G)$. Moreover, it is my opinion that we may be helped in proving this with the relationships between

treelength and treewidth in Section 3.5. Similar ideas can be found in [DFG11]. However, I conjecture that the problem of computing the treelength remains NP-complete on planar graphs. This conjecture is motivated by a hardness result on the problem of deciding on the existence of tree t-spanners in these graphs [FK01]. Proving or disproving this conjecture would make advance our understanding on the structure of bounded treelength graphs.

Part II

Privacy at large scale in social graphs

Unlike the previous part, the focus in the next two chapters is on *dynamic* processes on networks. The rules of these dynamics cause certain paths between the vertices to appear or to disappear, hence they impact on the information propagation in the graph. Our general purpose is to predict the outcome of these dynamics.

- Chapter 4 presents new results on the computation of equilibria for a large family of graphical games, that are exemplified by **coloring games**. Note that equilibria for these games have been proposed in [KL13] as a solution concept for the dynamics of communities in social graphs.
- Chapter 5 introduces a new model in order to detect the **targeting** of (potentially sensitive) data by an online advertiser, and to learn which data causes the reception of a given ad. Targeting can be regarded as a dynamic process on an "adgraph" [AMM10]: built from the data inputs and the ad allocation protocols.

Chapter 4

The computation of equilibria in coloring games

Summary

We establish new complexity results for computing k-strong Nash equilibria in coloring games. These results are partly generalized to some other graphical games.

In Section 4.3, we prove that for every fixed $k \geq 1$, it can be computed a k-strong Nash equilibrium for every coloring game with a better-response dynamic. We give the exact worst-case (polynomial) time of convergence for $k \leq 2$, that we prove through an original connection between the executions of the better-response dynamics and the chains (directed paths) in a DAG called the *Dominance lattice*. However, for every $k \geq 4$, we prove that the better-response dynamic converges in *superpolynomial* time in the worst-case. The latter result disproves a conjecture from [KL13, EGM12] that for every $k \geq 1$, this dynamic converges in polynomial time.

Then, in Section 4.4, we establish new results on the parallel and space complexity of computing a Nash equilibrium in coloring games. Precisely, we prove that this problem (that is polynomial-time solvable) is PTIME-hard under NC-reductions. This is hint that computing a Nash equilibrium in these games is a problem inherently sequential, that cannot be solved within limited (logarithmic) workspace, neither with an "efficient" distributed algorithm: with low local computational time and communication complexity.

In Section 4.5, we put the focus on a generalization of coloring games to edge-weighted graphs, sometimes called the additively separable symmetric Hedonic games. We give sufficient conditions for these games to admit a k-strong Nash equilibrium. Then, we prove that for every $k \geq 2$, and for every fixed set of edge-weights $\mathcal{W}$, the following dichotomy results holds true: either all the games played on a graph with edge-weights in $\mathcal{W}$ admit a k-strong Nash equilibrium, or the corresponding decision problem is NP-complete.

Finally, a broader set of graphical games, generalizing coloring games in their own way, is introduced in Section 4.6. For each of those, we discuss on the extent to which our results for coloring games still apply.

My papers on coloring games and their generalizations [DMC12, DMC13a, DMC17, Duc16] are collected in the appendix.

Contents

4.1 Introduction

In this chapter, we aim at better understanding how the rules of the dynamics affect the *privacy* of the users'information in social graphs, that is defined in [EDP] as "*a right which prevents public authorities from measures which are [invasive for the respect of private life], unless certain conditions have been met.*" Formal definitions of this notion of privacy, in game-theoretic terms, can be found, *e.g.*, in [Dwo08]. Note that if we consider a communication network such as a social graph, private information flows through the edges of the graphs. Hence, one important aspect in the study of privacy in these networks can be informally summarized at detecting where the information can be accessed to in the graph over time. As a partial answer to this question, we will study *coloring games* on graphs in this chapter.

Precisely, our aim is to compute equilibria for those games, that have been proposed in [KL13] as a solution concept for the outcome of the communities formation

process in social networks. Coloring games and some basic definitions for this chapter will be presented in Section 4.1.1. Then, the content of this chapter will be described in Section 4.1.2. In particular, in what follows, full definitions are given in Section 4.2, while the technical sections range from Sections 4.3 to 4.6.

4.1.1 Presentation of coloring games

A coloring game is played on an undirected graph with each vertex being an agent (formal definitions will be restated with details in Section 4.2). Agents must choose a color in order to construct a proper coloring of the graph, and the individual goal of each agent is to maximize the number of agents with the same color as hers. On a more theoretical side, coloring games have been introduced in [PS08] as a game-theoretic setting for studying the chromatic number in graphs. Precisely, the authors in [PS08] have shown that for every coloring game, there exists a Nash equilibrium where the number of colors is exactly the chromatic number of the graph. Since then, these games have been rediscovered many times, attracting attention on the way in the study of information sharing and propagation in graphs [CKPS10, EGM12, KL13].

4.1.1.1 Some applications of coloring games

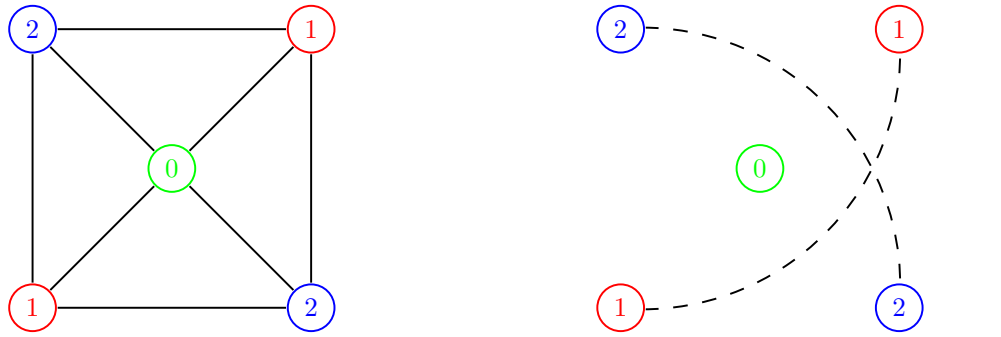
Distributed coloring in graphs. In particular, the authors in [CKPS10] base on coloring games in order to design distributed algorithms for coloring a graph, with applications to the frequency assignment problem and the design of sleep/awake protocols in Wireless Sensor Networks. The latter protocols are the cornerstone of energy saving methods in these networks, and they also serve as a routine for securing group communications.

Later on, in part motivated by the above applications, the authors in [MW13] presented a unifying framework for the so-called "distributed" welfare games. The goal with these games is to encode the solutions of a distributed resource allocation problem as the Nash equilibria of a given graph game. They are specified by assigning each agent an admissible utility function to optimize. Coloring games have been shown to fit in this generic framework.

Modeling of community formation in social networks. More recently, coloring games have been proposed in order to model community formation in social networks [KL13]. Indeed, let us assume that each community results from a group of users sharing about some information topic. Let us also assume for simplicity that each user shares about a given topic in only one community[1]. Therefore, given a *fixed* topic, communities partition the users. The dynamics of these communities is modeled with a coloring game, that is played on a "conflict graph" where each edge represents a conflicting opinion between two users.

[1]Note that by doing so, existing correlations between communities that are related to different topics are neglected [KBSP16].

This representation may be confusing because the communities are densely connected subsets in the social graph, whereas here in the coloring game they correspond to color classes, and so, to independent sets of the conflict graph. In this context, it may be more natural to define the game on the *complement* of the conflict graph: agents must construct a clique partition of this graph, and the individual goal of each agent is to maximize the size of her clique (see Figure 4.1 for an illustration).



(a) A coloring game played on a graph G. Agent are labeled with their colour.

(b) The corresponding clique partition in the complement of G.

Figure 4.1: Dual representations for coloring games.

Generalizations of coloring games have been proposed in the literature [ABK+16, BZ03, MW13]. In this chapter, we are particularly interested in a subclass of Hedonic games [Bal04], sometimes called the additively separable symmetric Hedonic games [BZ03]. We will call them *generalized coloring games* because, as shown below, they are a proper extension of the classical coloring games. A generalized coloring game is played on an *edge-weighted* graph, with each vertex being an agent. As before agents must choose a color, and the individual goal of each agent is now to maximize the sum of the weights of the edges that are incident to herself and to another agent with the same color as her.

Formally, let $G = (V, E, w)$ be an edge-weighted graph with $w : E \to \mathbb{Q} \cup \{-\infty\}$ be its weight function. A coloring $c : V \to \mathbb{N}$ of G is a partition of its vertex-set with each class (or group) being assigned a distinct integer, and for every vertex $v \in V$ we denote by $c(v)$ the integer corresponding to her group, also known as her color. Then, in the generalized coloring game that is played on G, the vertices of G are the agents of the game, and the strategy of an agent is her color. Every agent $v \in V$ aims at maximizing her utility function $\sum_{u \in N_G(v) | c(u)=c(v)} w_{uv}$. We refer to Figure 4.2 for an illustration.

Note that every coloring game that is played on an unweighted graph G^- can be transformed into a generalized coloring game, by creating an edge-weighted complete graph with vertex-set $V(G^-)$ where the edges of G^- have weight $-\infty$ and the nonedges of G^- have unit weight.

$\Longrightarrow$ From now on, we will assume the classical coloring games to be defined this

way, and all the definitions will be directly given for generalized coloring games.

4.1.2 Contributions

Our main purpose is to characterize the complexity of computing stable partitions for generalized coloring games. Those are configurations where no small subset of agents have an incentive to change their current strategy for the same new color. On a social network point of view, stable partitions ensure that no small coalition of users have an incentive to leave their current community for another one, thus preventing information leakage from a community to another.

More precisely, we carefully control the maximum size k of such a subset, and we aim at computing k-stable partitions, *a.k.a.* configurations of the game where no k-subset of agents have an incentive to deviate from their current strategy (*e.g.*, see Figure 4.3 for an illustration). Note that 1-stable partitions are exactly the Nash equilibria of the game.

Formally, for any $G = (V, E, w)$ and $c : V \to \mathbb{N}$, a k-deviation w.r.t. c is any subset $S \subseteq V$ with $|S| \leq k$ that satisfies the following property: there exists some color $i \in \mathbb{N}$ so that, for every $v \in S$, we have $c(v) \neq i$ and:

$$\sum_{u \in N_G(v) | c(u) = c(v)} w_{uv} < \sum_{u \in N_G(v) | u \in S} w_{uv} + \sum_{u \in N_G(v) | c(u) = i} w_{uv}.$$

The coloring c represents a k-stable partition if there is no k-deviation w.r.t. c[2].

We now describe our contributions in more details. Positive and negative results are obtained on the complexity of computing k-stable partitions for the classical (non generalized) coloring games with better-response dynamics (Section 4.1.2.1) and parallel or space efficient algorithms (Section 4.1.2.2). Our results on the existence of k-stable partitions in generalized coloring games are summarized in Section 4.1.2.3. Extensions of all these results to broader classes of games are finally announced in Section 4.1.2.4.

4.1.2.1 Convergence of better-response dynamics for coloring games

The first two technical sections (Sections 4.3 and 4.4) are devoted to (non generalized) coloring games. In particular, Section 4.3 is devoted to the computation of k-stable partitions for these games.

In [KL13], Kleinberg and Ligett prove that every coloring game with n agents admits a partition that is k-stable for every $k \leq n$, but that it is NP-hard to compute one (this result was also proved independently by Escoffier et al. [EGM12]). Indeed, a largest group in such a partition must be a maximum independent set of the underlying graph. In contrast, it can be computed a k-stable partition in polynomial

[2]There is a more general notion of k-deviations where the agents deviating from their current strategies are not imposed to choose the same color i. However, as shown in [EGM12] for any (non generalized) coloring game, there exists such a k-deviation if and only there is one where the at most k agents deviating choose the same color i.

time for every fixed $k \leq 3$, by using simple *better-response dynamics* [PS08, EGM12, KL13] that we will describe next. The latter results question the role of the value of k in the complexity of computing stable partitions.

Formally, a better-response dynamic proceeds as follows. We start from a trivial coloring of the graph where all the vertices have a different color and then, as long as there exists a k-deviation w.r.t. the current coloring, we pick any one of these k-deviations S and we assign a same new color i to all the vertices in S so that they strictly increase their respective utility function.

We prove in Section 4.3 that better-response dynamics can be used for computing a k-stable partition *for every fixed* $k \geq 1$ (but not necessarily in polynomial time). It shows already that for every fixed $k \geq 1$, the problem of computing a k-stable partition is in the complexity class PLS (Polynomial Local Search), that is conjectured to lie strictly between P and NP [JPY88].

Then, we relate the time of convergence of better-response dynamics with a combinatorial object that is called the *Dominance lattice* [Bry73], thereby obtaining a closed formula for the worst-case time of convergence of the better-response dynamics for $k \leq 2$. Finally, we will show how lower-bounds on the time of convergence for the better-response dynamics can be obtained for larger values of k. These bounds are obtained with a new representation of the Dominance lattice, that I will briefly sketch. In particular, the main result in this section is that for every fixed $k \geq 4$, better-response dynamics converge in *superpolynomial time* $\Omega(n^{\Theta(\log n)})$ in the worst-case. The latter result disproves conjectures of Kleinberg and Ligett [KL13] and Escoffier et al. [EGM12] that better-response dynamics always converge in polynomial time for every fixed k.

This is joint work with Dorian Mazauric and Augustin Chaintreau.

4.1.2.2 The parallel complexity of coloring games

The negative results of Section 4.3 do not preclude the possibility that a k-stable partition can be computed in polynomial time for every fixed $k \geq 4$. For instance, this could be achieved by using a different dynamic. In order to better understand the complexity of this problem, I gave a closer look at the simpler (polynomial-time solvable) problem of computing a Nash equilibrium in coloring games.

More precisely, I investigate in Section 4.4 on the parallel and space complexity of computing a Nash equilibrium in these games. This aspect is also important when considering the applications of coloring games: to serve as a basis for distributed algorithms or to model the social behaviour of users with limited memory and computing power.

I prove in Section 4.4 that the problem of computing a Nash equilibrium in coloring games is PTIME-hard under logspace reductions. The latter result suggests that this problem is inherently sequential, and that it cannot be solved within limited (logarithmic) workspace under the well-established complexity assumption $PTIME \neq LOGSPACE$.

4.1.2.3 Existence of stable partitions for generalized coloring games

We also study in Section 4.5 the existence of k-stable partitions in generalized coloring games, and on the complexity of the related decision problem. So far, it has been proved in [BZ03] that every generalized coloring game admits a Nash equilibrium. However, computing one is a PLS-complete problem. This complexity comes from the fact that edge-weights may be arbitrary. In Section 4.5, we fix in advance a set of admissible edge-weights $\mathcal{W}$. We investigate on how the choice of $\mathcal{W}$ impacts on the existence of stable partitions.

The main result in this section, found in collaboration with Dorian Mazauric and Augustin Chaintreau, is that for every fixed $\mathcal{W}$, there exists a sharp threshold $k(\mathcal{W})$ (possibly, $k(\mathcal{W}) = +\infty$) such that the following dichotomy result holds true:

- every coloring game that is played on a graph with edge-weights in $\mathcal{W}$ admits a k-stable partition for every fixed $k \leq k(\mathcal{W})$;
- however, for every fixed $k > k(\mathcal{W})$, deciding on the existence of a k-stable partition for these games is an NP-complete problem.

This sharp threshold is explicitly given for most sets $\mathcal{W}$. We complement this result with preliminary relationships between the existence of stable partitions and the structure of the underlying graph on which the game is played.

4.1.2.4 Generalization to other games

Finally, in Section 4.6 we discuss on more general games that also extend the coloring games, some of them have been already defined and studied in the literature with a slightly different terminology [KL13, DP94, BZ03]. We show that most of our results from the two previous Sections 4.3 and 4.5 can be extended to those games.

The results that are presented in Sections 4.3, 4.5 and 4.6 are grouped in a paper [DMC17] that has been submitted to *SIAM Journal of Discrete Mathematics* (see also [DMC13a, DMC13b]). Results summarized in Section 4.4 have been published independently in [Duc16].

4.2 Definitions

We refer to [OR94, Mye13] for the basics of game theory. In what follows, we restate the formal notions given in the introduction with more details.

Let $G = (V, E, w)$ be an edge-weighted graph, with $w : E \to \mathbb{Q} \cup \{-\infty\}$ be its weight function. We may assume that G is a clique by replacing the nonedges with null-weight edges, and so, we will write $G = (V, w)$ in what follows. An arbitrary partition of the vertices in G is named a *coloring*. Each group of the partition defines a color.

Every graph G defines a *generalized coloring game* whose agents are its vertices. Configurations of this game are the colorings of G. In particular, the strategy of an agent is her color. Furthermore, given a configuration of the game, every agent

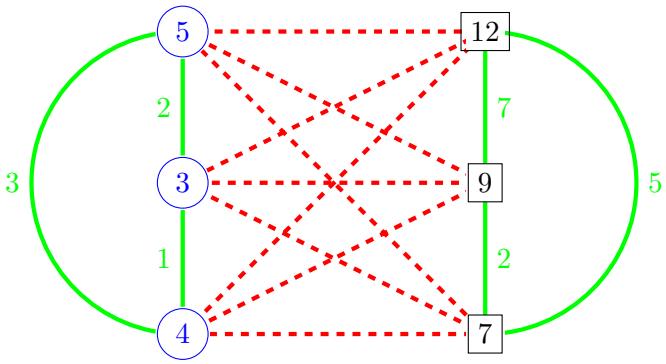


Figure 4.2: A bicoloring of a graph $G = (V, w)$. Agents that are represented by a circle (resp., by a square) have the same color. Red dashed edges have negative weight $-\infty$, while green continuous edges are labeled with their (positive) weight. Furthermore, each agent is labeled with her payoff.

$v \in V(G)$ receives payoff $\sum_{u \in V \setminus v | c(u)=c(v)} w_{uv}$, with $c(u)$ being the color of u. We refer to Figure 4.2 for an illustration.

Let us point out that classically, the non generalized coloring games are defined on an *unweighted* graph that is obtained from G by removing all edges with positive weight. We shall come back to this point later on in the section.

4.2.1 Stable partitions and better-response dynamics

Let us fix a configuration of the (generalized) coloring game that is played on G. A subset $S \subseteq V(G)$ with $|S| \leq k$ is a *k-deviation* if it can be assigned a same color to all the agents in S (different from their current color) so that their respective payoff is increased. Examples of 2-deviations are provided with Figure 4.3. When no k-deviation exists, we call the configuration a *k-stable partition*. The k-stable partitions correspond to the notion of *k-strong* Nash equilibria. In particular, 1-stable partitions correspond to the classical notion of Nash equilibria. Note that a k-stable partition might fail to exist, as shown with Figure 4.3.

The following *better response dynamics* are a classical approach in order to compute stable partitions. They are used in [KL13] in order to model the social choices of users in the community formation process.

Let $k \geq 1$ be fixed. We start from a trivial configuration where each agent has a different color. Then, as long as there exists a k-deviation, we pick any existing k-deviation S and we assign a same color c to all the agents in S so that they increase their respective payoff. Let us point out that c can be either a new colour (we make of S a new color class) or a color already assigned to some other agents not in S (we make the agents in S part of an existing color class). Furthermore, if this above dynamic converges then it stops on a k-stable partition.

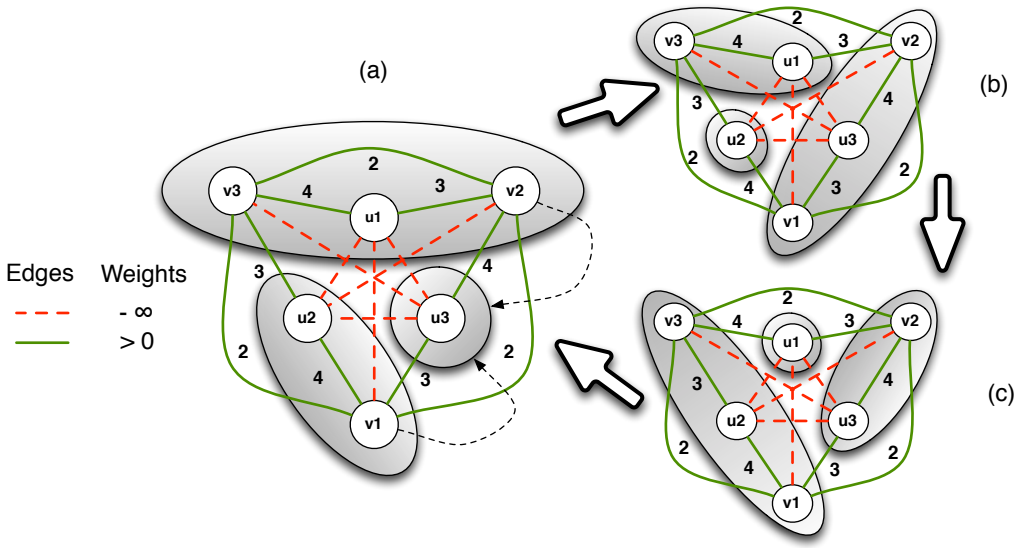


Figure 4.3: A graph $G = (V, w)$ with set of edge-weights $\mathcal{W} = \{-\infty, 2, 3, 4\}$. The coloring game played on G does not admit any 2-stable partition. Indeed, we here represent its 1-stable partitions, none of which is a 2-stable partition.

4.2.2 Friendship and conflict graphs

Finally, given an edge-weighted graph G, we define two unweighted graphs whose properties will be shown to be related to the properties of the generalized coloring game that is played on G.

- The *friendship graph* G^+ has for vertex-set $V(G)$ and for edge-set the edges of G with positive weight;
- Similarly, the *conflict graph* G^- has for vertex-set $V(G)$ and for edge-set the edges of G with negative weight.

As an example, given $G = (V, w)$ in Figure 4.2, the friendship graph G^+ is a disjoint union of two triangles, and the conflict graph G^- is a complete bipartite graph $K_{3,3}$.

Let us consider the particular case where the edges of G have weight either 1 or $-\infty$. In this situation, stable partitions for the coloring game that is played on G are *proper colorings* of the conflict graph G^-, *a.k.a.* colorings where no two adjacent vertices are assigned the same color. This justifies the terminology of coloring games.

4.3 Unweighted games: the time of convergence for better-response dynamics

The next two sections are devoted to the particular case of (non generalized) coloring games, *i.e.*, when the edge-weights of the underlying graph belong to $\{-\infty, 1\}$.

Classically [PS08, CKPS10, KL13, EGM12], these games are assumed to be played on the conflict graph G^- that is induced by the edges weighted $-\infty$. In particular, the goal of each agent is to construct a proper coloring of G^- while maximizing the number of agents with the same color as herself. Since the conflict graph is unweighted, we will sometimes call these games the unweighted coloring games in what follows.

Our purpose in this section is to (partly) characterize the complexity of computing a k-stable partition for these games, for every fixed k. Indeed, Kleinberg and Ligett [KL13] proved that for every k, every unweighted game admits a k-stable partition. However, finding a coloring that is a k-stable partition for every k is an NP-hard problem. In what follows, we will subdivide our contributions in three parts. Each part is devoted to the proofs of upper and lower bounds on the time of convergence for better-response dynamics.

- We first prove in Section 4.3.1 that for every fixed $k \geq 1$, better-response dynamics always converge to a k-stable partition. We discuss on the consequences of this result on the complexity of computing k-stable partitions.
- Then, we obtain in Section 4.3.2 the *exact* worst-case time of convergence for $k \leq 2$.
- Finally, we prove in Section 4.3.3 that better-response dynamics converge in *superpolynomial time* as soon as $k \geq 4$. The latter result answers negatively to an open question from [KL13, EGM12].

4.3.1 A finer-grained complexity for the problem of computing k-stable partitions

First, we prove that when applied to unweighted games, better-response dynamics always converge. Then, we discuss about the implications of this result on the complexity of computing a k-stable partition.

The following proof makes use of a *partition vector*, first introduced in [CKPS10].

Definition 62. Given a proper coloring of G^-, let λ_i be the number of colors c_i so that exactly i agents are colored by c_i. We denote by $\overrightarrow{\Lambda} = (\lambda_n, \ldots, \lambda_1)$ the partition vector of the coloring.

As an example, suppose that G^- is a complete bipartite graph with two sides of respective size n_1 and n_2, and we color all vertices on a same side with the same color. If $n_1 = n_2$ then $\lambda_{n_1} = 2$ and for every $i \neq n_1$, $\lambda_i = 0$. Otherwise, $\lambda_{n_1} = \lambda_{n_2} = 1$, and for every $i \notin \{n_1, n_2\}$, $\lambda_i = 0$.

Lemma 63. *For any (conflict) graph G^-, let us consider the unweighted game that is played on G^-. Then, for every $k \geq 1$, the better-response dynamic applied to this game converges to a k-stable partition.*

Proof. At each time we modify the current coloring of G^-, we also modify the corresponding partition vector $\overrightarrow{\Lambda}$. So, in order to prove that the better-response

dynamic converges, it suffices to prove that $\overrightarrow{\Lambda'}$, obtained after the coloring has changed, is lexicographically greater than $\overrightarrow{\Lambda}$. Let us prove this by fixing a k-deviation S (w.r.t. the current coloring). After the coloring has been changed – with respect to S –, all vertices in S have strictly increased their payoff. For unweighted games, this is equivalent to have all vertices in S increase the number of agents with the same color as theirs. In particular, let c be the color assigned to all the agents in S, and let j be the number of agents colored c before the coloring has been changed. By the hypothesis, the change of coloring results in $j + |S|$ vertices colored c. So, we get $\overrightarrow{\Lambda'} - \overrightarrow{\Lambda} = (0, \ldots, 0, \lambda'_{j+|S|} - \lambda_{j+|S|} = 1, \ldots)$, and so, $\overrightarrow{\Lambda} <_{Lex} \overrightarrow{\Lambda'}$. Finally, as the number of possible vectors is finite, we obtain the convergence of the better-response dynamic. □

Next, we discuss on the consequences of Lemma 63 on the complexity of computing k-stable partitions for unweighted games. Informally, an optimization problem is in PLS (Polynomial Local Search) if an optimal solution can be computed with a local-search algorithm, *i.e.*, an algorithm converging to an optimal solution by repeatedly improving a current solution with a slight pertubation of it[3]. Lemma 63 proves that for every fixed k, the problem of computing a k-stable partition for unweighted games is in PLS. This complexity class is strictly included in NP unless NP=coNP [JPY88].

Hence, to summarize this subsection, we have improved the best-known results on the complexity of computing a k-stable for unweighted games, for every fixed k.

4.3.2 Closed formula for the worst-case time of convergence of better-response dynamics ($k \leq 2$)

Polynomial-time solvable problems are conjectured to be strictly contained in PLS [JPY88]. In this section, we are interested in characterizing for which values of k the problem of computing a k-stable partition is in P. As a partial answer to this question, we aim at characterizing for which values of k the corresponding better-response dynamic converges within a polynomial number of steps.

It was proved in various papers [PS08, KL13, EGM12] that better-response dynamics converge in polynomial-time for every fixed $k \leq 3$. The proofs in these papers rely on a potential function argument. We now give an alternative proof of this result for the case $k \leq 2$. It is based on a more combinatorial argument and it allows us to obtain the *exact* worst-case time of convergence.

Theorem 64. *Let G^- be an n-vertex conflict graph. We consider the unweighted game that is played on G^-. Let m and r be the unique non negative integers such that $n = \frac{m(m+1)}{2} + r$, and $0 \leq r \leq m$.*

Then, for every $k \leq 2$, the better-response dynamic applied to this above game converges to a k-stable partition within no more than $2\binom{m+1}{3} + mr \sim \frac{2\sqrt{2}}{3} n\sqrt{n}$ steps.

[3] Each step of the algorithm takes polynomial time, but the number of steps may be superpolynomial.

Moreover this worst-case upper-bound is reached if the conflict graph G^- has no edges.

The remaining of this subsection is devoted to the proof of Theorem 64.

At first glance, it might look counter intuitive that the worst-case convergence time of the dynamic is reached for the edgeless conflict graph. Indeed, when the conflict graph has no edges there is a unique stable partition, with all agents having the same color. However, this can be better understood by noticing that every proper coloring of a conflict graph G^- is also a proper coloring of the edgeless conflict graph $G^\emptyset$ with same vertices. In particular, if we color accordingly G^- and $G^\emptyset$ then a k-deviation for G^- is also a k-deviation for $G^\emptyset$. It directly follows from this observation that the worst-case convergence time for better-response dynamics is always reached with $G^\emptyset$.

The proof of Theorem 64 also makes use of partition vectors. As for Lemma 63, we show that every time the coloring is changed by using a 1-deviation (resp., a 2-deviation), the corresponding partition vector increases with respect to some ordering. However, in order to prove a polynomial upper-bound for the time of convergence, we cannot use anymore the lexicographical ordering, since it is a *total ordering* and the number of partition vectors is superpolynomial [HW79]. Instead, we will use a partial ordering that was introduced by Brylawski in [Bry73], in a somewhat different context.

Integer partitions. We first observe that when the game is played on an n-vertex conflict graph, each partition vector of its colorings defines a unique way to write n as a sum of positive integers. The latter means that partition vectors are in bijective correspondance with the nonincreasing sequences of n nonnegative integers whose terms sum up to n. More precisely, every vector $\overrightarrow{\Lambda}$ is related to the nonincreasing sequence $Q(\overrightarrow{\Lambda})$, with its $n - \sum_{i=1}^{n} \lambda_i$ lowest terms equal to zero, and exactly λ_i terms equal to i for every $1 \leq i \leq n$. These sequences are called integer partitions in the literature [Bry73, HW79].

Dominance ordering. Brylawski has defined an ordering over the integer partitions, sometimes called the dominance ordering [Bry73]. Namely, given two partitions, one is greater than the other if and only if for every $1 \leq i \leq n$, the sum of its i largest terms is greater than or equal to the sum of the i largest terms of the other. The latter ordering is a direct application of the theory of majorization to integer partitions [OM16].

For instance, let us consider two trivial colorings of $G^\emptyset$: one with every agent having a different color, and another with every agent having the same color. In the first case, the partition vector is $\overrightarrow{\Lambda} = (0, \ldots, 0, n)$ so that $\lambda_1 = n$ and $\lambda_i = 0$ for every $i > 1$; in the second case, the partition vector is $\overrightarrow{\Lambda'} = (1, 0, \ldots, 0)$ so that $\lambda'_n = 1$ and $\lambda'_i = 0$ for every $i < n$. The corresponding integer partitions are $Q(\overrightarrow{\Lambda}) = (1, 1, 1, \ldots, 1)$ and $Q(\overrightarrow{\Lambda'}) = (n, 0, 0, \ldots, 0)$. In particular, the i largest

terms of $Q(\overrightarrow{\Lambda'})$ always equal n, whereas the i largest terms of $Q(\overrightarrow{\Lambda})$ equal $i \leq n$. Hence, $\overrightarrow{\Lambda'}$ is greater than $\overrightarrow{\Lambda}$ w.r.t. the dominance ordering.

Relationship with 2-deviations. The dominance ordering gives rise to a lattice on the integer partitions. Furthermore, it has been proved in [GK86] that a longest chain in this lattice has length $2\binom{m+1}{3} + mr$, with m and r being the unique non negative integers such that $n = \frac{m(m+1)}{2} + r$, and $0 \leq r \leq m$. Therefore, in order to prove Theorem 64 we have been left to prove a correspondance between the valid sequences of 2-deviations in $G^{\emptyset}$ and the chains of integer partitions in the Dominance lattice. Note that this correspondance holds true only for the edgeless conflict graph $G^{\emptyset}$. Below, we first prove this correspondance in the case of 1-deviations.

Lemma 65. *Assuming $G^{-} = G^{\emptyset}$ is edgeless, let Q, Q' be two integer partitions of $n = |V|$.*

Then, Q' dominates Q if and only if there exist two colorings c, c' of G^{-} with respective partition vectors $\overrightarrow{\Lambda}$ and $\overrightarrow{\Lambda'}$ such that: $\mathcal{Q}(\overrightarrow{\Lambda}) = Q$, $\mathcal{Q}(\overrightarrow{\Lambda'}) = Q'$, and there is a valid sequence of 1-deviations from c to c'.

Proof. ($\Rightarrow$) Suppose that Q' dominates Q. We may assume w.l.o.g. that there is no intermediate integer partition Q'' such that Q' dominates Q'' and Q'' dominates Q. Indeed, then we can prove the result in general by induction on the length of a longest chain from Q to Q'. In this situation, we say that Q' covers Q. Brylawski [Bry73] has proposed a combinatorial characterization of the covering relation. Precisely, Q' covers Q if and only if there exist indices j, k satisfying:

- $k = j + 1$ or $q_j = q_k$;
- $q'_j = q_j + 1$, $q'_k = q_k - 1$, and for all i such that $i \notin \{j, k\}$, $q'_i = q_i$.

In particular, since $k = j + 1$ or $q_j = q_k$, we get $q_j \geq q_k$.

Let c be any coloring with partition vector $\overrightarrow{\Lambda}$, so that $\mathcal{Q}(\overrightarrow{\Lambda}) = Q$. We order the color classes by nonincreasing size, naming L_i the i^{th} largest class, in a way so that $|L_i| = q_i$. Then, we pick any $v \in L_k$, that exists since $|L_k| = q_k > 0$. Since by construction there are $|L_j| = q_j \geq q_k$ agents with color j, and there is no edge incident to v and to another agent with color j by the hypothesis, therefore, we can increase the payoff of v by changing her color for j. By doing so, we obtain a new coloring c' with partition vector $\overrightarrow{\Lambda'}$ such that $\mathcal{Q}(\overrightarrow{\Lambda'}) = Q'$.

Conversely, ($\Leftarrow$) let c and c' be two colorings with respective partition vectors $\overrightarrow{\Lambda}$ and $\overrightarrow{\Lambda'}$ such that $\mathcal{Q}(\overrightarrow{\Lambda}) = Q$ and $\mathcal{Q}(\overrightarrow{\Lambda'}) = Q'$. Assume that c' can be obtained from c after a 1-deviation. In particular, let v change her color. We can order the color classes by nonincreasing size, naming L_i the i^{th} largest class, in a way so that:

- v changes her color $c(v) = k$ for $c'(v) = j$, with $j \leq k$;
- every color class L_i, $i < j$, has size $|L_i| > |L_j|$;
- every color class L_i, $i > k$, has size $|L_i| < |L_k|$.

Note that $Q = (|L_1|, |L_2|, \ldots, |L_j|, \ldots, |L_k|, \ldots, |L_n|)$ by construction. In particular, Q' is such that $q'_i = q_i = |L_i|$ if $i \notin \{j, k\}$, and $q'_j = |L_j| + 1$, $q'_k = |L_k| - 1$. As a

consequence, we have that Q' dominates Q by the hypothesis. Note that this second part of the proof holds for any conflict graph G^-. □

To complete the proof of Theorem 64, we need to show that 2-deviations cannot make the time of convergence of the dynamic increase. We prove this below with a finer-grained analysis of the partition vectors that are obtained after 2-deviations.

Lemma 66. *Assuming $G^- = G^\emptyset$ is edgeless, let Q, Q' be two integer partitions of $n = |V|$. Suppose that there exist two colorings c, c' of G^- with respective partition vectors $\overrightarrow{\Lambda}$ and $\overrightarrow{\Lambda'}$ such that: $\mathcal{Q}(\overrightarrow{\Lambda}) = Q$, $\mathcal{Q}(\overrightarrow{\Lambda'}) = Q'$, and c' is obtained from c after a 2-deviation. Then, Q' dominates Q.*

Proof. Let $S = \{u, v\}$ be a 2-deviation w.r.t. c so that c' is obtained from c by assigning a same color j to u and v. Furthermore, let $i = c(u)$ and let $i' = c(v)$. In what follows, we denote by $L_i, L_{i'}, L_j$ the color classes of c that correspond, respectively, to the colors i, i' and j.

We note that if $|L_j| \geq |L_i|$ then u can increase her payoff by changing her current color i for j. In this situation, c' can be obtained from c after a sequence of two 1-deviations, with u followed by v changing their respective colors for j sequentially. Therefore, Q' dominates Q by Lemma 65. Similarly, if $|L_j| \geq |L_{i'}|$ then c' can be obtained from c by changing the respective colors of v then u for color j sequentially. Therefore, we also have in this case that Q' dominates Q by Lemma 65. From now on, let us assume that $|L_j| = |L_i| - 1 = |L_{i'}| - 1$. There are two cases to be considered:

- Suppose that $i = i'$. Then the numbers of agents colored by i and j in c' are respectively $|L_i \setminus \{u, v\}| = |L_i| - 2$ and $|L_j \cup \{u, v\}| = |L_j| + 2 = |L_i| + 1$. In particular, another coloring c'' can be obtained from c with a 1-deviation as follows. We pick any agent $u_j \in L_j$ and we make her payoff increase from $|L_j| - 1 = |L_i| - 2$ to $|L_i| = |L_j| + 1$ by changing her current color j for i. By doing so, the coloring c'' so obtained has the same partition vector as c'. Therefore, since c'' is obtained from c after a 1-deviation, Q' dominates Q by Lemma 65.
- Otherwise, $i \neq i'$. Then the numbers of agents colored by i, i' and j in c' are respectively $|L_i \setminus \{u\}| = |L_i| - 1$, $|L_{i'} \setminus \{v\}| = |L_{i'}| - 1 = |L_i| - 1$ and $|L_j \cup \{u, v\}| = |L_j| + 2 = |L_i| + 1$. Again, another coloring c'' with the same partition vector as c' can be obtained from c after a 1-deviation, this time by modifying the color of v from i' to i. Since c'' is obtained from c after a 1-deviation, Q' dominates Q by Lemma 65.

□

By Lemma 66, the maximum number of consecutive 2-deviations in better-response dynamics is upper-bounded by the length of a longest chain in the Dominance lattice. Since Lemma 65 proves that it can be obtained a sequence of 1-deviations with exactly this length, Theorem 64 follows.

Perspectives. In [PS08], Panagopoulou and Spirakis proved that for every conflict graph G^- with independent number $\alpha(G^-)$, the better-response dynamic converges to a Nash equilibrium within $\mathcal{O}(n \cdot \alpha(G^-))$ steps. This improves upon the upper-bound of Theorem 64 for the graphs with independent set $\alpha(G^-) = o(\sqrt{n})$. I conjecture that the worst-case time of convergence of the dynamic is an $\mathcal{O}(n \cdot \sqrt{\alpha(G^-)})$, that would be the best possible.

4.3.3 Lower-bounds for the worst-case time of convergence of better-response dynamics ($k \geq 4$)

Finally, on the negative side we lower-bound the worst-case running-time of better-response dynamics for $k = 4$. It has been conjectured in [EGM12] that in the case of unweighted games, better-response dynamics always converge in polynomial time for every fixed k. Our results for $k = 4$ disprove this conjecture.

Theorem 67. *Let $G^\emptyset$ be an edgeless conflict graph with n vertices. We consider the unweighted game played on $G^\emptyset$.*

Then, for every $k \geq 4$, better-response dynamics applied to this above game converge in $\Omega(n^{\Theta(\log n)})$ steps in the worst-case.

Due to its high level of technicality, the proof of Theorem 67 will be only sketched in what follows. The full proof can be found in [DMC17].

4.3.3.1 Cascade sequences: overview

In order to give a flavor of the method, let us consider some coloring of $G^\emptyset$, with partition vector $\overrightarrow{\Lambda}$ so that $\lambda_p \geq 4$ and $\lambda_{p-3} \geq 1$ for some p. We take a subset S of four agents, each with a distinct color and receiving payoff $p-1$. Such a subset surely exists since $\lambda_p \geq 4$. Then, since $\lambda_{p-3} \geq 1$, there exists some color c that is assigned to exactly $p-3$ agents. Assigning color c to the four agents in S would increase their respective payoff from $p-1$ to p, so, S is a 4-deviation. This case is interesting because after the 4-deviation, the length of a longest chain, in the Dominance lattice, from the current coloring to the *unique* stable partition of $G^\emptyset$ (where all the agents have the same color) has been increased. Hence, we aim at using this type of 4-deviations in order to maximize the number of steps for the better-response dynamic.

With that goal in mind, we now define *cascade sequences*. Indeed, suppose now that for some p, we have as before $\lambda_p \geq 4$ and $\lambda_{p-3} \geq 1$, but also $\lambda_i \geq 1$ for every $i \leq p-4$. As it is described above, we modify the current coloring with a 4-deviation, thereby obtaining as the new partition vector $\overrightarrow{\Lambda'}$ so that:

$$\begin{cases} \lambda'_{p+1} & = \lambda_{p+1} + 1 \\ \lambda'_{p} & = \lambda_{p} - 4 \\ \lambda'_{p-1} & = \lambda_{p-1} + 4 \\ \lambda'_{p-3} & = \lambda_{p-3} - 1 \\ \lambda'_{i} & = \lambda_i \text{ otherwise.} \end{cases}$$

Then, since $\lambda'_{p-1} \geq 4$ and $\lambda'_{p-4} = \lambda_{p-4} \geq 1$, we can modify the new coloring with another 4-deviation, and so on. As an example, the following is a cascade sequence of size four. Each configuration is represented with an integer partition:

$$\mathcal{Q}_0 = (7, 7, 7, 7, 4, 3, 2, 1),$$

$$\mathcal{Q}_1 = (8, 6, 6, 6, 6, 3, 2, 1),$$

$$\mathcal{Q}_2 = (8, 7, 5, 5, 5, 5, 2, 1),$$

$$\mathcal{Q}_3 = (8, 7, 6, 4, 4, 4, 4, 1),$$

$$\mathcal{Q}_4 = (8, 7, 6, 5, 3, 3, 3, 3).$$

In order to lower-bound the time of convergence in the worst-case, we aim at maximizing the size and the number of cascade sequences during the steps of the dynamic. The latter is achieved through a complex recursive procedure, where we define larger and larger cascades (but in fewer and fewer number) by inserting complex sequences of "adaptive" 1-deviations in-between. In the following Section 4.3.3.2, we will introduce new technical notions that we use in [DMC17] in order to formally define this procedure.

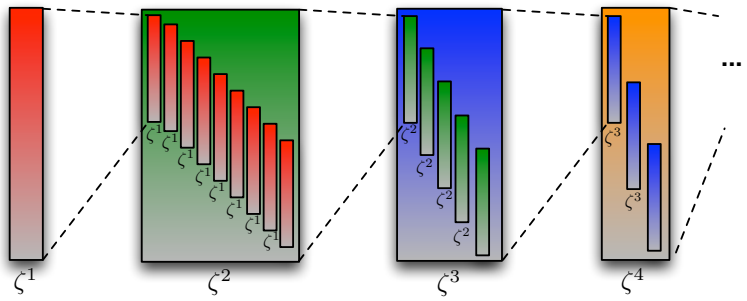


Figure 4.4: A recursive procedure in order to increase the size of cascade sequences (sketch).

4.3.3.2 Representing long sequences of 4-deviations with vectors

Our construction can be best defined by using a vectorial representation of 4-deviations. More precisely, when we change a coloring with partition vector $\overrightarrow{\Lambda}$ for another coloring with partition vector $\overrightarrow{\Lambda'}$, the deviation corresponding to that change can be represented with the difference vector $\overrightarrow{\Lambda'} - \overrightarrow{\Lambda}$. As an example, if after a 1-deviation some agent increases her payoff from $p-1$ to $p+1$ then she leaves a group of size p for some other group of size $p+1$. In particular, her former color class has size $p-1$ after her departure, and her new color class has size $p+2$ after

her arrival. Therefore, the corresponding difference vector $\overrightarrow{\Delta} = \overrightarrow{\Lambda'} - \overrightarrow{\Lambda}$ satisfies:

$$\begin{cases} \delta_{p+2} & = 1 \\ \delta_{p+1} & = -1 \\ \delta_{p} & = -1 \\ \delta_{p-1} & = 1 \\ \delta_{i} & = 0 \text{ otherwise.} \end{cases}$$

Symmetric property. Our recursive cascades are easier to represent this way, *i.e.*, as a vectorial sequence satisfying some "symmetric properties", that we define next.

Definition 68. The minimum-size sub-vector that contains all non-zero entries of a vector is called the *support* of the vector. We say a vector has the *symmetric property* if, and only if, the coordinates of its support are invariant under the reverse permutation (in which case, it is said "symmetric").

On the one hand, we show in [DMC17] that except for a few pathological cases, every 1-deviation yields an elementary vector that has the symmetric property. But the property does not hold in general for k-deviations whenever $k \geq 3$. This might give a hint of what changes in the nature of the problem when larger deviations are allowed.

On the other hand, the use of this above vectorial representation might lead to define vectorial sequences that do not truly represent sequences of 4-deviations. Thus, we need to define additional constraints in order to prevent that case from happening, which unfortunately level up the technicality of the proof. We give a flavour of it by introducing the notion of **balanced sequences**.

Definition 69. Given any integer $h > 0$, let $\overrightarrow{\varphi}^1, \overrightarrow{\varphi}^2, \ldots, \overrightarrow{\varphi}^t$ be vectors. We call this sequence h-balanced if, for any $1 \leq i \leq t$, the sum of the i first vectors, namely $\sum_{j=1}^{i} \overrightarrow{\varphi}^j$, has all its entries greater than or equal to $-h$.

As an example, since agents in a k-deviation can be in no more than k distinct color classes, the vector gotten after any k-deviation is always k-balanced.

Given a h-balanced sequence $(\overrightarrow{\varphi}^1, \overrightarrow{\varphi}^2, \ldots, \overrightarrow{\varphi}^t)$ of k-deviations, let $\overrightarrow{\Phi} = \sum_{i=1}^{t} \overrightarrow{\varphi}^i$ be the sum of all deviations. In what follows, we will say that $\overrightarrow{\Phi}$ *represents* the sequence. Let $p_{\max}$ be the largest index j that satisfies $\overrightarrow{\Phi}_j \neq 0$. Equivalently, $p_{\max}$ is the largest size of a group modified (hence created) after some deviation in the sequence happens (*i.e.*, $\forall l, \forall p > p_{\max}, \varphi_p^l = 0$). Then, one can observe that a sufficient condition so that the sequence is valid is that it starts from a coloring with at least h color classes of each size j, for $1 \leq j \leq p_{\max}$.

The following claim will be used in our sketch proof for Theorem 67.

Claim 70. *Suppose that* $\overrightarrow{\Phi^1}$ *and* $\overrightarrow{\Phi^2}$ *respectively represent a* h_1*-balanced sequence and a* h_2*-balanced sequence. Then,* $\overrightarrow{\Phi} = \overrightarrow{\Phi^1} + \overrightarrow{\Phi^2}$ *represents a* $\big(\max\{h_1, h_2 - \min_i \Phi^1_i\}\big)$*-balanced sequence, that is the concatenation of the two sequences represented by* $\overrightarrow{\Phi^1}$ *and* $\overrightarrow{\Phi^2}$*.*

Proof. Clearly, $\overrightarrow{\Phi}$ represents the sequence obtained by the concatenation of the two sequences that are respectively represented by $\overrightarrow{\Phi^1}$ and $\overrightarrow{\Phi^2}$. In particular, the subsequence represented by $\overrightarrow{\Phi^1}$ is h_1-balanced by the hypothesis. The remaining subsequence is represented by $\overrightarrow{\Phi^2}$, that is h_2-balanced by the hypothesis. Since it follows the first subsequence, and all the entries of $\overrightarrow{\Phi^1}$ are greater than or equal to $\min_i \Phi^1_i$, therefore, this second subsequence is $(h_2 - \min_i \Phi^1_i)$-balanced. ◇

Sketch of the construction. Our proof for Theorem 67 relies on a "shift" operator: given a vector $\overrightarrow{\varphi}$ whose support ranges between indices $p_{\min}, p_{\max}$, the vector ${}^{\mathtt{tr}(i)}\overrightarrow{\varphi}$, $i < p_{\min}$, is a vector of the same size *and the same support* as $\overrightarrow{\varphi}$, but whose support ranges between indices $p_{\max} - i, p_{\min} - i$. For instance, we have ${}^{\mathtt{tr}(1)}(0,1,-2,1,0,0,0) = (0,0,1,-2,1,0,0)$. In particular, if $\overrightarrow{\varphi}$ represents a k-deviation, then ${}^{\mathtt{tr}(i)}\overrightarrow{\varphi}$ represents the same k-deviation, up to a decrease by i of the size of all color classes involved.

One can extend the operator and its meaning to sequences of k-deviations as well. Formally, let $\overrightarrow{\varphi}^1, \ldots, \overrightarrow{\varphi}^t$ be a sequence of k-deviations, and let $\overrightarrow{\Phi} = \sum_{l=1}^{t} \overrightarrow{\varphi}^l$ represent this sequence. Then, if no group of size less than $i+1$ is modified nor created by the sequence (*i.e.*, $\forall l, \forall p \leq i, \varphi^l_p = 0$), we obtain by linearity of the operator that ${}^{\mathtt{tr}(i)}\overrightarrow{\Phi} = \sum_{l=1}^{t} {}^{\mathtt{tr}(i)}\overrightarrow{\varphi}^l$.

Let us prove two important properties of the so-called "shift operator":

Claim 71. *Let* $\overrightarrow{\phi}$ *be any vector that has a support of size* $s = p_{\max} - p_{\min} + 1$*, and with the symmetric property. For any positive integers* r *and* d *such that* $1+(r-1)d \leq p_{\min}$*, the vector* $\overrightarrow{\phi}' = \sum_{h=0}^{r-1} {}^{\mathtt{tr}(hd)}\overrightarrow{\phi}$ *also has the symmetric property.*

Proof. The support of vector $\overrightarrow{\phi}'$ has size $s' = (r-1)d + s$. In the following, we will assume up to padding the vector $\overrightarrow{\phi}$ with additional null entries that it is unbounded *i.e.*, it is indexed by $\mathbb{Z}$. By the hypothesis the vector $\overrightarrow{\phi}$ has the symmetric property and so, $\forall 1 \leq j \leq p_{\max} + p_{\min} - 1, \phi_j = \phi_{p_{\min}+p_{\max}-j}$. Let $0 \leq j \leq s'/2 - 1$. We have that:

$$\phi'_{p_{\max}-j} = \sum_{h=0}^{r-1} \phi_{p_{\max}-j+hd} = \sum_{h=0}^{r-1} \phi_{p_{\max}+p_{\min}-(p_{\max}-j+hd)}$$

$$= \sum_{h=0}^{r-1} \phi_{p_{\min}+j-(r-1-h)d} = \sum_{h=0}^{r-1} \phi_{p_{\min}-(r-1)d+j+hd} = \phi'_{p_{\min}-(r-1)d+j}.$$

Thus, $\overrightarrow{\phi}'$ also has the symmetric property. ◇

Claim 72. *For any positive integers r and d, if $\overrightarrow{\Phi}$ represents a h-balanced sequence then $\overrightarrow{\Phi}' = \sum_{j=0}^{r-1} \mathtt{tr}(jd)\overrightarrow{\Phi}$ represents a $(h + e_\Phi)$-balanced sequence, with $e_\Phi = -\min_{i_1 \leq i_2} \sum_{j=i_1}^{i_2} \Phi_j$.*

Proof. We prove the claim by induction on r. If $r = 0$ then $\overrightarrow{\Phi'} = \overrightarrow{\Phi}$ and so the claim holds vacuously in this base case. Otherwise, let us write $\overrightarrow{\Phi'} = \left(\sum_{j=0}^{r-2} \mathtt{tr}(jd)\overrightarrow{\Phi}\right) + \mathtt{tr}((r-1)d)\overrightarrow{\Phi} = \overrightarrow{\Phi''} + \mathtt{tr}((r-1)d)\overrightarrow{\Phi}$. Note that $\mathtt{tr}((r-1)d)\overrightarrow{\Phi}$ represents a h-balanced sequence, and by the induction hypothesis $\overrightarrow{\Phi''}$ represents a $(h + e_\Phi)$-balanced sequence. Since all entries of $\overrightarrow{\Phi''}$ are greater than or equal to $-e_\Phi$, therefore, $\overrightarrow{\Phi}$ represents a $(h + e_\Phi)$-balanced sequence by Claim 70. ◇

Finally, in order to prove Theorem 67, we construct vectors $\overrightarrow{\zeta}^i$ that represent sequences of deviations. The construction is recursive. To construct the vector $\overrightarrow{\zeta}^{i+1}$ from $\overrightarrow{\zeta}^i$, we follow a particular construction that we will show valid and that is illustrated in Figure 4.4. The construction is composed of a repetition of the sequence defined by $\overrightarrow{\zeta}^i$ a certain number of times (linear in some parameter $t = \Theta(\log n)$) shifting the "starting point" of each sequence by the same value. The construction then adds 1-deviations in order to get a technical generalization of the symmetric property, called *Good property* (see [DMC17]).

Claim 73. *There exists a sequence of vectors $\overrightarrow{\zeta}^i$ such that the following hold true for every i:*

- *There exist two positive integers denoted by a_i, t_1^i, and there exists a sequence of 1-deviations represented by $\overrightarrow{\xi}^{i+1}$ so that:*

$$\overrightarrow{\zeta}^{i+1} = \sum_{j=0}^{a_i} \mathtt{tr}(jt_1^i)\overrightarrow{\zeta}^i + \overrightarrow{\xi}^{i+1}.$$

- *If $\overrightarrow{\zeta}^i$ represents a h_i-balanced sequence then $\overrightarrow{\zeta}^{i+1}$ represents a (h_i+1)-balanced sequence.*
- *Furthermore, $s_i \leq s_{i+1} < \frac{3}{2}s_i$ where s_i and s_{i+1} denote the respective sizes of the support of $\overrightarrow{\zeta}^i$ and $\overrightarrow{\zeta}^{i+1}$, and $\overrightarrow{\zeta}^{i+1}$ represents a sequence of at least $(\frac{s_i}{2^{i+2}} - 5)$-times more deviations than in the sequence represented by $\overrightarrow{\zeta}^i$.*

Sketch Proof of Claim 73. Our constructions will ensure that every $\overrightarrow{\zeta}^i$ satisfies a so-called *Good Property*, namely:

- $\overrightarrow{\zeta}^i$ has the symmetric property, with its nonzero entries being equal to $1, -1, -1, 1$ and (by symmetry) $1, -1, -1, 1$;
- it has a support $\overrightarrow{supp(\zeta^i)}$ of even size s_i with its two middle entries being equal to 1;
- last, the two least entries of $\overrightarrow{supp(\zeta^i)}$ that are valued -1 are indexed by t_1^i, t_2^i with $1 < t_1^i < t_2^i < 2t_1^i$, and $t_2^i \leq 2^{i+1}$.

Before entering in the details of the construction, let us sketch how we use this Good property in what follows. Let h_i be the least integer such that the vector $\overrightarrow{\zeta}^i$ represents a h_i-balanced sequence. Our main objective is to maximize the size of this sequence while minimizing h_i.

- In particular, if $\overrightarrow{\zeta}^i$ satisfies the Good property then its nonzero entries are constrained to $1, -1, -1, 1, 1, -1, -1, 1$, and so, $\overrightarrow{\zeta}^i$ is h_i-balanced implies that $\sum_{j=0}^{a_i} \mathtt{tr}(jt_1^i)\overrightarrow{\zeta}^i$ is (h_i+2)-balanced by Claim 72. We will ensure in addition that $\overrightarrow{\xi}^{i+1}$ represents a 1-balanced sequence of 1-deviations, so, altogether combined this will show that $\overrightarrow{\zeta}^{i+1} = \sum_{j=0}^{a_i} \mathtt{tr}(jt_1^i)\overrightarrow{\zeta}^i + \overrightarrow{\xi}^{i+1}$ is (h_i+2)-balanced by Claim 70 (a little more work is needed in order to prove that $\overrightarrow{\zeta}^{i+1}$ is (h_i+1)-balanced).

- Moreover, we note that since $\overrightarrow{\zeta}^{i+1} = \sum_{j=0}^{a_i} \mathtt{tr}(jt_1^i)\overrightarrow{\zeta}^i + \overrightarrow{\xi}^{i+1}$, it represents a sequence of at least a_i-times more deviations than $\overrightarrow{\zeta}^i$. We will choose a_i (used for the shiftings) to be the largest even integer j such that $jt_1^i + t_2^i < s_i/2+1$. The latter choice implies that $a_i \leq \frac{s_i-4-2t_2^i}{2t_1^i}$. Then, since $1 < t_1^i < t_2^i < 2t_1^i$, and $t_2^i \leq 2^{i+1}$, we obtain that $\frac{s_i-4-2t_2^i}{2t_1^i} > \frac{s_i}{2^{i+2}} - \frac{1}{2^i} - 1$, and so, $a_i \geq \left\lfloor \frac{s_i}{2^{i+2}} - \frac{1}{2^i} - 1 \right\rfloor - 2 \geq \frac{s_i}{2^{i+2}} - 5$, as desired.

As a result, the Good property is a sufficient condition for the two requirements of the claim. Let us now sketch the construction of the vectors $\overrightarrow{\zeta}^i$.

Base case. Let $L = \Theta(\sqrt{n})$ and $t = \Theta(\log n)$. We initiate the sequence with a cascade of 4-deviations. This cascade has size t^2 and it starts with four agents in different color classes of size $L-1$ leaving for a new color class of size $L-4$ (until four agents in different classes of size $L-t^2$ leave for a new color class of size $L-3-t^2$). Then, in order to satisfy some technical requirements we complete the cascade with a small sequence of 1-deviations. Let $\overrightarrow{\Phi}^1$ represent this subsequence. By the calculation, all its entries are equal to zero except for: $\Phi^1_L = \Phi^1_{L-5} = \Phi^1_{L-t^2-1} = \Phi^1_{L-t^2-6} = 1$, and $\Phi^1_{L-1} = \Phi^1_{L-2} = \Phi^1_{L-t^2-4} = \Phi^1_{L-t^2-5} = -1$. Note that this intermediate sequence does not satisfy the Good property.

We finally construct $\overrightarrow{\zeta}^1$ by repeating $\overrightarrow{\Phi}^1$ many times, namely from $\sum_{i=0}^{t^2-5} \mathtt{tr}(i)\overrightarrow{\Phi}^1$, and then ending with "adjusting" sequences of 1-deviations.

To better understand the role played by the latter sequences, let p, q, h be three nonnegative integers such that $p > q + 2h$. Let us consider the sequence where an agent leaves a group of size $q+1$ for a group of size $p-1$, then another agent leaves a group of size $q+2$ for a group of size $p-2$, and so on util a final agent leaves a group of size $q+h$ for a group of size $p-h$. This sequence is represented by a vector $\overrightarrow{\phi}$ such that: $\phi_p = \phi_q = 1$, and $\phi_{p-h} = \phi_{q+h} = -1$. We use this type of sequence so as to position the nonzero entries of $\overrightarrow{\zeta}^1$ as desired in order to satisfy the Good property.

Inductive step. It turns out that all the main ideas for the construction are already present in the base case. Indeed, suppose the vector $\overrightarrow{\zeta}^i$ to be constructed in order to satisfy the Good property. As already stated, we choose a_i to be the

largest even integer j such that $jt_1^i + t_2^i < s_i/2+1$. Then, let $\overrightarrow{\Phi}^{i+1} = \sum_{j=0}^{a_i} \mathtt{tr}(jt_1^i)\overrightarrow{\zeta}^i$, that is a vector with the symmetric property by Claim 71. By construction, this vector has a support of size $s_{i+1} = s_i + a_i t_i^1 < \frac{3}{2}s_i$, that is even because s_i and a_i are even, and that will also be the size of the support of $\overrightarrow{\zeta}^{i+1}$.

In fact, $\overrightarrow{\Phi}^{i+1}$ "almost" satisfies the Good property, but is has more nonzero entries than what is required. So, we set to zero this surplus of nonzero entries using four sequences of 1-deviations, thereby obtaining $\overrightarrow{\zeta}^{i+1}$. □

It can be proved by induction on i that the above-defined sequence $\overrightarrow{\zeta}^i$ is $\mathcal{O}(i)$-balanced and with support of size $o\left(\left(\frac{3}{2}\right)^i\right)$. In particular, this sequence is valid if we start from a coloring with $\mathcal{O}(i)$ color classes of size s for every $1 \leq s \leq o\left(\left(\frac{3}{2}\right)^i\right)$ — in which case, we must ensure $n \geq \mathcal{O}\left(i \cdot \left(\frac{3}{2}\right)^i\right)$. Hence, we can construct the sequence $\overrightarrow{\zeta}^i$ for some polynomial $i = \Omega(n^{1/6}/\log n)$. Altogether combined with the lower-bound on the size of the sequence that is represented by $\overrightarrow{\zeta}^i$, the latter achieves proving Theorem 67. We refer to [DMC17] for the full calculation.

Discussion and open questions. To sum up this section, we have by Theorem 64 that better-response dynamics cannot be used in order to compute 4-stable partitions in polynomial time. As a byproduct of our vectorial approach, we also get an $\Omega(n^2)$ lower-bound on the convergence time of the dynamic for $k = 3$ (see [DMC17]). We conjecture that the worst-case convergence time of the dynamic in this case is indeed $\mathcal{O}(n^2)$, that would improve upon the known $\mathcal{O}(n^3)$ upper-bound.

Finally, it is open whether the problem of computing a 4-stable partition can be solved in polynomial time. In particular, is this problem complete for the complexity class PLS ?

4.4 The parallel complexity of coloring games

In the line of prior Section 4.3, we keep studying the complexity of computing stable partitions for unweighted games. However, the present section is focused on the complexity of computing Nash equilibria (1-stable partitions).

By Theorem 64, for every unweighted game, the better-response dynamic converges to a Nash equilibrium in polynomial time. However, we know by Theorem 67 that the same does not hold for k-stable partitions, with $k \geq 4$. Therefore, it might be desirable to have a better understanding of the complexity of computing Nash equilibria for these games.

4.4.1 Overall approach and main result

I shall investigate on the belonging of the problem – the computation of a Nash equilibrium in coloring games – to some complexity classes that are related to parallel and space complexity. The goal in doing so is to bring more insights on the complexity of the problem.

Complexity classes. In what follows, computations are performed on a parallel random-access machine (PRAM, see [GHR95]) with an unlimited amount of (numbered) processors. We will handle with read/write conflicts between processors with the strategy CREW-PRAM (concurrent read, exclusive write).

Let PTIME contain the decision problems that can be solved in *sequential* polynomial-time, that is with a single processor. Problem A reduces to problem B if given an oracle to solve B, then A can be solved in polylogarithmic-time with a polynomial number of processors. In particular, a problem B is PTIME-hard if every problem in PTIME reduces to B (this is formally defined as quasi-PTIME-hardness in [GHR95]). Such reductions are finer-grained than the more standard logspace reductions.

On the applicative point of view, we recall that coloring games have been proposed in order to design distributed algorithms on graphs, and to model the behaviour of social network users with limited memory and computing power. We note that any distributed algorithm on graphs can be simulated on a parallel machine with one processor per edge and per vertex. Furthermore, there are strong and well-known relationships between space and parallel complexity [Pap03]. Hence, the following result also brings more insights on the feasability of the proposed applications for coloring games in the literature.

The main result in this section can be stated as follows.

Theorem 74. *Computing a Nash equilibrium for coloring games is PTIME-hard.*

This theorem paves the way to a deepening of the complexity of computing Nash equilibria in graph games. I think that similar investigations should be pursued for other games where it can be computed a Nash equilibrium in polynomial time.

The reduction for proving Theorem 74 is from the standard Monotone Circuit Value problem. However, the gadgets needed are technically challenging, and we will need to leverage nontrivial properties of coloring games in order to prove its correctness. I detail this reduction in what follows.

4.4.2 The reduction

4.4.2.1 The Monotone Circuit Value problem

In order to prove Theorem 74, we will reduce from a variation of the well-known Monotone Circuit Value problem, defined as follows.

Problem 1 (Monotone Circuit Value).

Input: *A boolean circuit $\mathcal{C}$ with m gates and n entries, a word $w \in \{0,1\}^n$ such that:*

- *the gates are either AND-gates or OR-gates;*
- *every gate has exactly two entries (in-degree two);*
- *a topological ordering of the gates is given, with the m^{th} gate being the output gate.*

Question: *Does $\mathcal{C}$ output 1 when it takes w as input ?*

This variation of Monotone Circuit Value is proved to be PTIME-complete in [GHR95]. On the technical point of view, it requires a topological ordering of the gates as part of the input. This non standard add up will be shown to be a key element in the reduction to coloring games.

In what follows, let $\langle \mathcal{C}, w \rangle$ be any instance of Monotone Circuit Value. We will reduce it to a coloring game as follows. Let $\mathcal{G} := (g_1, g_2, \ldots, g_m)$ be the gates of the circuit, that are topologically ordered.

4.4.2.2 Construction of the gate-gadgets

For every $1 \leq j \leq m$, the j^{th} gate will be simulated by a subgraph $G_j = (V_j, E_j)$ with $12(n+j) - 9$ vertices. We refer to Figure 4.5 for an illustration.

Let us give some intuition for the following construction of G_j. We aim at simulating the computation of the (binary) output of all the gates in $\mathcal{C}$ when it takes w as input. To do that, we will construct a supergraph G^- of G_j (to be defined later), then we will consider the unweighted game that is played on G^-. The goal of the construction will be to ensure that given a fixed Nash equilibrium for this game, we can guess the output of the j^{th} gate from the subcoloring of G_j. More precisely, the subcoloring will encode a "local certificate" that indicates which values on the two entries of g_j cause the output.

Observe that to certify that an OR-gate outputs 1, it suffices to show that it receives 1 on any of its two entries, whereas for an AND-gate it requires to show that it receives 1 on its two entries. Since by de Morgan's laws [DM47], the negation of an AND-gate can be transformed into an OR-gate and vice-versa, therefore, we need to distinguish between three cases in order to certify the output of the gate. So, the vertices in V_j are partitioned in three subsets of equal size $4(n+j) - 3$, denoted by V_j^1, V_j^2, V_j^3. Furthermore, for every $1 \leq t \leq 3$, every vertex in V_j^t is adjacent to every vertex in $V_j \setminus V_j^t$.

Let us now describe the structure of the three (isomorphic) subgraphs $G_j[V_j^t] = (V_j^t, E_j^t)$ with $1 \leq t \leq 3$. Informally, we will need this internal structure in order to ensure that every of the three subsets V_j^t will behave as a "truthful" certificate to decide on the output of the gate; *i.e.*, only a few vertices of V_j will be used to

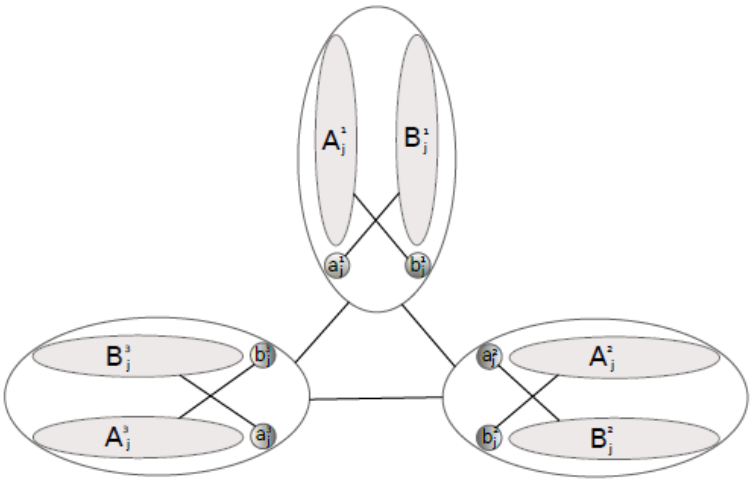


Figure 4.5: Gadget subgraph G_j representing the j^{th} gate. An edge between two subsets of vertices (delimited by an ellipse) denotes the existence of a complete bipartite subgraph.

certify the output of the j^{th} gate, while all others will be divided into artificial aggregates that we name "private groups" whose role is to ensure "truthfulness" of the certificate (this will be made clearer in the following). There are two nonadjacent vertices $a_j^t, b_j^t \in V_j^t$ playing a special role. The other vertices in $V_j^t \setminus \{a_j^t, b_j^t\}$ are partitioned in two subsets A_j^t, B_j^t of respective size $2(n+j)-3$ and $2(n+j)-2$. The sets A_j^t, B_j^t are called the *private groups* of a_j^t, b_j^t. Furthermore, every vertex in A_j^t is adjacent to every vertex in $V_j^t \setminus (A_j^t \cup \{a_j^t\})$, similarly every vertex in B_j^t is adjacent to every vertex in $V_j^t \setminus (B_j^t \cup \{b_j^t\})$.

Computation. Since all edges are defined above independently the one from the other, the graph $G_j[V_j^1] = (V_j^1, E_j^1)$ (encoded by its adjacency lists) can be constructed with $|V_j^1| + |E_j^1| = 4(n+j)^2 - 2(n+j) - 2$ processors simply by assigning the construction of each vertex and each edge to a different processor. Note that each processor can decide on the vertex, resp. the edge, it needs to compute from its number. Overall, it takes $\mathcal{O}(\log(n+j))$-time in order to construct $G_j[V_j^1]$ in parallel. The latter can be easily generalized in order to construct G_j in $\mathcal{O}(\log(n+j))$-time with $|V_j|+|E_j|$ processors. Therefore, the graphs $G_1, G_2, \ldots, G_m$ can be constructed in parallel in $\mathcal{O}(\log(n+m))$-time with $\sum_{j=1}^{m}(|V_j| + |E_j|)$ processors, that is (huge!) polynomial in $n+m$.

4.4.2.3 Construction of the graph

Let $X = \{x_1, x'_1, \ldots, x_i, x'_i, \ldots, x_n, x'_n\}$ contain $2n$ nonadjacent vertices, that are two vertices per letter in the binary word w. The (conflict) graph $G^- = (V, E)$ for the reduction has vertex-set $V = X \cup \left(\bigcup_{j=1}^{m} V_j\right)$. In particular, it has $2n - 9m + 6m(m+2n+1)$ vertices. Furthermore, $G^-[V_j]$ is isomorphic to G_j for every

$1 \leq j \leq m$. In order to complete our reduction, let us now describe how our gadgets are connected the one with the other.

For technical reasons, we will need to make adjacent every vertex in the private group A_j^t (resp. B_j^t), with $1 \leq j \leq m$ and $1 \leq t \leq 3$, to every vertex in $V \setminus V_j$. By doing so, note that every vertex in $V \setminus (A_j^t \cup \{a_j^t\})$ is adjacent to every vertex in A_j^t (resp., every vertex in $V \setminus (B_j^t \cup \{b_j^t\})$ is adjacent to every vertex in B_j^t). Furthermore, each edge is defined independently the one from the other. Hence, similarly as above, $\sum_{j=1}^{m} \sum_{t=1}^{3} (|A_j^t| + |B_j^t|)|V \setminus V_j|$ processors are sufficient in order to construct these edges in $\mathcal{O}(\log(n+m))$-time, that is polynomial in $n+m$.

Finally, we recall that for every j, there are three cases to distinguish in order to decide on the output of the j^{th} gate, with each case being represented with some subset V_j^t. The union of subsets representing a *positive* certificate (output 1) is named Y_j, while the union of those representing a *negative* certificate (output 0) is named N_j. In particular, if the j^{th} gate is an OR-gate, let $Y_j := \{a_j^1, b_j^1, a_j^2, b_j^2\}$ and $N_j := \{a_j^3, b_j^3\}$ (it suffices to receive 1 on one input). Else, the j^{th} gate is an AND-gate, so, let $Y_j := \{a_j^1, b_j^1\}$ and $N_j := \{a_j^2, b_j^2, a_j^3, b_j^3\}$.

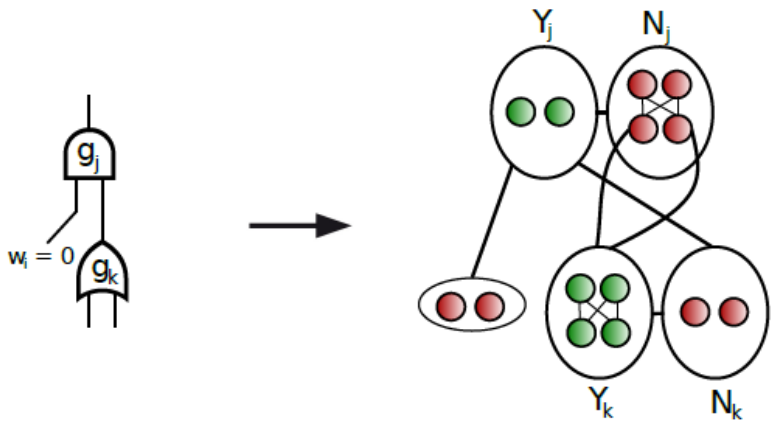


Figure 4.6: Edges in G^- to simulate the two connections of an AND-gate in the circuit.

Suppose the j^{th} gate is an OR-gate (the case where it is an AND-gate follows by symmetry, up to interverting Y_j with N_j, see also Figure 4.6). Let us consider the first entry of the gate. There are two cases.

- Suppose that it is the i^{th} entry of the circuit, for some $1 \leq i \leq n$.
  If $w_i = 0$ then we make both x_i, x_i' adjacent to both a_j^1, b_j^1.
  Else, $w_i = 1$, we make both x_i, x_i' adjacent to both a_j^3, b_j^3.
- Otherwise, the entry is some other gate of the circuit, and so, since gates are topologically ordered, it is the k^{th} gate for some $k < j$. We make every vertex in N_k adjacent to both a_j^1, b_j^1, and we make every vertex in Y_k adjacent to both a_j^3, b_j^3.

The second entry of the gate is similarly considered, up to replacing above the two vertices a_j^1, b_j^1 with a_j^2, b_j^2. We refer to Figure 4.6 for an illustration.

Let us point out that the graph G^-, obtained with our reduction, is *undirected*, whereas the original circuit $\mathcal{C}$ is a DAG (directed acyclic graph). However, since the sizes of private groups are proportional to the positions of the gates in the topological ordering of the circuit, this orientation of the edges can be easily retrieved, from the certificates with smaller privates groups to those with larger ones. Therefore, we do not lose any information.

Computation. Observe that there is only a constant number of edges that are added at this step for each gate. Furthermore, the construction of these new edges only requires to read the two in-neighbours of the gate in the circuit $\mathcal{C}$. As a result, the last step can be done in parallel in $\mathcal{O}(\log(n+m))$-time with m processors.

4.4.3 Proof of the main result

4.4.3.1 Structure of a Nash equilibrium

The (conflict) graph $G^- = (V, E)$ of the reduction defines an unweighted coloring game. Let us fix any Nash equilibrium for this game (that exists by Theorem 64). We will show that it is sufficient to know the color of every vertex in $Y_m \cup N_m$ in order to decide on the output of the circuit $\mathcal{C}$ (recall that the m^{th} gate is the output gate). To prove it, we will need the following technical claims in order to gain more insights on the structure of the equilibrium.

More precisely, we will prove that there are exactly $6m+1$ color classes, that are one color class per private group A_j^t or B_j^t and one additional color for the vertices in X. The intuition is that there are $2(n+m)$ vertices in one special color class (including X) that simulates the computation of the output of $\mathcal{C}$, whereas all other vertices are "trapped" with the vertices in their respective private group. We refer to Figure 4.7 for an illustration.

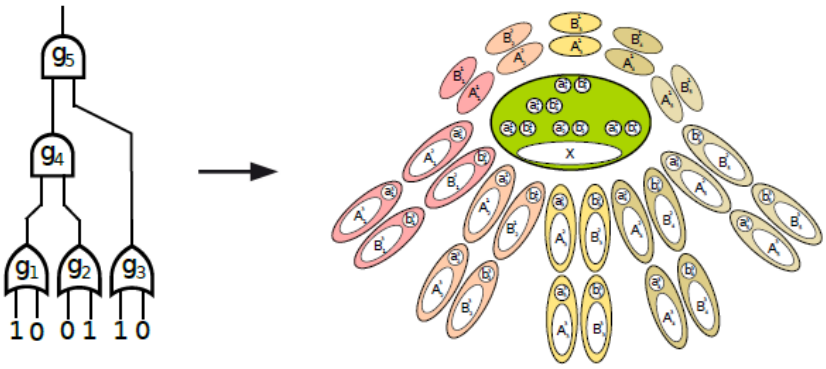


Figure 4.7: A boolean circuit (left) with a Nash equilibrium of the coloring game from our reduction (right). For ease of reading, edges of the graph are not depicted. Each color class is represented with an ellipse. Intuitively, vertices in the central color class simulate the computation of the output. Other color classes contain a private group and they are "inactive".

Full proofs of the claims are delayed to my publication [Duc16]. In what follows, we will denote by $L_c \subseteq V$ the subset of agents colored by c.

Claim 1. *For every j, any color class does not contain more than two vertices in every $Y_j \cup N_j$. Furthermore, if it contains exactly two vertices in $Y_j \cup N_j$ then these are a_j^t, b_j^t for some $1 \leq t \leq 3$.*

Proof. A Nash equilibrium is a proper coloring of G^-. Therefore, since any two vertices in different subsets among V_j^1, V_j^2, V_j^3 are adjacent by construction, they cannot have the same color. Since $Y_j \cup N_j = \{a_j^1, b_j^1, a_j^2, b_j^2, a_j^3, b_j^3\}$ and $a_j^t, b_j^t \in V_j^t$ for every $1 \leq t \leq 3$, the claim follows directly. ◇

Claim 2. *Any two vertices that are in a same private group have the same color. Similarly, x_i and x_i' have the same color for every $1 \leq i \leq n$.*

Proof. Let S be either a private group ($S = A_j^t$ or $S = B_j^t$ for some $1 \leq j \leq m$ and $1 \leq t \leq 3$), or a pair representing the same letter of word w (*i.e.*, $S = \{x_i, x_i'\}$ for some $1 \leq i \leq n$). Let $v \in S$ maximize her payoff and let c be her color. Note that v receives payoff $|L_c| - 1$ with L_c being the color class composed of all the vertices with color c. Furthermore, every $u \in S$ receives payoff lower than or equal to $|L_c| - 1$ by the choice of v. In such case, every $u \in S$ must be colored c, or else, since the adjacency and the nonadjacency relations are the same for u and v (they are twins), furthermore u, v are nonadjacent, the agent u would increase her payoff to $|L_c|$ by choosing c as her new color, thus contradicting the hypothesis that we are in a Nash equilibrium. ◇

The argument we use in Claim 2 is that twin vertices, *i.e.*, nonadjacent agents with the same neighbourhood, must have the same color. In order to prove the following Claim 3, we had to use the same argument under different disguises.

More precisely, consider a union $U \subseteq V$ of color classes. Then, $G^- \setminus U$ defines a coloring subgame, and the constriction of the coloring to the subgraph must be a Nash equilibrium for this subgame. it follows that twin vertices in $G^- \setminus U$ must have the same color, that was the key observation for proving Claim 3.

Claim 3. *Let $1 \leq j \leq m$ and $1 \leq t \leq 3$. Either A_j^t or $A_j^t \cup \{a_j^t\}$ is a color class, and in the same way either B_j^t or $B_j^t \cup \{b_j^t\}$ is a color class. Furthermore, either $B_j^t \cup \{b_j^t\}$ is a color class, or a_j^t and b_j^t have the same color.*

We recall that we aim at simulating the computation of the output of all the gates in $\mathcal{C}$. To do that, we will prove the existence of a special color class containing X and some pair of vertices in $Y_j \cup N_j$ for every j (Claim 5). Intuitively, the two vertices of $Y_j \cup N_j$ are used to certify the output of the j^{th} gate. However, this certificate is "local" in the sense that it assumes the output of the $j-1$ smaller gates to be already certified. Therefore, we need to prove that there can be no "missing gate", *i.e.*, every gate is represented in the special color class. This is where the topological ordering over the gates comes into play. In what follows, we recall that L_c denotes the subset of agents colored by c.

Claim 4. *Let c be a color such that $L_c \not\subseteq X$ and L_c does not intersect any private group (A^t_j or B^t_j for any $1 \leq j \leq m$ and $1 \leq t \leq 3$).*

Then, $X \subseteq L_c$ and there exists an index j_0 such that the following holds true: $|L_c \cap (Y_j \cup N_j)| = 2$ for every $1 \leq j \leq j_0$, and $L_c \cap (Y_j \cup N_j) = \emptyset$ for every $j_0 + 1 \leq j \leq m$.

Proof. By the hypothesis $L_c \not\subseteq X$ and L_c does not intersect any private group, so, there is at least one vertex of $\bigcup_{j=1}^{m}(Y_j \cup N_j)$ with color c. Let j_0 be the largest index j such that there is a vertex in $Y_j \cup N_j$ with color c. Since by Claim 1, there can be no more than two vertices of $Y_j \cup N_j$ that are in L_c for every j, therefore, by maximality of j_0 we get $|L_c| \leq |X| + 2j_0 = 2(n + j_0)$. In particular, observe that if $|L_c| = 2(n+j_0)$ then $X \subseteq L_c$ and for every $1 \leq j \leq j_0$ there are exactly two vertices in $Y_j \cup N_j$ with color c. So, let us prove that $|L_c| = 2(n + j_0)$, that will prove the claim.

By the choice of j_0, there is some $1 \leq t \leq 3$ such that $a^t_{j_0} \in L_c$ or $b^t_{j_0} \in L_c$. In particular, $|L_c| \geq \min\{|A^t_{j_0}|, |B^t_{j_0}|\} + 1 = 2(n + j_0) - 2$ or else, every vertex $v^t_{j_0} \in L_c \cap \{a^t_{j_0}, b^t_{j_0}\}$ would increase her payoff by choosing the color of the vertices in her private group (that is a color class by Claim 3), thus contradicting the hypothesis that we are in a Nash equilibrium.

We prove as an intermediate subclaim that for any $1 \leq j \leq j_0 - 1$ such that $L_c \cap (Y_j \cup N_j) \neq \emptyset$, there is some $1 \leq t' \leq 3$ such that $a^{t'}_j, b^{t'}_j \in L_c$. Indeed, in this situation, there is some t' such that $a^{t'}_j \in L_c$ or $b^{t'}_j \in L_c$. If $b^{t'}_j \in L_c$ then we are done as by Claim 3, $a^{t'}_j \in L_c$. Otherwise, $b^{t'}_j \notin L_c$ and we prove this case cannot happen. First observe that $a^{t'}_j \in L_c$ in this case. Furthermore, since $a^{t'}_j$ and $b^{t'}_j$ do not have the same color we have by Claim 3 that $B^{t'}_j \cup \{b^{t'}_j\}$ is a color class. In this situation, $b^{t'}_j$ receives payoff $2(n + j) - 2 \leq 2(n + j_0 - 1) - 2 < |L_c|$. Since in addition $a^{t'}_j$ and $b^{t'}_j$ are twins in $G \setminus (A^{t'}_j \cup B^{t'}_j)$, vertex $b^{t'}_j$ could increase her payoff by choosing color c, thus contradicting that we are in a Nash equilibrium. This proves $a^{t'}_j, b^{t'}_j \in L_c$, and so, the subclaim.

By the subclaim, there is an even number $2k$ of vertices in $\bigcup_{j=1}^{j_0-1}(Y_j \cup N_j)$ with color c, for some $k \leq j_0 - 1$. Similarly, since by Claim 2 the vertices x_i, x'_i have the same color for every $1 \leq i \leq n$, $|X \cap L_c| = 2n'$ for some $n' \leq n$. Now there are two cases to be considered.

- Suppose that $b^t_{j_0} \in L_c$. Then, by Claim 3 $a^t_{j_0} \in L_c$. Furthermore $|L_c| \geq 2(n + j_0) - 1$ or else, vertex $b^t_{j_0}$ would increase her payoff by choosing the color of the vertices in $B^t_{j_0}$ (that is a color class by Claim 3), thus contradicting the hypothesis that we are in a Nash equilibrium. As a result, $|L_c| = 2(n' + k + 1) \geq 2(n + j_0) - 1$, that implies $n' + k \geq n + j_0 - 1$, and so, $|L_c| \geq 2(n + j_0)$, as desired.
- Else, $b^t_{j_0} \notin L_c$ and we prove this case cannot happen. First observe that $a^t_{j_0} \in L_c$. Furthermore, $|L_c| = 2(n' + k) + 1 \geq 2(n + j_0) - 2$, that implies $n' + k \geq n + j_0 - 1$, and so, $|L_c| \geq 2(n + j_0) - 1$. However, since $a^t_{j_0}$ and $b^t_{j_0}$ do not have the same color, $B^t_{j_0} \cup \{b^t_{j_0}\}$ is a color class by Claim 3. In particular, $b^t_{j_0}$ receives payoff

$2(n+j_0)-2 < |L_c|$. Since $a^t_{j_0}, b^t_{j_0}$ are twins in $G \setminus (A^t_{j_0} \cup B^t_{j_0})$, vertex $b^t_{j_0}$ could increase her payoff by choosing color c, thus contradicting that we are in a Nash equilibrium.

Altogether, $|L_c| \geq 2(n+j_0)$, that proves the claim. ◇

We point out that by combining Claim 1 with Claim 4, one obtains that for every $1 \leq j \leq m$, there are either zero or two vertices in $Y_j \cup N_j$ in each color class not containing a private group, and in case there are two vertices then these are a^t_j, b^t_j for some $1 \leq t \leq 3$. We elaborate on this property in order to prove the following Claim 5.

Claim 5. *Any two vertices in X have the same color. Furthermore, for every $1 \leq j \leq m$, every vertex in $Y_j \cup N_j$ either has the same color as vertices in X or as vertices in her private group.*

Finally, we will need a "truthfulness" property to prove correctness of our reduction. Namely, the value of the output of any gate in the circuit must be correctly guessed from the agents with the same color as vertices in X. We prove this, as for Claim 1, by elaborating on the property that every Nash equilibrium is a proper coloring of G^-. In this situation, the edges added at the last step of the reduction ensure that the agents of two "uncompatible certificates" cannot be assigned the same color.

Claim 6. *Let $1 \leq j_0 \leq m$ such that for every $1 \leq j \leq j_0$, there is at least one vertex in $Y_j \cup N_j$ with the same color c_0 as all vertices in X. Then for every $1 \leq j \leq j_0$, $L_{c_0} \cap Y_j \neq \emptyset$ if and only if the output of the j^{th} gate is 1.*

4.4.3.2 Proof of Theorem 74

Proof of Theorem 74. Let $\langle \mathcal{C}, w \rangle$ be any instance of Monotone Circuit Value. Let $G^- = (V, E)$ be the conflict graph obtained with our reduction. It can be constructed in polylogarithmic-time with a polynomial number of processors. The graph G^- defines an unweighted coloring game. We fix any Nash equilibrium for this game, that exists by Theorem 64. By Claim 5, any two vertices in X have the same color c_0. We will prove that there is at least one vertex in Y_m with color c_0 if and only if the circuit $\mathcal{C}$ outputs 1 when it takes w as input. Since Monotone Circuit Value is PTIME-complete [GHR95], the latter will prove that computing a Nash equilibrium for coloring games is PTIME-hard.

By Claim 6, we only need to prove that for every $1 \leq j \leq m$, there is at least one vertex in $Y_j \cup N_j$ with color c_0. To prove it by contradiction, let j_0 be the smallest index j such that no vertex in $Y_j \cup N_j$ has color c_0. By Claim 5, every vertex in $Y_{j_0} \cup N_{j_0}$ has the same color as her private group. In particular, the three of $a^1_{j_0}, a^2_{j_0}, a^3_{j_0}$ receive payoff $2(n+j_0)-3$. We will prove that one of these three agents could increase her payoff by choosing c_0 as her new color, thus contradicting that we are in a Nash equilibrium.

Indeed, by the minimality of j_0, it follows by Claim 4 that for any $1 \leq j \leq j_0 - 1$, there are exactly two vertices of $Y_j \cup N_j$ with color c_0, while for every $j_0 \leq j \leq m$ there is no vertex in $Y_j \cup N_j$ with color c_0. As a result, $|L_{c_0}| = 2(n + j_0) - 2$. In particular, any agent among $a^1_{j_0}, a^2_{j_0}, a^3_{j_0}$ could increase her payoff by choosing c_0 as her new color — provided she is nonadjacent to every vertex in L_{c_0}. We will show it is the case for at least one of the three vertices, that will conclude the proof of the theorem. Assume w.l.o.g. that the j_0^{th} gate is an OR-gate (indeed, since by de Morgan's laws, the negation of an AND-gate can be transformed into an OR-gate and vice-versa, both cases are symmetrical). There are two cases.

- Suppose that the output of the j_0^{th} gate is 1. In such case, there must be an entry of the gate such that: it is the i^{th} entry of the circuit, for some $1 \leq i \leq n$, and $w_i = 1$; or it is the k^{th} gate of the circuit for some $k < j_0$ and the output of that gate is 1. In the latter case, we have by Claim 6 that the two vertices of $Y_k \cup N_k$ with color c_0 are in the set Y_k.
  Assume w.l.o.g. that the above-mentioned entry is the first entry of the gate. By construction, the two vertices $a^1_{j_0}, b^1_{j_0}$ are nonadjacent to every vertex in L_{c_0}.
- Else, the output of the j_0^{th} gate is 0. Therefore, for every entry of the gate: either it is the i^{th} entry of the circuit, for some $1 \leq i \leq n$, and $w_i = 0$; or it is the k^{th} gate of the circuit for some $k < j_0$ and the output of that gate is 0. In the latter case, we have by Claim 6 that the two vertices of $Y_k \cup N_k$ with color c_0 are in the set N_k. By construction, the two vertices $a^3_{j_0}, b^3_{j_0}$ are nonadjacent to every vertex in L_{c_0}.

  In both cases, it contradicts our assumption that we are in a Nash equilibrium. □

Conclusion. Theorem 74 proves that computing a Nash equilibrium for coloring games is PTIME-hard. This may be hint that these games are a too powerful computational mechanism design for "lightweight" distributed applications. In this respect, an interesting open problem would be to determine the classes of conflict graphs for which this hardness result holds.

Furthermore, we recall that computing a Nash equilibrium for generalized coloring games is PLS-complete [BZ03]. Hence, this result reinforces the view that for many PLS-hard "weighted" games, the corresponding "unweighted" game is PTIME-hard [Sch91].

4.5 Weighted games: existence of equilibria

Next, we go back to generalized (weighted) coloring games in this section. Every generalized coloring game admits a Nash equilibrium [BZ03]. So, we are more interested in the existence of k-stable partition, for $k \geq 2$. However, as shown with Figure 4.3, not all generalized coloring games admit a 2-stable partition. Since in contrast, unweighted games admit a k-stable partition for every fixed k, it looks

natural to investigate on the impact of a *fixed* set of edge-weights $\mathcal{W}$ on the existence of stable partitions.

We are particularly interested in the special case where all edge-weights of the underlying graph G are comprised in $\mathcal{W} = \{-\infty, 0, 1\}$. Roughly, in this modest extension of the unweighted games, we now allow *indifference* relationships between some agents. More formally, the goal of the agents is now to construct a proper coloring of the conflict graph G^- while maximizing their number of neighbours in the friendship graph G^+ with the same color as theirs. Perhaps surprisingly, we shall prove that even in this slight extension, the existence of stable partitions is much more constrained than it is for the unweighted games.

This is joint work with Dorian Mazauric and Augustin Chaintreau.

4.5.1 Positive results

On the one hand, we relate some structural properties of the underlying graph G with the existence of stable partitions. In particular, we relate the existence of stable partitions with the *girth* (size of a smallest cycle) in the friendship graph:

Theorem 75. *Let $G = (V, w)$ have all its edge-weights in $\{-\infty, 0, 1\} \cup -\mathbb{N}$. If the friendship graph G^+ has girth at least $k + 1$ then the generalized coloring game that is played on G admits a k-stable partition.*

Furthermore, the better-response dynamic applied to this above game converges to a k-stable partition within a quadratic number of steps.

Theorem 75 follows from a potential function argument. More precisely, let us define the global utility of a given coloring as the sum of the individual payoff of every agent. See Figure 4.8 for an example. We prove that it is a potential function which increases after any k-deviation.

In order to see the difficulty, we emphasize that even for unweighted games, this above potential function might decrease after a k-deviation (*e.g.*, see the example of 4-deviation that is given in Section 4.3.3 and the related illustration of Figure 4.8). In fact, if j denotes the color assigned to all the agents in the k-deviation then the global utility increases only if all the agents deviating increase their respective payoff in large part due to the agents already colored j. This may not be the case if there are many agents of this k-subset that are pairwise connected by an edge with positive weight. However, if we now assume that the friendship graph G^+ has a large girth then we can upper-bound the number of edges with positive weights among any small subset of agents (because such small subsets must induce a forest in G^+), thereby preventing that case from happening.

In particular, since any friendship graph has girth at least three, we obtain the following corollary:

Corollary 76. *Let $G = (V, w)$ have all its edge-weights in $\{-\infty, 0, 1\} \cup -\mathbb{N}$. Then, the generalized coloring game that is played on G admits a 2-stable partition.*

Furthermore, the better-response dynamic applied to this above game converges to a 2-stable partition within a quadratic number of steps.

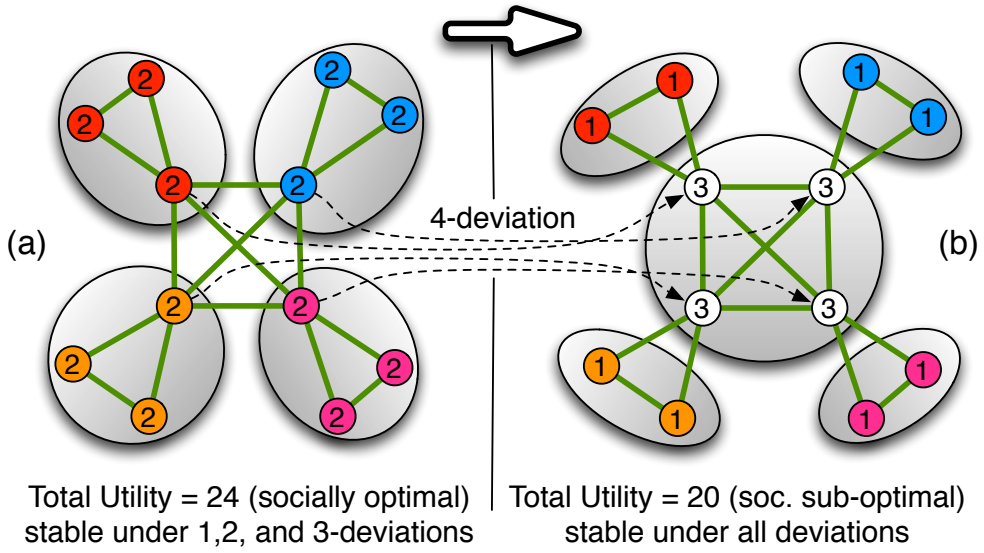


Figure 4.8: Change of configuration for an unweighted game after a 4-deviation. For ease of readability, only the edges of the friendship graph are represented. Agents are labeled with their payoff.

Perspectives. It is open whether similar results can be obtained for a larger family of sets $\mathcal{W}$. In particular, can it be obtained similar results for some $\mathcal{W}$ with two distinct positive weights ?

4.5.2 The hardness of recognizing games with k-stable partitions

We finally present a more complex construction of weighted games with no k-stable partition for some small value of k. Furthermore, we will explain how the mere existence of a single counter example impacts on the complexity of the recognition of games with k-stable partitions.

On the one hand, as shown with Figure 4.3, there are generalized coloring games that do not admit a 2-stable partition. On the other hand, we proved with Corollary 76 that by constraining the set $\mathcal{W}$ of admissible edge-weights, one obtains a large class of weighted games that admit a 2-stable partition. Surprisingly, this latter result cannot be improved already for $\mathcal{W} = \{-\infty, 0, 1\}$. Precisely, we give in Figure 4.9 an example of a graph G with weights in $\mathcal{W} = \{-\infty, 0, 1\}$ so that the coloring game that is played on G does not admit a 3-stable partition!

The construction in Figure 4.9 borrows from the one of Figure 4.3 (*i.e.*, the nonexistence of 2-stable partitions in generalized coloring games). Roughly, we impose the friendship graph and the conflict graph to be highly symmetric, that ensures that 2-stable partitions for the game are isomorphic. Then, we show that the isomorphism between two distinct 2-stable partitions translates to a 3-deviation from one to the other.

Proposition 77. *There is a graph $G = (V, w)$ whose edge-weights are constrained to $\mathcal{W} = \{-\infty, 0, 1\}$ and such that there does not exist a 3-stable partition for the coloring game defined on G.*

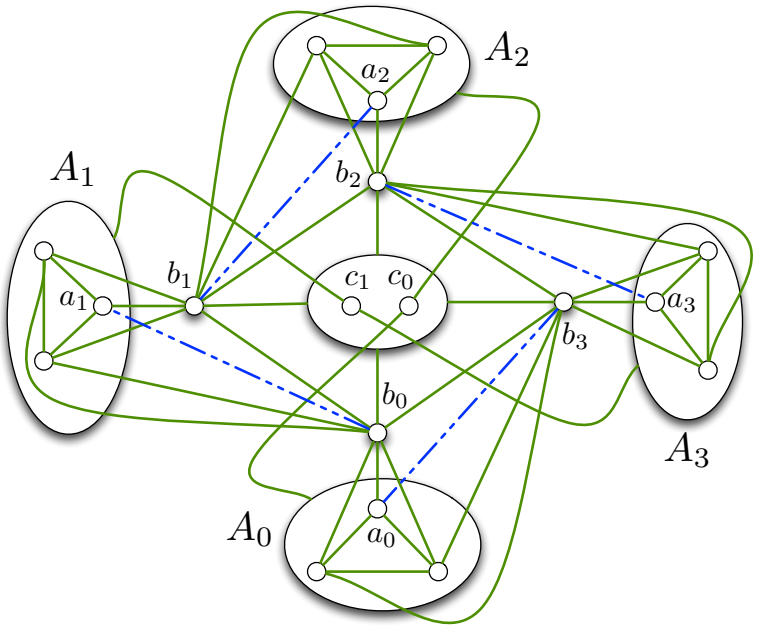


Figure 4.9: A graph $G = (V, w)$ with edge-weights in $\mathcal{W} = \{-\infty, 0, 1\}$. The coloring game played on G does not admit a 3-stable partition. To keep the graph readable, we use conventions. (1) Some sets of nodes are grouped within a circle; an edge from another node to that circle denotes an edge to *all* elements of this set. (2) Edges of the conflict graph are not represented. In particular, all nodes that are not connected by an edge on the figure are connected by an edge with negative weight $-\infty$. (3) Green solid edges represent edges with weight 1, whereas blue dashed edges represent edges with weight 0.

Proof. The set of vertices consists of four sets A_i, $0 \leq i \leq 3$, each of equal size $h \geq 2$ and with a special vertex a_i, plus four vertices b_i, $0 \leq i \leq 3$, and two vertices c_0 and c_1. In what follows, indices are taken modulo 2 for c_j, $j \in \{0, 1\}$, and they are taken modulo 4 everywhere else. Figure 4.9 represents the example with $h = 3$. The friendship graph G^+ here consists of all the edges with weight 1; it contains:

1. all the edges between nodes in A_i $(0 \leq i \leq 3)$;
2. edges between b_i and A_i $(0 \leq i \leq 3)$;
3. edges between b_i and $A_{i+1} \setminus \{a_{i+1}\}$ $(0 \leq i \leq 3)$;
4. edges between b_i and b_{i-1} and b_{i+1} $(0 \leq i \leq 3)$;
5. edges between c_0 and all the b_i, and edges between c_1 and all the b_i;
6. edges between c_0 and $A_0 \cup A_2$, and edges between c_1 and $A_1 \cup A_3$.

Moreover, there are four edges with weight 0, namely the edges $\{b_i, a_{i+1}\}$. All the other pairs of agents represent "*enemies*" (they are pairwise connected by an edge with negative weight $-\infty$). That is two nodes in different $A_i, A_{i'}$ are enemies; a user b_i is enemy of b_{i+2} and of the nodes in A_{i+2} and A_{i+3}; c_0 and c_1 are enemies; c_0 is enemy of the nodes in A_1 and A_3, and c_1 is enemy of the nodes in A_0 and A_2. We now assume by contradiction there exists a 3-stable partition for the coloring game defined on $G = (V, w)$.

Full proofs for the following claims are postponed to our paper [DMC17].

Claim 78. *Every agent in A_i picks the same color.*

Our key instrument for proving this claim is a generalization of *false twins* to weighted graphs. We recall that given an *unweighted game* that is played on the conflict graph G^-, for any Nash equilibrium for this game, false twins in G^- must have the same color (see Claim 2).

Now, given an edge-weighted graph $G = (V, w)$, we say that u and u' are *quasi-twins* if $w_{uu'} > 0$ and for all nodes $v \in V \setminus \{u, u'\}$, $w_{uv} = w_{u'v}$ except maybe for one v_0 for which $|w_{uv_0} - w_{u'v_0}| = 1$. We can observe that for unweighted games, quasi-twins are exactly the false twins in the conflict graph. In [DMC17], we prove that for any Nash equilibrium of the coloring game that is played on G, quasi-twins must have the same color. Since in the above construction for Proposition 77, the agents in A_i are pairwise quasi-twins, Claim 78 follows from this result directly.

Claim 79. *b_i picks the same color as the agents in A_i or the agents in A_{i+1}.*

Claim 80. *There is an i such that agents in A_i, b_i and b_{i-1} pick the same color.*

It follows by Claim 80 that there is an i such that the agents in A_i, b_i, b_{i-1}, c_i all pick the same color. Moreover, such a color class is unique in the 3-stable partition due to the conflict graph in G (induced by the conflict edges). In what follows, let L_{i_0} be the color class of a_0 in the 3-stable partition. By symmetry, we will assume $L_{i_0} = \{b_0, b_3, c_0\} \cup A_0$.

Case 1: the agents a_2, b_1, b_2 all have the same color. In particular, by Claims 78 and 79 their color class is $A_2 \cup \{b_1, b_2\}$.

Then, there are two subcases. Suppose that a_1 and c_1 have the same color, in which case their color class is $A_1 \cup \{c_1\}$. In this situation, the agent b_1 would increase her payoff from $1 + (|A_2| - 1) = |A_2| = h$ to $1 + |A_1| = h + 1$ by choosing the same color as a_1 and c_1. So, there is a 1-deviation. Otherwise, a_1 and c_1 do not have the same color, so, their respective color classes are A_1 and either $\{c_1\}$ or $A_3 \cup \{c_1\}$. Then, the agents b_1 and c_1 would increase their respective payoff from $1 + (|A_2| - 1) = |A_2| = h$ and $\leq |A_3| = h$ to $1 + |A_1| = h + 1$ by choosing the same color as a_1. So, there is a 2-deviation.

Case 2: both agents a_2 and b_2 have the same color, but b_1 has a different color. In particular, by Claims 78 and 79 their respective color classes are $A_2 \cup \{b_2\}$ and either $A_1 \cup \{b_1\}$ or $A_1 \cup \{b_1, c_1\}$.

Then, there are two subcases. Suppose that the agents a_3 and c_1 have the same color, in which case their color class is $A_3 \cup \{c_1\}$. Then, both b_2 and b_3 would increase their respective payoff from $|A_2| = h$ and $2 + (|A_0| - 1) = 1 + |A_0| = h + 1$ to $2 + (|A_3| - 1) = |A_3| + 1 = h + 1$ and $2 + |A_3| = h + 2$ by choosing the color of a_3. Otherwise, a_3 and c_1 do not have the same color, in which case the color class of c_1 is either $\{c_1\}$ or $A_1 \cup \{b_1, c_1\}$. But then the three of b_2, b_3, c_1 would increase their respective payoff from $|A_2| = h$, $2 + (|A_0| - 1) = 1 + |A_0| = h + 1$, and $\leq 1 + |A_1| = h + 1$ to $2 + (|A_3| - 1) = 1 + |A_3| = h + 1$, $2 + |A_3| = h + 2$, and $2 + |A_3| = h + 2$ by choosing the same color as a_3.

Case 3: both agents a_2 and b_1 have the same color, but b_2 has a different color, in which case their respective color classes are $A_2 \cup \{b_1\}$ and either $A_3 \cup \{b_2\}$ or $A_3 \cup \{b_2, c_1\}$ by Claim 79. In that case, b_1 would increase her payoff from $|A_2| - 1 = h - 1$ to $|A_1| = h$ by choosing the color of a_1, so, there is a 1-deviation.

Case 4: the agent a_2 has a different color than b_1 and b_2. In this case, their respective color classes are: A_2, either $A_1 \cup \{b_1\}$ or $A_1 \cup \{b_1, c_1\}$, either $A_3 \cup \{b_2\}$ or $A_3 \cup \{b_2, c_1\}$. In particular, b_2 and a_3 have the same color.

Then, there are two subcases. Suppose that c_1 and a_3 have the same color. In this situation, their color class is $A_3 \cup \{b_2, c_1\}$. So, the agent b_3 would increase her payoff from $2 + (|A_0| - 1) = h + 1$ to $2 + |A_3| = h + 2$ by choosing this color, so, there is a 1-deviation. Otherwise, c_1 and a_3 do not have the same color, in which situation their respective color classes are: either $\{c_1\}$ or $A_1 \cup \{b_1, c_1\}$, and $A_3 \cup \{b_2\}$. But then both b_3 and c_1 would increase their respective payoff from $\leq h + 1$ to $h + 2$ by choosing the color of a_3.

Finally, since in all cases there is a 3-deviation, there does not exist a 3-stable partition for the coloring game defined on G. □

Let us define, for every fixed set $\mathcal{W}$, $k(\mathcal{W})$ to be the largest k such that every coloring game which is played on a graph with edge-weights in $\mathcal{W}$ admits a k-stable partition. As an example, for the special case of unweighted games, we have by [KL13] that $k(\{-\infty, 1\}) = +\infty$. In contrast, we have by the combination of Corollary 76 and Proposition 77 that $k(\{-\infty, 0, 1\}) = 2$. In Table 4.1, we report on the value of $k(\mathcal{W})$ for most sets $\mathcal{W}$.

$\mathcal{W}$	$k(\mathcal{W})$
$\{-\infty, a\}, a > 0$	∞
$\{-\infty, 0, a\}, a > 0$	2
$\{-\infty, a, b\}, b > a > 0$	1
$\{-a, b\}, a > 0, b > 0$	$\leq 2 \cdot \lceil \frac{a+1}{b} \rceil + 1$

Table 4.1: Values of $k(\mathcal{W})$ for different $\mathcal{W}$.

Surprisingly, this above threshold $k(\mathcal{W})$ fully characterizes the complexity of recognizing coloring games with a k-stable partition. More precisely, we have obtained the following dichotomy result for generalized coloring games:

Theorem 81. *Let $\mathcal{W}$ contain $-\infty$ and $k \geq 1$ be fixed. Then, the problem of deciding whether a given coloring game, played on a graph with edge-weights in $\mathcal{W}$, admits a k-stable partition is either:*

- *trivial if $k \leq k(\mathcal{W})$;*
- *or NP-complete if $k > k(\mathcal{W})$.*

In order to get a better intuition for the above Theorem 81, let us consider a minimum-size counter-example $G_0 = (V_0, w_0)$ such that the coloring game played on G_0 does not admit a k-stable partition. Our reduction constructs, from any unweighted graph $G = (V, E)$, an edge-weighted supergraph of G_0 (that is illustrated with Figure 4.10).

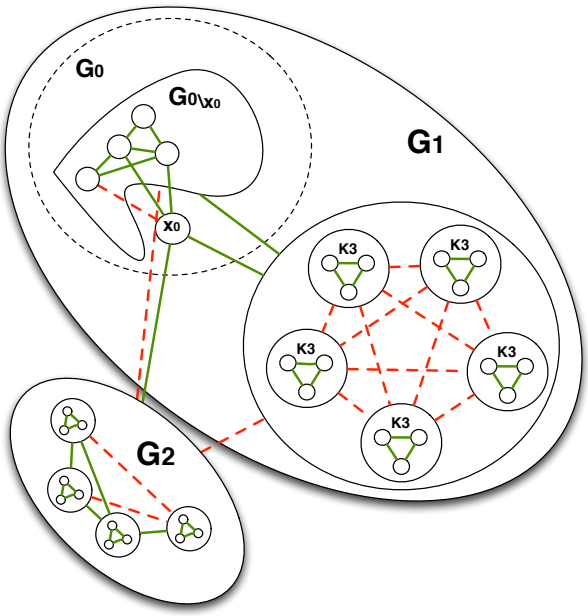


Figure 4.10: Reduction from Maximum independent set. The graph G_0 represents a minimum-size counter-example. Conflict edges with negative weight $-\infty$ are drawn in dashed red whereas all edges drawn in bold green have the same positive weight.

For this graph to have a k-stable partition, one needs a way to force some special agent $x_0 \in V_0$ to pick a different color than the other agents in $V_0 \setminus x_0$. Then, by minimality of the counter-example, the coloring subgame that is played on $G_0 \setminus x_0$ admits a k-stable partition and we are able to extend this subcoloring to a k-stable partition for the game played on the supergraph. Altogether combined, this game played on the supergraph admits a k-stable partition if and only if some agent x_0

can be forced to take a different colour than all other agents in $V_0 \setminus x_0$. Finally, we prove that x_0 indeed takes a different color than the agents of $V_0 \setminus x_0$ if and only if it is part of a large clique in the friendship graph. The latter is shown to correspond to a large independent set in the unweighted graph G that we use for the reduction. As a result, since the MAXIMUM INDEPENDENT SET problem is NP-complete [Dai80], this achieves proving that the problem of recognizing coloring games with a k-stable partition is NP-hard.

4.6 Extensions of coloring games

This section finally covers other games that encompass more aspects of coalition and group formation. We discuss on the extent to which our results for coloring games can be applied to this broader setting. In particular, we intend the following to be a high level description, and so, we made the choice to postpone the proofs of all the results to the research report [DMC12]. These results have not been published elsewhere.

4.6.1 Gossiping

Coloring games with gossip have been introduced by Kleinberg and Ligett in [KL13] for their study of community formation. Such game is still played on an edge-weighted graph $G = (V, E, w)$, with the vertices of G being the agents of the game. However, two agents with distinct colors may now "gossip", in which case both color classes they are part of are merged. Obviously, and as before, this deviation will only take place if it makes increase the utility of the two agents.

Formally, given $G = (V, E, w)$ and $c : V \to \mathbb{N}$, a gossip-deviation w.r.t. c is a 2-subset $\{u, v\}$ such that $c(u) \neq c(v)$ and:

$$\sum_{x|c(x)=c(u)} w_{vx} > 0, \quad \sum_{y|c(y)=c(v)} w_{uy} > 0.$$

The color c represents a k-stable partition for the coloring game with gossip if it is a k-stable partition for the generalized coloring game played on G (without gossip) *and in addition* there is no gossip deviation.

It actually turns out that *unweighted* coloring games with gossip are equivalent to the classical unweighted coloring games. Indeed, consider an unweighted game played on the conflict graph G^-, c a proper coloring of G^-, and suppose that there are two agents u and v gossiping. In particular, u and v cannot be adjacent in G^- to any agent colored by $c(v)$ or $c(u)$ (or else, they would not benefit from merging the two color classes $L_{c(u)}$ and $L_{c(v)}$). Let us assume w.l.o.g. that $|L_{c(u)}| \geq |L_{c(v)}|$. Then, the agent v would also strictly increase her payoff by changing her current color $c(v)$ for $c(u)$. As a result, if there exists a gossip deviation then there is a 1-deviation.

However, in the more general case of weighted games, we prove that there may not exist a 2-stable partition already when there is a unique and fixed positive weight in $\mathcal{W}$. This is in sharp contrast with Corollary 76.

4.6.2 Asymmetry

Another natural variation of coloring games is to make them play on a directed graph. In this situation, colorings of the game and strategies and utility functions of the agents can be defined similarly as before. However, it may now be the case that $w_{uv} \neq w_{vu}$ for some pairs u, v. These games are sometimes called additively separable (asymmetric) Hedonic games [BZ03]. We refer to Figure 4.11 for an illustration.

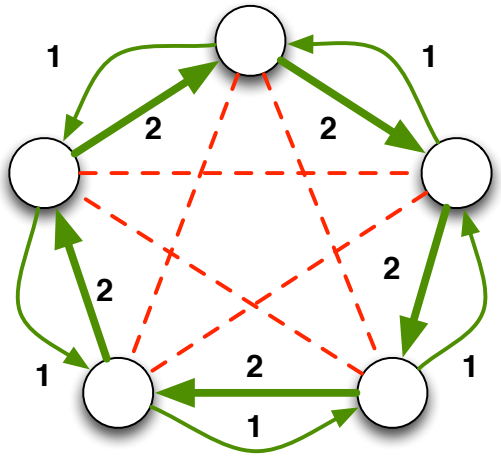


Figure 4.11: A coloring game played on a directed graph. Bidirectional arcs with negative weight $-\infty$ are drawn in dashed red. This game can be shown not to admit a Nash equilibrium.

Even if modest generalization of coloring games, the addition of asymmetrical weights leads to much stronger form of intractability. This can be seen with a simple digraph $D = (\{u, v\}, w)$ such that $w_{uv} > 0$ whereas $w_{vu} < 0$. Clearly, there does not exist any Nash equilibrium for the game played on D.

On the complexity point of view, the problem of deciding whether an asymmetric game admits a Nash equilibrium is NP-hard [SD10]. We prove that this result holds already when there can be no more than two color classes at equilibrium (we prove this by reducing from the well-known Partition problem). We recall that in contrast, every generalized coloring games admits a Nash equilibrium, and that such an equilibrium can be computed in quasi-polynomial time with better-response dynamics.

4.6.3 List coloring games

In [DMC12], we introduced a third variation of coloring games, where the strategy of an agent is no more her color, but rather a list of q colors with $q \geq 1$ being a fixed constant. On the social network analysis point of view, our aim in doing so was to allow every user to be part of different communities in order to better represent the community formation process.

In particular, given $G = (V, E, w)$, a configuration of the q-list coloring game played on this graph is a list coloring of G with each vertex having a list of at most q colors, and we name by $\ell(v)$ the list of any agent $v \in V$. Given a fixed q-list coloring of G, the utility function of v now depends on the number of colors that v shares with each peer, that can be written as:

$$\sum_{u \in V} h\left(|\ell(u) \cap \ell(v)|, w_{uv}\right) \tag{4.1}$$

where $h(g, w)$ is a function measuring the utility of sharing g colors with an agent when it is connected to v by an edge with weight w. Note that we assume, without loss of generality, that:

$$h(0, .) = 0,\ h(., 0) = 0 \text{ and } \forall w \in \mathbb{Q},\ h(1, w) = w$$

$$\forall g \in \mathbb{N},\ w \mapsto h(g, w) \text{ is a non-decreasing function,}$$

$$\forall w \in \mathbb{Q},\ g \mapsto w \cdot h(g, w) \text{ is a non-decreasing function.}$$

The last property simply ensures that $h(g, w)$ increases with g when w is positive, and decreases with g when w is negative. In practice, most of our results are proved in the simpler case where $h : (g, w) \mapsto (1 + \varepsilon g)w$, where ε is a small constant.

On the positive side, every q-list coloring game admits a Nash equilibrium. This can be shown by noticing that a q-list coloring game is a potential game, with its potential function being the global utility (sum of the utility functions of every agent). However, for every $q > 1$, we prove that there exist *unweighted* q-list coloring games that do not admit a 3-strong Nash equilibrium (robust to any coalition of at most three agents). The latter result is in sharp contrast with [KL13, EGM12], where the authors prove that every unweighted coloring game admits a k-stable partition for every fixed $k \geq 1$.

Last, we want to emphasize, perhaps counter-intuitively, that a decrease of the parameter q does not preserve the existence of k-strong equilibria. Namely, for every q, there exists $G_q = (V, E, w)$ such that:

- the q-list coloring game played on G_q *does not* admit any 2-strong Nash equilibrium;
- whereas for any other $q' \neq q$, the q'-list coloring game that is played on this same graph G_q does admit a 2-strong Nash equilibrium.

4.6.4 Coloring games on hypergraphs

Finally, we briefly consider the case where we replace the underlying graph $G = (V, E, w)$ by a *hypergraph* $H = (V, E, w)$, with $w : E \mapsto \mathbb{Q} \cup \{-\infty\}$ being a weight function on the hyperedges. On the social network point of view, hyperedges allow one to account for more complex types of relationships between the users.

Formally, given $H = (V, E, w)$ and $c : V \to \mathbb{N}$ a coloring of H, the utility function of any agent $v \in V$ can now be written as the sum of the weights w_e, for all hyperedge e to which v is incident and such that every vertex $u \in e$ satisfies $c(u) = c(v)$. In short, it is:

$$\sum_{e \in E | \{v\} \subseteq e \subseteq L_{c(v)}} w_e,$$

with $L_{c(v)} = \{u \in V \mid c(u) = c(v)\}$. This game was studied by Deng and Papadimitriou in [DP94], but with transferable utilities[4].

On the positive side, every coloring game played on a hypergraph is a potential game, with its potential function being the sum of the weights w_e for all *monocolored* hyperedges e (*i.e.*, every two vertices in e must be assigned the same color). If the coloring game is played on a graph then the latter function is equal to half of the global utility. However, this is not true anymore for coloring games on hypergraphs, because hyperedges may now be of arbitrary size.

In particular, we get that every coloring game played on a hypergraph admits a Nash equilibrium, and that one such equilibrium can be computed in quasi-polynomial time with better-response dynamics. We can also extend the positive result of Theorem 75 by taking for cycles the notion of Berge cyclicity (cycles in the incidence graph). Unfortunately, there exist hypergraphs with girth two (w.r.t. Berge cyclicity), so, this extended Theorem 75 has weaker consequences for hypergraphs than it has for graphs. As an example, Corollary 76 does not hold for coloring games on hypergraphs.

4.7 Concluding remarks

Our results in this section shed new lights on the complexity of coloring games. In particular, our results for generalized coloring games in Section 4.5 reinforce the relationship between these games and the MAXIMUM INDEPENDENT SET problem in graphs.

Furthermore, we presented in Section 4.3 an interesting relationship between unweighted games (non generalized coloring games) and the lattice of integer partitions. I believe that an in-depth study of this relationship will help to better understand the structure of stable partitions for unweighted games, and the com-

[4]Informally, there are transferable utilities if arbitrary subsets of agents can share their respective utility functions together, whose total sum is then reparted to these agents w.r.t. some rules.

plexity for computing their equilibria. In particular, the main open question in this field is whether the problem of computing 4-stable partitions is PLS-complete.

My investigations on the parallel and space complexity for computing Nash equilibria, in Section 4.4, have been firstly motivated by this above question. Indeed, I hope that the reduction from the Monotone Circuit Value problem to the computation of Nash equilibria can be transformed into a reduction from FLIP – a circuit computation problem that is the standard PLS-complete problem [JPY88] – to the computation of 4-stable partitions.

On a more general side, an interesting question would be to determine whether conversely, PLS-completeness for a "weighted" game implies PTIME-completeness for some corresponding 'unweighted" game ? Relationships between PLS-completeness and PTIME-completeness have been investigated since the original paper [JPY88] (introducing the complexity class PLS). It was conjectured that PLS-completeness for a search problem implies that checking for the local optimality of a solution is PTIME-complete. However, this conjecture was disproved in [Kre89]. Since for many PLS-complete problems, there exists a local-search algorithm that runs in quasi polynomial time (polynomial in the size, but exponential in the weights), any variation of these games where the weights are bounded is trivially in PTIME. Thus, proving or disproving that these variations are PTIME-hard would make advance our understanding of what makes a search problem PLS-hard.

Chapter 5

Learning formulas in a noisy model

Summary

We introduce a new learning model in Section 5.2. This model is motivated by some applications in *Web's transparency*, that is a nascent field where there is a need for uncovering data misuse online. Our objective is to learn an unknown Boolean function that represents the (potentially sensitive) data targeted by a given advertiser.

In Section 5.3, we describe an algorithm for learning the function in the particular case where it depends on a single data input. The cornerstone of this algorithm is a reduction to a Set Cover problem, that is also at the basis of our work in the subsequent sections.

In Section 5.4, we present sufficient conditions – w.r.t. the classification noise in our model – in order to generalize this algorithm for learning every *monotonic* function that only depends on a fixed number of inputs. We also propose an improved algorithm that runs in quasi-linear time, but that can only be applied assuming more restrictive hypotheses on the noise.

Finally, we question in Section 5.5 what can be learnt within our model. On the positive side, we prove that if the function only depends on a fixed number of inputs, positively or negatively, then all these inputs can be computed in quasi-polynomial time with high probability. Under one additional assumption on the classification noise, this algorithm can be extended for learning the function. However, we prove that in general, not all functions can be learnt within our model. Actually, it is impossible to distinguish a conjunction from a disjunction, even if they only depend on two inputs.

My papers on this learning problem [LDL+14, DLCG15, DTC17, CD17] are collected in the appendix.

Contents

5.1 Introduction

This chapter is now devoted to a learning problem on Boolean functions, that we motivate next. Roughly, we aim at making possible for every user online to uncover any misuse of her data. Although *Big Data* promises important societal progress, it exacerbates at the same time the need for algorithmic accountability as more and more decisions affecting millions of users are being automated using personal and private information. Examples of such practices have begun to surface. In a recent incident, Google was found to have used institutional emails from ad-free Google Apps for Education to target ads in users' personal accounts [Gou14, Saf13]. MySpace was found to have violated its privacy policy by leaking personally identifiable information to advertisers [KW10]. Several consumer sites, such as Orbitz and Staples, were found to have adjusted their product pricing based on user location [Mat12, VDSVS12]. And Facebook's 2010 ad targeting was shown to be vulnerable to micro-targeted ads specially crafted to reveal a user's private profile data [Kor11].

The recent area of *Web's transparency* has developed generic methods to reveal which information item or input generates personalization and differentiated treatments [DTD15, LDL+14, LSS+15]. Their output should not be regarded as absolute

truth, but rather as evidence for further investigation. In this work, we aim at giving a theoretical framework in order to analyse these methods. We also describe new core algorithms for these methods that are formally analysed in our setting.

Our contributions in this chapter are summarized in Section 5.1.1. Then, an outline of the chapter is provided in Section 5.1.2.

5.1.1 Our results

Simply put, we aim at describing the core algorithms for Web's transparency tools, and to provide the theoretical framework in order to analyse these algorithms. We detail this a bit more below.

5.1.1.1 A theory for ad targeting identification

Let Ad Targeting Detection be defined as the problem of deciding whether some specific input is targeted by a given ad. Similarly, let Ad Targeting Identification be defined as the problem of deciding which inputs are targeted by this ad. First, based on recent experiments [DTD15, LDL+14], we model the problems of Ad Targeting Detection and Ad Targeting Identification as a learning problem, where the hypothesis is a Boolean function that represents the (potentially sensitive) data inputs targeted by a given advertiser[1].

We report on this model and on its motivations in Section 5.2. This is joint work with Augustin Chaintreau.

Furthermore, all the other results that are presented in this chapter are proved in the learning model of Section 5.2.

5.1.1.2 A general approach reducing to Set Cover

In the following two Sections 5.3 and 5.4, we present algorithms for learning a function that only depends on a constant number of inputs and that is *monotonic* (increasing the number of data inputs cannot make decrease the likelihood to receive an ad). These algorithms are based on a reduction to a natural variation of Set Cover, where we seek for a minimum-size family of subsets (*each representing an input that is targeted*) covering a large fraction of a given universal set (*representing all the accounts that receive a given ad*).

This general approach is presented in Section 5.3, along with an algorithm for learning a function when it depends on a single input. Then, this algorithm is generalized in Section 5.4 for learning a monotonic function under the hypothesis that it depends on at most k inputs, for some fixed k. However, this generalized algorithm is proved to be correct only under a technical assumption, namely, if the *classification noise* of the oracle is bounded. The latter assumption implies that it

[1] Note that Ad Targeting Detection can be reduced to a particular case of Ad Targeting Identification where it is asked whether the targeting can be represented by the null-function.

is much likelier for an account within scope to receive an ad than for an account out of scope.

This is joint work with Mathias Lécuyer, Francis Lan, Andrei Papancea, Theofilos Petsios, Riley Spahn, Max Tucker, Augustin Chaintreau and Roxana Geambasu.

5.1.1.3 Necessary and sufficient conditions for learnability

Finally, we give in Section 5.5 a more general algorithmic proof that any function depending on a fixed number of inputs can be learnt — if we make additional assumptions on the oracle. More precisely, we prove that all the relevant inputs on which the function depends can be learnt if an upper-bound on their number is fixed in advance. The latter can be extended to an algorithm for learning any function, but that is proved to be correct only under an additional technical assumption (we call it "strong positive variance" of the oracle). Roughly, we suppose that there can be no population of accounts within scope that are significantly likelier to receive a given ad than all other accounts within scope.

Last, we prove that in general, if no additional assumption is given then only the functions depending on a *single* input can be learnt in our model.

5.1.2 Outline of the chapter

We first introduce a new learning model for Boolean functions in Section 5.2. In Section 5.3, we introduce a generic method in order to design learning algorithms in this model, and to formally analyse these algorithms. We apply this method to the particular case where the function to be learnt only depends on a single input. Then, in Section 5.4 we extend this approach to more general (monotonic) functions, that requires a more in-depth analysis of our probabilistic tools. Finally, in Section 5.5, we delineate the minimal hypotheses to incoporate in the model in order to make *any* function learnable. Note that these hypotheses are not part of the core assumptions for our learning model because they have not been confirmed experimentally. We then conclude this chapter in Section 5.6.

5.2 Learning model

The following presentation of our learning model is kept generic on purpose in order to apply to a broad set of scenarii of online targeting. Let $\mathcal{D} = \{D_1, D_2, \ldots, D_N\}$ be a set of N inputs representing individual information from a given user (typically, keywords extracted from emails in an account, see also [LDL+14]). Our main objective is to identify how these inputs affect a given output of interest (say, an ad or a recommendation). In order to achieve the goal, we here assume that each output is affected through an unknown targeting function f_{output}, that we simply denote by f in the following. The targeting function f is a mapping from the family of all *combinations* (subsets of $\mathcal{D}$) to the Boolean set $\{0; 1\}$. By convention, $f(\mathcal{C}) = 1$ indicates that an account *exactly* containing the inputs in $\mathcal{C}$ is targeted,

and we denote $f(.) = 0$ if the ad is untargeted. We aim at learning f subject to diverse requirements, each representing one aspect of our experiments for doing so in practice.

A generic framework from learning theory is first presented in Section 5.2.1. Then, we detail how we adapt this framework to our needs in the subsequent Sections 5.2.2, 5.2.3 and 5.2.4. This model is part of our paper [CD17], that is joint work with Augustin Chaintreau.

5.2.1 PAC learning

We refer to [Ang88] for basics of computational learning theory and query complexity. A *hypothesis* $\mathcal{H}$ is a class of Boolean functions. Let $f : \{0,1\}^N \mapsto \{0,1\}$ be an (unknown) function, possibly not in $\mathcal{H}$. In what follows, we are given:

- a function $\mathrm{O}_f : \{0,1\}^N \mapsto \{0,1\}$ (possibly randomized), that is called an *oracle* and whose outputs are assumed to depend on the outputs of f.
  Example: a call to the oracle can represent an observation whether a given account has received the ad;
- a random generator of pairs $\langle x, \mathrm{O}_f(x)\rangle$, that is called a *sampler* and for which every $x \in \{0,1\}^N$ is picked at random w.r.t. some fixed probability distribution Π (denoted by $x \sim \Pi$).
  Example: The sampler can represent our experimental setting. In order to learn the targeting function, we are bound to rely on experiments — to see how it reacts to various inputs. For instance, in [LDL+14] these experiments consist in collecting the ads from Gmail accounts with different subsets of emails.

Let ε, δ be nonnegative[2]. A *PAC-learning* algorithm for f under $\mathcal{H}$ hypothesis (*a.k.a.*, probably approximately correct learning algorithm) is given constant-time access to the sampler, and it must compute, in time polynomial in N and $1/\delta$, the representation of a function $h \in \mathcal{H}$ such that $\mathbb{P}r[h(x) \neq f(x) \mid x \sim \Pi] \leq \varepsilon$. The *query complexity* of the algorithm is its number of calls to the sampler. It is preferrable to keep this complexity small, say, polylogarithmic in N.

In what follows, we will always assume that $\varepsilon = 0$, *i.e.*, we aim at learning f exactly.

There is a vast literature on this problem [Ang88, AR07, FGKP09, MOS04, Val12], with different choices made for: the dependencies between the oracle and the function to be learnt, the distribution for the sampler, the representation of a function, the hypothesis, etc. The main novelty in this work is the set of assumptions on the oracle, and to some extent the choice for the representation of the functions. All the choices made for this work will be presented and discussed in this section.

Outline. In Section 5.2.2, we introduce basic terminology for a specific class of functions called *juntas*, that will be our hypothesis. Our choice for the representation of a function is also discussed in this section. Then, we formally describe our set of

[2]Note that here, δ is no longer related to graph hyperbolicity (defined in Chapter 2).

assumptions on the oracle in Section 5.2.3. In particular, we briefly report on some experiments in Section 5.2.3.1 that have supported the choices made in this work. The axioms on the oracle are given in Section 5.2.3.2. We end this section with our choices made for the sampler in Section 5.2.4.

5.2.2 Juntas

Our choices for the hypothesis $\mathcal{H}$ and the representation of a function are presented in this subsection. Complementary information for the case of monotonic functions is given in Section 5.2.2.1.

The following presentation differs from the standard terminology in the literature of Boolean function learning, but it is shown to be equivalent to it. This change of terminology is motivated by our interpretation of a Boolean word $w \in \{0,1\}^N$ as denoting the content of an online account.

Let $\mathcal{D} = \{D_1, D_2, \ldots, D_N\}$ be a fixed ground set. There is a natural one-to-one mapping between $\{0,1\}^N$ and $2^{\mathcal{D}}$ (power-set of $\mathcal{D}$), defined as $\phi : w \in \{0,1\}^N \mapsto \{D_i \in \mathcal{D} \mid w_i = 1\}$. For simplicity, we will identify f with $f \circ \phi^{-1}$ in what follows. Furthermore, we will call a subset of $\mathcal{D}$ a *combination*. The function f is said to depend on D_i if there exists a combination $\mathcal{C} \subseteq \mathcal{D} \setminus D_i$ such that $f(\mathcal{C}) \neq f(\mathcal{C} \cup \{D_i\})$.

Definition 82 ([BL97]). For every $k \geq 1$, f is a k-junta if it depends on at most k inputs $D_i \in \mathcal{D}$.

In what follows, we will select the class of k-juntas, for some constant k, as our hypothesis. Note that in practice, it is recommended to advertisers to select k in some range between 5 and 20 [Goo].

Representation of a junta. An *implicant* of f is a pair $\langle \mathcal{C}_{in}, \mathcal{C}_{out} \rangle$ of two disjoint combinations of $\mathcal{D}$ with the property that $f(\mathcal{C}) = 1$ for every combination $\mathcal{C}$ such that $\mathcal{C}_{in} \subseteq \mathcal{C}$ and $\mathcal{C} \cap \mathcal{C}_{out} \neq \emptyset$. It is a *prime implicant* of f if for every strict subsets $\mathcal{C}'_{in} \subsetneq \mathcal{C}_{in}$ and $\mathcal{C}'_{out} \subsetneq \mathcal{C}_{out}$, the pair $\langle \mathcal{C}'_{in}, \mathcal{C}'_{out} \rangle$ is not an implicant of f. Every k-junta has $\mathcal{O}(3^k/\sqrt{k})$ prime implicants [CM78].

In what follows, we choose as a representation for any function f the set $\mathcal{S}^{(core)}$ of its prime implicants. Note that we have, for any f:

$$f(\mathcal{C}) = \max_{\langle \mathcal{C}_{in}, \mathcal{C}_{out} \rangle \in \mathcal{S}^{(core)}} \left(\prod_{D_i \in \mathcal{C}_{in}} \mathbb{I}_{\{D_i \in \mathcal{C}\}} \right) \cdot \left(\prod_{D_i \in \mathcal{C}_{out}} (1 - \mathbb{I}_{\{D_i \in \mathcal{C}\}}) \right),$$

with $\mathbb{I}_{\{D_i \in \mathcal{C}\}}$ being an indicator function that takes value 1 only if $D_i \in \mathcal{C}$ (otherwise it is equal to 0). The latter is sometimes called the Blake canonical form of f [Bla38].

5.2.2.1 Case of monotonic functions

A function f is called *monotonic* if for every combination $\mathcal{C} \subseteq \mathcal{D}$ such that $f(\mathcal{C}) = 1$, we have that $f(\mathcal{C}') = 1$ for every superset $\mathcal{C}' \supseteq \mathcal{C}$. Monotonic functions naturally

arise in some settings where negative keywords are unavailable, such as (until recently) Facebook [FBE].

In this situation, we simplify the representation of f as follows. A *family* $\mathcal{S}$ of *size* l is any collection of l distinct combinations. The *order* of the family is defined as the largest order of a combination it contains. Interestingly, there is a duality between families and monotonic functions. Indeed on the one hand, one can define for any family $\mathcal{S}$ a function $f : \mathcal{C} \to \max_{\mathcal{C}_j \in \mathcal{S}} \mathbb{I}_{\{\mathcal{C}_j \subseteq \mathcal{C}\}}$ that takes value $f(\mathcal{C}) = 1$ whenever the subset $\mathcal{C}$ contains at least one combination in $\mathcal{S}$. In such case we say that $\mathcal{S}$ *explains* the function. On the other hand, we now show that the converse also holds: given a monotonic function f, there is a *unique* family explaining f that is both of minimum order and minimum size:

Lemma 83. *For each monotonic function f there exists a unique family $\mathcal{S}^{(core)}$ satisfying:*

(i) $\mathcal{S}^{(core)}$ has size l and order r and it explains *f.*

(ii) No family of size $l' < l$ explains f.

(iii) No family of order $r' < r$ explains f.

Proof. We define $\mathcal{S}^{(\text{in})} = \{\mathcal{C} \subseteq \mathcal{D} \mid f(\mathcal{C}) = 1\}$ the set of all combinations for which f takes value 1. Let $\overrightarrow{D}_f$ be the digraph with vertex-set $\mathcal{S}^{(\text{in})}$ and with arc-set $\{(\mathcal{C}, \mathcal{C}') \mid \mathcal{C} \subsetneq \mathcal{C}'\}$. We have that $\overrightarrow{D}_f$ is a DAG (Directed Acyclic Graph) because the subset-containment relation defines a partial order. So, let $\mathcal{S}^{(core)}$ be the non-empty set of combinations with null in-degree in $\overrightarrow{D}_f$. By construction, each combination in $\mathcal{S}^{(\text{in})}$ contains some combination of $\mathcal{S}^{(core)}$ and $\mathcal{S}^{(core)} \subseteq \mathcal{S}^{(\text{in})}$, hence $\mathcal{S}^{(core)}$ explains f. Furthermore, we claim that $\mathcal{S}^{(core)}$ is contained in *any* family $\mathcal{S}'$ explaining f: indeed, since $\mathcal{S}'$ is required to contain a subset of any combination $\mathcal{C} \in \mathcal{S}^{(core)}$, and no combination of $\mathcal{S}^{(\text{in})}$ is strictly contained in $\mathcal{C}$, then it must contain $\mathcal{C}$. This shows that $\mathcal{S}^{(core)}$ satisfies all conditions of Lemma 83. Finally, since another family explaining f needs to include $\mathcal{S}^{(core)}$, then it will necessarily have a higher size l, hence $\mathcal{S}^{(core)}$ is the unique with both minimum size and order. □

For every monotonic function f, the family whose existence is proved in Lemma 83 is called its *core family* and we choose this family as the representation of f.

5.2.3 The oracle

We now introduce specific assumptions on the oracle O_f. We recall that the latter formalizes the observations gathered from different online accounts, *i.e.*, the collection of advertisements received w.r.t. the data inputs contained in the accounts. So, we first report on some experiments in Section 5.2.3.1 in order to motivate our choices for the oracle, presented in Section 5.2.3.2. Finally, an idealized oracle (formerly used in our papers [LDL+14, DLCG15, DTC17]) is discussed in Section 5.2.3.3.

5.2.3.1 Supporting experiments

We briefly report on some experiments whose results and interpretations have motivated our choices for the oracle.

Experiment 1: Correlation of the outcomes with the function to be learnt. In [LDL+14], we posted four Google AdWords campaigns targeted on very specific keywords (Chaldean Poetry, Steampunk, Cosplay, and Falconry). Then, we placed in more than 800 Gmail accounts some emails including these keywords. Overall, the corresponding ads were received by more than 97% of the accounts. The latter shows, as expected, a positive correlation between the outcomes of the experiments and the scope of the campaign.

Experiment 2: Limited coverage. The *coverage* is defined as the true positive rate (*i.e.*, the average probability for an account within the scope of some advertisement campaign to receive this ad). By varying the number N of inputs in our experiments, we have observed that the coverage is a decreasing function in the number of data inputs contained in the accounts. This might come from a larger pool of advertising campaigns for which the accounts are within scope, that makes obtaining an ad slot more competitive. In particular, the probability of receiving an ad cannot be assumed to be a constant that is independent from N.

Experiment 3: Cross-unit effects. The authors in [TDDW15] showed that multiple browser instances running in parallel affect one another. They did so by comparing the diversity of the ads received by browsers running in isolation w.r.t. browsers running in parallel (see [TDDW15] for details). This result suggests that the outcomes of different observations are correlated.

5.2.3.2 Axiomatisation

Let us now introduce our assumptions on the oracle. Formally, O_f is a membership oracle with (asymmetric) classification noise. That is, it outputs the Boolean $f(\mathcal{C})$ for any combination $\mathcal{C}$ with some probability to flip the result. Unlike prior work [Ang88], we do not assume the classification noise to be symmetric, *i.e.*, the oracle may flip the result with some propability *depending on the combination.* Nonetheless, we will assume a few properties for the noise distribution. To our best knowledge, the following assumptions that are made on this probability have not been studied before in the literature.

Histories. Experiment 3 in Section 5.2.3.1 have evidenced that the noise distribution is subject to cross-unit effects. So, in order to handle with these correlations, we find it more suitable to generalize our oracle O_f so that it can take *families of combinations* as inputs. More precisely, let a family be any vector of combinations, denoted by $\mathcal{F} = \langle A_1, A_2, \ldots, A_t \rangle$. The outcome $\mathrm{O}_f(\mathcal{F})$ is simply defined as the

binary vector $\mathrm{O}_f(\mathcal{F}) = \langle \mathrm{O}_f(A_1), \mathrm{O}_f(A_2), \ldots, \mathrm{O}_f(A_t) \rangle$. Furthermore, let the pair $\mathrm{H}_{\mathcal{F}} = (\mathcal{F}; \mathrm{O}_f(\mathcal{F}))$ be the history of $\mathcal{F}$.

Let $\mathcal{F}_{-i} = \langle A_1, \ldots, A_{i-1}, A_{i+1}, \ldots, A_t \rangle$. We will assume that each individual outcome $\mathrm{O}_f(A_i)$ is correlated to the partial history $\mathrm{H}_{\mathcal{F}_{-i}}$. However, it may and must be the case that some natural properties hold independently from any history, that we now detail as follows. Let us point out that $\mathcal{S}^{(\mathrm{in})}$ stands for the set of all combinations $\mathcal{C}$ such that $f(\mathcal{C}) = 1$.

Assumption 1 (targeting lift)**.** There exists a universal constant $\varphi \in]0; 1[$, called the targeting lift and such that for any $\mathcal{C}_0, \mathcal{C}_1$ with $f(\mathcal{C}_0) = 0, f(\mathcal{C}_1) = 1$:

$$\mathbb{P}r[\mathrm{O}_f(A_i) = 1 \mid A_i = \mathcal{C}_0, \mathrm{H}_{\mathcal{F}_{-i}}] < \varphi \cdot \mathbb{P}r[\mathrm{O}_f(A_i) = 1 \mid A_i = \mathcal{C}_1, \mathrm{H}_{\mathcal{F}_{-i}}].$$

This Assumption 1 is local and it simply ensures that it is more likely to receive an ad for an account within scope than out of scope (conditioned on any fixed history $\mathrm{H}_{\mathcal{F}_{-i}}$). In particular, it implies that the targeting function f is related to the outcome we study.

As we will show in Section 5.4, our most efficient algorithms are proved to be valid only if the targeting lift is bounded.

Assumption 2 (polynomial-growth)**.** There exist positive universal constant α, β, γ with $\alpha \leq 1$ and such that:

$$\mathbb{P}\left[\sum_{i=1}^{t} \mathrm{O}_f(A_i) < \left(\beta \cdot |\mathcal{F} \cap \mathcal{S}^{(\mathrm{in})}|^{\alpha}\right)\right] \leq e^{-\gamma \cdot t}$$

In accordance with Experiment 1 in Section 5.2.3.1, we properly state with Assumption 2 that the amount of accounts receiving an ad must be at least a significant fraction of the account population *within scope* , except on some small event with low probability like, for instance, when the targeting campaign runs out of budget.

Let us point out that if we were assuming that there is some minimum *constant* probability p_{in} for an account within scope to be targeted, Assumption 2 could be shown to be satisfied for $\alpha = 1$ by using standard concentration inequalities. By considering the case $\alpha \leq 1$, we may consider the case where this minimum probability slowly tends to zero when N grows, say, $p_{in} \sim p_o / \log^{\mathcal{O}(1)}(N)$ where p_0 is a constant. The latter case seems to be what happens in practice, as supported by Experiment 2 in Section 5.2.3.1.

Assumption 3 (noninterference)**.** Let the function f only depend on inputs in $V \subseteq \mathcal{D}$. Furthermore, let $A'_i = A_i \cap V$ and let $\mathcal{F}' = \langle A'_1, \ldots, A'_t \rangle$.

$$\mathbb{P}r[\mathrm{O}_f(\mathcal{F})] = \mathbb{P}r[\mathrm{O}_f(\mathcal{F}')].$$

Finally, we formalize with Assumption 3 that none of the input that does not affect the function f can impact on the outcome.

5.2.3.3 Discussion: idealized model with independence

For simplicity, we were assuming in [LDL+14] an idealized learning model where the outputs of the oracle were independent random variables and there were two constant p_{in}, p_{out} such that:

$$\mathbb{P}r[\mathrm{O}_f(\mathcal{C})=1 \mid f(\mathcal{C})=1] = p_{in} > p_{out} = \mathbb{P}r[\mathrm{O}_f(\mathcal{C}')=1 \mid f(\mathcal{C}')=0].$$

Limitations. Independence in the model contradicts Experiment 3 in Section 5.2.3.1. Similarly, a constant probability p_{in} to be targeted contradicts Experiment 2 in Section 5.2.3.1. These are the reasons why we are now considering the more general assumptions in Section 5.2.3.2 for the oracle.

Nonetheless, we will see in what follows that our former analysis in the idealized model still holds under more general assumptions. Precisely, the approach presented in this chapter leaves us to analyse a random counting process whose outcome can be lower and upper-bounded by estimating the sum of *independent* random variables (see Lemma 86). In particular, by choosing p_{in}, p_{out} so that:

$$\begin{cases} p_{in} = (1+\mathcal{O}(1)) \cdot \frac{\beta}{\log^{1/\alpha-1}(N)} \\ p_{out} < \varphi \cdot p_{in} \end{cases}$$

all the results obtained with the simpler model in [LDL+14, DLCG15, DTC17] can be generalized to the more general model that is presented in this Section 5.2.

5.2.4 Distribution for the sampler

Last, we present our choices for the distribution and the sampler. The latter formalizes our experimental process, that consists in creating fake Gmail accounts and filling in them with random data.

Exchangeability is defined in [GR86] as the probability that if two accounts were exchanging their data inputs, the probability distribution of the outcome would not be impacted. So, in order to get exchangeability, we take a Bernouilli distribution $\Pi = B(p, N)$, *i.e.*, for every random combination that is sampled, each input $D_i \in \mathcal{D}$ must be present independently at random with probability p.

Interestingly, our process is related with the so-called random intersection model [KSSC99], that can be defined as follows. Let N, M and h be positive integers, and let p be some probability. In order to create a random intersection graph, we first create a bipartite graph B randomly with two sides of respective size N and M, and with each edge being present independently at random with probability p. Then, a new graph is created from B by taking as vertex-set the side of size M and adding an edge between every two vertices that share at least h common neighbours in B, for some constant h. Random intersection graphs have been proposed as a model for complex networks [GL06]. So, it makes sense to mimic this process in order to create random Gmail accounts.

5.3 Single-input targeting

This section addresses the detection and identification of *single-input* targeting, that is when the reception of the output is caused by the presence (or the absence) of a single input. More formally, we propose a PAC-learning algorithm with 1-juntas as hypothesis.

This is joint work with Mathias Lécuyer, Francis Lan, Andrei Papancea, Theofilos Petsios, Riley Spahn, Augustin Chaintreau and Roxana Geambasu.

Outline. Our main result is stated in Section 5.3.1, where we also discuss on its positioning in the nascent field of Web's transparency. Then, the following Sections 5.3.2 and 5.3.3 cover the main tools used in our study. The first tool is algorithmic: we present in Section 5.3.2 a classical technique for learning Boolean functions, of which we use a natural variation as the main brick basis of our algorithm (presented in Section 5.3.2.1). Second, we adapt in Section 5.3.3 standard concentration inequalities to our learning model. The latter will be our main tool in the analysis of the algorithm. Finally, we sketch this algorithm for learning 1-juntas in Section 5.3.4.

Full proofs can be found in our paper [LDL+14]. The version presented in this section also borrows from our paper [CD17] (in preparation).

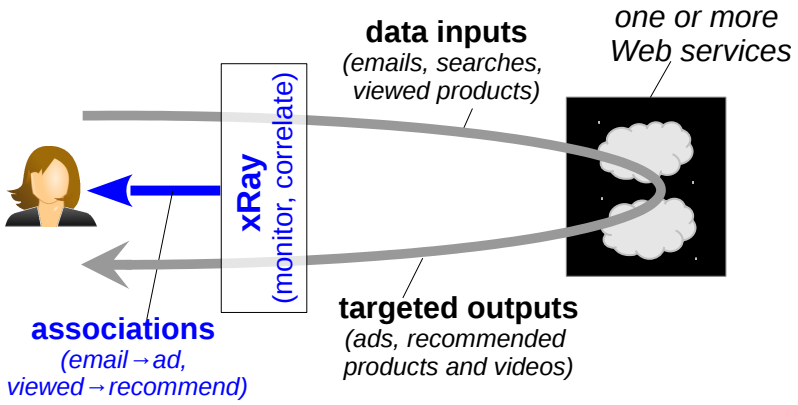


Figure 5.1: Xray suggests plausible associations between the emails of a user and the ads she receives, using the core algorithm presented in this section.

5.3.1 Our results

Below, we state our main result in this section.

Theorem 84. *Let $\alpha \leq 1$ be the polynomial-growth (Assumption 2). There is a PAC-learning algorithm such that, for every $\varepsilon > 0$, the targeting function can be learnt with probability $1 - \varepsilon$ under 1-juntas hypothesis, in $\mathcal{O}(N \cdot \log^{1/\alpha}(N/\varepsilon))$-time and $\mathcal{O}(\log^{1/\alpha}(N/\varepsilon)))$ queries.*

Theorem 84 is the core algorithm of a prototype called *Xray*, that we introduced in [LDL+14] and on which we now give more emphasis. Roughly, Xray predicts through the help of its core algorithm which data in an arbitrary Web account (such as emails, searches, or viewed products) is being used to target which outputs (such as ads, recommended products, or prices). We refer to Figure 5.1 for an illustration of its functioning. This problem has received some attention in the literature [DTD15, HSMK+13, HSL+14, LDL+14, MGEL12]. However, concurrently with [DTD15], our work on Xray has been the first, to the best of our knowledge, to provide *theoretical* guarantees on the predictions made under plausible assumptions. Furthermore, our core algorithm has (poly)logarithmic query complexity, whereas the authors in [DTD15] (using a different model than in Section 5.2) have proposed an algorithm with *linear* query complexity.

5.3.2 Reduction to Set Cover

The following reduction has been proposed in [AMK03] in order to infer Boolean functions from positive and negative examples. We detail how we can adapt this work to our setting in Section 5.3.2.1

First, suppose that $\mathrm{O}_f = f$ (*i.e.*, there is no classification noise). In this situation, the following lemma holds:

Lemma 85 ([MOS04]). *Let f be a nonconstant k-junta. Suppose that with confidence $1-\varepsilon$, it can be computed an input D_i on which the function f depends in time $n^c \cdot poly(2^k, 1/\varepsilon)$. Then there is an algorithm for exactly learning f which runs in time $n^c \cdot poly(2^k, 1/\varepsilon)$.*

By Lemma 85, if there is no classification noise then learning f can be reduced to compute the inputs on which this function depends. In order to do so, let $\langle \mathcal{C}_1, f(\mathcal{C}_1)\rangle, \langle \mathcal{C}_2, f(\mathcal{C}_2)\rangle, \ldots, \langle \mathcal{C}_m, f(\mathcal{C}_m)\rangle$ be drawn from the sampler. Let us take as our universal set $\mathcal{U} = \{(j_1, j_2) \mid j_1 < j_2 \text{ and } f(\mathcal{C}_{j_1}) \neq f(\mathcal{C}_{j_2})\}$. For every pair $(j_1, j_2) \in \mathcal{U}$, since $f(\mathcal{C}_{j_1}) \neq f(\mathcal{C}_{j_2})$ the two combinations $\mathcal{C}_{j_1}, \mathcal{C}_{j_2}$ must differ in at least one input on which the function f depends. In particular, let us define, for every input $D_i \in \mathcal{D}$, the set $\mathcal{S}_i = \{(j_1, j_2) \mid j_1 < j_2 \text{ and } D_i \in \mathcal{C}_{j_1} \Delta \mathcal{C}_{j_2}\}$, where Δ denotes the symmetric difference between two combinations. Then, it can be proved under some assumptions on the distribution Π of the sampler and the Boolean function f that a minimum set cover for $\mathcal{U}$ with the $\mathcal{S}_i$'s is in one-to-one correspondance with the inputs D_i's on which this function f depends [AMK03].

The obvious drawback of this approach is that computing a minimum set cover is NP-hard [Kar72]. Therefore, greedy heuristics should be used, and the analysis of their output is more delicate [FA05]. However, if the number of relevant variables is assumed to be a constant (that is the case for 1-juntas and more generally, for k-juntas with fixed k), there is no need to rely on such approximations.

5.3.2.1 Set Intersection algorithm

This above machinery cannot be applied to our setting directly because it is strongly dependent on the assumption $\mathrm{O}_f = f$. Indeed, without this assumption, the correspondance between the relevant variables and minimum set covers does not hold. However, based on Assumption 1 (*i.e.*, accounts within scope are much likelier to be targeted than accounts out of scope), it looks intuitive that the relevant variables should still correspond to a set cover for a *large fraction* of the universal set. Hence, in our setting, we propose an alternative reduction to a Set Cover problem with threshold, where we now seek for $x\_$ *intersecting subsets* of small size (defined as the subsets intersecting a fraction x of the universal set), for some parameter $x < 1$. The latter problem is formally described with Algorithm 1.

Input: a family $\mathcal{F}$; a threshold parameter $x < 1$; a size parameter s.
Output: the family $\mathcal{S}_x$ of $x_$intersecting combinations of size at most s.

$\mathcal{S}_x \leftarrow \{\}$;
foreach $\mathcal{C} \subseteq \mathcal{D}$ **s.t.** $|\mathcal{C}| \leq s$ **do**
  if $\mathcal{C}$ *intersects* $\geq x \cdot |\mathcal{F}|$ *accounts in* $\mathcal{F}$ **then**
    $\mathcal{S}_x \leftarrow \mathcal{S}_x \cup \{\mathcal{C}\}$;
  end
end

Algorithm 1: Set-intersection algorithm.

Discussion: parameter tuning. The reader may observe that Algorithm 1 requires a threshold parameter x as input. For simplicity, we will assume that a good estimate on the *targeting lift* (Assumption 1) is given, and we will show in the following that this information is enough in order to tune x. Nonetheless, we point out that finding this parameter in practice might be cumbersome.

Two methods have been proposed for doing so in [DTC17]. If we are given a ground-truth, *i.e.*, a set of functions f with their representation, then we can use it in order to tune the parameter x by reverse-engineering. However, a ground-truth is not always available. In this situation, we are limited to pick uniformally at random different values for x in a given interval, then to make our algorithms run in parallel. This interval can be chosen so that there can be no false negative (*i.e.*, all the relevant inputs are detected). Then, a final filtering process is used to elect the run whose output has to be taken into account.

5.3.3 Concentration inequalities

We complete Section 5.3.2 by introducing the main tool that will be used to analyse our subsequent algorithms (Lemma 86). Roughly, when using Algorithm 1 as a routine, we aim at approximating some random counting process in order to determine the existence or nonexistence of $x_$intersecting subsets. Classically, concentration

inequalities such as Chernoff bounds [Hoe63] are used in the analysis. However, standard concentration inequalities apply to the sum of *independent* variables, so, they cannot be used in our setting directly. The following is a tedious (but classical) analysis where we show how to adapt Chernoff bounds to our needs.

Lemma 86. *Let $X_1, \ldots, X_m$ be random Boolean variables satisfying:*

$$p_{\min} \leq \mathbb{P}r[X_i = 1 \mid X_1, \ldots, X_{i-1}] \leq p_{\max}$$

for some constant $p_{\min}, p_{\max}$. Then the following hold for any $0 < \delta < 1$:

$$\mathbb{P}r[\sum_{i=1}^{m} X_i \geq (1+\delta) \cdot p_{\max} \cdot m] \leq e^{-\delta^2 m p_{\max}/3}$$

$$\mathbb{P}r[\sum_{i=1}^{m} X_i \leq (1-\delta) \cdot p_{\min} \cdot m] \leq e^{-\delta^2 m p_{\min}/2}$$

Proof. By symmetry, we will only consider the first inequality. Let $t > 0$. Let us show that:

$$\mathbb{E}[\Pi_{i=1}^{m} e^{t \cdot X_i}] \leq \left(p_{\max}(e^t - 1) + 1\right)^m.$$

The proof is by induction. By the hypothesis,

$$\begin{aligned}
&\mathbb{E}[e^{t \cdot X_m} \mid X_1, \ldots, X_{m-1}] \\
&\qquad = e^t \cdot \mathbb{P}r[X_m = 1 \mid X_1, \ldots, X_{m-1}] + 1 \cdot \mathbb{P}r[X_m = 0 \mid X_1, \ldots, X_{m-1}] \\
&\qquad \leq p_{\max}(e^t - 1) + 1,
\end{aligned}$$

that is the base case. Suppose for the induction hypothesis that:

$$\mathbb{E}[\Pi_{j=i+1}^{m} e^{t \cdot X_j} \mid X_1, \ldots, X_i] \leq \left(p_{\max}(e^t - 1) + 1\right)^{m-i}.$$

Then by the law of total probability:

$$\begin{aligned}
&\mathbb{E}[\Pi_{j=i}^{m} e^{t \cdot X_j} \mid X_1, \ldots, X_{i-1}] \\
&\qquad = e^t \cdot \mathbb{P}r[X_i = 1 \mid X_1, \ldots, X_{i-1}] \cdot \mathbb{E}[\Pi_{j=i+1}^{m} e^{t \cdot X_j} \mid X_1, \ldots, X_{i-1}, X_i = 1] \\
&\qquad + 1 \cdot \mathbb{P}r[X_i = 0 \mid X_1, \ldots, X_{i-1}] \cdot \mathbb{E}[\Pi_{j=i+1}^{m} e^{t \cdot X_j} \mid X_1, \ldots, X_{i-1}, X_i = 0] \\
&\qquad \leq \left(\mathbb{P}r[X_i = 1 \mid X_1, \ldots, X_{i-1}] \cdot (e^t - 1) + 1\right) \cdot \left(p_{\max}(e^t - 1) + 1\right)^{m-i} \\
&\qquad \leq \left(p_{\max}(e^t - 1) + 1\right)^{m-i+1},
\end{aligned}$$

which proves the induction hypothesis. The remaining of the proof is now classical

computation of Chernoff Bound. By Markoff inequality:

$$\begin{aligned}\mathbb{P}r[\sum_{i=1}^{m} X_i \geq (1+\delta)\cdot p_{\max}\cdot m] &= \mathbb{P}r[e^{t\cdot\sum_{i=1}^{m} X_i} \geq e^{t\cdot(1+\delta)\cdot p_{\max}\cdot m}] \\ &\leq \mathbb{E}[e^{t\cdot\sum_{i=1}^{m} X_i}]/e^{t\cdot(1+\delta)\cdot p_{\max}\cdot m} \\ &= e^{-t\cdot(1+\delta)\cdot p_{\max}\cdot m}\cdot\mathbb{E}[\Pi_{i=1}^{m} e^{t\cdot X_i}] \\ &\leq e^{-t\cdot(1+\delta)\cdot p_{\max}\cdot m}\cdot\left(p_{\max}(e^t-1)+1\right)^m \\ &\leq e^{-t\cdot(1+\delta)\cdot p_{\max}\cdot m}\cdot e^{p_{\max}(e^t-1)\cdot m} \\ &= e^{p_{\max}\cdot m\cdot\left(e^t-1-t\cdot(1+\delta)\right)}\end{aligned}$$

Finally, set $t = \ln(1+\delta)$. One obtains:

$$\mathbb{P}r[\sum_{i=1}^{m} X_i \geq (1+\delta)\cdot p_{\max}\cdot m] \leq \left(\frac{e^\delta}{(1+\delta)^{1+\delta}}\right)^{mp_{\max}} \leq e^{-\delta^2 m p_{\max}/3}.$$

□

To summarize Sections 5.3.2 and 5.3.3, we aim at learning the targeting function f by reducing to a natural variation of Set Cover (Section 5.3.2), of which we will analyse the outcome by using concentration inequalities (Section 5.3.3).

5.3.4 Proof overview

Finally, let us sketch the proof of Theorem 84. It is based on the correctness proof of the following Algorithm 2. Note that the ground-set $\mathcal{D}$ and its size N, the targeting lift φ (Assumption 1) and the constants α, β, γ (Assumption 2) are known parameters, so, we don't include them in the input of the algorithms.

Roughly, we use the Set-intersection algorithm (Algorithm 1) in order to detect an input that is significantly present (or missing) among the positive examples (*i.e.*, combinations $\mathcal{C}_i$ such that $\mathrm{O}_f(\mathcal{C}_i) = 1$). On the one hand, since irrelevant inputs (on which the targeting function does not depend) cannot affect the outcome, they are present or missing among the positive examples by mere chance. In particular, these inputs can be neither significantly present nor absent among these examples. On the other hand, since it is likelier for the oracle to output 1 for a combination within scope (in $\mathcal{S}^{(\mathrm{in})}$) than for a combination out of scope, there should be slightly more positive examples that are correctly classified than misclassified. Since any input on which f depends positively (resp., negatively) must be present (resp., missing) in all the positive examples that have been correctly classified, this relevant input will be detected by the Set-intersection algorithm with high probability.

Sketch Proof of Theorem 84. Let us set the distribution Π of the sampler to the uniform distribution $B(1/2, N)$. We first make $\Theta(\log^{1/\alpha}(N/\varepsilon))$ queries to the sampler. Note that since f is assumed to depend on only one variable, we have in expectation that $f(\mathcal{C}_i) = 1$ for half of the combinations $\mathcal{C}_i$ queried. Hence, by Chernoff

Input: accuracy ε.
Output: the representation $\mathcal{S}^{(core)}$ of f under 1-juntas hypothesis.

/* *Parameters tuning* */;
Let $x \in]\frac{1}{2}; \frac{1}{1+\varphi}[$;
Let $m \in \Omega\left(\log^{1/\alpha}(N/\varepsilon)\right)$ /**m depends on x* */;

/* *Uniform sampling* */;
Draw $\langle \mathcal{C}_1, \mathrm{O}_f(\mathcal{C}_1)\rangle, \langle \mathcal{C}_2, \mathrm{O}_f(\mathcal{C}_2)\rangle, \ldots, \langle \mathcal{C}_m, \mathrm{O}_f(\mathcal{C}_m)\rangle$ with $\Pi = B(1/2, N)$;
$\mathcal{F} \leftarrow \{\mathcal{C}_i \mid 1 \leq i \leq m \text{ and } \mathrm{O}_f(\mathcal{C}_i) = 1\}$;

/* *Reduction to* Set Cover */ ;
$\mathcal{S}_x \leftarrow$ `Set-intersection`$(\mathcal{F}, x, 1)$;
if $\exists i, \mathcal{S}_x = \{\{D_i\}\}$ **then**
  //positive targeting;
  $\mathcal{S}^{(core)} \leftarrow \{\langle \{D_i\}, \emptyset\rangle\}$;
end
else
  $\overline{\mathcal{F}} \leftarrow \{\mathcal{D} \setminus \mathcal{C}_i \mid 1 \leq i \leq m \text{ and } \mathrm{O}_f(\mathcal{C}_i) = 1\}$;
  $\mathcal{S}_x \leftarrow$ `Set-intersection`$(\overline{\mathcal{F}}, x, 1)$;
  if $\exists i, \mathcal{S}_x = \{\{D_i\}\}$ **then**
    //negative targeting;
    $\mathcal{S}^{(core)} \leftarrow \{\langle \emptyset, \{D_i\}\rangle\}$;
  end
  else
    //null function;
    $\mathcal{S}^{(core)} \leftarrow \{\langle \emptyset, \emptyset\rangle\}$;
  end
end

Algorithm 2: PAC-learning under 1-juntas hypothesis.

Bound, the number of combinations queried that are in $\mathcal{S}^{(\text{in})}$ (within scope) is also an $\Theta(\log^{1/\alpha}(N/\varepsilon))$. By Assumption 2, this is the correct order of magnitude in order to ensure that, with probability $1-\Theta(\varepsilon)$, the oracle O_f will output 1 for at least $\Theta(\log(N/\varepsilon))$ queries.

In particular, let $\mathcal{F}$ be the set of all random combinations $\mathcal{C}_i$ such that $\mathrm{O}_f(\mathcal{C}_i) = 1$. By Assumption 3, the inputs on which the function f does not depend cannot affect the outcome, so, they are contained in half of the combinations of $\mathcal{F}$ in expectation. Furthermore, since these inputs are independently distributed and $|\mathcal{F}| = \Omega(\log(N/\varepsilon))$ is sufficiently large, we can prove by Chernoff bounds that these irrelevant inputs can be neither contained (nor absent) in a large fraction $x > 1/2$ of the combinations in $\mathcal{F}$, with high probability $1-\Theta(\varepsilon)$.

Conversely, it remains to prove that we can detect and identify the unique input on which the function f depends. We claim that for every $A_i \in \mathcal{F}$, $\mathbb{P}r[f(A_i) = 1 \mid \mathrm{O}_f(A_i) = 1, \mathrm{H}_{\mathcal{F}_{-i}}] > 1/(1+\varphi)$, with φ being the targeting lift. Indeed, since f is assumed to be a 1-junta, it is equally likely for a random combination to be in $\mathcal{S}^{(\text{in})}$ than to be out of $\mathcal{S}^{(\text{in})}$. By Assumption 1, combinations out of $\mathcal{S}^{(\text{in})}$ have φ less chances to be in $\mathcal{F}$ than those in $\mathcal{S}^{(\text{in})}$, and so, the claim follows. Then, by using the concentration inequalities of Lemma 86, we obtain that $|\mathcal{F} \cap \mathcal{S}^{(\text{in})}| \geq x|\mathcal{F}|$ with high probability $1-\Theta(\varepsilon)$, provided x is chosen such that $x < 1/(1+\varphi)$. In particular, if f only depends on some input $D_j \in \mathcal{D}$ then there is a large fraction x of the combinations in $\mathcal{F}$ such that either D_j is present (if f depends on the input positively) or absent (if f depends on the input negatively) in these combinations. □

Perspectives. Our work shows that single-input targeting can be *always* detected and identified, under some plausible assumptions. As I mentioned earlier, similar results have been proved under different assumptions, but up to the price of an exponentially larger query complexity [DTD15]. To derive a unifying model where similar results can be proved is, to my mind, an important issue.

5.4 Complex targeting: the case of monotonic functions

This section now addresses complex targeting, *i.e.*, when the targeting function depends on at least two inputs. This is joint work with Mathias Lécuyer, Max Tucker, Augustin Chaintreau and Roxana Geambasu.

It has been argued in [DTD15] that this more challenging case could be reduced to the simpler case of single-input targeting (Section 5.3). In particular, assuming that f strongly depends on some input, this relevant input may still be identified with the algorithms crafted for single-input targeting. However, there is no reason a priori why the targeting function should depend more on some input than on the others. In [DTC17], we describe an experiment where we show that in some cases where two inputs are *simultaneously* targeted (*e.g.*, "programming

interview" with "new job"), the corresponding output is misclassified by our prototype Xray [LDL+14] as being untargeted. We detail this a bit more in Figure 5.2. This result has motivated the study of PAC-learning algorithms with k-juntas as hypothesis, for $k > 1$.

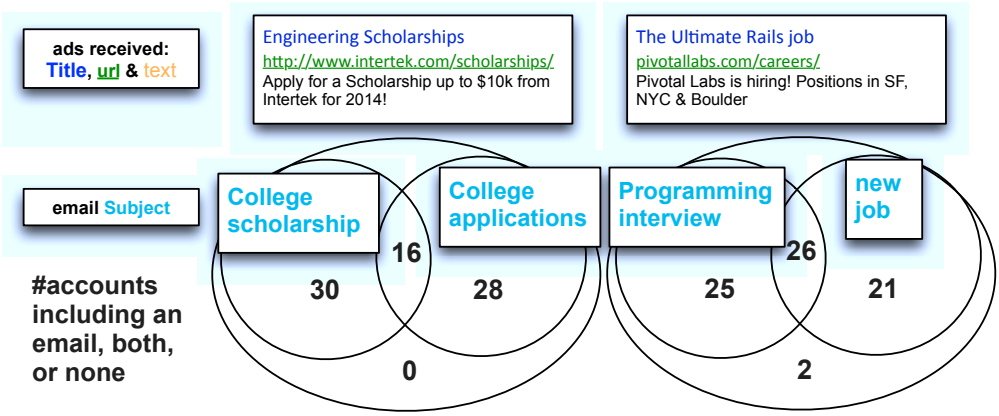


Figure 5.2: A correlation study between random placements of inputs (bottom) and outputs received (top).

More precisely, in this section we only consider *monotonic* k-juntas, that are the functions f such that $\mathcal{C} \subseteq \mathcal{C}'$ implies $f(\mathcal{C}) \leq f(\mathcal{C}')$ (see Section 5.2.2.1). By doing so, we do not pretend to cover *all* the cases of complex targetings that happen in real-life. Our purpose is to generalize our positive results on single-input detection to a broader set of targeting functions, and to investigate on the theoretical limitations of our set cover approach in Section 5.3. Furthermore, we note that in settings such as Facebook (until recently) [FBE], negative keywords are unavailable to the advertisers, and so, every targeting function should be monotonic.

In what follows, our results will be proved under one additional assumption on the oracle. Namely:

Assumption 4 (Nondiscrimination)**.** Let $\mathcal{F}$ be any family. For any $\mathcal{C}_1, \mathcal{C}'_1$ with $f(\mathcal{C}_1) = f(\mathcal{C}'_1) = 1$:

$$\mathbb{P}r[\mathrm{O}_f(A_i) = 1 \mid A_i = \mathcal{C}_1, \mathrm{H}_{\mathcal{F}_{-i}}] = \mathbb{P}r[\mathrm{O}_f(A_i) = 1 \mid A_i = \mathcal{C}'_1, \mathrm{H}_{\mathcal{F}_{-i}}].$$

Outline. In Section 5.4.1, we present a PAC-learning algorithm with monotonic k-juntas as hypothesis that generalizes our work on single-input targeting. This algorithm is proved to be correct only if some upper-bound on the targeting lift is assumed. Then, we describe more efficient algorithms in Section 5.4.2, but that are proved to be correct under stronger assumptions on the targeting lift. Further discussions on this work are given in the conclusion (Section 5.4.3).

5.4.1 Beyond single-input: the influence of the targeting lift

We first present an extended version of Theorem 84 that can be applied to monotonic k-juntas with additional assumptions. This result has been published in [DLCG15], and it is a joint work with Mathias Lécuyer, Augustin Chaintreau and Roxana Geambasu.

Theorem 87. *Let φ be the targeting lift (Assumption 1) and let $\alpha \leq 1$ be the polynomial-growth (Assumption 2). For every fixed positive integers s and w, there exists a constant $M_{s,w}$ such that the following holds if $\varphi < M_{s,w}$:*

There exists a PAC-learning algorithm such that, for every $\varepsilon > 0$, the targeting function can be learnt with probability $1 - \varepsilon$ under the hypothesis that it has a core with size at most s and order at most w. Furthermore, this algorithm runs in $\mathcal{O}(N^{s+w} \cdot \log^{1/\alpha}(N/\varepsilon))$-time and it has $2^{\mathcal{O}(s+w)} \cdot \log^{1/\alpha}(N/\varepsilon)$ query complexity.

Theorem 87 can be restated under monotonic k-juntas hypothesis by setting $s = w = k$. The main steps of its proof are now sketched. We describe a PAC-learning algorithm under monotonic juntas hypothesis and we prove its correctness.

Sketch of the algorithm. As for Theorem 84, we set the distribution of the sampler to be a binomial distribution $B(p, N)$, and then we make a polylogarithmic number of queries. However, note that here, the probability p will depend on the values for s and w.

The algorithm iterates on all the $\mathcal{O}(N^w)$ subsets $\mathcal{C}$ of size at most w, and it aims at deciding whether $f(\mathcal{C}) = 1$. In order to do so, the following key question is answered: what can be said among the supersets of $\mathcal{C}$ for which the oracle outputs 1 ? By definition, all of those subsets contain all inputs in $\mathcal{C}$ so we are interested in understanding how other inputs affect them. This is where exactly two cases emerge: Firstly, if we assume that $\mathcal{C}$ does contain a combination of the core, it automatically implies that independently of any other inputs, they all receive the ad with the same probability (by Assumption 4 on nondiscrimination). Secondly, if we assume on the other hand the opposite, then among all accounts including $\mathcal{C}$, there will be specific sets of inputs that may complete a combination from the core family and hence be targeted more heavily than others. This latter case resembles the situation of single-input targeting. The former case resembles a situation where f is untargeted (ads appear randomly). We can therefore design a new test as follows, that is sound and complete to determine in which case we are.

Roughly, the algorithms considers all the queried combinations $\mathcal{C}_i$ such that $\mathcal{C} \subseteq \mathcal{C}_i$ and $\mathrm{O}_f(\mathcal{C}_i) = 1$. Then, for all these combinations $\mathcal{C}_i$, the subset $\mathcal{C}_i \setminus \mathcal{C}$ is placed in some family $\Delta^{(ad)}(\mathcal{C})$. The test concludes that $f(\mathcal{C}) = 1$ if and only if $\Delta^{(ad)}(\mathcal{C})$ has no x_intersecting subset of size at most s, for some predefined choice of x. It can be verified in $\mathcal{O}(N^s \cdot \log^{1/\alpha}(N/\varepsilon))$-time by calling upon Algorithm 1. Furthermore, if a combination $\mathcal{C}$ passes the test and it is inclusion wise minimal w.r.t. this property then it is part of the core family of f with high probability.

In what follows (sections 5.4.1.1 and 5.4.1.2), we introduce the two propositions that are the cornerstone of our analysis for this above algorithm. We also explain in Section 5.4.1.3 why there is a need for assuming an upper-bound on the targeting lift in order to prove the correctness of this algorithm.

5.4.1.1 Soundness of the algorithm

First, we need to show that in any family of random subsets, almost asymptotically surely (a.a.s.) there can be no $x_$intersecting subset of small size (Lemma 88). The following technique is similar to the one used in [NRS04] in order to prove that a.a.s. the minimum size of a dominating set in any n-vertex random graph is $\Theta(\log n)$.

Lemma 88. *Let $1 > x > 0$, $s \in \mathbb{N}$, $p < 1-(1-x)^{\frac{1}{s}}$, and $\mathcal{B}$ a family of combinations that are drawn randomly from a binomial distribution $B(p, N)$. There exists $C > 0$ such that for any $\varepsilon > 0$ and polynomial P, if $m \geq C \cdot (s \ln(n) + \ln P(n) + \ln(1/\varepsilon))$ then with probability $(1 - \varepsilon/P(n))$ no $x\_$ intersecting subset of size s exists for this family.*

Proof. Let us consider an arbitrary combination $\mathcal{C} \subseteq \mathcal{D}$ of size s. We introduce Y the variable counting how many random subsets in $\mathcal{B}$ this combination $\mathcal{C}$ intersects, and we note that $\mathcal{C}$ is an $x_$intersecting subset exactly if $Y \geq xm$. We also observe that Y is a sum of binary independent variables and so, since the probability that $\mathcal{C}$ intersects an arbitrary subset in $\mathcal{B}$ is $1-(1-p)^s$, it has expectation $\mu = (1-(1-p)^s)\, m$. Assuming $p < 1-(1-x)^{\frac{1}{s}}$ as we do, μ is multiplicatively smaller than xm. Hence we can apply Chernoff Bound to conclude that $\mathbb{P}r[Y \geq xm] \leq \frac{\varepsilon}{P(N)N^s}$ when

$$m \geq C \cdot \ln\left(N^s P(N)/\varepsilon\right) \text{ with } C = \frac{3\,(1-(1-p)^s)}{\left(x-(1-(1-p)^s)\right)^2}.$$

Furthermore, since there are $\binom{N}{l} \leq N^s$ choices of $\mathcal{C}$, by the union bound the probability that at least one of them is an $x_$intersecting subset is at most $\frac{\varepsilon}{P(N)}$. □

By Lemma 88, the test routine of the algorithm will detect all the combinations $\mathcal{C}$ such that $f(\mathcal{C}) = 1$ with high probability. However, the latter result requires a predetermined lower-bound on the threshold parameter x (that depends on the probability p). In particular, the larger the size s of the core family of f, the larger the lower-bound on x.

5.4.1.2 Completeness of the algorithm

Second, we aim at proving that the test routine of the algorithm will reject all the combinations $\mathcal{C}$ such that $f(\mathcal{C}) = 0$ with high probability. This case is quite similar to single-input targeting, as it suffices to show that w.h.p., positive examples in $\Delta^{(ad)}(\mathcal{C})$ (defined above for the test) are likelier to be correctly classified than misclassified. Indeed, if this holds then taking one input in each combination of the core will leave an $1_$intersecting subset of size at most s for the subfamily

$\Delta^{(ad)}(\mathcal{C}) \cap \mathcal{S}^{(\text{in})}$ (correctly classified), that will be an $x_$intersecting subset for the whole family $\Delta^{(ad)}(\mathcal{C})$, for some well-chosen x.

The main difficulty is that it is not equally likely for a random combination to be within scope than out of scope. In particular, the larger the order w (size of a largest combination in the core), the lesser the probability for a random combination to be in $\mathcal{S}^{(\text{in})}$. So, we need to tune the probability p (used for the distribution Π of the sampler) in order to increase in turn the probability for a random account to be within scope. Formally, let us introduce the following function on this probability:

$$\hat{\varphi}^{(k)}_{s,w}(p) = \frac{1 - \frac{s}{k}(1-(1-p)^s)}{\frac{s}{k}(1-(1-p)^s)} \frac{(1-(1-p)^{s/k})^w}{1-(1-(1-p)^{s/k})^w} \tag{5.1}$$

Proposition 89. *Suppose that f has a core family of size at most s and order at most w, and that $\varphi \leq \hat{\varphi}^{(k)}_{s,w}(p)$ for some $k \leq s$. Then, there exist two positive constants x and C (independent of f) such that the following holds for any choice of $\varepsilon > 0$, polynomial P and combination $\mathcal{C}$:*

Let $\mathcal{B}$ be a family of m combinations that are drawn randomly from a binomial distribution $B(p, N)$. If $m \geq p^{-|\mathcal{C}|} \cdot C \cdot (\ln(N) + \ln P(N) + \ln(1/\varepsilon))$, then with probability $(1 - \varepsilon/P(N))$ exactly <u>one</u> of the following claims holds:

(i) $\mathcal{C}$ contains a core combination, i.e. it is in $\mathcal{S}^{(in)}$.

(ii) an $x\_$ intersecting subset of size k exists for

$$\Delta(\mathcal{C}) = \{\mathcal{S} \cap \overline{\mathcal{C}} \mid \mathcal{S} \in \mathcal{B}, \mathrm{O}_f(\mathcal{S}) = 1, \mathcal{C} \subseteq \mathcal{S}\}$$

5.4.1.3 Upper-bounds on the targeting lift

Altogether combined, Lemma 88 and Proposition 89 can be proved to be simultaneously correct only if x is chosen in some fixed interval, whose length depends on: the size s of the core, its order w, the targeting lift φ (Assumption 1) and the probability p that is used for the distribution Π of the sampler. Note that the probability p can be tuned in order to maximize the length of this interval, but even then, there will be an upper-limit (depending on φ, s, w) beyond which this interval will be empty. Let us express this limit as an upper-bound $M_{s,w}$ on the targeting lift (only depending on s and w).

Lemma 90. *Let $M_{s,w} = \sup_{p \in]0;1[} \hat{\varphi}^{(s)}_{s,w}(p)$, we have:*

$$\begin{cases} \text{if } s = 1, & M_{1,w} = 1/w, \\ \text{if } w = 1, & M_{s,1} = 1/s, \\ \text{for all } s, w, & \frac{1}{(2^{\max(s,w)}-1)^2} \leq M_{s,w} \leq \frac{1}{(2^{\min(s,w)}-1)^2}, \\ \text{for all } w, s, & M_{s,w} = M_{w,s}, \\ \text{if } s = w = n, & M_{n,n} = 1/(2^n-1)^2. \end{cases}$$

Moreover, if $w > 1$ and $s > 1$ then we have $M_{s,w} = \frac{(p^*)^w}{1-(p^*)^w} \frac{(1-p^*)^s}{1-(1-p^*)^s}$ where p^* is the only solution in $]0;1[$ of:

$$wp^{w+1} - s(1-p)^{s+1} - (w+s)p + w = 0.$$

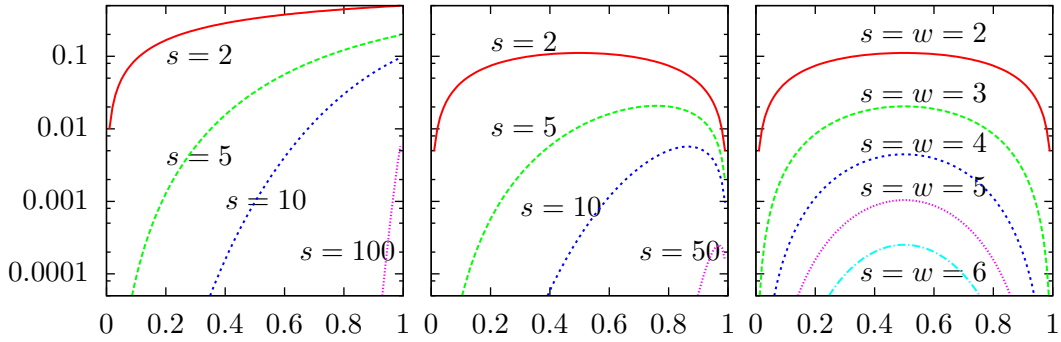


Figure 5.3: Value of $\hat{\varphi}^{(s)}_{s,w}$ as a function of $1-p$ for $w = 1$ (left), $w = 2$ (middle) and $w = s$ (right).

Discussion and perspectives. It is particularly informative to see how this *undetected targeting lift* (upper-bound on the targeting lift) grows with the complexity of the formula used. Figure 5.3 presents the value of the function $\hat{\varphi}^{(s)}_{s,w}$ that defines it (it is drawn up to a change of variable to make it easier to read). As proved in Lemma 90 and shown in the figure, if the targeting solely uses disjunction (resp. conjunction) of inputs, *i.e.*, $w = 1$ (resp., $s = 1$) as shown in Figure 5.3 (left), the undetected targeting lift behaves as $1/s$ and hence it is polynomial. This polynomial expansion remains if s grows while w remains small. Figure 5.3 (middle) presents an example. In contrast, when s and w grow simultaneously, *e.g.*, if they are equal as shown in Figure 5.3 (right) one can show that undetected targeting lift can be decreasing *exponential* fast. While this demonstrates the hardness of *Web's transparency*, we note that such forms of complex targeting combining so many inputs to decide may be relatively rare in practice.

5.4.2 Faster algorithms and tradeoffs

Theorem 87 proves that the set cover approach of Section 5.3 can be generalized for learning the monotonic k-juntas — but under some assumptions on the targeting lift. In this subsection, we present algorithms for this task that achieve a better running-time, but under stronger assumptions on the lift. This is joint work with Max Tucker and Augustin Chaintreau.

Namely, in Section 5.4.2.1 we replace the exact test of Algorithm 1 with a greedy approximation algorithm, thereby decreasing the running time from $\mathcal{O}(N^s \cdot \log^{1/\alpha}(N/\varepsilon))$ to $\mathcal{O}(sN \cdot \log^{1/\alpha}(N/\varepsilon))$. In Section 5.5.1, we introduce a new PAC-learning algorithm that is more intricate than the one presented for Theorem 87 but runs in $\tilde{\mathcal{O}}((sw)! \cdot N)$-time.

5.4.2.1 Faster detection algorithm

Our purpose is to describe a faster test of recognition for the combinations $\mathcal{C}$ such that $f(\mathcal{C}) = 1$.

Theorem 91. *Let φ be the targeting lift (Assumption 1) and let $\alpha \leq 1$ be the polynomial-growth (Assumption 2). For every fixed positive integers s and w, there exists a constant $\hat{M}_{s,w}$ such that the following holds if $\varphi < \hat{M}_{s,w}$:*

Suppose that f has a core family of size at most s and order at most w. Then, for every combination $\mathcal{C}$ and for every $\varepsilon > 0$, it can be decided in $2^{\mathcal{O}(|\mathcal{C}|+s+w)} \cdot N \cdot \log^{1/\alpha}(N/\varepsilon)$-time and with probability $1-\varepsilon$ whether $f(\mathcal{C}) = 1$.

Let us sketch the proof of Theorem 91. We recall that in order to decide whether $f(\mathcal{C}) = 1$ with high probability, it suffices to verify whether some family $\Delta(\mathcal{C})$ has an x_intersecting subset of size at most s, where x is a well-chosen parameter depending on s and w. It can be done in $\tilde{\mathcal{O}}(N^s)$-time by using Algorithm 1. The gist of Theorem 91 is to make this step faster by replacing the (exact) Set-intersection algorithm with a greedy approximation algorithm that we describe next.

Formally, let us set $\mathcal{S} = \Delta(\mathcal{C})$, $\mathcal{C}' = \emptyset$. While $\mathcal{S} \neq \emptyset$ and $|\mathcal{C}'| < s$, we pick any input D_j that maximizes the number of intersected subsets in $\mathcal{S}$. This input D_j is added in $\mathcal{C}'$, then every combination containing D_j is removed from $\mathcal{S}$. This process has already received some attention in the literature of Boolean function learning [FA05], but under a different learning model.

On the one hand, if the resulting combination $\mathcal{C}'$ is an x_intersecting subset then we can conclude, as before for Theorem 87, that $f(\mathcal{C}) = 0$ with high probability. On the other hand, if $\Delta(\mathcal{C})$ does admit an x_intersecting subset of this size then we can prove by using standard arguments on submodular functions that $\mathcal{C}'$ must be an $(1-(1-\frac{1}{s})^{1/s})x$_intersecting subset. It can be proved by Lemma 88 that no such a subset can exist if $f(\mathcal{C}) = 1$ and x is large enough. So overall, we are left to test whether the resulting combination $\mathcal{C}'$ is an x'_intersecting subset with $x' = (1-(1-\frac{1}{s})^{1/s})x$.

As a direct consequence of Theorem 91, we obtain a faster algorithm than the one presented for Theorem 87:

Corollary 92. *Let φ be the targeting lift (Assumption 1) and let $\alpha \leq 1$ be the polynomial-growth (Assumption 2). For every fixed positive integers s and w, there exists a constant $\hat{M}_{s,w}$ such that the following holds if $\varphi < \hat{M}_{s,w}$:*

There exists a PAC-learning algorithm such that, for every $\varepsilon > 0$, the targeting function can be learnt with probability $1-\varepsilon$ under the hypothesis that it has a core with size at most s and order at most w. Furthermore, this algorithm runs in $\mathcal{O}(2^s \cdot N^w \cdot \log^{1/\alpha}(N/\varepsilon))$-time and it has $2^{\mathcal{O}(s+w)} \cdot \log^{1/\alpha}(N/\varepsilon)$ query complexity.

Let us point out that this new algorithm is Fixed-Parameter Tractable in the size s of the core family, but it still depends on its order w exponentially.

5.4.2.2 Faster identification algorithm

We have shown in Section 5.4.2.1 how to improve the detection test in order to recognize the combinations of $\mathcal{S}^{(\text{in})}$. Getting rid of the exhaustive search of all possible subsets of small size (at most the order w of the core) is much more difficult.

We show how to do so by using more properties of the intersecting subsets and a significantly more elaborate algorithm, that has similarities with the one described in [Ang88, Sec. 3.1, Theorem 1].

Theorem 93. *Let φ be the targeting lift (Assumption 1) and let $\alpha \leq 1$ be the polynomial-growth (Assumption 2). For every fixed positive integers s and w, there exists a constant $\overline{M}_{s,w}$ such that the following holds if $\varphi < \overline{M}_{s,w}$:*

There exists a PAC-learning algorithm such that, for every $\varepsilon > 0$, the targeting function can be learnt with probability $1-\varepsilon$ under the hypothesis that it has a core with size at most s and order at most w. Furthermore, this algorithm has $2^{\mathcal{O}(sw\cdot\log(sw))} \cdot \log^{1/\alpha}(N/\varepsilon)$ query complexity, and with probability $1-\varepsilon$ it runs in $2^{\mathcal{O}(sw\cdot\log(sw))} \cdot N \cdot \log^{1/\alpha}(N/\varepsilon)$-time.

Input: a family $\mathcal{F}$; and a threshold parameter x.
Output: the representation $\mathcal{S}^{(core)}$ of f.

$\mathcal{S}^{(core)} \leftarrow \{\}$;
$\mathcal{S}_x \leftarrow$ `Set-intersection`$(\mathcal{F}, x, 1)$; $\mathcal{S} \leftarrow \bigcup_{\{D_i\}\in\mathcal{S}_x} \{D_i\}$;
if $\mathcal{S} \neq \emptyset$ **then**
  /* *Removal of inputs until obtaining a core combination* */;
  foreach $D_i \in \mathcal{S}$ **do**
    $\hat{\mathcal{S}} = \mathcal{S} \setminus D_i$;
    $\hat{\mathcal{F}} = \{A_j \setminus \hat{\mathcal{S}} \mid A_j \in \mathcal{F}, \hat{\mathcal{S}} \subseteq A_j\}$;
    $\hat{\mathcal{S}}_x \leftarrow$ `Set-intersection`$(\hat{\mathcal{F}}, x, 1)$;
    if $\hat{\mathcal{S}}_x = \emptyset$ **then**
      /* *the subset $\hat{\mathcal{S}}$ is still in $\mathcal{S}^{(in)}$* */;
      $\mathcal{S} \leftarrow \hat{\mathcal{S}}$;
    end
  end
  $\mathcal{S}^{(core)} \leftarrow \mathcal{S}^{(core)} \cup \{\mathcal{S}\}$;
  foreach $D_i \in \mathcal{S}$ **do**
    /* *Recursive call to the algorithm* */;
    $\mathcal{S}_i \leftarrow$ `GSI`$(\mathcal{F} \setminus \{A_j \in \mathcal{F} \mid D_i \in A_j\}, x)$;
    $\mathcal{S}^{(core)} \leftarrow \mathcal{S}^{(core)} \cup \mathcal{S}_i$;
  end
end

Algorithm 3: Generalized set-intersection algorithm (GSI).

The proof of Theorem 93 is based on Algorithm 3, that is an intricate variation of the Set-intersection algorithm. Let us first give a better analysis of the detection test that is used in order to recognize the combinations of $\mathcal{S}^{(\text{in})}$ (presented earlier in Section 5.4.1). Given a combination $\mathcal{C}$, this test either asserts that $f(\mathcal{C}) = 1$ or it outputs an x_intersecting subset of size at most s for some family of subsets $\Delta(\mathcal{C})$.

The latter subset can be regarded as a "negative certificate" on which we can extract more information as follows:

Lemma 94. *Suppose that f has a core family of size at most s and order at most w, and let $k \leq s$. Then, provided $\varphi < \overline{M_{s,w}}^{(k)}$ for some constant $\overline{M_{s,w}}^{(k)}$ (only depending on s, k and w), there exist positive constants x (threshold), p (probability for the sampler) and C such that the following holds for any choice of $\varepsilon > 0$, polynomial P and combination $\mathcal{C}$:*

Let $\mathcal{B}$ be a family of m combinations that are drawn randomly from a binomial distribution $B(p, N)$, with $m \geq \alpha^{-|\mathcal{C}|} \cdot C \cdot (\ln(N) + \ln P(N) + \ln(1/\varepsilon))$. The two following claims hold for $S_{x,k} = \{\mathcal{S} \mid \mathcal{S}\ x\_\text{intersecting for } \Delta(\mathcal{C}), |\mathcal{S}| \leq k\}$ with probability $(1 - \varepsilon/P(N))$:

(i) All combinations in $S_{x,k}$ intersect $\bigcup_{\mathcal{S} \in \mathcal{S}^{(core)}} \mathcal{S}$.

(ii) $\mathcal{C} \cup \bigcup_{\mathcal{S} \in S_{x,k}} \mathcal{S}$ is empty or contains a core combination.

From now on, this lemma will be used with $k = 1$.

Let us sketch the main principles behind the algorithm. As before, we set the distribution Π for the sampler to be a binomial distribution $B(p, N)$, for some predetermined value of p. Then, we make a polylogarithmic number of queries to the sampler, and we let $\mathcal{F}$ to be the set of all the combinations $\mathcal{C}_i$ queried such that $\mathrm{O}_f(\mathcal{C}_i) = 1$. If f is not the null-function then under the conditions of Theorem 93, this family $\mathcal{F}$ has $x_$intersecting subsets of size 1. Furthermore, by calling upon Algorithm 1 (with $s = 1$), all such intersecting subsets of unit size can be computed, in quasi-linear time.

At this step, Lemma 94 comes into play. Indeed, let $\mathcal{S}$ be the union of all $x_$intersecting subsets of $\mathcal{F}$ of unit size. By Lemma 94(i), every input in $\mathcal{S}$ is in a core combination, hence $|\mathcal{S}| \leq sw$. In addition, we have by Lemma 94(ii) that $\mathcal{S}$ contains a core combination with high probability. The main idea behind Theorem 93 is to extract a core combination $\mathcal{C}$ from $\mathcal{S}$, then to call Algorithm 3 recursively on a constant number of subsets of $\mathcal{F}$ in order to obtain the remaining of the core family. More precisely, there is one recursive call for every input $D_i \in \mathcal{C}$, before which we remove from $\mathcal{F}$ all the combinations that contain D_i in order to obtain different core combinations than $\mathcal{C}$. The main difficulty is to bound the depth of the recursion. This is where we use once more Lemma 94. Indeed, since at each call to Algorithm 3, the superset $\mathcal{S}$ has all its inputs contained in a core combination, and we virtually remove one of them before each recursive call, the depth of the recursion is bounded w.h.p. by $\sum_{\mathcal{C} \in \mathcal{S}^{(core)}} |\mathcal{C}| \leq sw$.

5.4.3 Conclusion and open perspectives

We have generalized the results obtained for single-input targeting to a larger class of targeting functions. A main drawback of this set cover approach, when applied to complex targeting, is that it can be proved to be correct only if the targeting lift is close to zero (the larger the size and the order of the core, the closer the lift to zero). So far, our learning model does not make any assumption on what a "realistic" value

for the lift should be. We aim at closing this gap in a near future by using some advertising models in the literature [GEC+13] in order to better evaluate the order of magnitude for this value.

On a more positive side, our approach can be modified in order to give efficient algorithms with quasi-linear time. This is a neat advantage for Web transparency tools such as Xray that, for now, do not cope with complex targeting. To the best of our knowledge, the only prototype which goes beyond the case of single-input targeting is Sunlight [LSS+15], where a different subclass of targeting functions is adressed. Roughly, the core algorithm of this tool is able to detect and to identify a class of "threshold functions" f, where each input is assigned a weight and f outputs 1 only if the sum of the weights of the inputs in presence is greater than some predetermined threshold.

Finally, let us point out that generalizing our approach in this section to *non-monotonic* functions is challenging, at best. Indeed, we recall that our detection test concludes that a combination $\mathcal{C}$ is in $\mathcal{S}^{(\text{in})}$ if and only if there is no x_intersecting subset of small size in a given family $\Delta(\mathcal{C})$. This test fails if the targeting function is non-monotonic. As an example, consider a combination $\mathcal{C}$ such that for every prime implicant $\langle \mathcal{C}_{in}, \mathcal{C}_{out} \rangle$ of f, we have $\mathcal{C} \cap \mathcal{C}_{out} \neq \emptyset$. By construction, there can be no superset of $\mathcal{C}$ that is in $\mathcal{S}^{(\text{in})}$. Hence, it may be the case that all these supersets are equally likely to be targeted by mistake, with all the inputs not in $\mathcal{C}$ being present by mere chance. The latter would imply that the family $\Delta(\mathcal{C})$ that is used for the test would not have any x_intersecting subset of small size, and so, that $\mathcal{C}$ is mistakenly identified as part of $\mathcal{S}^{(\text{in})}$.

5.5 General case

Finally, this section is about the theoretical limitations of the learning model of Section 5.2. That is, we aim at characterizing what can be learnt in our model. Full proofs can be found in [CD17], which is joint work with Augustin Chaintreau.

Outline. On the positive side, we prove in Section 5.5.1 that all the relevant inputs (on which the targeting depends) can be computed. Furthermore, under one additional assumption on the oracle (generalizing Assumption 4 on nondiscrimination), *any* targeting function can be learnt. However in general (without an additional assumption), it is proved in Section 5.5.3 that monotonic 2-juntas cannot be learnt in our setting. In fact, it is already impossible to distinguish between a conjunction or a disjunction!

Although the proofs in Sections 5.5.1 and 5.5.2 are algorithmic, they do not lead to efficient PAC-learning algorithms.

5.5.1 Identification of the relevant inputs

This subsection presents an algorithm for computing the at most k inputs on which a k-junta depends. Roughly, the relevant inputs will be inferred by virtually "fixing"

$k-1$ inputs from the ground-set $\mathcal{D}$. Such removal will reduce the problem to single-input targeting, and so, the set cover approach of Section 5.3 can be used. This is formalized with the following Algorithm 4.

Input: accuracy parameter ε; upper-bound k on the number of relevant inputs.
Output: the set of relevant inputs V.

$V \leftarrow \{\}$;
/* *Parameters tuning* */;
Let $x \in]\frac{1}{2}; \frac{1}{1+\varphi}[$;
Let $m \in \Omega\left(2^k \cdot \log^{1/\alpha}(N/\varepsilon)\right)$ /**m depends on x* */;
/* *Uniform sampling* */;
Draw $\langle \mathcal{C}_1, \mathrm{O}_f(\mathcal{C}_1)\rangle, \langle \mathcal{C}_2, \mathrm{O}_f(\mathcal{C}_2)\rangle, \ldots, \langle \mathcal{C}_m, \mathrm{O}_f(\mathcal{C}_m)\rangle$ with $\Pi = B(1/2, N)$;
$\mathcal{F} \leftarrow \{\mathcal{C}_i \mid 1 \leq i \leq m \text{ and } \mathrm{O}_f(\mathcal{C}_i) = 1\}$;
/* *Exhaustive search for prime implicants* */;
foreach $\langle \mathcal{C}_{in}, \mathcal{C}_{out}\rangle$ **with** $|\mathcal{C}_{in}| + |\mathcal{C}_{out}| \leq k-1$ **do**
  $\hat{\mathcal{F}} \leftarrow \{A_p \in \mathcal{F} \mid \mathcal{C}_{in} \subseteq A_p \text{ and } A_p \cap \mathcal{C}_{out} = \emptyset\}$;
  $\overline{\hat{\mathcal{F}}} \leftarrow \{\mathcal{D} \setminus A_p \mid A_p \in \hat{\mathcal{F}}\}$;
  if $|\hat{\mathcal{F}}| \geq \Omega\left(k \cdot \log(N/\varepsilon)\right)$ **then**
    $\mathcal{S}_x \leftarrow$ `Set-intersection`$(\hat{\mathcal{F}}, 1, x)$ // *positive dependency*;
    $\overline{\mathcal{S}}_x \leftarrow$ `Set-intersection`$(\overline{\hat{\mathcal{F}}}, 1, x)$ // *negative dependency*;
    $\hat{S} \leftarrow \left(\bigcup_{\{D_i\} \in \mathcal{S}_x \cup \overline{\mathcal{S}}_x} \{D_i\}\right) \setminus \mathcal{C}_{in}$;
    $V \leftarrow V \cup \hat{S}$;
  end
end

Algorithm 4: Inference algorithm for the relevant inputs.

Theorem 95. *Let $\alpha \leq 1$ be the polynomial-growth (Assumption 2). There is an algorithm such that, for every $\varepsilon > 0$, the set of relevant variables $V = \{D_i \in \mathcal{D} \mid f$ depends on $D_i\}$ can be learnt with probability $1-\varepsilon$ under k-juntas hypothesis. This algorithm runs in $\mathcal{O}(N^k \cdot \log^{1/\alpha}(N/\varepsilon))$-time and it has $\mathcal{O}(2^k \cdot \log^{1/\alpha}(N/\varepsilon)))$ complexity query.*

Sketch Proof of Theorem 95. We give a correctness proof of Algorithm 4. Let $\langle \mathcal{C}_{in}, \mathcal{C}_{out}\rangle$ be fixed, with $|\mathcal{C}_{in}| + |\mathcal{C}_{out}| \leq k-1$. Let us define $\hat{\mathcal{B}}$ as the family of all the combinations that have been queried with the sampler, that contain $\mathcal{C}_{in}$ and that do not intersect $\mathcal{C}_{out}$. Furthermore, let $\hat{\mathcal{D}} = \mathcal{D} \setminus (\mathcal{C}_{in} \cup \mathcal{C}_{out})$, let $\hat{f}(\hat{\mathcal{C}}) = f(\hat{\mathcal{C}} \cup \mathcal{C}_{in})$ and $\hat{\mathrm{O}}_f(\hat{\mathcal{C}}) = \mathrm{O}_f(\hat{\mathcal{C}} \cup \mathcal{C}_{in})$ for every combination $\hat{\mathcal{C}} \subseteq \hat{\mathcal{D}}$. In order to reuse the results from

Section 5.3, we will base on the property that $\hat{\mathrm{O}}_f$ "almost" behaves like an oracle for the targeting function $\hat{f}$. That is, it satisfies Assumptions 1 and 2 (trivially), but it only satisfies Assumption 3 partially. More precisely, if f (but not necessarily $\hat{f}$) only depends on some inputs in $\hat{V} \cup \mathcal{C}_{in} \cup \mathcal{C}_{out}$, then Assumption 3 applies for $\hat{V}$. By Lemma 88 (nonexistence of x_intersecting subsets of small size in random families), this weaker version of Assumption 3 implies that the targeting function f depends on every input that is computed by Algorithm 4 with high probability.

In order to complete the proof of the theorem, let D_j be any input on which f depends. Since D_j is relevant, there is a bipartition $\langle \mathcal{C}_{in}, \mathcal{C}_{out} \rangle$ of the relevant inputs from $\mathcal{D} \setminus D_j$ so that $f(\mathcal{C}_{in} \cup \{D_j\}) \neq f(\mathcal{C}_{in})$. So, let us fix any such bipartition $\langle \mathcal{C}_{in}, \mathcal{C}_{out} \rangle$. In such case, $\hat{f}$ is a 1-junta that only depends on D_j, furthermore $\hat{\mathrm{O}}_f$ satisfies Assumption 3 for $\hat{f}$. The average size of $\hat{\mathcal{B}}$ is $m/2^{k-1}$, where m is the number of queries. This case is thus reduced to single-input targeting (Theorem 84). Finally, by taking a union bound over the relevant inputs, every input on which f depends is in V (computed by Algorithm 4) with high probability. □

5.5.2 Filtering technique

In this subsection we now present an algorithm for learning the targeting function f exactly. Suppose that we are given the relevant inputs for the targeting function f. In order to learn $\mathcal{S}^{(\mathrm{in})}$, it suffices to learn all the subsets $\mathcal{C}$ on these (at most k) inputs so that $f(\mathcal{C}) = 1$. Intuitively, this can be achieved by comparing any two combinations $\mathcal{C}_0, \mathcal{C}_1$ and testing whether containing one of these two subsets, say, $\mathcal{C}_1$, increases the chance to be targeted (compared to $\mathcal{C}_0$). On may expect that the latter certifies $f(\mathcal{C}_1) = 1$ and $f(\mathcal{C}_0) = 0$. Algorithm 5 (introduced next) builds upon this intuition.

Input: a set of inputs V; a family $\mathcal{F}$; a threshold parameter t.
Output: the class $\mathcal{T}_k$ of all bipartitions $\langle \mathcal{C}_{in}, \mathcal{C}_{out} \rangle$ of V s.t. $\langle \mathcal{C}_{in}, \mathcal{C}_{out} \rangle \in \mathcal{S}^{(\mathrm{in})}$.

$k \leftarrow |V|$;

/ Partition of the family w.r.t. the relevant inputs */*;
Partition $\mathcal{F}$ into $\mathcal{F}_1, \mathcal{F}_2, \ldots, \mathcal{F}_{2^k}$ s.t.:
- $\forall 1 \leq i < 2^k,\ |\mathcal{F}_i| \geq |\mathcal{F}_{i+1}|$;
- $\forall A_p, A_q \in \mathcal{F},\ A_p \cap V = A_q \cap V \iff A_p, A_q \in \mathcal{F}_i$ for some i;

/ Ordering of the bipartitions of V by decreasing presence in the family */*;
for $i \in \{1, \ldots, 2^k\}$ **do**
  $V_i \leftarrow A_p \cap V$ with $A_p \in \mathcal{F}_i$;
end

/ Identification of the targeting lift */*;
$i_{\lim} \leftarrow \min\{1 \leq i \leq 2^k \mid |\mathcal{F}_i| \geq t \cdot |\mathcal{F}_{i+1}|\}$;
$\mathcal{T}_k \leftarrow \{\langle V_i, V \setminus V_i \rangle \mid 1 \leq i \leq i_{\lim}\}$;

Algorithm 5: Recognition algorithm for the targeting function.

However, it turns out that subtle complications occur which may lead our approch in this section to failure. The reason is that we obtain an ordering over all the bipartitions of the relevant inputs, that somewhat represents the combinations of these inputs by nonincreasing importance, but we have no clue on where the non-targeted combinations should start in this ordering. So, intuitively, the targeting function f can be learnt only if the *targeting lift* can be detected, for the latter delineates the border between combinations within scope and those out of scope.

We next introduce a new parameter on the oracle that will be used in order to prove correctness of our approach under some additional assumptions.

Definition 96. The oracle has *positive variance* ψ if for any family $\mathcal{F} = \langle A_1, \ldots, A_t \rangle$ the following holds for any $\mathcal{C}_1, \mathcal{C}_1' \in \mathcal{S}^{(\text{in})}$:

$$\mathbb{P}r[\mathrm{O}_f(A_i) = 1 \mid A_i = \mathcal{C}_1, \mathrm{H}_{\mathcal{F}_{-i}}] \geq \psi \cdot \mathbb{P}r[\mathrm{O}_f(A_i) = 1 \mid A_i = \mathcal{C}_1', \mathrm{H}_{\mathcal{F}_{-i}})].$$

Assumption 4 (introduced in Section 5.4 corresponds to the extremal case where the oracle has positive variance $\psi = 1$. Roughly, a large positive variance implies that there cannot exist a combination within scope that is significantly more targeted than the other combinations of $\mathcal{S}^{(\text{in})}$. If so then the targeting lift can be detected using a simple leftmost approach (see Algorithm 5).

Theorem 97. *Let φ be the targeting lift (Assumption 1) and let $\alpha \leq 1$ be the polynomial-growth (Assumption 2). The following holds if the oracle has positive variance $\psi > \varphi$:*

There exists a PAC-learning algorithm such that, for every $k \geq 1$ and for every $\varepsilon > 0$, the targeting function can be learnt with probability $1 - \varepsilon$ under the k-juntas hypothesis. Furthermore, this algorithm runs in $\mathcal{O}(N^k \cdot \log^{1/\alpha}(N/\varepsilon))$-time and it has $\mathcal{O}(2^k \cdot \log^{1/\alpha}(N/\varepsilon))$ query complexity.

Let us point out that even when there is no classification noise ($\mathrm{O}_f = f$), the best known PAC-learning algorithm under the k-juntas hypothesis has time complexity $N^{\mathcal{O}(k)}$ [MOS04]. Therefore, improving upon this time complexity will probably require additional assumptions.

5.5.3 Impossibility results

We end up this section with a proof that not all targeting functions can be learnt with respect to our learning model.

Proposition 98. *It is impossible to learn the targeting function f in general. In particular, there is a given monotone 2-junta that cannot be learnt even if the targeting lift is arbitrarily small.*

Proof. In order to prove the result, we will construct an oracle O_f that satisfies Assumptions 1, 2 and 3 for two distinct targeting functions. The latter is enough to prove the proposition since in such case, O_f could be used in our model for any

of the two functions, and so, these cannot be distinguished with high probability. More precisely, fix $0 < p_0 < 1/5$. Let us define O_f such that for any combination $\mathcal{C}$:

$$\mathbb{P}r[\mathrm{O}_f(\mathcal{C}) = 1] = p_0 \cdot (1 + 2 \cdot \mathbb{I}_{\{D_1 \in \mathcal{C}\}} + 2 \cdot \mathbb{I}_{\{D_2 \in \mathcal{C}\}}).$$

Note that for any combination $\mathcal{C}$, we have $\mathbb{P}r[\mathrm{O}_f(\mathcal{C}) = 1] \leq 5p_0 < 1$.

Since every combination has positive probability to be targeted and the above oracle considers the combinations independently, by Chernoff bound, O_f satisfies Assumption 2 with $\alpha = 1$ for any targeting function[3]. Furthermore, O_f satisfies Assumption 3 for any targeting function that only depends on the two inputs D_1, D_2. In particular, let $f_1(\mathcal{C}) = \max\{\mathbb{I}_{\{D_1 \in \mathcal{C}\}}, \mathbb{I}_{\{D_2 \in \mathcal{C}\}}\}$ (disjunction) and let $f_2(\mathcal{C}) = \mathbb{I}_{\{D_1 \in \mathcal{C}\}} \cdot \mathbb{I}_{\{D_2 \in \mathcal{C}\}}$ (conjunction). These two functions are monotone. In fact, they are linear combinations since $f_1(\mathcal{C}) = 1 \iff \mathbb{I}_{\{D_1 \in \mathcal{C}\}} + \mathbb{I}_{\{D_2 \in \mathcal{C}\}} \geq 1$, and similarly $f_1(\mathcal{C}) = 1 \iff \mathbb{I}_{\{D_1 \in \mathcal{C}\}} + \mathbb{I}_{\{D_2 \in \mathcal{C}\}} \geq 2$. For both functions, the oracle O_f satisfies Assumption 1 with any targeting lift $\varphi > 1/2$.

Finally, note that in order to extend this negative result to lifts arbitrarily smaller than $1/2$, one may just consider a slightly more complex oracle for the above two functions f_1, f_2, namely:

$$\mathbb{P}r[\mathrm{O}_f(\mathcal{C}) = 1] = q_0^{3 - \mathbb{I}_{\{D_1 \in \mathcal{C}\}} - \mathbb{I}_{\{D_2 \in \mathcal{C}\}}},$$

for some nonzero probability q_0 that can be taken arbitrarily small. □

Let us emphasize that the negative result of Proposition 98 already holds for *monotonic* and *threshold* functions (see Section 5.4.3). In particular, it applies to the theory behind the Web's transparency tools XRay [LDL+14] and Sunlight [LSS+15].

5.6 Conclusion

We have proved in this chapter that the theory behind two recent Web's transparency tools – namely, Xray [LDL+14] and Sunlight [LSS+15] – can be applied to cope with complex targeting, but that it requires stronger assumptions on the oracle. When these assumptions do not hold, we show that not all targeting functions can be learnt.

It is equally likely that not all (learnable) functions can be learnt efficiently. Indeed, even in a simpler learning model where there is no classification noise, learning k-juntas takes $N^{\mathcal{O}(k)}$-time [MOS04, Val12]. Furthermore, given a set of positive and negative examples, learning a function that is compatible with these examples under k-juntas hypothesis is $W[2]$-hard [AKL07]. The same negative result holds under monotonic k-juntas hypothesis [AKL07], which makes the existence of $N^{o(k)}$-time PAC-learning algorithms unlikely. I believe that the existence or

[3]Note that we must also choose the same constants β and γ. Here, the two functions considered are such that the scope of one is contained into the scope of the other. So, we can choose β, γ w.r.t. the function with smallest scope.

nonexistence of quasi-linear time algorithms in our setting is strongly related to the value of the targeting lift — as supported by the results of Section 5.4.

For this reason, I would find it interesting to mix up our learning model with some advertising models of the literature (*e.g.*, [GEC+13]) in order to fix some "plausible" estimate for the lift. This project is part of my on-going work.

Another interesting problem would be to enhance our learning model by including a natural graph structure between the accounts. Namely, as we stated in Section 5.2.4, the online accounts of users can be seen as the vertices of a random intersection graph [KSSC99], where an edge represents two accounts sharing a certain number of inputs. The influence of these edges on the outcome is largely ignored by our model, but it has received some attention in [LMT17]. Combining our model with the approach proposed in [LMT17] is thus an important issue.

Finally, on the practical point of view, one hidden drawback of the algorithms that are proposed in this chapter is that each query to the sampler represents, in real-life, a fake account which needs to be created and maintained [LDL+14]. This task is hard to automate, because online accounts have to respect a policy and they can be quickly closed if they don't. Ideally, we would like to find a different implementation of our algorithms that would require fewer account creations. For instance, could we avoid creating fake accounts by making some existing accounts collaborate (with each of them representing one query to the sampler) ? This simple idea would raise privacy concerns, that would require a computational mechanism design in order to handle with the communications between two accounts.

Overall, new guidelines than our "Set Cover approach" should be investigated, such as for instance the use of different oracles (that would represent accounts on different online platforms) [AM10]. Our work may also benefit from a bio inspired algorithm named Ant-miner [HLC07, PLF02] whose objective is to uncover classification rules from a dataset.

Chapter 6

Conclusion

Contents

As networks more and more impact our lives, the world is turned to be ruled by algorithms. This situation has motivated the need for more efficiency in algorithmic and more transparency in the use of algorithms. In this thesis, we have adressed these two issues by studying metric tree-likeness in graphs – related to the way the information flows in complex networks – and a collection of privacy oriented problems. We provide a finer-grained analysis for the complexity of all the problems studied, so as to question their scalability.

The contributions of my thesis are summarized in Section 6.1, with an emphasis on future work.

6.1 Open perspectives

We summarize our results in Chapters 2–5 and raise interesting questions for future work. In what follows, we borrow from the concluding sections of these different chapters.

First, we have given a survey on graph hyperbolicity in Chapter 2 where known lower and upper bounds are collected and the best known results on the complexity for computing this parameter are covered. In particular, we have introduced a general framework in order to sharply estimate the distortion of hyperbolicity that may be caused by various graph operations and so, conversely, by various graph decompositions. We also have proved new sufficient conditions in order to lower or upper-bound the hyperbolicity in some graph classes that are used for the design of data center interconnection networks. We expect our techniques to apply to even more graph classes, that is left as an interesting open question.

Furthermore, on the complexity point of view, we have proposed a preprocessing method that is based on clique-decomposition. Interestingly, the core arguments that are used for our analysis of the preprocessing can be applied to any other "tree-like" decomposition: where the subgraphs are the bags of a tree decomposition with bounded-diameter adhesion sets. Now, the question is whether they can be applied to more general decompositions, say where the subgraphs are the sets of some family with a "tree-representation" [BXHR12]? If so, then the latter result would subsume

all known results on the preservation of hyperbolicity under modular, split and clique decomposition. Another interesting open question is whether the recognition of graphs with "large" hyperbolicity can be done faster than computing this parameter for general graphs. Indeed, we recall that all the hardness results proved for this parameter have been obtained for graphs with small hyperbolicity (bounded by a constant). This is further supported by the experiments presented in [CCL15, BCC15]: where the practical running-time is dominated by the computation of the all-pairs shortest-paths except for some hard instances that have been observed to have a small hyperbolicity.

Chapter 3 is devoted to an in-depth (complexity) study of some tree decompositions in graphs where the bags must satisfy metric constraints. More precisely, we have proved that computing the clique-decomposition is computationally equivalent to Triangle Detection under the standard assumption that the latter problem is equivalent to Matrix Multiplication. We have also proved that computing the parameters treebreadth, pathbreadth and pathlength (recently introduced in [DK14, DKL14]) is NP-hard and not FPT. However, on a more positive side, the clique-decomposition of planar graphs and bipartite graphs can be computed in linear time, and in the same way it can be decided in polynomial time whether a given bipartite graph or planar graph has treebreadth one.

In particular, let us point out that all our hardness results have been obtained for classes of graphs with a large clique or clique-minor, whereas all our positive results have been proved for classes of graphs with bounded clique-number or clique-minor. Therefore, it would be interesting to find new results (positive or negative) that could better clarify the role of the clique-number and the Hadwiger number (size of a largest clique-minor) in the parameterized complexity of the tree decomposition problems studied in this chapter. In this respect, we might be helped by a central result in Chapter 3: stating that treewidth and treelength can only differ by at most a constant-factor in the classes of graphs with bounded-length isometric cycles and bounded genus.

Furthermore, whereas our tools in Part I have been mainly graph-theoretical, we have used a more diverse toolkit in Part II based on combinatorics, game theory and learning theory. This diversification of our techniques has conduced to open questions of a different nature, that we will expose next.

In particular, we have studied coloring games in Chapter 4, exhibiting new results on the complexity of computing strong Nash equilibria in these games and in some of their variations. More precisely, we have proved that a k-strong Nash equilibrium can be computed with better-response dynamics for every fixed $k \geq 1$, however the dynamics do not converge in polynomial time as soon as $k \geq 4$. So, it may be the case that computing a k-strong Nash equilibrium in coloring games is PLS-complete for some fixed $k \geq 4$. Such a result would be interesting because coloring games are played on an *unweighted* graph whereas the classical PLS-complete problems are "weighted", *i.e.*, they can be solved in quasi-polynomial time with respect to some set of input weights.

Note that as a way to deepen our understanding of coloring games, we have proved that computing a Nash equilibrium in these games is PTIME-hard. It has been proved that computing a Nash equilibrium in generalized coloring games (played on an edge-weighted graph) is PLS-complete. Can we prove that conversely, if a "weighted" game is PLS-hard then any corresponding "unweighted" game (where all the input weights are bounded) is PTIME-hard ? Such a result would make advance our understanding of the complexity of search problems.

Finally, we have studied in Chapter 5 the problem of learning a Boolean function that only depends on a fixed number of variables, under new hypotheses on the classification noise. Our problem is motivated by online advertising in the Internet (we aim at unveiling data misuse), and so, it has a natural graph structure. However, this structure has not been exploited in our algorithms. Hence, an important issue would be to better account for this underlying graph.

As an example, is this graph hyperbolic ? If it were the case, could we take advantage of this fact in order to obtain efficient approximation algorithms for Set Cover, that could be used in our approach for learning the function ?

Bibliography

[AAD16] M. Abu-Ata and F. F. Dragan. Metric tree-like structures in real-world networks: an empirical study. *Networks*, 67(1):49–68, 2016. (Cited in pages 24, 36, 37, 40, 58, 73, 74, 77, 22, 34, 35, 38, 56, 71, 72 and 75.)

[ABC+91] J. M. Alonso, T. Brady, D. Cooper, V. Ferlini, M. Lustig, M. Mihalik, M. Shapiro, and H. Short. Notes on word hyperbolic groups. In *Group theory from a geometrical viewpoint*. Singapore: World Scientific, 1991. (Cited in pages 15, 31, 34, 13, 29 and 32.)

[ABF+98] R. Agarwala, V. Bafna, M. Farach, M. Paterson, and M. Thorup. On the approximability of numerical taxonomy (fitting distances by tree metrics). *SIAM Journal on Computing*, 28(3):1073–1085, 1998. (Cited in pages 15 and 13.)

[ABK+07] I. Abraham, M. Balakrishnan, F. Kuhn, D. Malkhi, V. Ramasubramanian, and K. Talwar. Reconstructing approximate tree metrics. In *Proceedings of the twenty-sixth annual ACM symposium on Principles of distributed computing*, pages 43–52. ACM, 2007. (Cited in pages 73 and 71.)

[ABK+16] E. Angel, E. Bampis, A. Kononov, D. Paparas, E. Pountourakis, and V. Zissimopoulos. Clustering on k-edge-colored graphs. *Discrete Applied Mathematics*, 2016. (Cited in pages 120 and 118.)

[ACHK16] A. Abboud, K. Censor-Hillel, and S. Khoury. Near-linear lower bounds for distributed distance computations, even in sparse networks. Technical report, ArXiv, 2016. (Cited in pages 68 and 66.)

[ACP87] S. Arnborg, D.G. Corneil, and A. Proskurowski. Complexity of finding embeddings in a k-tree. *SIAM Journal on Algebraic Discrete Methods*, 8(2):277–284, 1987. (Cited in pages 77, 103, 75 and 101.)

[AD15] H. Alrasheed and F. F. Dragan. Core-periphery models for graphs based on their δ-hyperbolicity: An example using biological networks. In *Complex Networks VI*, pages 65–77. Springer, 2015. (Cited in pages 35, 73, 33 and 71.)

[ADM14] R. Albert, B. DasGupta, and N. Mobasheri. Topological implications of negative curvature for biological and social networks. *Physical Review E*, 89(3):032811, 2014. (Cited in pages 24, 73, 22 and 71.)

[AF84] M. Aigner and M. Fromme. A game of cops and robbers. *Discrete Applied Mathematics*, 8(1):1–12, 1984. (Cited in pages 50 and 48.)

[AGCFV] D. Aguirre-Guerrero, M. Camelo, L. Fabrega, and P. Vila. Word-metric-based greedy routing scheme for data center networks. Submitted. (Cited in pages 15, 22, 14 and 20.)

[AJ13] A. G Aksoy and S. Jin. The apple doesn't fall far from the (metric) tree: The equivalence of definitions. Technical report, ArXiv, 2013. (Cited in pages 25, 29, 34, 23, 27 and 32.)

[AK89] S. B. Akers and B. Krishnamurthy. A group-theoretic model for symmetric interconnection networks. *IEEE transactions on Computers*, 38(4):555–566, 1989. (Cited in pages 17, 52, 15 and 50.)

[AKL07] V. Arvind, J. Köbler, and W. Lindner. Parameterized learnability of k-juntas and related problems. In *International Conference on Algorithmic Learning Theory*, pages 120–134. Springer, 2007. (Cited in pages 188, 186 and 187.)

[ALNP15] C. Avin, Z. Lotker, Y. Nahum, and D. Peleg. Core size and densification in preferential attachment networks. In M. M. Halldórsson, K. Iwama, N. Kobayashi, and B. Speckmann, editors, *ICALP 2015, Kyoto, Japan*, pages 492–503. Springer Berlin Heidelberg, 2015. (Cited in page 24.)

[ALP11] C. Avin, Z. Lotker, and Y. A. Pignolet. On the elite of social networks. Technical report, ArXiv, 2011. (Cited in pages 24 and 22.)

[ALPT16] C. Avin, Z. Lotker, D. Peleg, and I. Turkel. On social networks of program committees. *Social Network Analysis and Mining*, 6(1):18, 2016. (Cited in page 24.)

[AM10] J. Arpe and E. Mossel. Application of a generalization of russo's formula to learning from multiple random oracles. *Combinatorics, Probability and Computing*, 19(02):183–199, 2010. (Cited in pages 189 and 187.)

[AMK03] T. Akutsu, S. Miyano, and S. Kuhara. A simple greedy algorithm for finding functional relations: efficient implementation and average case analysis. *Theoretical Computer Science*, 292(2):481–495, 2003. (Cited in pages 170 and 168.)

[AMM10] N. Archak, V. S. Mirrokni, and S. Muthukrishnan. Mining advertiser-specific user behavior using adfactors. In *Proceedings of the 19th international conference on World wide web*, pages 31–40. ACM, 2010. (Cited in pages 115 and 113.)

[Ang88] D. Angluin. Queries and concept learning. *Machine Learning*, 2(4):319–342, 1988. (Cited in pages 163, 166, 182, 161, 164 and 180.)

[APW12] P. Austrin, T. Pitassi, and Y. Wu. Inapproximability of treewidth, one-shot pebbling, and related layout problems. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 13–24. Springer, 2012. (Cited in pages 73, 103, 71 and 101.)

[AR07] J. Arpe and R. Reischuk. Learning juntas in the presence of noise. *Theoretical Computer Science*, 384(1):2–21, 2007. (Cited in pages 163 and 161.)

[ASHM13] A. B. Adcock, B. D. Sullivan, O. R. Hernandez, and M. W. Mahoney. Evaluating openmp tasking at scale for the computation of graph hyperbolicity. In *International Workshop on OpenMP*, pages 71–83. Springer, 2013. (Cited in pages 55 and 53.)

[ASM13] A. B. Adcock, B. D. Sullivan, and M. W. Mahoney. Tree-like structure in large social and information networks. In *2013 IEEE 13th International Conference on Data Mining*, pages 1–10. IEEE, 2013. (Cited in pages 16, 73, 14 and 71.)

[ASM16] A. B. Adcock, B. D. Sullivan, and M. W. Mahoney. Tree decompositions and social graphs. *Internet Mathematics*, 12(5), 2016. (Cited in pages 69, 77, 108, 67, 75 and 106.)

[AVWW16] A. Abboud, V. Vassilevska Williams, and J. Wang. Approximation and fixed parameter subquadratic algorithms for radius and diameter in sparse graphs. In *Proceedings of the twenty-seventh annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 377–391. SIAM, 2016. (Cited in pages 88 and 86.)

[BAJ00] A. Barabási, R. Albert, and H. Jeong. Scale-free characteristics of random networks: the topology of the world-wide web. *Physica A: Statistical Mechanics and its Applications*, 281(1):69–77, 2000. (Cited in pages 2, 16, 46, 220, 14 and 44.)

[Bal04] C. Ballester. NP-completeness in hedonic games. *Games and Economic Behavior*, 49(1):1–30, 2004. (Cited in pages 120 and 118.)

[Ban90] H.-J. Bandelt. Recognition of tree metrics. *SIAM Journal on Discrete Mathematics*, 3(1):1–6, 1990. (Cited in pages 14, 29, 12 and 27.)

[BBGM15] A. Berry, A. Brandstädt, V. Giakoumakis, and F. Maffray. Efficiently decomposing, recognizing and triangulating hole-free graphs without diamonds. *Discrete Applied Mathematics*, 184:50–61, 2015. (Cited in pages 86 and 84.)

[BC03] H.-J. Bandelt and V. Chepoi. 1-hyperbolic graphs. *SIAM Journal on Discrete Mathematics*, 16(2):323–334, 2003. (Cited in pages 25, 66, 23 and 64.)

[BCC15] M. Borassi, A. Chessa, and G. Caldarelli. Hyperbolicity measures democracy in real-world networks. *Physical Review E*, 92(3):032812, 2015. (Cited in pages 24, 192, 22 and 190.)

[BCCM15] M. Borassi, D. Coudert, P. Crescenzi, and A. Marino. On computing the hyperbolicity of real-world graphs. In *Algorithms-ESA 2015*, pages 215–226. Springer, 2015. (Cited in pages 41, 56, 73, 39, 54 and 71.)

[BCF94] M.-F. Bélanger, J. Constantin, and G. Fournier. Graphes et ordonnés démontables, propriété de la clique fixe. *Discrete Mathematics*, 130(1):9–17, 1994. (Cited in pages 51 and 49.)

[BCH16] M. Borassi, P. Crescenzi, and M. Habib. Into the square: On the complexity of some quadratic-time solvable problems. *Electronic Notes in Theoretical Computer Science*, 322:51–67, 2016. (Cited in pages 65, 67, 111, 63 and 109.)

[BDCV98] A. Brandstädt, F. Dragan, V. Chepoi, and V. Voloshin. Dually chordal graphs. *SIAM Journal on Discrete Mathematics*, 11(3):437–455, 1998. (Cited in pages 37, 44, 82, 83, 35, 42 and 80.)

[Ben98] I. Benjamini. Expanders are not hyperbolic. *Israel Journal of Mathematics*, 108(1):33–36, 1998. (Cited in pages 35, 50, 72, 33, 48 and 70.)

[Ben13] I. Benjamini. The hyperbolic plane and hyperbolic graphs. In *Coarse Geometry and Randomness*, pages 23–31. Springer, 2013. (Cited in pages 14 and 12.)

[BF06] M. Bonk and T. Foertsch. Asymptotic upper curvature bounds in coarse geometry. *Mathematische Zeitschrift*, 253(4):753–785, 2006. (Cited in pages 33, 34, 31 and 32.)

[BFGR15] R. Belmonte, F. V. Fomin, P. A. Golovach, and M. S. Ramanujan. Metric dimension of bounded width graphs. In *International Symposium on Mathematical Foundations of Computer Science*, pages 115–126. Springer, 2015. (Cited in pages 104 and 102.)

[BFÖ+03] G. S. Brodal, R. Fagerberg, A. Östlin, C. N. S. Pedersen, and S. S. Rao. Computing refined buneman trees in cubic time. In *International Workshop on Algorithms in Bioinformatics*, pages 259–270. Springer, 2003. (Cited in pages 69 and 67.)

[BFW92] H.L. Bodlaender, M.R. Fellows, and T. Warnow. Two strikes against perfect phylogeny. In *ICALP'92, Vienna, Austria*, pages 273–283, 1992. (Cited in pages 95 and 93.)

[BH11] M. R. Bridson and A. Haefliger. *Metric spaces of non-positive curvature*, volume 319. Springer Science & Business Media, 2011. (Cited in pages 15, 30, 31, 33, 34, 49, 13, 28, 29, 32 and 47.)

[BH12] A. E. Brouwer and W. H. Haemers. Distance-regular graphs. In *Spectra of Graphs*, pages 177–185. Springer, 2012. (Cited in pages 52, 53 and 51.)

[BHO+11] I. Benjamini, C. Hoppen, E. Ofek, P. Prałat, and N. Wormald. Geodesics and almost geodesic cycles in random regular graphs. *Journal of Graph Theory*, 66(2):115–136, 2011. (Cited in pages 45 and 43.)

[BHV06] A. Berry, P. Heggernes, and Y. Villanger. A vertex incremental approach for maintaining chordality. *Discrete Mathematics*, 306(3):318–336, 2006. (Cited in pages 102 and 101.)

[BK96] H. L. Bodlaender and T. Kloks. Efficient and constructive algorithms for the pathwidth and treewidth of graphs. *Journal of Algorithms*, 21(2):358–402, 1996. (Cited in pages 111 and 110.)

[BK06] H. L. Bodlaender and A. Koster. Safe separators for treewidth. *Discrete Mathematics*, 306(3):337–350, 2006. (Cited in pages 102 and 101.)

[BKC09] M. Boguná, D. Krioukov, and K. C. Claffy. Navigability of complex networks. *Nature Physics*, 5(1):74–80, 2009. (Cited in pages 2, 16, 220 and 14.)

[BKK95] H. L. Bodlaender, T. Kloks, and D. Kratsch. Treewidth and pathwidth of permutation graphs. *SIAM Journal on Discrete Mathematics*, 8(4):606–616, 1995. (Cited in pages 78 and 76.)

[BKKM98] H. L. Bodlaender, T. Kloks, D. Kratsch, and H. Müller. Treewidth and minimum fill-in on d-trapezoid graphs. *J. Graph Algorithms Appl*, 2(5):1–28, 1998. (Cited in pages 78 and 76.)

[BKM01] G. Brinkmann, J. H. Koolen, and V. Moulton. On the hyperbolicity of chordal graphs. *Annals of Combinatorics*, 5(1):61–69, 2001. (Cited in pages 25, 41, 44, 23, 39 and 42.)

[BL97] A. L. Blum and P. Langley. Selection of relevant features and examples in machine learning. *Artificial intelligence*, 97(1):245–271, 1997. (Cited in pages 164 and 162.)

[BL05] Y. Bilu and N. Linial. Monotone maps, sphericity and bounded second eigenvalue. *Journal of Combinatorial Theory, Series B*, 95(2):283–299, 2005. (Cited in pages 23 and 21.)

[Bla38] A. Blake. *Canonical Expressions in Boolean Algebra*. [Chicago], 1938. (Cited in pages 164 and 162.)

[BM86] H.-J. Bandelt and H. M. Mulder. Distance-hereditary graphs. *Journal of Combinatorial Theory, Series B*, 41(2):182–208, 1986. (Cited in pages 41 and 39.)

[BM93] H. L. Bodlaender and R. H. Möhring. The pathwidth and treewidth of cographs. *SIAM Journal on Discrete Mathematics*, 6(2):181–188, 1993. (Cited in pages 78 and 76.)

[BM08] J. A. Bondy and U. S. R. Murty. *Graph theory*. Grad. Texts in Math., 2008. (Cited in page 5.)

[Bod06] H.L. Bodlaender. Treewidth: Characterizations, applications, and computations. In *WG 2006, Bergen, Norway*, pages 1–14, 2006. (Cited in pages 39, 77, 81, 83, 38, 75, 79 and 82.)

[BPK10] M. Boguná, F. Papadopoulos, and D. Krioukov. Sustaining the internet with hyperbolic mapping. *Nature communications*, 1:62, 2010. (Cited in pages 23 and 21.)

[BPS10] A. Berry, R. Pogorelcnik, and G. Simonet. An introduction to clique minimal separator decomposition. *Algorithms*, 3(2):197–215, 2010. (Cited in pages 3, 18, 55, 61, 62, 77, 78, 85, 89, 221, 16, 53, 58, 60, 75, 76, 83 and 87.)

[BPS11] A. Berry, R. Pogorelcnik, and A. Sigayret. Vertical decomposition of a lattice using clique separators. In *Proceedings of The Eighth International Conference on Concept Lattices and Their Applications, Nancy, France, October 17-20, 2011*, pages 15–29, 2011. (Cited in pages 90 and 88.)

[BPS14] A. Berry, R. Pogorelcnik, and G. Simonet. Organizing the atoms of the clique separator decomposition into an atom tree. *Discrete Applied Mathematics*, 177:1–13, 2014. (Cited in pages 62, 85, 89, 59, 83 and 87.)

[Bra01] U. Brandes. A faster algorithm for betweenness centrality. *Journal of mathematical sociology*, 25(2):163–177, 2001. (Cited in pages 23 and 21.)

[BRSV13] S. Bermudo, J. M. Rodríguez, J. M. Sigarreta, and J.-M. Vilaire. Gromov hyperbolic graphs. *Discrete Mathematics*, 313(15):1575–1585, 2013. (Cited in pages 16 and 14.)

[Bry73] T. Brylawski. The lattice of integer partitions. *Discrete Mathematics*, 6(3):201 – 219, 1973. (Cited in pages 122, 128, 129, 120, 126 and 127.)

[BS11] M. Bonk and O. Schramm. Embeddings of gromov hyperbolic spaces. In *Selected Works of Oded Schramm*, pages 243–284. Springer, 2011. (Cited in pages 15 and 14.)

[BS12] I. Benjamini and O. Schramm. Finite transitive graph embeddings into a hyperbolic metric space must stretch or squeeze. In *Geometric aspects of functional analysis*, pages 123–126. Springer, 2012. (Cited in pages 52 and 50.)

[BT97] H. L. Bodlaender and D. M. Thilikos. Treewidth for graphs with small chordality. *Discrete Applied Mathematics*, 79(1):45–61, 1997. (Cited in pages 73 and 71.)

[BT12] Y. Baryshnikov and G. H. Tucci. Asymptotic traffic flow in a hyperbolic network. In *Communications Control and Signal Processing (ISCCSP), 2012 5th International Symposium on*, pages 1–4. IEEE, 2012. (Cited in pages 23 and 21.)

[Bun74] P. Buneman. A note on the metric properties of trees. *Journal of Combinatorial Theory, Series B*, 17(1):48–50, 1974. (Cited in pages 14, 25, 69, 12, 23 and 67.)

[BW12] A. Berry and A. Wagler. Triangulation and clique separator decomposition of claw-free graphs. In *International Workshop on Graph-Theoretic Concepts in Computer Science*, pages 7–21. Springer, 2012. (Cited in pages 86 and 84.)

[BXHR12] B.-M. Bui-Xuan, M. Habib, and M. Rao. Tree-representation of set families and applications to combinatorial decompositions. *European Journal of Combinatorics*, 33(5):688–711, 2012. (Cited in pages 191 and 189.)

[BZ03] N. Burani and W. S. Zwicker. Coalition formation games with separable preferences. *Mathematical Social Sciences*, 45(1):27–52, 2003. (Cited in pages 120, 123, 146, 154, 118, 121, 145 and 152.)

[CCDL17] N. Cohen, D. Coudert, G. Ducoffe, and A. Lancin. Applying clique-decomposition for computing gromov hyperbolicity. Submitted (Research Report on HAL, hal-00989024), 2017. (Cited in pages 3, 13, 18, 61, 62, 63, 221, 11, 16, 58, 59 and 60.)

[CCL15] N. Cohen, D. Coudert, and A. Lancin. On computing the gromov hyperbolicity. *Journal of Experimental Algorithmics (JEA)*, 20:1–6, 2015. (Cited in pages 41, 43, 46, 56, 73, 192, 39, 40, 44, 54, 71 and 190.)

[CCNV11] J. Chalopin, V. Chepoi, N. Nisse, and Y. Vaxès. Cop and robber games when the robber can hide and ride. *SIAM Journal on Discrete Mathematics*, 25(1):333–359, 2011. (Cited in pages 32, 34, 46, 50, 30, 44 and 48.)

[CCPP14] J. Chalopin, V. Chepoi, P. Papasoglu, and T. Pecatte. Cop and robber game and hyperbolicity. *SIAM Journal on Discrete Mathematics*, 28(4):1987–2007, 2014. (Cited in pages 29, 33, 34, 50, 57, 27, 30, 31, 32, 48 and 55.)

[CD00] V. Chepoi and F. Dragan. A note on distance approximating trees in graphs. *European Journal of Combinatorics*, 21(6):761–766, 2000. (Cited in pages 39, 41, 69, 37 and 67.)

[CD14] D. Coudert and G. Ducoffe. Recognition of c_4-free and 1/2-hyperbolic graphs. *SIAM Journal of Discrete Mathematics*, 28(3):1601–1617, 2014. (Cited in pages 3, 13, 18, 57, 65, 221, 11, 16 and 63.)

[CD16a] D. Coudert and G. Ducoffe. Data center interconnection networks are not hyperbolic. *Journal of Theoretical Computer Science*, 639(1):72–90, 2016. (Cited in pages 3, 13, 17, 48, 50, 51, 53, 54, 60, 221, 11, 15, 46, 49, 52 and 58.)

[CD16b] D. Coudert and G. Ducoffe. On the hyperbolicity of bipartite graphs and intersection graphs. *Discrete Applied Mathematics*, 214:187–195, 2016. (Cited in pages 3, 13, 17, 42, 43, 221, 11, 15, 40 and 41.)

[CD17] A. Chaintreau and G. Ducoffe. A theory for ad targeting identification. In preparation, 2017. (Cited in pages 5, 159, 163, 169, 184, 223, 157, 161, 167 and 182.)

[CDE+08] V. Chepoi, F. F. Dragan, B. Estellon, M. Habib, and Y. Vaxès. Notes on diameters, centers, and approximating trees of δ-hyperbolic geodesic spaces and graphs. *Electronic Notes in Discrete Mathematics*, 31:231–234, 2008. (Cited in pages 16, 55, 69, 70, 14, 53, 67 and 68.)

[CDE+12] V. Chepoi, F. F. Dragan, B. Estellon, M. Habib, Y. Vaxès, and Y. Xiang. Additive spanners and distance and routing labeling schemes for hyperbolic graphs. *Algorithmica*, 62(3-4):713–732, 2012. (Cited in pages 15 and 14.)

[CDHH16] J. Carmesin, R. Diestel, M. Hamann, and F. Hundertmark. Canonical tree-decompositions of finite graphs I. existence and algorithms. *Journal of Combinatorial Theory, Series B*, 116:1–24, 2016. (Cited in pages 85 and 83.)

[CDN16] D. Coudert, G. Ducoffe, and N. Nisse. To approximate treewidth, use treelength! *SIAM Journal of Discrete Mathematics*, 30(3):1424–1436, 2016. (Cited in pages 4, 40, 75, 79, 104, 105, 107, 108, 109, 222, 38, 73, 77, 102, 103 and 106.)

[CDV16] V. Chepoi, F. Dragan, and Y. Vaxès. Core congestion is inherent in hyperbolic networks. Technical Report arXiv:1605.03059, ArXiv, 2016. (Cited in pages 15, 21, 23, 50, 71, 13, 19, 48 and 69.)

[CE07] V. Chepoi and B. Estellon. Packing and covering δ-hyperbolic spaces by balls. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 59–73. Springer, 2007. (Cited in pages 55, 70, 53 and 68.)

[CFHM13] W. Chen, W. Fang, G. Hu, and M. W. Mahoney. On the hyperbolicity of small-world and treelike random graphs. *Internet Mathematics*, 9(4):434–491, 2013. (Cited in pages 16, 35, 37, 38, 45, 46, 14, 33, 36, 43 and 44.)

[Cisa] The Zettabyte Era – Trends and Analysis. `http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/vni-hyperconnectivity-wp.html`. (Cited in pages 1 and 219.)

[Cisb] The Zettabyte Era Officially Begins (How Much is That?). `http://blogs.cisco.com/sp/the-zettabyte-era-officially-begins-how-much-is-that`. (Cited in page 1.)

[CKK97] J. Chen, S. P. Kanchi, and A. Kanevsky. A note on approximating graph genus. *Information processing letters*, 61(6):317–322, 1997. (Cited in pages 105 and 103.)

[CKPS10] I. Chatzigiannakis, C. Koninis, P. N. Panagopoulou, and P. G. Spirakis. Distributed game-theoretic vertex coloring. In *OPODIS'10*, pages 103–118, 2010. (Cited in pages 4, 119, 126, 222, 117 and 124.)

[CM78] A. K. Chandra and G. Markowsky. On the number of prime implicants. *Discrete Mathematics*, 24(1):7–11, 1978. (Cited in pages 164 and 162.)

[CN04] I. Chatterji and G. A. Niblo. A characterization of hyperbolic spaces. Technical report, ArXiv, 2004. (Cited in pages 34 and 32.)

[COS97] D. G. Corneil, S. Olariu, and L. Stewart. Asteroidal triple-free graphs. *SIAM Journal on Discrete Mathematics*, 10(3):399–430, 1997. (Cited in pages 41 and 39.)

[Cou90] B. Courcelle. The monadic second-order logic of graphs. I. recognizable sets of finite graphs. *Information and computation*, 85(1):12–75, 1990. (Cited in pages 77, 103, 75 and 101.)

[CPFV14] M. Camelo, D. Papadimitriou, L. Fàbrega, and P. Vilà. Geometric routing with word-metric spaces. *IEEE Communications Letters*, 18(12):2125–2128, 2014. (Cited in pages 23 and 21.)

[CRS15] W. Carballosa, J. M. Rodríguez, and J. M. Sigarreta. Hyperbolicity in the corona and join of graphs. *Aequationes mathematicae*, 89(5):1311–1327, 2015. (Cited in pages 44 and 42.)

[Cun82] William H. Cunningham. Decomposition of directed graphs. *SIAM Journal on Algebraic Discrete Methods*, 3(2):214–228, 1982. (Cited in pages 60 and 58.)

[Dah98] E. Dahlhaus. Minimal elimination of planar graphs. In *Scandinavian Workshop on Algorithm Theory*, pages 210–221. Springer, 1998. (Cited in pages 90 and 88.)

[Dah02] E. Dahlhaus. Minimal elimination ordering for graphs of bounded degree. *Discrete applied mathematics*, 116(1):127–143, 2002. (Cited in pages 90 and 88.)

[Dai80] D. P. Dailey. Uniqueness of colorability and colorability of planar 4-regular graphs are NP-complete. *Discrete Mathematics*, 30(3):289 – 293, 1980. (Cited in pages 153 and 150.)

[Dam73] R. M. Damerell. on moore graphs. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 74, pages 227–236. Cambridge Univ Press, 1973. (Cited in pages 52 and 51.)

[Dam16] P. Damaschke. Computing giant graph diameters. In *International Workshop on Combinatorial Algorithms*, pages 373–384. Springer, 2016. (Cited in pages 73 and 71.)

[DC17] G. Ducoffe and D. Coudert. Clique-decomposition revisited. In revision (Research Report on HAL, hal-01266147), 2017. (Cited in pages 4, 75, 78, 86, 87, 88, 90, 111, 222, 73, 76, 84, 85 and 109.)

[DCG14] P. De Caria and M. Gutierrez. On the correspondence between tree representations of chordal and dually chordal graphs. *Discrete Applied Mathematics*, 164:500–511, 2014. (Cited in pages 37 and 35.)

[DCM] Datacenters internationaux | Microsoft. `http://www.microsoft.com/fr-fr/server-cloud/cloud-os/global-datacenters.aspx`. (Cited in pages 1 and 219.)

[DDGY07] Y. Dourisboure, F. F. Dragan, C. Gavoille, and C. Yan. Spanners for bounded tree-length graphs. *Theoretical Computer Science*, 383(1):34–44, 2007. (Cited in pages 23, 77, 21 and 75.)

[DFG11] F. F. Dragan, F. V. Fomin, and P. A. Golovach. Spanners in sparse graphs. *Journal of Computer and System Sciences*, 77(6):1108–1119, 2011. (Cited in pages 112 and 110.)

[DG07] Y. Dourisboure and C. Gavoille. Tree-decompositions with bags of small diameter. *Discrete Mathematics*, 307(16):2008–2029, 2007. (Cited in pages 39, 77, 79, 82, 91, 103, 105, 109, 110, 38, 75, 80, 81, 89, 101 and 108.)

[DG09] Y. Dieng and C. Gavoille. On the tree-width of planar graphs. *Electronic Notes in Discrete Mathematics*, 34:593–596, 2009. (Cited in pages 75, 104, 73 and 102.)

[DGM06] S. N. Dorogovtsev, A. V. Goltsev, and J. F. F. Mendes. K-core organization of complex networks. *Physical review letters*, 96(4):040601, 2006. (Cited in pages 2, 16, 220 and 14.)

[DH04] E. D. Demaine and M. Hajiaghayi. Equivalence of local treewidth and linear local treewidth and its algorithmic applications. In *Proceedings of the fifteenth annual ACM-SIAM Symposium on Discrete algorithms (SODA)*, pages 840–849. Society for Industrial and Applied Mathematics, 2004. (Cited in pages 105 and 104.)

[DH08] E. D. Demaine and M. Hajiaghayi. The bidimensionality theory and its algorithmic applications. *The Computer Journal*, 51(3):292–302, 2008. (Cited in pages 77, 109, 75 and 107.)

[DHH+05] A. Dress, B. Holland, K. T. Huber, J. H. Koolen, V. Moulton, and J. Weyer-Menkhoff. δ additive and δ ultra-additive maps, gromov's trees, and the farris transform. *Discrete Applied Mathematics*, 146(1):51–73, 2005. (Cited in pages 22 and 20.)

[Die10] Reinhard Diestel. Graph theory. *Heidelberg, Graduate Texts in Mathematics*, 173:451 pp., 2010. 4*th* edition. (Cited in pages 5, 81, 106, 107, 79 and 105.)

[Dir61] G. A. Dirac. On rigid circuit graphs. In *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, volume 25, pages 71–76. Springer, 1961. (Cited in pages 37 and 35.)

[DK14] F.F. Dragan and E. Köhler. An approximation algorithm for the tree t-spanner problem on unweighted graphs via generalized chordal graphs. *Algorithmica*, 69(4):884–905, 2014. (Cited in pages 39, 77, 78, 82, 91, 92, 192, 38, 75, 76, 80, 89, 90 and 190.)

[DKL14] F.F. Dragan, E. Köhler, and A. Leitert. Line-distortion, bandwidth and path-length of a graph. In *Algorithm Theory–SWAT 2014*, pages 158–169. Springer, 2014. (Cited in pages 78, 91, 92, 192, 76, 89, 90 and 190.)

[DKMY15] B. DasGupta, M. Karpinski, N. Mobasheri, and F. Yahyanejad. Node expansions and cuts in gromov-hyperbolic graphs. Technical report, ArXiv, 2015. (Cited in pages 35, 55, 72, 73, 33, 53, 70 and 71.)

[DL07] F. F. Dragan and I. Lomonosov. On compact and efficient routing in certain graph classes. *Discrete applied mathematics*, 155(11):1458–1470, 2007. (Cited in pages 81 and 80.)

[DL15] F.F. Dragan and A. Leitert. On the minimum eccentricity shortest path problem. In *Algorithms and Data Structures – WADS*, pages 276–288. Springer, 2015. (Cited in pages 92 and 90.)

[DLCG15] G. Ducoffe, M. Lécuyer, A. Chaintreau, and R. Geambasu. Web's transparency for complex targeting: Algorithms, limits and tradeoffs. In *SIGMETRICS'15 Proceedings of the 2015 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, pages 465–466, 2015. (Cited in pages 5, 159, 165, 168, 177, 223, 157, 164, 166 and 175.)

[DLN16a] G. Ducoffe, N. Legay, and N. Nisse. On the complexity of computing treebreadth. In *IWOCA 2016 – 27th International Workshop on Combinatorial Algorithms*, pages 3–15, 2016. (Cited in pages 4, 75, 78, 91, 92, 93, 94, 98, 111, 222, 73, 76, 89, 90, 96, 109 and 110.)

[DLN16b] G. Ducoffe, S. Legay, and N. Nisse. On computing tree and path decompositions with metric constraints on the bags. Technical Report arXiv:1601.01958, arXiv, 2016. (Cited in pages 91, 93, 94, 96, 97, 89, 92 and 95.)

[DLVM86] P. Duchet, M. Las Vergnas, and H. Meyniel. Connected cutsets of a graph and triangle bases of the cycle space. *Discrete Mathematics*, 62(2):145–154, 1986. (Cited in pages 108 and 107.)

[DM47] A. De Morgan. *Formal logic: or, the calculus of inference, necessary and probable*. Taylor and Walton, 1847. (Cited in pages 139 and 137.)

[DM15] Reinhard Diestel and Malte Müller. Connected tree-width. Technical Report arXiv preprint arXiv:1211.7353, ArXiv, oct 2015. (Cited in pages 108 and 106.)

[DMC12] G. Ducoffe, D. Mazauric, and A. Chaintreau. Can Selfish Groups be Self-Enforcing? Technical Report arXiv:1212.3782, ArXiv, 2012. (Cited in pages 5, 117, 153, 155, 223, 115, 151 and 152.)

[DMC13a] G. Ducoffe, D. Mazauric, and A. Chaintreau. Can selfish groups be self-enforcing? In *Workshop on Social Computing and User Generated Content at EC'13*, pages 1–47, 2013. (Cited in pages 4, 5, 117, 123, 222, 223, 115 and 121.)

[DMC13b] G. Ducoffe, D. Mazauric, and A. Chaintreau. De la difficulté de garder ses amis (quand on a des ennemis)! In *ALGOTEL 2013 – 15èmes Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications*, pages 1–4, 2013. (Cited in pages 123 and 121.)

[DMC17] G. Ducoffe, D. Mazauric, and A. Chaintreau. The complexity of hedonic coalitions under bounded cooperation. Submitted (Research Report on ArXiv, arXiv:1212.3782), 2017. (Cited in pages 4, 5, 117, 123, 131, 132, 133, 135, 137, 150, 222, 223, 115, 121, 129, 130, 147 and 148.)

[dMSV11] F. de Montgolfier, M. Soto, and L. Viennot. Treewidth and hyperbolicity of the internet. In *Network Computing and Applications (NCA), 2011 10th IEEE International Symposium on*, pages 25–32, Aug 2011. (Cited in pages 46, 77, 44 and 75.)

[DMT96] A. Dress, V. Moulton, and W. Terhalle. T-theory: An overview. *European Journal of Combinatorics*, 17(2):161–175, 1996. (Cited in pages 15, 22, 14 and 20.)

[DP94] X. Deng and C. Papadimitriou. On the Complexity of Cooperative Solution Concepts. *Mathematics of Operations Research*, 19(2):257–266, 1994. (Cited in pages 123, 156, 121 and 154.)

[DP09] R. Duan and S. Pettie. Fast algorithms for (max, min)-matrix multiplication and bottleneck shortest paths. In *Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 384–391. Society for Industrial and Applied Mathematics, 2009. (Cited in pages 56 and 54.)

[DRB99] H.N. De Ridder and H.L. Bodlaender. Graph automorphisms with maximal projection distances. In *Fundamentals of Computation Theory*, pages 204–214. Springer, 1999. (Cited in pages 51 and 49.)

[DTC17] G. Ducoffe, M. Tucker, and A. Chaintreau. Can web's transparency tools cope with complex targeting? In preparation, 2017. (Cited in pages 5, 159, 165, 168, 171, 175, 223, 157, 164, 166, 169 and 173.)

[DTD15] A. Datta, M.C. Tschantz, and A. Datta. Automated experiments on ad privacy settings. *Proceedings on Privacy Enhancing Technologies*, 2015(1):92–112, 2015. (Cited in pages 160, 161, 170, 175, 159, 168 and 173.)

[Dua14] R. Duan. Approximation algorithms for the gromov hyperbolicity of discrete metric spaces. In *Latin American Symposium on Theoretical Informatics*, pages 285–293. Springer, 2014. (Cited in pages 57 and 55.)

[Duc16] G. Ducoffe. The parallel complexity of coloring games. In *SAGT 2016 – 9th International Symposium on Algorithmic Game Theory*, pages 27–39, 2016. (Cited in pages 4, 117, 123, 143, 223, 5, 115, 121 and 140.)

[Dwo08] C. Dwork. Differential privacy: A survey of results. In *International Conference on Theory and Applications of Models of Computation*, pages 1–19. Springer, 2008. (Cited in pages 118 and 116.)

[DX09] F. F. Dragan and Y. Xiang. How to use spanning trees to navigate in graphs. In *International Symposium on Mathematical Foundations of Computer Science*, pages 282–294. Springer, 2009. (Cited in pages 16 and 14.)

[EDP] European Data Protection Supervisor. https://secure.edps.europa.eu/EDPSWEB/edps/EDPS/Dataprotection/Legislation. (Cited in pages 2, 118, 219 and 116.)

[EGM12] B. Escoffier, L. Gourvès, and J. Monnot. Strategic coloring of a graph. *Internet Mathematics*, 8(4):424–455, 2012. (Cited in pages 4, 117, 119, 121, 122, 126, 127, 131, 155, 222, 115, 120, 124, 125, 129 and 154.)

[EKS16] K. Edwards, W. S. Kennedy, and I. Saniee. Fast approximation algorithms for p-centres in large *delta*-hyperbolic graphs. Technical report, ArXiv, 2016. (Cited in pages 55, 70, 71, 53, 68 and 69.)

[Ema] Email Statistics Report, 2015-2019. http://www.radicati.com/wp/wp-content/uploads/2015/02/Email-Statistics-Report-2015-2019-Executive-Summary.pdf. (Cited in pages 1 and 219.)

[EPC+92] D. Epstein, M. S. Paterson, J. W. Cannon, D. F. Holt, S. V. Levy, and W. P. Thurston. *Word processing in groups*. AK Peters, Ltd., 1992. (Cited in pages 15 and 13.)

[EPL72] S. Even, A. Pnueli, and A. Lempel. Permutation graphs and transitive graphs. *Journal of the ACM (JACM)*, 19(3):400–410, 1972. (Cited in pages 41 and 39.)

[Epp00] D. Eppstein. Diameter and treewidth in minor-closed graph families. *Algorithmica*, 27(3-4):275–291, 2000. (Cited in pages 110 and 109.)

[FA05] D. Fukagawa and T. Akutsu. Performance analysis of a greedy algorithm for inferring boolean functions. *Information Processing Letters*, 93(1):7–12, 2005. (Cited in pages 170, 181, 168 and 179.)

[Fan11] W. Fang. On hyperbolic geometry structure of complex networks. Technical report, IRIF – Institut de Recherche en Informatique Fondamentale, 2011. Report of M1 internship in Microsoft Research Asia. (Cited in pages 65 and 63.)

[Far72] J. S. Farris. Estimating phylogenetic trees from distance matrices. *American Naturalist*, pages 645–668, 1972. (Cited in pages 29 and 27.)

[Far83] M. Farber. Characterizations of strongly chordal graphs. *Discrete Mathematics*, 43(2):173–189, 1983. (Cited in pages 41 and 39.)

[FBE] facebook business – What are Custom Audiences from your website? https://www.facebook.com/business/help/610516375684216. (Cited in pages 165, 176, 163 and 174.)

[FBN] Facebook newsroom – Company info. http://newsroom.fb.com/company-info/. (Cited in pages 1 and 219.)

[FCM14] L. Ferretti, M. Cortelezzi, and M. Mamino. Duality between preferential attachment and static networks on hyperbolic spaces. *EPL (Europhysics Letters)*, 105(3):38001, 2014. (Cited in pages 53 and 51.)

[FG01] J. Flum and M. Grohe. Fixed-parameter tractability, definability, and model-checking. *SIAM Journal on Computing*, 31(1):113–145, 2001. (Cited in pages 77 and 75.)

[FGKP09] V. Feldman, P. Gopalan, S. Khot, and A. K. Ponnuswami. On agnostic learning of parities, monomials, and halfspaces. *SIAM Journal on Computing*, 39(2):606–645, 2009. (Cited in pages 163 and 161.)

[FGL+15] M. Farrell, T. D. Goodrich, N. Lemons, F. Reidl, F. S. Villaamil, and B. D. Sullivan. Hyperbolicity, degeneracy, and expansion of random intersection graphs. In *International Workshop on Algorithms and Models for the Web-Graph*, pages 29–41. Springer, 2015. (Cited in pages 45 and 43.)

[FGT11] F. V. Fomin, P. Golovach, and D. M. Thilikos. Contraction obstructions for treewidth. *Journal of Combinatorial Theory, Series B*, 101(5):302–314, 2011. (Cited in pages 109 and 107.)

[FH76] S. Foldes and P. L. Hammer. *Split graphs*. Universität Bonn. Institut für Ökonometrie und Operations Research, 1976. (Cited in pages 37 and 34.)

[Fin15] E. Fink. Hyperbolicity via geodesic stability. Technical report, ArXiv, 2015. (Cited in pages 20 and 18.)

[FIV15] H. Fournier, A. Ismail, and A. Vigneron. Computing the gromov hyperbolicity of a discrete metric space. *Information Processing Letters*, 115(6):576–579, 2015. (Cited in pages 56, 57, 64, 53, 55 and 62.)

[FK01] S. P. Fekete and J. Kremer. Tree spanners in planar graphs. *Discrete Applied Mathematics*, 108(1):85–103, 2001. (Cited in pages 112 and 110.)

[FKLL15] P. Floderus, M. Kowaluk, A. Lingas, and E.-M. Lundell. Detecting and counting small pattern graphs. *SIAM Journal on Discrete Mathematics*, 29(3):1322–1339, 2015. (Cited in pages 67 and 65.)

[FKP02] A. Fabrikant, E. Koutsoupias, and C. H. Papadimitriou. Heuristically optimized trade-offs: A new paradigm for power laws in the internet. In P. Widmayer, S. Eidenbenz, F. Triguero, R. Morales,

R. Conejo, and M. Hennessy, editors, *ICALP Proceedings*, pages 110–122. Springer Berlin Heidelberg, 2002. (Cited in page 46.)

[FLP+15] F. V. Fomin, D. Lokshtanov, M. Pilipczuk, S. Saurabh, and M. Wrochna. Fully polynomial-time parameterized computations for graphs and matrices of low treewidth. Technical Report arXiv:1511.01379, arXiv, 2015. (Cited in pages 88, 90 and 86.)

[FM71] M. J. Fischer and A. R. Meyer. Boolean matrix multiplication and transitive closure. In *Switching and Automata Theory, 1971., 12th Annual Symposium on*, pages 129–131, Oct 1971. (Cited in pages 64 and 62.)

[Gal67] Tibor Gallai. Transitiv orientierbare graphen. *Acta Mathematica Hungarica*, 18(1):25–66, 1967. (Cited in pages 60 and 58.)

[Gav72] F. Gavril. Algorithms for minimum coloring, maximum clique, minimum covering by cliques, and maximum independent set of a chordal graph. *SIAM Journal on Computing*, 1(2):180–187, 1972. (Cited in pages 89 and 88.)

[Gav74] F. Gavril. The intersection graphs of subtrees in trees are exactly the chordal graphs. *Journal of Combinatorial Theory, Series B*, 16(1):47–56, 1974. (Cited in pages 37, 82, 83, 35, 80 and 81.)

[GdLH90] E. Ghys and P. de La Harpe. *Sur les groupes hyperboliques d'apres Mikhael Gromov*. Birkhauser Boston, Inc., Science Press, 1990. (Cited in pages 15, 31, 49, 59, 68, 13, 29, 47, 57 and 66.)

[GEC+13] P. Gill, V. Erramilli, A. Chaintreau, B. Krishnamurthy, K. Papagiannaki, and P. Rodriguez. Follow the money: understanding economics of online aggregation and advertising. In *Proceedings of the Internet measurement conference (IMC)*, pages 141–148. ACM, 2013. (Cited in pages 184, 189, 182 and 187.)

[GG78] M. C. Golumbic and C. F. Goss. Perfect elimination and chordal bipartite graphs. *Journal of Graph Theory*, 2(2):155–163, 1978. (Cited in pages 37 and 35.)

[GHR95] R. Greenlaw, H. J. Hoover, and W. L. Ruzzo. *Limits to parallel computation: P-completeness theory*. Oxford University Press, 1995. (Cited in pages 138, 139, 145, 136, 137 and 144.)

[GJ85] M. C. Golumbic and R. .E Jamison. The edge intersection graphs of paths in a tree. *Journal of Combinatorial Theory, Series B*, 38(1):8–22, 1985. (Cited in pages 64 and 62.)

[GK86] C. Greene and D. J. Kleitman. Longest chains in the lattice of integer partitions ordered by majorization. *European Journal of Combinatorics*, 7(1):1–10, jan 1986. (Cited in pages 129 and 127.)

[GK14] R. D. Gray and M. Kambites. A strong geometric hyperbolicity property for directed graphs and monoids. *Journal of Algebra*, 420:373–401, 2014. (Cited in pages 74 and 72.)

[GKR13] M. Grohe, K. Kawarabayashi, and B. Reed. A simple algorithm for the graph minor decomposition: logic meets structural graph theory. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 414–431. Society for Industrial and Applied Mathematics, 2013. (Cited in pages 105 and 104.)

[GKS95] M.C. Golumbic, H. Kaplan, and R. Shamir. Graph sandwich problems. *Journal of Algorithms*, 19(3):449–473, 1995. (Cited in pages 94, 95, 92 and 93.)

[GL05] C. Gavoille and O. Ly. Distance labeling in hyperbolic graphs. In *International Symposium on Algorithms and Computation*, pages 1071–1079. Springer, 2005. (Cited in pages 15, 23, 68, 14, 21 and 66.)

[GL06] J.-L. Guillaume and M. Latapy. Bipartite graphs as models of complex networks. *Physica A: Statistical Mechanics and its Applications*, 371(2):795–813, 2006. (Cited in pages 168 and 166.)

[Glo67] F. Glover. Maximum matching in a convex bipartite graph. *Naval Research Logistics Quarterly*, 14(3):313–316, 1967. (Cited in pages 37 and 35.)

[GM00] C. Gutwenger and P. Mutzel. A linear time implementation of SPQR-trees. In *International Symposium on Graph Drawing*, pages 77–90. Springer, 2000. (Cited in pages 85, 90, 83 and 89.)

[GMN15] A. C. Giannopoulou, G. B. Mertzios, and R. Niedermeier. Polynomial fixed-parameter algorithms: A case study for longest path on interval graphs. Technical Report arXiv:1506.01652, arXiv, 2015. (Cited in pages 88 and 86.)

[GMT84] M. C. Golumbic, C. L. Monma, and W. T. Trotter. Tolerance graphs. *Discrete Applied Mathematics*, 9(2):157–170, 1984. (Cited in pages 41 and 39.)

[GO95] A. Gajentaan and M. H. Overmars. On a class of $\wr(n^2)$ problems in computational geometry. *Computational geometry*, 5(3):165–185, 1995. (Cited in pages 65 and 63.)

[Gol71] A.J. Goldman. Optimal center location in simple networks. *Transportation science*, 5(2):212–221, 1971. (Cited in pages 21, 62, 82, 19, 60 and 80.)

[Goo] AdWords Help: About Keyword Planner. `https://support.google.com/adwords/answer/2999770`. (Cited in pages 164 and 162.)

[Gou14] J. Gould. SafeGov.org - Google admits data mining student emails in its free education apps, 2014. (Cited in pages 2, 160, 219 and 158.)

[GR86] S. Greenland and J. M. Robins. Identifiability, exchangeability, and epidemiological confounding. *International journal of epidemiology*, 15(3):413–419, 1986. (Cited in pages 168 and 166.)

[GR13] C. Godsil and G. F. Royle. *Algebraic graph theory*, volume 207. Springer Science & Business Media, 2013. (Cited in pages 53 and 51.)

[Gro87] M. Gromov. Hyperbolic groups. In *Essays in group theory*, pages 75–263. Springer, 1987. (Cited in pages 2, 14, 15, 19, 25, 28, 29, 30, 31, 34, 35, 57, 68, 69, 220, 12, 13, 17, 23, 26, 27, 32, 55, 66 and 67.)

[Gro16] M. Grohe. *Tangles and Connectivity in Graphs*, pages 24–41. Springer International Publishing, 2016. (Cited in pages 85 and 83.)

[GS10] M. Groshaus and J.L. Szwarcfiter. Biclique graphs and biclique matrices. *Journal of Graph Theory*, 63(1):1–16, 2010. (Cited in pages 17, 44, 15 and 42.)

[Ham68] R.C. Hamelink. A partial characterization of clique graphs. *Journal of Combinatorial Theory*, 5(2):192–197, 1968. (Cited in pages 17, 42, 15 and 40.)

[Hay85] R. B. Hayward. Weakly triangulated graphs. *Journal of Combinatorial Theory, Series B*, 39(3):200–208, 1985. (Cited in pages 41 and 39.)

[Heg06] P. Heggernes. Minimal triangulations of graphs: A survey. *Discrete Mathematics*, 306(3):297–317, 2006. (Cited in pages 84 and 82.)

[HLC07] J. He, D. Long, and C. Chen. An improved ant-based classifier for intrusion detection. In *Third International Conference on Natural Computation (ICNC 2007)*, volume 4, pages 819–823. IEEE, 2007. (Cited in pages 189 and 187.)

[HLW06] S. Hoory, N. Linial, and A. Wigderson. Expander graphs and their applications. *Bulletin of the American Mathematical Society*, 43(4):439–561, 2006. (Cited in pages 50 and 48.)

[Hoe63] W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American statistical association*, 58(301):13–30, 1963. (Cited in pages 172 and 170.)

[Hor87] J. D. Horton. A polynomial-time algorithm to find the shortest cycle basis of a graph. *SIAM Journal on Computing*, 16(2):358–366, 1987. (Cited in pages 106 and 104.)

[How79] E. Howorka. On metric properties of certain clique graphs. *Journal of Combinatorial Theory, Series B*, 27(1):67–74, 1979. (Cited in pages 65 and 63.)

[HPR14] V. Hernández, D. Pestana, and J. M. Rodríguez. Bounds on the hyperbolicity constant. *Electronic Notes in Discrete Mathematics*, 46:137–144, 2014. (Cited in pages 16, 41, 14 and 38.)

[HSL+14] A. Hannak, G. Soeller, D. Lazer, A. Mislove, and C. Wilson. Measuring price discrimination and steering on e-commerce web sites. In *Proceedings of the 2014 conference on internet measurement conference*, pages 305–318. ACM, 2014. (Cited in pages 170 and 168.)

[HSMK+13] A. Hannak, P. Sapiezynski, A. Molavi Kakhki, B. Krishnamurthy, D. Lazer, A. Mislove, and C. Wilson. Measuring personalization of

web search. In *Proceedings of the 22nd international conference on World Wide Web*, pages 527–538. ACM, 2013. (Cited in pages 170 and 168.)

[HT73] J. E. Hopcroft and R. E. Tarjan. Dividing a graph into triconnected components. *SIAM Journal on Computing*, 2(3):135–158, 1973. (Cited in pages 77, 85, 75 and 83.)

[HTV05] P. Heggernes, J. A. Telle, and Y. Villanger. Computing minimal triangulations in time $o(n^\alpha \log n) = o(n^{2.376})$. In *Proceedings of the sixteenth annual ACM-SIAM Symposium on Discrete algorithms (SODA)*, pages 907–916. Society for Industrial and Applied Mathematics, 2005. (Cited in pages 90 and 88.)

[HW79] G. H. Hardy and E. M. Wright. *An introduction to the theory of numbers*. Oxford University Press, 1979. (Cited in pages 128 and 126.)

[IDC] IDC – Extracting Value from Chaos. `http://www.emc.com/collateral/analyst-reports/idc-extracting-value-from-chaos-ar.pdf`. (Cited in pages 1 and 219.)

[IPZ98] R. Impagliazzo, R. Paturi, and F. Zane. Which problems have strongly exponential complexity? In *Foundations of Computer Science, 1998. Proceedings. 39th Annual Symposium on*, pages 653–662. IEEE, 1998. (Cited in pages 64 and 62.)

[JL04] E. Jonckheere and P. Lohsoonthorn. Geometry of network security. In *American Control Conference, 2004. Proceedings of the 2004*, volume 2, pages 976–981. IEEE, 2004. (Cited in pages 15, 23, 24, 14, 21 and 22.)

[JLB08] E. Jonckheere, P. Lohsoonthorn, and F. Bonahon. Scaled gromov hyperbolic graphs. *Journal of Graph Theory*, 57(2):157–180, 2008. (Cited in pages 16, 46, 73, 14, 44 and 71.)

[JLBB11] E. Jonckheere, M. Lou, F. Bonahon, and Y. Baryshnikov. Euclidean versus hyperbolic congestion in idealized versus experimental networks. *Internet Mathematics*, 7(1):1–27, 2011. (Cited in pages 23 and 21.)

[Jor69] C. Jordan. Sur les assemblages de lignes. *J. Reine Angew. Math*, 70(185):81, 1869. (Cited in pages 70 and 68.)

[JPY88] D. S. Johnson, C. H. Papadimitriou, and M. Yannakakis. How easy is local search? *Journal of computer and system sciences*, 37(1):79–100, 1988. (Cited in pages 122, 127, 157, 120, 125 and 155.)

[Kar72] R. M. Karp. Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Springer, 1972. (Cited in pages 170 and 168.)

[KBSP16] K.-K. Kleineberg, M. Boguñá, M. Serrano, and F. Papadopoulos. Hidden geometric correlations in real multiplex networks. *Nature Physics*, 2016. (Cited in pages 119 and 118.)

[Kho02] S. Khot. On the power of unique 2-prover 1-round games. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing (STOC)*, pages 767–775. ACM, 2002. (Cited in pages 72 and 71.)

[KK95] T. Kloks and D. Kratsch. Treewidth of chordal bipartite graphs. *Journal of Algorithms*, 19(2):266–281, 1995. (Cited in pages 78 and 76.)

[KL06] R. Krauthgamer and J.R. Lee. Algorithms on negatively curved spaces. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 119–132. IEEE, 2006. (Cited in pages 17, 55, 71, 15, 53 and 69.)

[KL13] J. Kleinberg and K. Ligett. Information-sharing in social networks. *Games and Economic Behavior*, 82:702–716, 2013. (Cited in pages 4, 115, 117, 118, 119, 121, 122, 123, 124, 126, 127, 151, 153, 155, 222, 113, 120, 125, 150 and 154.)

[Kle07] R. Kleinberg. Geographic routing using hyperbolic space. In *IEEE INFOCOM 2007-26th IEEE International Conference on Computer Communications*, pages 1902–1909. IEEE, 2007. (Cited in pages 23 and 21.)

[KLNS15] A. Kosowski, B. Li, N. Nisse, and K. Suchan. k-Chordal Graphs: from Cops and Robber to Compact Routing via Treewidth. *Algorithmica*, 72(3):758–777, 2015. (Cited in pages 23, 77, 21 and 75.)

[Klo96] T. Kloks. Treewidth of circle graphs. *International Journal of Foundations of Computer Science*, 7(2):111–120, 1996. (Cited in pages 78 and 76.)

[KM02] J. H. Koolen and V. Moulton. Hyperbolic bridged graphs. *European Journal of Combinatorics*, 23(6):683–699, 2002. (Cited in pages 25, 40, 65, 23, 38 and 64.)

[KNS13] W. Sean Kennedy, Onuttom Narayan, and Iraj Saniee. On the hyperbolicity of large-scale networks. Technical Report arXiv:1307.0031, ArXiv, 2013. (Cited in pages 55, 57, 58, 59, 73, 53, 56 and 71.)

[Kor11] A. Korolova. Privacy Violations Using Microtargeted Ads: A Case Study. *Journal of Privacy and Confidentiality*, 3(1):27–49, 2011. (Cited in pages 160 and 159.)

[KPK+10] D. Krioukov, F. Papadopoulos, M. Kitsak, A. Vahdat, and M. Boguná. Hyperbolic geometry of complex networks. *Physical Review E*, 82(3):036106, 2010. (Cited in pages 16, 53, 14 and 51.)

[Kre89] M. W. Krentel. Structure in locally optimal solutions. In *Foundations of Computer Science, 1989., 30th Annual Symposium on*, pages 216–221. IEEE, 1989. (Cited in pages 157 and 155.)

[KS06a] D. Kratsch and J. Spinrad. Between $\wr(nm)$ and $\wr(n^\alpha)$. *SIAM Journal on Computing*, 36(2):310–325, 2006. (Cited in pages 65, 86, 88, 63 and 84.)

[KS06b] D. Kratsch and J. Spinrad. Minimal fill in $o(n^{2.69})$ time. *Discrete mathematics*, 306(3):366–371, 2006. (Cited in pages 86 and 84.)

[KS15] K. Kawarabayashi and A. Sidiropoulos. Beyond the euler characteristic: approximating the genus of general graphs. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing (STOC)*, pages 675–682. ACM, 2015. (Cited in pages 105 and 103.)

[KSSC99] M. Karonski, E. R. Scheinerman, and K. B. Singer-Cohen. On random intersection graphs: The subgraph problem. *Combinatorics, Probability and Computing*, 8(1&2):131–159, 1999. (Cited in pages 168, 189, 166 and 187.)

[KW10] B. Krishnamurthy and C. E. Wills. On the leakage of personally identifiable information via online social networks. *SIGCOMM Computer Communication Review*, 40(1), jan 2010. (Cited in pages 160 and 158.)

[Lan14] A. Lancin. *Study of complex networks properties for the optimization of routing models*. Theses, Université Nice Sophia Antipolis, December 2014. (Cited in pages 18 and 16.)

[LB62] C Lekkeikerker and J Boland. Representation of a finite graph by a set of intervals on the real line. *Fundamenta Mathematicae*, 51(1):45–64, 1962. (Cited in pages 37 and 34.)

[LDL+14] M. Lécuyer, G. Ducoffe, F. Lan, A. Papancea, T. Petsios, R. Spahn, A. Chaintreau, and R. Geambasu. Xray: Enhancing the web's transparency with differential correlation. In *USENIX Security Symposium*, pages 49–64, 2014. (Cited in pages 5, 159, 160, 161, 162, 163, 165, 166, 168, 169, 170, 176, 188, 189, 223, 157, 164, 167, 174, 186 and 187.)

[LdMS98] C. L. Lucchesi, C. P. de Mello, and J. L. Szwarcfiter. On clique-complete graphs. *Discrete Mathematics*, 183(1):247–254, 1998. (Cited in pages 44 and 42.)

[Leh74] P. Lehot. An optimal algorithm to detect a line graph and output its root graph. *Journal of the ACM (JACM)*, 21(4):569–575, 1974. (Cited in pages 64 and 62.)

[Lei93] H.-G. Leimer. Optimal decomposition by clique separators. *Discrete mathematics*, 113(1-3):99–123, 1993. (Cited in pages 86 and 84.)

[LG14] F. Le Gall. Powers of tensors and fast matrix multiplication. In *Proceedings of the 39th international symposium on symbolic and algebraic computation*, pages 296–303. ACM, 2014. (Cited in pages 56 and 54.)

[LLDM09] J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney. Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters. *Internet Mathematics*, 6(1):29–123, 2009. (Cited in pages 2, 16, 220 and 14.)

[LMT17] E. Le Merrer and G. Trédan. Uncovering influence cookbooks : Reverse engineering the topological impact in peer ranking services. In *The 20th ACM Conference on Computer-Supported Cooperative Work and Social Computing*. ACM, 2017. (Cited in pages 189 and 187.)

[Lok10] D. Lokshtanov. On the complexity of computing treelength. *Discrete Applied Mathematics*, 158(7):820–827, 2010. (Cited in pages 78, 83, 91, 94, 95, 103, 111, 76, 82, 89, 92, 93, 101 and 109.)

[Lot15] Z. Lotker. Voting algorithm in the play julius caesar. In *ASONAM '15*, pages 848–855. ACM, 2015. (Cited in page 24.)

[LSS+15] M. Lecuyer, R. Spahn, Y. Spiliopolous, A. Chaintreau, R. Geambasu, and D. Hsu. Sunlight: Fine-grained targeting detection at scale with statistical confidence. In *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security*, CCS '15, pages 554–566, New York, NY, USA, 2015. ACM. (Cited in pages 160, 184, 188, 159, 182 and 186.)

[LT15] S. Li and G. H. Tucci. Traffic congestion in expanders and (p, δ)–hyperbolic spaces. *Internet Mathematics*, 11(2):134–142, 2015. (Cited in pages 23, 73, 21, 71 and 72.)

[Mal15] A. Malyshev. Expanders are order diameter non-hyperbolic. Technical report, ArXiv, 2015. (Cited in pages 50, 72, 48 and 70.)

[Mat12] D. Mattioli. WSJ.com - On Orbitz, Mac Users Steered to Pricier Hotels, 2012. (Cited in pages 2, 160, 219 and 158.)

[MGEL12] J. Mikians, L. Gyarmati, V. Erramilli, and N. Laoutaris. Detecting price and search discrimination on the internet. In *Proceedings of the 11th ACM Workshop on Hot Topics in Networks*, pages 79–84. ACM, 2012. (Cited in pages 170 and 168.)

[MGHB15] H. Miao, P. Gao, M. Hajiaghayi, and J. Baras. Hypercubemap: Optimal social network ad allocation using hyperbolic embedding. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*, pages 357–362. ACM, 2015. (Cited in pages 15 and 14.)

[MLH08] Clémence Magnien, Matthieu Latapy, and Michel Habib. Fast computation of empirically tight bounds for the diameter of massive graphs. *ACM Journal of Experimental Algorithmics*, 13, 2008. (Cited in pages 16, 70, 14 and 68.)

[MOS04] E. Mossel, R. O'Donnell, and R. A. Servedio. Learning functions of k relevant variables. *Journal of Computer and System Sciences*, 69(3):421–434, 2004. (Cited in pages 163, 170, 187, 188, 161, 168, 185 and 186.)

[MP14] D. Mitsche and P. Prałat. On the hyperbolicity of random graphs. *The Electronic Journal of Combinatorics*, 21(2):2–39, 2014. (Cited in pages 40, 45, 38 and 43.)

[MP15] A. Martínez-Pérez. Chordality properties and hyperbolicity on graphs. Technical report, ArXiv, 2015. (Cited in pages 41 and 39.)

[MRSV10] J. Michel, J. M. Rodríguez, J. M. Sigarreta, and M. Villeta. Gromov hyperbolicity in cartesian product graphs. *Proceedings-Mathematical Sciences*, 120(5):593–609, 2010. (Cited in pages 44 and 42.)

[MS99] V. Moulton and M. Steel. Retractions of finite distance functions onto tree metrics. *Discrete Applied Mathematics*, 91(1):215–233, 1999. (Cited in pages 15, 22, 14 and 20.)

[MT01] B. Mohar and C. Thomassen. *Graphs on surfaces*, volume 10. JHU Press, 2001. (Cited in pages 105 and 103.)

[MW13] J. R. Marden and A. Wierman. Distributed welfare games. *Operations Research*, 61(1):155–168, 2013. (Cited in pages 119, 120, 117 and 118.)

[Mye13] R. B. Myerson. *Game theory*. Harvard university press, 2013. (Cited in pages 123 and 121.)

[New02] M. E. J. Newman. Assortative mixing in networks. *Physical review letters*, 89(20):208701, 2002. (Cited in pages 24 and 22.)

[Nis14] N. Nisse. *Algorithmic complexity: Between Structure and Knowledge How Pursuit-evasion Games help.* Accreditation to supervise research, Université Nice Sophia Antipolis, May 2014. (Cited in pages 32 and 30.)

[NRS04] S. Nikoletseas, C. Raptopoulos, and P. Spirakis. The existence and efficient construction of large independent sets in general random intersection graphs. In *International Colloquium on Automata, Languages, and Programming*, pages 1029–1040. Springer, 2004. (Cited in pages 178 and 176.)

[NS95] W. D. Neumann and M. Shapiro. Automatic structures, rational growth, and geometrically finite hyperbolic groups. *Inventiones mathematicae*, 120(1):259–287, 1995. (Cited in pages 20 and 18.)

[NS11] O. Narayan and I. Saniee. Large-scale curvature of networks. *Physical Review E*, 84(6):066108, 2011. (Cited in pages 2, 11, 23, 220, 9 and 21.)

[NST15] O. Narayan, I. Saniee, and G. H. Tucci. Lack of hyperbolicity in asymptotic erdös–renyi sparse random graphs. *Internet Mathematics*, 11(3):277–288, 2015. (Cited in pages 45 and 43.)

[NW83] R. Nowakowski and P. Winkler. Vertex-to-vertex pursuit in a graph. *Discrete Mathematics*, 43(2):235–239, 1983. (Cited in pages 32 and 30.)

[OM16] I. Olkin and A. W. Marshall. *Inequalities: theory of majorization and its applications*, volume 143. Academic press, 2016. (Cited in page 128.)

[OR94] M. J. Osborne and A. Rubinstein. *A course in game theory*. MIT press, 1994. (Cited in pages 123 and 121.)

[Pap95] P. Papasoglu. Strongly geodesically automatic groups are hyperbolic. *Inventiones mathematicae*, 121(1):323–334, 1995. (Cited in pages 32 and 30.)

[Pap96] P. Papasoglu. An algorithm detecting hyperbolicity. *Geometric and computational perspectives on infinite groups*, 25:193–200, 1996. (Cited in page 55.)

[Pap03] C. H. Papadimitriou. *Computational complexity*. John Wiley and Sons Ltd., 2003. (Cited in pages 138 and 136.)

[PLF02] R. S. Parpinelli, H. S. Lopes, and A. A. Freitas. Data mining with an ant colony optimization algorithm. *IEEE transactions on evolutionary computation*, 6(4):321–332, 2002. (Cited in pages 189 and 187.)

[PR05] C. H. Papadimitriou and D. Ratajczak. On a conjecture related to geometric routing. *Theoretical Computer Science*, 344(1):3–14, 2005. (Cited in pages 22 and 21.)

[Pri94] E. Prisner. A common generalization of line graphs and clique graphs. *Journal of Graph Theory*, 18(3):301–313, 1994. (Cited in pages 44 and 42.)

[Pri95] E. Prisner. *Graph dynamics*, volume 338. CRC Press, 1995. (Cited in pages 44 and 42.)

[PRST13] A. Portilla, J. M. Rodrıguez, J. M. Sigarreta, and E. Tourıs. Gromov hyperbolic directed graphs. *Appl. Sinica*, 2013. (Cited in pages 74 and 72.)

[PS97] A. Parra and P. Scheffler. Characterizations and algorithmic applications of chordal graph embeddings. *Discrete Applied Mathematics*, 79:171–188, 1997. (Cited in pages 84 and 82.)

[PS08] P. N. Panagopoulou and P. G. Spirakis. A game theoretic approach for efficient graph coloring. In *ISAAC'08*, pages 183–195, 2008. (Cited in pages 119, 122, 126, 127, 131, 117, 120, 124, 125 and 129.)

[Qui83] A. Quilliot. *Problèmes de jeux, de point fixe, de connectivité et de représentation sur des graphes, des ensembles ordonnés et des hypergraphes*. PhD thesis, Thèse de doctorat d'état, Université de Paris VI, France, 1983. (Cited in pages 32 and 30.)

[Reu16] Bernhard Reus. Robustness of p. In *Limits of Computation*, pages 173–181. Springer, 2016. (Cited in pages 17 and 15.)

[RS84] N. Robertson and P. D. Seymour. Graph minors. III. planar tree-width. *Journal of Combinatorial Theory, Series B*, 36(1):49–64, 1984. (Cited in pages 109 and 107.)

[RS86] N. Robertson and P.D. Seymour. Graph minors. II. algorithmic aspects of tree-width. *Journal of algorithms*, 7(3):309–322, 1986. (Cited in pages 39, 76, 37 and 74.)

[RS10] P. Raghavendra and D. Steurer. Graph expansion and the unique games conjecture. In *ACM STOC*, pages 755–764. ACM, 2010. (Cited in pages 72 and 70.)

[RST94] N. Robertson, P. Seymour, and R. Thomas. Quickly excluding a planar graph. *Journal of Combinatorial Theory, Series B*, 62(2):323–348, 1994. (Cited in pages 109 and 107.)

[RW09] R. T. Rockafellar and R. J.-B. Wets. *Variational analysis*, volume 317. Springer Science & Business Media, 2009. (Cited in pages 20 and 18.)

[Saf13] SafeGov.org. Declaration of Kyle C. Wong in Support of Google Inc.'s Opposition to Plaintiffs' Motion for Class Certification, 2013. (Cited in pages 160 and 158.)

[Sch91] A. A. Schäffer. Simple local search problems that are hard to solve. *SIAM journal on Computing*, 20(1):56–87, 1991. (Cited in pages 146 and 145.)

[Sch92] P. Scheffler. Optimal embedding of a tree into an interval graph in linear time. *Annals of Discrete Mathematics*, 51:287–291, 1992. (Cited in pages 105 and 103.)

[SD10] S.-C. Sung and D. Dimitrov. Computational complexity in additive hedonic games. *European Journal of Operational Research*, 203(3):635–639, 2010. (Cited in pages 154 and 152.)

[Sei74] D. Seinsche. On a property of the class of n-colorable graphs. *Journal of Combinatorial Theory, Series B*, 16(2):191–193, 1974. (Cited in pages 41 and 39.)

[Sey16] P. Seymour. Tree-chromatic number. *Journal of Combinatorial Theory, Series B*, 116:229–237, 2016. (Cited in pages 77 and 75.)

[SG11] Mauricio A. Soto Gómez. *Quelques propriétés topologiques des graphes et applications à internet et aux réseaux*. PhD thesis, Univ. Paris Diderot (Paris 7), 2011. (Cited in pages 30, 34, 38, 60, 61, 28, 32, 36, 58 and 59.)

[Sha11] Y. Shang. Lack of gromov-hyperbolicity in colored random networks. *PanAmerican Mathematical Journal*, 21(1):27–36, 2011. (Cited in pages 45 and 43.)

[Sha12] Y. Shang. Lack of gromov-hyperbolicity in small-world networks. *Open Mathematics*, 10(3):1152–1158, 2012. (Cited in pages 45 and 43.)

[Sha13] Y. Shang. Non-hyperbolicity of random graphs with given expected degrees. *Stochastic Models*, 29(4):451–462, 2013. (Cited in pages 45, 46, 43 and 44.)

[Shc13a] V. Shchur. A quantitative version of the morse lemma and quasi-isometries fixing the ideal boundary. *Journal of Functional Analysis*, 264(3):815–836, 2013. (Cited in pages 20 and 18.)

[Shc13b] V. Shchur. *Quasi-isometries between hyperbolic metric spaces, quantitative aspects*. PhD thesis, Université Paris-Sud, 2013. (Cited in pages 49, 59, 68, 47, 57 and 66.)

[Shi77] D. R. Shier. A min-max theorem for p-center problems on a tree. *Transportation Science*, 11(3):243–252, 1977. (Cited in pages 70 and 68.)

[ST08] Y. Shavitt and T. Tankel. Hyperbolic embedding of internet graph for distance estimation and overlay construction. *IEEE/ACM Transactions on Networking (TON)*, 16(1):25–36, 2008. (Cited in pages 23 and 21.)

[Sys79] M.M. Sysło. Characterizations of outerplanar graphs. *Discrete Mathematics*, 26(1):47–53, 1979. (Cited in pages 63 and 61.)

[Tar72] R. E. Tarjan. Depth-first search and linear graph algorithms. *SIAM journal on computing*, 1(2):146–160, 1972. (Cited in pages 77, 85, 90, 75, 83 and 88.)

[Tar85] R. E. Tarjan. Decomposition by clique separators. *Discrete Mathematics*, 55(2):221 – 232, 1985. (Cited in pages 64, 78, 85, 86, 62, 76, 83 and 84.)

[TDDW15] M.C. Tschantz, A. Datta, A. Datta, and J.M. Wing. A methodology for information flow experiments. In *Computer Security Foundations Symposium (CSF), 2015 IEEE 28th*, pages 554–568, July 2015. (Cited in pages 166 and 164.)

[Ten16] S.-H. Teng. Scalable algorithms for data and network analysis. *Foundations and Trends in Theoretical Computer Science*, 12(1–2):1–274, 2016. (Cited in pages 1 and 219.)

[The14] The Guardian. Snapchat's expired snaps are not deleted, just hidden, 2014. (Cited in pages 2 and 219.)

[Tho89] C. Thomassen. The graph genus problem is NP-complete. *Journal of Algorithms*, 10(4):568–576, 1989. (Cited in pages 105 and 103.)

[Tuc13] G. H. Tucci. Non-hyperbolicity in random regular graphs and their traffic characteristics. *Central European Journal of Mathematics*, 11(9):1593–1597, 2013. (Cited in pages 45 and 43.)

[Twi] Twitter Usage Statistics. `http://www.internetlivestats.com/twitter-statistics/`. (Cited in pages 1 and 219.)

[TY84] R. E. Tarjan and M. Yannakakis. Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs. *SIAM Journal on Computing*, 13(3):566–579, 1984. (Cited in pages 89 and 87.)

[Ueh99] R. Uehara. Tractable and intractable problems on generalized chordal graphs. *Models of Computation and Algorithms*, 1093:27–32, 1999. (Cited in pages 41 and 39.)

[Val12] G. Valiant. Finding correlations in subquadratic time, with applications to learning parities and juntas. In *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*, pages 11–20. IEEE, 2012. (Cited in pages 163, 188, 161 and 186.)

[VDSVS12] J. Valentino-Devries, J. Singer-Vine, and A. Soltani. WSJ.com - Websites Vary Prices, Deals Based on Users' Information, 2012. (Cited in pages 2, 160, 219 and 158.)

[VS14] K. Verbeek and S. Suri. Metric embedding, hyperbolic space, and social networks. In *Proceedings of the thirtieth annual symposium on Computational geometry*, page 501. ACM, 2014. (Cited in pages 15, 17, 22, 45, 46, 48, 49, 55, 68, 14, 20, 43, 44, 53 and 66.)

[VWW10] V. Vassilevska Williams and R. Williams. Subcubic equivalences between path, matrix and triangle problems. In *51st Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 645–654. IEEE, 2010. (Cited in pages 65, 87, 63 and 85.)

[VWWWY15] V. Vassilevska Williams, J. R. Wang, R. Williams, and H. Yu. Finding four-node subgraphs in triangle time. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1671–1680. SIAM, 2015. (Cited in pages 67 and 65.)

[WAPL14] Y. Wu, P. Austrin, T. Pitassi, and D. Liu. Inapproximability of treewidth and related problems. *J. Artif. Intell. Res. (JAIR)*, 49:569–600, 2014. (Cited in pages 77 and 75.)

[Whi92] H. Whitney. Congruent graphs and the connectivity of graphs. In *Hassler Whitney Collected Papers*, pages 61–79. Springer, 1992. (Cited in pages 3, 17, 42, 221, 15 and 40.)

[Wil16] Vassilevska Williams. Fine-grained algorithms and complexity (invited talk). In *33rd Symposium on Theoretical Aspects of Computer Science (STACS 2016)*, volume 47, page 3. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2016. (Cited in pages 17, 64, 67, 87, 16, 62, 65 and 85.)

[WLJX12] C. Wang, T. Liu, W. Jiang, and K. Xu. Feedback vertex sets on tree convex bipartite graphs. In *COCOA 2012, Banff, AB, Canada*, pages 95–102, 2012. (Cited in pages 92, 98, 90 and 96.)

[WS98] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *nature*, 393(6684):440–442, 1998. (Cited in pages 2, 16, 220 and 14.)

[WZ11] Y. Wu and C. Zhang. Hyperbolicity and chordality of a graph. *the Electronic Journal of Combinatorics*, 18(1):P43, 2011. (Cited in pages 27, 35, 40, 41, 47, 58, 63, 65, 25, 33, 38, 39, 45, 56, 61 and 64.)

[Yan15] M. Yancey. An investigation into graph curvature's ability to measure congestion in network flow. Technical Report arXiv:1512.01281, ArXiv, 2015. (Cited in pages 23, 39, 69, 73, 21, 36, 67 and 71.)

Appendix A

Résumé de la thèse

A.1 Contexte

Le *partage d'information* en ligne a gagné en importance au cours de ces dernières décennies. Les chiffres parlent d'eux mêmes: en 2015 il y a eu 205 milliards de courriels échangés quotidiennement [Ema]; sur la même période, près de 500 millions de tweets par jour ont été envoyés [Twi]; on observe plus généralement une hausse du trafic Internet, qui est passé de 100 GB par jour en 1992 à 20 235 GBps en 2015 [Cisa]. De même le volume des données stockées a explosé, avec des prévisions de l'ordre de 40 zettaoctets pour 2020 [IDC].

Alors que nous entrons de plain-pied dans cette "ère du zettaoctet", les techniciens de l'information se retrouvent confrontés à diverses problématiques qui sont régulièrement relayées par les médias. Dans cette thèse, nous nous intéressons à deux de ces problématiques:

- **Le Passage à l'échelle** – défini dans [Ten16] comme l'impératif d'obtenir des algorithmes en temps quasi-linéaire en la taille des réseaux. Dans une perspective plus large, il y a une demande croissante d'algorithmes efficaces pour gérer les réseaux de communication. Ces demandes émanent de nombreux domaines scientifiques tels que ceux des télécommunications, de la bio-informatique, de la vision artificielle et de l'économie. La difficulté principale est qu'avec l'essor des échanges d'information et des collectes de données en ligne la taille des réseaux a augmenté, avec à présent des millions de serveurs dans certains centres de données [DCM], des milliards d'utilisateurs sur les réseaux sociaux [FBN], etc. Devant pareilles tailles mêmes les algorithmes cas d'école ne passent pas tous à l'échelle, ce qui accroît le fossé entre ce qui est calculable et ce qu'on veut calculer. Il y a donc lieu de redéfinir ce que signifie un calcul efficace, ou qui passe à l'échelle, dans ce contexte.

  Nous proposons des avancées dans cette direction en nous basant sur des outils de la théorie des graphes et de la complexité.

- **Le respect de la vie privée** – est défini dans [EDP] comme "*un droit qui interdit aux autorités publiques [et à toute autre organisation ou individu] d'exercer des mesures [de nature à rendre publique la vie privée des gens] à moins que certaines conditions soient vérifiées.*" Plus précisément, des inquiétudes naissent de la collecte effrénée de données par les entreprises en ligne, dont les dérives se font jour fréquemment [Gou14, Mat12, VDSVS12, The14]. De là le besoin

de modèles prédictifs afin que chaque individu puisse détecter ces dérives, voire même les identifier.

Nos principaux outils pour cette tâche sont ceux de la théorie de l'apprentissage et la théorie algorithmique des jeux.

Avant d'annoncer nos résultats dans la Section A.2, nous commençons par esquisser notre approche pour cette thèse. Le travail présenté est la somme de plusieurs problèmes combinatoires sur les graphes, dont l'étude est motivée par les deux problématique exposées ci-dessus. Puisque les solutions proposées pour ces problèmes doivent passer à l'échelle sur les grands réseaux, on s'intéresse tout particulièrement à l'étude *fine* de leur complexité.

En particulier, la Partie I est dédiée à l'étude de paramètres dans les graphes dont les relations avec les problématiques ci-dessus ont été montrées dans d'autres travaux [NS11]. L'étude des propriétés des "réseaux complexes", ainsi que de leurs applications, est un sujet bien établi [LLDM09, BAJ00, BKC09, WS98, DGM06]. Dans notre cas, l'accent est mis sur la proximité des métriques de graphes avec les métriques d'arbres [Gro87]. Ce sujet a reçu une attention croissante au cours des dernières décennies. Nous détaillons dans la Partie I comment les avantages et les inconvénients des arbres (avec d'un côté d'importantes applications algorithmiques mais de l'autre côté des vulnérabilités bien connues) peuvent s'étendre aux graphes qui sont (métriquement) proches des arbres.

Cette ligne principale de la thèse sera complétée dans la Partie II par l'analyse de deux processus dynamiques sur les graphes. Ces deux processus modélisent des aspects fondamentaux de la problématique du respect de la vie privée dans les réseaux de communications. En d'autres termes, l'objectif dans cette ligne secondaire de la thèse est de concevoir des outils (qui passent à l'échelle) afin de renforcer le respect de la vie privée dans ces réseaux.

A.2 Contributions

Notre travail est exposé dans deux parties disjointes et indépendantes l'une de l'autre. Leur contenu est présenté dans les Sections A.2.1 et A.2.2 ci-dessous.

Par ailleurs, l'annexe regroupe l'ensemble des articles par lesquels les résultats de cette thèse ont été publiés. En effet, nous avons fait le choix de ne pas inclure toutes les preuves dans le corps des chapitres, en partie pour des raisons de lisibilité car certaines d'entre elles dépassent allègrement la douzaine de pages. Seront seulement données les preuves qui, de notre point de vue, illustrent le mieux les techniques utilisées. Le tout accompagné d'esquisses des preuves les plus longues.

A.2.1 Partie I: Sur les graphes dont la métrique est proche de celle d'un arbre

Nous étudions dans la Partie I des propriétés métriques des graphes et des décompositions de graphes. L'objectif majeur de cette partie est l'étude fine de la complexité

de leur calcul. En particulier, ces propriétés peuvent-elles être calculées sur de très grands graphes, avec parfois des millions de noeuds et des milliards d'arêtes ? Nos pistes pour répondre à ces questions nous ont amené à étudier les relations entre les propriétés métriques d'un graphe et ses propriétés structurelles, topologiques, algébriques, etc.

A.2.1.1 Chapitre 2: Une vue d'ensemble sur l'hyperbolicité dans les graphes

Ce chapitre introduit la notion d'hyperbolicité dans les graphes. Ce paramètre donne des bornes sur la meilleure distorsion possible des distances dans un graphe quand il est plongé dans un arbre.

Tout d'abord nous démontrons plusieurs résultats, positifs comme négatifs, sur la complexité du calcul de ce paramètre. Plus précisément, sur le plan positif nous proposons une méthode de pré calcul afin de réduire la taille des graphes en entrée de nos algorithmes. Cette méthode utilise une décomposition des graphes bien connue, selon les cliques-séparatrices [BPS10]. Nous en faisons une analyse poussée. Sur un plan plus négatif, nous prouvons que reconnaître les graphes de petite hyperbolicité (au plus 1/2) est un problème de complexité équivalente à la détection de carrés induits dans un graphe. Ce résultat implique, sous certaines hypothèse de complexité standard, que calculer l'hyperbolicité d'un graphe est impossible en temps sous-cubique. Ces travaux ont été réalisés en collaboration avec Nathann Cohen, David Coudert et Aurélien Lancin [CD14, CCDL17].

Ensuite, nous établissons de nouvelles bornes sur ce paramètre dans des classes de graphes utilisées dans la conception des réseaux d'interconnexion de centre de données. Dans la pratique, nous avons utilisé ce résultat pour estimer fidèlement l'hyperbolicité de graphes de très grande taille sans le moindre calcul. Nous complétons ce résultat par une analyse fine des variations de l'hyperbolicité sous certaines opération bien connues sur les graphes telles que le graphe adjoint, le graphe des cliques, etc. Cette analyse prend une tout autre saveur dans les cas où l'opération peut facilement s'inverser (par exemple, la racine d'un graphe adjoint se calcule en temps linéaire [Whi92]), car elle donne alors de nouvelles méthodes de pré calcul pour le calcul de l'hyperbolicité d'un graphe. Ce travail a été réalisé en commun avec David Coudert [CD16a, CD16b].

A.2.1.2 Chapitre 3: Décompositions arborescentes avec des contraintes sur les distances dans les sacs

De nouveaux résultats sont présentés sur la complexité du calcul de *décompositions arborescentes* avec des contraintes sur les distances dans les sacs (*ç.a.d.* les sous-graphes résultant de la décomposition).

D'abord, on présente une analyse fine de la complexité du calcul de la décomposition d'un graphe selon ses cliques-séparatrices. La complexité de ce problème est prouvée équivalente, sous des hypothèses de complexité standard, à la détection de

triangles dans un graphe et au produit de deux matrices carrées. Sur un plan plus positif nous montrons que cette décomposition est calculable en temps quasi-linéaire pour des classes de graphes où la taille d'une plus grande clique est bornée par une constante. Ce travail a été réalisé en commun avec David Coudert [DC17].

Dans un deuxième temps, nous répondons à des questions ouvertes de la littérature sur la complexité du calcul de différents paramètres métriques des graphes, tous reliés à l'hyperbolicité (treebreadth, pathbreadth et pathlength). Nous prouvons que pour tous ces paramètres, leur calcul est un problème NP-difficile. En particulier, nous montrons que reconnaître les graphes de treebreadth au plus une est NP-complet. Cependant, nous prouvons que ce dernier problème devient polynomial si l'on se restreint aux graphes bipartis et aux graphes planaires. Ce travail a été réalisé en collaboration avec Sylvain Legay et Nicolas Nisse [DLN16a].

Enfin, nous étudions les relations entre une autre propriété métrique des graphes: la *treelength*, et une propriété *structurelle* bien connue: la *treewidth* des graphes. Nous bornons la treewidth par deux fonctions affines de la treelength dans les classes de graphes sans long cycle isométrique et de genre borné. Sur le plan algorithmique, on mentionne plusieurs applications de ce résultat. C'est un travail effectué en commun avec David Coudert et Nicolas Nisse [CDN16].

A.2.2 Partie II: Le respect de la vie privée à grande échelle dans les réseaux sociaux

Deux problèmes autour du respect de la vie privée sont introduits et étudiés dans cette partie. Notre objectif est d'obtenir une analyse fine de leur complexité.

A.2.2.1 Chapitre 4: le calcul d'équilibres dans les jeux de coloration

Nous considérons un jeu de coloration sur les graphes. Ce jeu a été proposé dans [KL13] pour modéliser la dynamique des communautés dans les réseaux sociaux. D'autres applications avaient été précédemment suggérées pour ce jeu, dont la sécurisation de groupes de communication [CKPS10].

Nous présentons de nouveaux résultats sur la complexité du calcul d'équilibres dans ce jeu. Pour être plus précis, nous nous concentrons tout d'abord sur le calcul d'équilibres par *meilleure réponse*. Cette méthode de recherche locale permet de calculer, pour tout entier k, un équilibre de Nash robuste contre toutes les coalitions d'agents possibles de taille au plus k. Sur le plan positif, nous établissons le temps de convergence exact de cette méthode dans le pire cas, pour $k \leq 2$. Toutefois, sur le plan négatif, nous prouvons que ce temps de convergence n'est pas polynomialement borné dès que $k \geq 4$. Ce résultat négatif répond aux questions ouvertes de [EGM12, KL13]. Ce travail a été effectué en commun avec Dorian Mazauric et Augustin Chaintreau [DMC13a, DMC17].

Nous complétons ce dernier résultat par une analyse plus fine de la complexité du calcul d'un équilibre de Nash dans le jeu de coloration (robuste contre les déviations de n'importe quel agent). On montre que ce problème est P-difficile, ce qui suggère

qu'il est intrinsèquement séquentiel [Duc16].

Enfin, le reste du chapitre est dédié à une généralisation naturelle du jeu de coloration sur les graphes arêtes-pondérés. Nous donnons des conditions suffisantes pour l'existence d'équilibres dans ces jeux qui dépendent de la structure du graphe sous-jacent (notamment, de sa maille). Nous proposons également des constructions surprenantes de jeux pour lesquels de tels équilibres n'existent pas. Pour finir, il est prouvé que reconnaître les jeux de coloration généralisés qui admettent de tels équilibres est un problème NP-complet. Des extensions de tous ces résultats à des classes de jeux plus générales sont aussi discutées. C'est un travail commun avec Dorian Mazauric et Augustin Chaintreau [DMC12, DMC13a, DMC17].

A.2.2.2 Chapitre 5: Apprentissage de formules logiques dans un modèle bruité

Nous consacrons le dernier chapitre à un problème d'apprentissage dont le contexte peut s'énoncer comme suit. Soient un ensemble $\mathcal{D}$ (qui représente des mots-clefs) et un graphe où chaque sommet est étiqueté par un sous-ensemble de $\mathcal{D}$ (*ç.a.d.* une collection de mots-clefs présents dans les courriels d'un utilisateur). On assigne un Booléen à chaque sommet selon un processus (boîte noire) aléatoire, qui lui-même est corrélé à une certaine fonction Booléenne (inconnue) sur les étiquettes. Le problème posé est l'apprentissage de cette fonction. Nous entendons ainsi modéliser le problème de la détection de l'utilisation des données utilisateur dans les campagnes publicitaires en ligne.

D'abord nous proposons un algorithme pour apprendre la fonction dans un cas simple où elle dépend d'au plus une variable. Cet algorithme est à la base de méthodes d'apprentissage plus sophistiquées pour d'autres classes de fonctions — mais sous des hypothèses plus contraignantes. Par ailleurs il est montré que sans ces hypothèses supplémentaires, il est impossible d'apprendre la fonction dès qu'elle dépend d'au moins deux variables. Ce travail a été effectué en collaboration avec Mathias Lécuyer, Francis Lan, Max Tucker, Riley Sphan, Andrei Papancea, Theofilos Petsios, Augustin Chaintreau et Roxana Geambasu [LDL+14, DLCG15, DTC17, CD17].