Algorithmic complexity: Between Structure and Knowledge How Pursuit-evasion Games help.

Nicolas Nisse

Sophia Antipolis
N° d'ordre :

Habilitation à Diriger la Recherche

présentée pour obtenir le grade de
HDR
Spécialité : Informatique

—

Algorithmic complexity Between Structure and Knowledge

How Pursuit-evasion Games help

Nicolas Nisse

—

Soutenue le 26 mai 2014 devant le jury composé de :

Rapporteurs :	**Cyril GAVOILLE**	Professeur
	Dimitrios M. THILIKOS	Directeur de Recherche
	Peter WIDMAYER	Professeur
Examinateurs :	**Victor CHEPOI**	Professeur
	David COUDERT	Chargé de Recherche
	Fedor V. FOMIN	Professeur
	Pierre FRAIGNIAUD	Directeur de Recherche
	Jean-Charles RÉGIN	Professeur

Contents

Introduction

This document describes the work I did since I obtained my Ph.D. in 2007. Following my Ph.D. thesis where I mainly worked on *graph searching games* and *graph decompositions* [t-Nis07], my main contributions relate to *pursuit-evasion games* and their relationship with *graph structural properties*.

The theoretical studies I am dealing with have applications for solving telecommunication problems, mainly related to *routing*. Such problems are generally difficult (NP-hard) but may become easier when restricted to classes of graphs with specific structural properties[1]. Therefore, *understanding graph structures* became one important topic of my research. *Pursuit-evasion games* - where mobile entities try to capture other ones - are one of my main tools for this purpose. Indeed, several variants of these games are closely related to graph structures. For instance, *graph searching* is an algorithmic interpretation of *graph decompositions* and, therefore, it is an interesting (and fruitful) way to study them. Moreover (and somehow surprisingly), pursuit-evasion games are again of a great help to study problems of telecommunication networks, by providing new models.

Another important issue of my research is to deal with the increasing size of nowadays networks (traffic, data, etc.) that makes unpractical most of the existing solutions. In particular, centralized algorithms with a full knowledge of the environment are no longer adequate to deal with current telecommunication problems while achieving expected performance. Therefore, part of my research focuses on simple local algorithms that may be used in real large-scale networks. I also study models of *distributed computation* by themselves. I mainly focus on their limits: what can be computed when nodes or mobile agents have very little knowledge and/or few abilities?

The manuscript is divided into 4 chapters. Chapter 1 is dedicated to the study of several variants of turn-by-turn Pursuit-Evasion Games, mostly to the *Cops and Robber* games. Chapter 2 focuses on graph decompositions and their relationship with graph searching. Chapter 3 treats another aspect of Pursuit-Evasion games with a study of variants of *Graph Searching* games, both in centralized and distributed settings. Finally, Chapter 4 deals with routing problems in various environments and with distributed computing. In the whole manuscript, three complementary aspects of my research are deeply interleaving:

- **Characterization of graph structural properties.** I study various graph properties through new point of views that give rise to alternative definitions and characterizations;
- **Recognition of graph properties.** I take advantage of the proposed characterizations to design efficient algorithms that either decide whether a graph satisfies some property, or compute some structures in graphs.

[1] For basics notions on graphs, algorithms and computational complexity, see [CLRS01, BM08]

- **Application** of my theoretical study on problems arising **in telecommunication networks.**

The work presented here is the result of collaborations with colleagues of the "MASCOATI" project-team and with colleagues from various French or foreign universities[2]. In particular, some of the work has been done during the Ph.D. theses of Ronan P. Soares [Soa13] and Bi Li under my co-supervision.

Before describing more precisely the organization of the manuscript, I start with a general introduction on Pursuit-Evasion games. Then, I present the context of our work on telecommunication networks, focusing on the algorithmic challenges that are arising nowadays.

Pursuit-evasion games

Pursuit-evasion refers to the problems in which one team of mobile entities, the *cops* or *searchers* or *hunters* or *pursuers* etc. attempts to *track down* another group of mobile entities, the *robbers* or *fugitives* or *rabbits* or *evaders* etc. The dual problem for the evaders is to avoid being caught. These are natural games arising from a wide range of applications, from the practical ones such as search and rescue (e.g., rescuing a speleologist in a cave, Breish (1967) [Bre12]), surveillance, monitoring, military strategy to trajectory tracking, etc. to the handling of abstract mathematical and theoretical computer science concepts (e.g., Graph Minor Theory, Robertson and Seymour (1983-2004)). Hence, Pursuit-evasion games have been studied in various divergent disciplines, e.g., graph theory, differential games, robotics, control theory, geometric algorithms, etc. (see [FT08, CHI11] for recent surveys).

One famous such game is the *Lion-and-Man problem* stated by Rado in 1930 [Lit53] where a lion and a man with the same maximum speed are confined in a closed arena. Surprisingly, Besicovitch showed that the man can indefinitely evade the lion while the lion can get arbitrarily close to him [Lit53]. On the other hand, when the man and the lion are moving within the non-negative quadrant of the plane (both initial coordinates of the lion are larger than those of the man), then the lion will eventually be sated [Sga01]. Many other variations of the problem have been considered (e.g., [KR05, BBH08, KI09]). This simple example illustrates the fact that many parameters (such as the environment where the game is played, or whether the robber must be touched or only approached to be captured, etc.) may impact the solution of the Pursuit-evasion games. Therefore, the studies of these games can be categorized into many variants, approaches and techniques.

In the *differential* approach, the motions of the players are described by differential equations well reflecting the physical constraints of the players (embodied by robots, drones, etc.). The solutions of these equations are used as control inputs to achieve the objective of the game. Game theory and probabilistic approaches are

[2]Since most of the contributions presented in this manuscript is the result of collaboration with other colleagues, I use "we" throughout the document. I will use "I" when personal opinions are presented or when further work is described.

widely used to address these games (see the textbook of Basar and Olsder [BO99b]). The complexity of the differential equations generally limits the solutions to heuristics. In this manuscript, I consider only combinatorial techniques and exact (i.e., achieving optimal solutions) or approximation algorithms.

In the combinatorial approach, research on Pursuit-evasion games falls into two main categories depending on whether the environment is represented geometrically using *polygonal models* or by *graphs*. In the former case, the space is continuous and the complexity comes from obstacles (holes) that impede visibility. Some recent results in this area can be found in [SY92, GLL$^+$99, CHI11, KS12, BKIS12, KS13]. In my work, I consider only the graphical models, i.e., both players move in a graph.

Another important distinction that I would like to mention deals with the objective of the games. Pursuit-evasion problems have been addressed either regarding the *worst-case behavior* of the evader or using an *average-case approach*. In the latter approach, the objective is generally to minimize the expected time of capture. In all cases, the games highly depend on the assumptions about the knowledge of the players. I consider only the worst-case approach, i.e., when the evader must be captured regardless of its strategy.

To sum up, in Chapters 1 to 3, I address Pursuit-evasion games played on graphs. I consider worst-case behavior of the evader, that is, I mainly consider deterministic strategies ensuring that the pursuers eventually capture the evader whatever it does. I survey the literature on two families of such games: the *Cops and Robber games* (Sections 1.1 and 1.2) and the *Graph Searching games* (Sections 3.1 and 3.2, and Section 2.3 for directed graphs). My contributions on Pursuit-evasion games mainly depend on the rules of the games considered, e.g., visibility of the players, their speed, their knowledge of the graph, etc. and on the class of graphs in which the games are played. This allows to provide new characterizations for several structural properties of graphs that may be used for algorithmic purposes.

Another important aspect of my study of Pursuit-evasion games is that they allow us to model problems arising in telecommunication networks, e.g., prefetching (Section 1.3) or routing reconfiguration problems (Section 2.3). Using the proposed models and the studied graph structural properties allows the design of efficient algorithms for several applications in telecommunication networks. As shown below, structural properties of current large-scale telecommunication networks play a crucial role for facing the applications efficiently.

Telecommunication networks and algorithmic perspectives

My work comes from problems arising in telecommunication networks. An important aspect is to design efficient algorithms for solving these problems in networks of large size.

Both the growth of (tele)communication networks and the changes in their usage lead to new algorithmic challenges. The size of large-scale networks (social networks, the Internet) is exponentially growing and their dynamics is increasing. For instance, recent telecommunication networks consist of several thousands or

even millions of nodes and they are still growing. For example, more than $65,000$ Autonomous Systems (AS) compose the Internet [fIDAC]. On the other hand, due to the growth of networks and the emergence of new applications and uses (video streaming, data sharing, cloud computing, social networking (Facebook), etc.), the traffic drastically grows. For instance, in optical networks the traffic is growing exponentially and this trend is expected to continue [cis12]. Those evolutions lead to new algorithmic challenges.

Solutions that have been proposed so far generally do not scale up to the magnitude of today's networks and cannot be straightly adapted to the dynamics of both topologies and traffic. For instance, due to time or space constraints, most polynomial-time algorithms cannot be used efficiently in large networks: A naive implementation of an algorithm with running time $O(n^4)$ takes several months (on a standard laptop) when $n = 10,000$. One possible approach to design efficient algorithms is to specialize the algorithms to particular network classes, that is, to take advantage of the structural properties of real life networks in order to design algorithms with better practical performance on such networks. Indeed, it is well known that most large-scale networks (the Internet, twitter, citation graphs, youtube links) share structural properties such as logarithmic diameter, power-law degree distribution and high clustering coefficient [BA99, AJB99, FFF99, SFFF03]. A lot of work has been done to discover/measure the specific properties of large-scale networks and then use this information for the design of efficient algorithms. For instance, distributed routing has been widely studied in various environments such as greedy geometric routing in *small-worlds* [WS98, Kle00b, Kle00a, FG10, GS11] or compact routing in power-law networks [KFY04, CSTW12] (see Section 4.2). In Chapter 4, among other contributions on routing in several networking environments, we provide some new results on compact routing using graph structures.

On the theoretical side, difficult (NP-complete) problems have been studied in order to design efficient algorithms solving them (almost) independently of the size of the inputs. More formally, the Fixed Parameterized Tractability (FPT) complexity paradigm aims at understanding where the difficulty of a problem lies if not in the size of the instance [DF99, FG06, Nie06]. That is, given a difficult problem, the question is to capture the "structure" (formalized by a parameter k) for which the problem may be easy to solve. A typical example of such a parameter is the *treewidth* of graphs that measures the proximity of a graph with a tree [Ros74, RS86a, Bod98, Bod07]. Roughly, a graph with small treewidth "looks" like a tree, i.e., can be decomposed in a tree-like manner using small separators. Hence, problems that can be efficiently solved in trees are generally tractable in bounded treewidth graphs by using dynamic programming on the tree-decomposition (e.g., [Cou90]). Importantly and surprisingly, the algorithmic counterpart of tree-decompositions is precisely the *graph searching game* where a team of searchers aims at capturing a visible and arbitrarily fast fugitive in a graph. In addition to its importance in the Graph Minor Theory [RS86a, RS90, RS91, RS03a, RS03b], treewidth plays a crucial role in many recent breakthrough results in parameterized complexity (see Section 2.1). Chapter 2 is devoted to our contributions on tree-decompositions that mainly rely on a

graph searching approach.

Another aspect of large-scale networks is their high dynamicity (for instance, peer-to-peer networks are characterized by an important churn: nodes arrive or leave the network at any time, etc.). Hence, it is important to design algorithms that can easily cope with the changes and that tolerate some faults, or at least can recover quickly (back) to a normal behaviour. Moreover, recent networks are not defined in a global way. For instance, a social network is emerging through local interactions growing thanks to the acquaintances of each person, a priori without any global structure. This aspect makes irrelevant several algorithms based on global knowledge/structure (e.g., tree-decompositions for dynamic programming, path-vector routing tables of the Border Gateway Protocol (BGP) in the Internet, etc.) or intrinsically centralized. Hence, current research concentrates on distributed or even local solutions, that is, solutions where nodes have limited memory and local knowledge of the network. It is important to note that "simple" problems (with polynomial or even linear complexity) may become difficult or even impossible to solve when communication is restricted. In particular, efficient solutions that have been proposed in a centralized setting highly rely on the knowledge of some structures of the network that are intrinsically global (hyperbolicity, chordality, treewidth, etc.) and thus are difficult to compute in a distributed (and even more in a local) way. For instance, most of the properties of the Internet have been estimated by some heuristics on partial maps of CAIDA. Moreover, most of the efficient algorithms that take advantage of them either are highly centralized algorithms (e.g., FPT algorithms), or rely on a global and centralized structure (tree-decompositions, sparse cover, etc.). The mobile agents paradigm is a natural and interesting approach to understand how information is spread (or gathered) in networks and to design distributed algorithms. In Chapters 3 and 4, we study several tasks requiring the coordination of mobile agents in various distributed settings, and we also investigate the computation of graph properties in a local setting.

Organization of the manuscript and personal contributions

In this document[3], several topics such as pursuit-evasion games, graph decompositions, properties of graphs and applications to telecommunication networks (in particular routing) are deeply interleaving. The order chosen to present them mainly depends on the computational setting: from the study of graph properties and of several theoretical pursuit-evasion games in a centralized setting to the study of routing in various distributed models. While our work mainly concerns theoretical study of graphs and algorithms, several sections throughout the manuscript are directly related to applications in telecommunication networks.

In the remaining of this section, I summarize our main contributions that are structured in four chapters.

[3]**Notations.** Throughout the document, my contributions are cited in the margins and with some keys: [j-X] (international journals), [c-X] (international conferences), [s-X] (submitted papers) and [t-Nis07] (Ph.D. thesis).

Chapter 1: Turn-by-turn Two-Player Games in Graphs. This chapter is dedicated to the study of several turn-by-turn two-player games in graphs: the famous *Cops and Robber games* (Sections 1.1 and 1.2), the *surveillance game* that we introduced as a model for Web-prefetching (Section 1.3), and a fractional game that we introduced as a generic relaxation of many existing turn-by-turn two-player games (Section 1.4).

Sections 1.1 and 1.2 focus on the *Cops and Robber ($\mathcal{C}\&\mathcal{R}$) games*. In this game, Player 1 first places k tokens (the *cops*) on some vertices of a graph, then Player 2 places its token (the *robber*) at some vertex. Then, in turn, starting with the cops, each player may move each of its tokens along an edge of the graph. Player 1 wins if it manages to move one of its tokens on the node occupied by the robber. Player 2 wins otherwise. The *cop-number* of a graph G is the minimum integer k such that Player 1 using k cops wins in G whatever Player 2 does. The famous Meyniel conjecture asks whether the cop-number of any n-node graph is $O(\sqrt{n})$ [Fra87a]. In **Section 1.1**, we first propose a survey of the literature on this game. In particular, the Meyniel's conjecture has been solved for many graph classes. We prove it is asymptotically true in the class of graphs with bounded chordality (Th. 9). In **Section 1.2**, we introduce the variant in which each player has a *speed* (i.e., a token with speed $s \geq 1$ may move along at most s edges at each turn). We fully characterize the *cop-win* graphs in this variant, i.e., the graphs in which one cop with speed s wins against a robber with speed s' (Th. 13 and 14). In the classical game, 3 cops are sufficient in any planar graph [AF84]. We prove that, in contrast, when the robber is faster than the cops, the cop-number of planar graphs becomes unbounded (Th. 15). We then survey some other variants of Cops and Robber games including one where the cops have no permanent visibility or when the cops only need to be at some distance of the robber to capture it. We provide partial characterization of the cop-win graphs in the latter two variants (Th. 17-18 and Lemma 5).

survey on $\mathcal{C}\&\mathcal{R}$ game

[c-KLNS12, j-KLNS14]

[j-CCNV11]

[c-NS08, j-FGK+10]

[j-CCNV11]

In **Section 1.3**, we introduce the *surveillance game* in which Player 2 starts from a given *marked* node of a graph G. Alternatively, Player 1 may first mark k nodes of G and then Player 2 may move to a neighbor of its position. Player 2 wins if it reaches an unmarked node, while Player 1 wins if it manages to mark all nodes before this happens. The *surveillance number* of G from $v_0 \in V(G)$ is the minimum k such that Player 1 marking at most k nodes at each turn wins whatever Player 2 does starting from v_0. This parameter is related to the minimum amount of resources (e.g., bandwidth) required during a prefetching process (e.g., Web-caching). We show that computing the surveillance number is not FPT and NP-hard in many graph classes and even PSPACE-complete in DAGs (Th. 20-22). On the other hand, we design polynomial-time algorithms for trees and interval graphs (Th. 24 and 26). We then consider two restrictions of the surveillance game that are closer to the practical application of Web-prefetching: in the *connected surveillance game*, the set of marked vertices must always induce a connected subgraph, and in the *online surveillance game*, Player 1 discovers the graph as it marks the nodes. We first study the cost of connectivity, i.e., give some upper and lower bound on the

[c-FGJM+12, j-FGJM+14]

ratio between the connected and unrestricted surveillance numbers (Th. 27 and 28). We then prove that the best online strategy is actually the trivial one that consists for Player 1 to mark all neighbors of Player 2 at each step (Th. 29). *[c-GMN+ 13]*

Finally, **Section 1.4** is devoted to some preliminary results on our on-going work about *fractional games*. We define a framework generalizing and relaxing many games (including the ones mentioned above) where Player 1 may mark or move fraction of tokens at each turn while Player 2 may move fractions of its token at each turn. We design an algorithm for solving the fractional games. In particular, our algorithm runs in polynomial-time when the length of the game is bounded by 2 (in contrast, computing the surveillance game is NP-hard even when the game is limited to 2 turns). For some games, we also prove that the fractional variant provides some good "approximation". Unfortunately, this does not hold for Cops and Robber games. *[s-GNPS13]*

This research has been done in collaboration with J. Chalopin, V. Chepoi, F. V. Fomin, F. Giroire, P. Golovach, A. Jean-Marie, A. Kosowski, J. Kratochvil, I. Lamprou, B. Li, D. Mazauric, S. Pérennes, R. P. Soares, K. Suchan and Y. Vaxès.

Chapter 2: Tree Decomposition, Graph Searching and Applications. This chapter focuses on graph decompositions (in particular tree-decompositions), their algorithmic applications (Section 2.1) and their relationship with graph searching games (Section 2.2). A directed graph decomposition is studied through the problem of routing reconfiguration in networks (Section 2.3).

In **Section 2.1**, we first recall the formal definition of tree-decompositions and give a brief overview of recent work using tree-decompositions for designing efficient algorithms for various optimization problems. In particular, we detail several aspects of bidimensionality and of kernelization theories. We then focus on the problem of computing the *chordality* of graphs. In particular, we prove that the treewidth of a graph is at most its chordality times its maximum degree (Corollary 2). This result follows our definition of a restriction of tree-decomposition that can be efficiently computed in a wide family of graphs (Th. 33). ***Brief survey on algorithmic applications of treewidth*** *[c-KLNS12, j-KLNS14]*

Section 2.2 is devoted to the study of graph searching games and their relationship with tree-decompositions. In graph searching, a team of searchers aims at capturing an arbitrarily fast fugitive which is either visible or invisible. Searchers' strategies are equivalent to tree-decompositions (visible case) or to path-decompositions (invisible case). Non-deterministic graph searching (defined in [t-Nis07]) allows us to establish a bridge between treewidth and pathwidth through the notion of branched treewidth. We provide several results on non-deterministic graph searching in trees. In particular, we design a polynomial-time algorithm for approximating the branched treewidth of trees up to a factor of two (Th. 37). We then extend the notion of branched tree-decomposition to the notion of partitioning-trees that allows us to generalize many distinct width parameters of graphs. In particular, we generalize the duality results of Seymour and Thomas [ST93] (Th. 41) and the FPT-algorithms to compute width parameters. We conclude the section by a survey *[c-FFN05, c-MN07, j-MN08, j-FFN09]* *[s-ACN07]* *[j-AMNT09, s-BBM+ 13]*

Survey on digraph decompositions

on digraph decompositions which, unfortunately, do not fall into our generalization framework.

Section 2.3 addresses the problem of digraph decompositions through the *processing game* (variant of digraph searching) introduced by Coudert *et al.* [CPPS05] for modeling the routing reconfiguration problem in WDM networks. Given a network with a given *routing* (paths and wavelengths) for a set of requests, the goal is to reach a new routing configuration without disturbing too much the network's customers. The *process number* of the dependency digraph (w.r.t. to the initial and final routings) is related to the number of interrupted requests during a routing reconfiguration process. We show that the processing game is monotone and therefore is equivalent to a digraph decomposition (Th. 43). Then, we study the tradeoff between the total number and the maximum number of concurrent interruptions during the reconfiguration process, which is equivalent to study the tradeoff between the number of pursuers and the number of nodes occupied by the pursuers during the processing game (Th. 44-46). We conclude this chapter by proving several complexity results in the reconfiguration problem when more realistic (physical) constraints are considered.

Introduction on routing reconfiguration

[c-NS13]

[c-CCM+10, j-CCM+11]

[c-CHM+09, c-CMN09, c-BCM+12]

This work has been done in collaboration with O. Amini, S. Belhareth, P. Berthomé, N. Cohen, D. Coudert, F. Huc, D. Mazauric, F. Mazoit, N. Nepomuceno, J.-S. Sereni, R. P. Soares, S. Thomassé, and I. Tahiri.

Chapter 3: Distributed Graph Searching. In this chapter, we address the problem of "clearing" a network through several variants of graph searching games. There, we turn ourselves toward the coordination of mobile agents in a distributed setting. We propose various algorithms for mobile agents with very limited capabilities (Sections 3.2 and 3.3).

About graph searching

In graph searching, capturing an invisible fugitive is equivalent to clearing a network. **Section 3.1** briefly recalls the formal definition of graph searching using the "clearing" terminology. We also give a brief overview of some of the main results of the literature.

Survey on connected searching

Section 3.2 starts with a survey of recent results on connected graph searching where the clear part (i.e., the part of the graph that cannot be occupied by the fugitive) is always connected. We then consider connected graph searching from a distributed point of view. That is, the searchers are autonomous mobile entities that must clear the network without having a global knowledge about it. Among other things, we show that, in this setting, the number of searchers required to clear a network drastically increases when compared with the number of searchers needed in a centralized setting (Th. 50 and 51).

[c-BFNV06, c-NS07, c-INS07, j-BFNV08, j-NS09, j-INS09]

In **Section 3.3**, we introduce a new variant of graph searching, namely the *exclusive graph searching*. In this variant, the nodes of the graph can be occupied by at most one searcher at a time. We first give several results on this new variant such that the NP-hardness in the general case, the polynomiality in trees, etc. (Th. 52-55). Finally, we study exclusive graph searching and other robots coordination problems in the *Look-Compute-Move model* (a.k.a. CORDA) [FPS12]. This

[c-BBN12, c-BBN13, s-MNP14]

distributed model allows us to explore the feasibility of several tasks using robots with very poor abilities. We design algorithms in several topologies such as trees and rings (Th. 56-61).

About CORDA *[c-BBN12, c-BBN13, c-DDN+13b, j-DDN+14b, c-DNN14]*

This research has been done in collaboration with G. d'Angelo, L. Blin, J. Burman, G. Di Stephano, D. Ilcinkas, E. Markou, A. Navarra, S. Pérennes, D. Soguet and K. Suchan.

Chapter 4: Complexity of Locality: Routing and Graph Properties. In this Chapter, we focus on routing problems. We consider various constraints that depend on the environment considered (wireless networks, AS-network, etc.) and that are mainly related to the degree of locality of the knowledge of the nodes that must perform the routing. We also investigate the computation of graph properties with new models of local computations.

Section 4.1 is devoted on information spreading in different contexts. For wireless networks, i.e., in the presence of interference, we design a polynomial-time approximation algorithm that computes schedules for gathering information at some node in a grid (Th. 62). The computed schedules are optimal up to a small additive constant and the complexity of the problem remains open. Then, we focus on routing in queuing networks where nodes have no routing tables (they have no knowledge about the network). We propose a greedy algorithm that we prove to be stable, i.e., the number of packets in the network remains bounded (Th. 63). Finally, we focus on streaming networks, for which we design a distributed algorithm for maintaining a diffusion tree (Th. 64 and 65).

[c-BNRR09, s-BLN+13]

[c-CHN+10]

[c-GMNP13]

Section 4.2 starts with the basics on compact routing. In distributed routing, packets are routed toward their destination using the local information they meet on their way. This information is stored locally by each node in a *routing table*. A key challenge in large scale networks is to reduce the size of these routing tables, i.e., to make the tables as compact as possible, without degrading the quality (the length) of the routes. We give an overview on the results obtained in this area, focusing on the performance that may be achieved when considering structural properties of graphs. We then present our contributions on compact routing in graphs having no long induced cycles (Th. 66). It is worth to note that these results deeply rely on the characterizations we obtained in Sections 1.1 and 2.1. To conclude the section, we consider routing when routing tables may be faulty. We design randomized algorithms that achieve good performance in grids and random regular graphs (Tables 4.2 and 4.3).

Brief overview on compact routing *[c-NRS09, j-NRS12, c-KLNS12, j-KLNS14]* *[c-HIKN10]*

Finally, in **Section 4.3**, we define a new model of distributed computation in which nodes have only local knowledge and can share only a compressed view of it. In this setting, we investigate which graph properties can be computed through the composition of this partial local information. By introducing a reduction-like technique, we show that several problems such as detecting triangles or deciding if the graph has a small diameter cannot be solved using only local information (Th. 67). On the other hand, the complete knowledge can be recovered in this very simple model in the case of sparse graphs (Th. 68). We then derive other related

[c-BMN+11, j-BKM+14]

models of computation where the information is shared sequentially and prove that our models form a strict hierarchy in terms of computational power (Th. 69 and Table 4.4).

[c-BKN+12, j-BKM+14]

This work has been done with F. Becker, J.-C. Bermond, C. Caillouet, F. Giroire, N. Hanusse, D. Ilcinkas, A. Kosowski, B. Li, M. Matamala, R. Modrzejewski, S. Pérennes, I. Rapaport, H. Rivano, K. Suchan, I. Todinca and M.-L. Yu.

Appendix: Other contributions related to complexity and graph structures. In this Appendix, we present two additional contributions (that are a bit marginal but still related with complexity and graph properties computation). We first present several complexity results on the computation of the *hull number* in various graph classes (Section A.1). We then study the computational complexity of *weighted coloring* in trees (Section A.2): we show that, unless the Exponential Time Hypothesis fails, the problem cannot be solved in time $n^{o(\log n)}$ in n-node trees (Th. 70).

[j-ACG+13]

[c-ANP14]

This work has been done in collaboration with J. Araújo, V. Campos, F. Giroire, S. Pérennes, L. Sampaio and R. P. Soares.

Chapter 1

Turn-by-turn Two-Player Games in Graphs

Content

This chapter is devoted to present our contributions in the area of *Cops and Robber* ($\mathcal{C}\&\mathcal{R}$) games and, more generally, of some positional games played on graphs or digraphs. These are two-player games where players alternatively move their token along edges (arcs) of a (di)graph, or act on nodes/links of the (di)graph. The goal is to compute winning strategies for one of the players, optimizing specific criteria.

Generally, strategies may take advantage of the structural properties of the graph where the game is played. Therefore, studying these games offers an alternative point of view for understanding graph properties. Another motivation for this study is that some of these games provide nice models for problems arising in telecommunication networks. In particular, we introduce the *Surveillance* game for modeling prefetching problems. Last but not least, these games are of theoretical interest by themselves and are fun.

Section 1.1 presents the definition of the $\mathcal{C}\&\mathcal{R}$ games as initially defined by Winkler and Nowakowski [NW83] and Quilliot [Qui83, Qui86, Qui93]. Then, the main results and techniques of $\mathcal{C}\&\mathcal{R}$ games are surveyed, focusing on their complexity issues and how they relate to graph structural properties [c-KLNS12, j-KLNS14]. Section 1.2 is dedicated to the study of variations of $\mathcal{C}\&\mathcal{R}$ games [c-NS08, j-FGK$^+$10, j-CCNV11]. In section 1.3, We introduce the *Surveillance* game and present the results we obtained about it [c-FGJM$^+$12, j-FGJM$^+$14, c-GMN$^+$13]. Finally, in section 1.4, we present a new abstract game generalizing numerous positional games and that looks promising for a better understanding of such games, and possibly for designing approximation algorithms, using linear programming techniques.

1.1 Cop-number in a nutshell

Cops and Robber ($\mathcal{C}\&\mathcal{R}$) games have been initially introduced by Winkler and Nowakowski [NW83] and independently by Quilliot [Qui83]. We try, in this section, to give an exhaustive overview of the results obtained along thirty years in this area. For further details, see the surveys [Als04, Hah07, BB12] and the recent book [BN11].

The Cops and robber game is played on a n-node m-edge connected graph $G = (V, E)$ by two players. One player $\mathcal{C}$ controls a team of $k \geq 1$ *cops* while the second player $\mathcal{R}$ has one *robber* that must avoid to be caught by the cops. Initially, $\mathcal{C}$ places its cops on vertices of G. Several cops may occupy the same node. Then, $\mathcal{R}$ chooses a vertex to be occupied by its robber. Then, alternatively, $\mathcal{C}$ may move each of its cops along an edge, and then the robber may move to an adjacent node of its position. In particular, the players may pass. This is a full information game. That is, at each step, the robber has the full information concerning the position(s) of the cops and the other way around. Player $\mathcal{C}$ wins if at some step of the game, one of its cops occupies the same vertex as the robber. If the robber perpetually avoids this situation, then $\mathcal{R}$ wins.

Strategies: On the algorithmic point of view, the goal is therefore to design *strategies* that allow one of the players to win. A strategy for one player define how it must behave at each step. Because this is a full information game, deterministic strategies do not depend on the past. Therefore, the way both players decide their next move only depends on the current *configuration*, i.e., the multi-set of vertices occupied by the cops and the position of the robber.

More formally, a (positional) strategy for Player $\mathcal{C}$ using $k \geq 1$ cops is defined as follows. A *k-strategy* for $\mathcal{C}$ is defined by a pair $(I, \sigma_{\mathcal{C}})$ where $I \subseteq V$ is the multi-set of initial positions of the k cops ($|I| = k$) and $\sigma_{\mathcal{C}} : V^k \times V \to V^k$ is a function that associates $\sigma_{\mathcal{C}}(S) = (c'_1, \cdots, c'_k)$ to any configuration $S = ((c_1, \cdots, c_k), r) \in V^k \times V$, where $c'_i \in N[c_i]$ represents the new position of the i^{th} cop, for any $i \leq k$.

Similarly, a strategy for $\mathcal{R}$ against k cops is defined as a pair $(r_0 \in V, \sigma_{\mathcal{R}} : V^k \times V \to V)$.

A strategy σ is *winning* for one player if this player following σ wins whatever be the strategy followed by the other player.

Examples: As a warm-up, it is easy to see that Player $\mathcal{C}$ wins using one cop ($k = 1$) in any tree T. Indeed, a winning strategy for $\mathcal{C}$ can be defined as follows. The cop starts at any vertex of T. Then, at each step, the cop moves along the (unique) path between its position and the one of the robber. Clearly, after a finite number of steps, the robber is eventually caught whatever it does. On the other hand, Player $\mathcal{R}$ can win in any cycle of length at least 4 against one cop. Indeed, a winning strategy for $\mathcal{R}$ consists in, at each step, for the robber to reach a vertex at distance at least 2 from the cop without being caught.

1.1.1 Cop-number and Meyniel Conjecture

In this section, we present the main algorithmic and computational complexity aspects of $\mathcal{C}\&\mathcal{R}$ games.

Cop-win graphs

The seminal work on $\mathcal{C}\&\mathcal{R}$ games has been the characterization of graphs in which one single cop can always win [NW83, Qui83]. The *cop-win* graphs are the graphs in which there exists a winning 1-strategy for $\mathcal{C}$. Nowakowski and Winkler [NW83] and Quilliot [Qui83] have provided a nice combinatorial characterization of cop-win graphs. In section 1.2, we show how we have generalized this result for further variants of $\mathcal{C}\&\mathcal{R}$ games [j-CCNV11]. Below, we briefly present the cop-win characterization because on the one hand, it is very instructive for the study of $\mathcal{C}\&\mathcal{R}$ games, and on the other hand some of our contributions presented throughout this chapter take advantage of similar techniques.

Consider the last step of a game played by a single cop, before the capture of the robber. At this step, a cop moves to a vertex $y \in V$, and then, the robber at $x \in V$ has no way to escape (otherwise it would do it and the game would continue). Hence, $N[x] \subseteq N(y)$, i.e., vertex y dominates the neighborhood of x[1]. Pushing this argument a step forward, the cop arrives at some vertex z when the robber is in w, such that for any node x in $N[w]$, the neighborhood of x is dominated by some $y_x \in N[z]$. Then, whatever be the next move of the robber toward a vertex $x \in N[w]$, the cop will then move to y_x and then capture the robber during the next step. This naturally leads to the following definition. A graph is *dismantable* if its vertices can be ordered $(v_1, \cdots, v_n)$ such that, for any $i < n$, there is $j > i$ with $N_{X_i}[v_i] \subseteq N(v_j)$, $X_i = \{v_i, \cdots, v_n\}$.

Theorem 1 *[NW83, Qui83] A graph is cop-win if and only if it is dismantable.*

Before sketching some proofs of this result, let us mention some graph classes that are dismantable such as *chordal* graphs (graphs without *induced* cycles of length at least 4) and *bridged* graphs (graphs without *isometric* cycles of length at least 4) [Far87, AF88, Che97, LS04].

Several proofs have been proposed for the above result. One of them is by induction on the number n of vertices. The induction consists in identifying a particular vertex v and transforming a winning strategy in $G[V \setminus \{v\}]$ into a strategy in G, and vice versa. We detail a bit and unformally how to show that a cop-win graph is dismantable. Since G is cop-win, we can pick a vertex v occupied by the robber one step before its capture. As mentionned above, there is a vertex w dominating the neigborhood of v in G. Hence, it remains to prove that $G' = G[V \setminus \{v\}]$ is cop-win, and then the induction hypothesis says that the vertices of

[1] Throughout the document, given a graph G, a subgraph H of G and a node $v \in V(H)$, $N_H(v)$ denotes the set of nodes of H that are adjacent to v. $N_H[v] = N_H(v) \cup \{v\}$ and the subscript is omitted whenever $H = G$.

G' can be ordered in the desired way. Adding v as first vertex in this order gives a dismantable ordering for G. To prove that G' is cop-win, we consider a winning strategy σ for the cop in G and, roughly, we define a strategy σ' in G' that is the same as strategy σ, but each time the cop should go to v, following σ in G, it goes to w instead remembering it is 'virtually" on v. It is worth mentioning that the defined strategy is not positionnal but should take advantage of one bit of memory. We let the reader prove that such a strategy is winning in G'.

Assume that G is dismantable, then the dismantling ordering offers a winning strategy for $\mathcal{C}$ (see, e.g., [IKK06]). Let T be the spanning tree of G rooted in v_n and such that, for any $i < n$, the parent of v_i in the tree is the vertex v_j, with greatest index $j > i$, dominating the neighborhood of v_i in $G[X_i]$. The cop starts in v_n. Then, its strategy consists of going to its neighbor that is an ancestor of the current position of the robber in T, and that is closest to the robber. To prove that this strategy is winning, we need to prove that (1) there always exists such a vertex in the neighborhood of the cop, so that the cop always occupies an ancestor of the position of the robber, and (2) the distance between the cop and v_n never decreases and strictly increases after a finite number of steps, so that the strategy terminates.

About complexity

Next to the seminal work of Winkler and Nowakovski and Quilliot, Player $\mathcal{C}$ has been allowed to use more than one cop by Aigner and Fromme [AF84]. Clearly, using n cops[2] makes the task easy for $\mathcal{C}$. Therefore, the question is to minimize the number of cops that $\mathcal{C}$ has to use to ensure its victory in a given graph. Given a graph G, the *cop-number* of G, denoted by $cn(G)$, has been defined in [AF84] as the minimum number of cops required to capture a robber in G. The question of the complexity of computing the cop-number of a graph arises naturally. As seen above, whether a graph G is cop-win, i.e., whether $cn(G) = 1$, can be decided in quadratic time: it is sufficient to decide whether G is dismantable. This section is devoted to summarize the complexity results related to the cop-number of graphs.

In the early eighties, the complexity of several other variants on pursuit problems on directed graphs was investigated in [CS76, KAI79, Rei79, CKS81, Joh83, AIK84, Rei84, FG87]. For instance, in some variants, the robber does not need to be caught but it must be prevented to reach a particular vertex. Following this work, Goldstein and Reingold proved that deciding if $cn(G) \leq k$ (k part of the input) is EXPTIME-complete when initial positions are given [GR95]. Their reduction uses the Alternating Boolean Formula, where two players alternatively modify the variables of a conjunctive normal boolean formula and the goal of the first player is the formula to become true. Using a similar argument, they proved that deciding if $cn(G) \leq k$ is EXPTIME-complete in strongly connected digraphs.

On the other hand, k being fixed, deciding whether the cop-number of an n-node graph is at most k can be decided in polynomial time $n^{O(k)}$ [BI93, HM06]. One

[2]Unless state otherwise, n always denotes the number of nodes of the considered graph.

important ingredient for this result is the notion of configurations graph. Recall that a configuration represents the set of vertices occupied by the cops and the robber. When k cops are playing, there are $O(n^{k+1})$ configurations. The *graph of configurations* has as vertex-set the set of all configurations and two configurations $(c_1, \cdots, c_k, r)$ and $(c'_1, \cdots, c'_k, r')$ are adjacent if it is possible to go to one from the other during one step of the game, i.e., $r' \in N[r]$ and $c'_i \in N[c_i]$ for any $i \leq k$. The configurations $(c_1, \cdots, c_k, r)$ with $r \in N[c_i]$ for some $i \leq k$ are called *final* and are labelled with 0. Now, a configuration $(c_1, \cdots, c_k, r)$ is labelled with $i > 0$ if for any $r' \in N[r]$ there is $(c'_1, \cdots, c'_k)$, $c'_i \in N[c_i]$ for any $i \leq k$, such that $(c'_1, \cdots, c'_k, r')$ is labelled at most $i - 1$. With such a labeling, it is easy to see that k cops can win in G if and only if there are $\{c_1, \cdots, c_k\} \in V^k$ such that for any $r \in V$, the configuration $(c_1, \cdots, c_k, r)$ has received a finite label. This can clearly be checked in time polynomial in the size $n^{O(k)}$ of the configurations graph. The problem of generalizing the characterization of cop-win graphs to graphs with cop-number k has been extensively studied (e.g., [Bea04]). Recently, graphs with cop-number at most $k \geq 1$ have been characterized in [CM12]. I confess that I don't understand what brought this recent result compared to the ones in [BI93, HM06].

Surprisingly, a long time went by before further complexity results appeared. The problem of computing the cop-number of graphs has been shown to be NP-hard, and even W[2]-hard in [FGK08, j-FGK$^+$10]. In particular, the problem is not FPT (no algorithm in running time $f(k)n^{O(1)}$ to decide if an n-node graph has cop-number at most k) unless the complexity hierarchy collapses. Also, [FGK08, j-FGK$^+$10] proved that the cop-number of n-node graphs cannot be approximated up to a ratio $O(\log n)$ in polynomial time. The question of approximate it up to a ratio $O(n^{1-\epsilon})$ for $\epsilon > 0$ is still open. Contrary to many "classical" optimization problems, proving that the computation of the cop-number is NP-hard is not the last step. Indeed, what about the NP membership? In other words, the question to know whether this problem belongs to NP is quite difficult. It has been first proved that the $\mathcal{C}\&\mathcal{R}$ game is PSPACE-complete when each cop is allowed to move a bounded number of times [FGL10]. Then, Mamino proved recently that the $\mathcal{C}\&\mathcal{R}$ game is PSPACE-hard [Mam13]. The key point of Mamino's proof is the definition of a generalized game on edge-labelled graphs where edges are labelled *protected* or *unprotected* and the robber is captured if a cop reaches its position through an unprotected edge [Mam13]. The protected edges allow to give more freedom to the robber, which was the missing element to obtain this complexity result (until then, most of the complexity results were in directed graphs because it allowed to "control" the trajectory of the robber). Very recently, Bill Kinnersley proved that the problem is XPTIME-complete [Kin13].

[j-FGK$^+$10]

From minimum degree, girth and Lower bounds, to Meyniel conjecture

While previously mentioned results show that computing the cop-number of graphs is difficult in general, the cop-number highly depends on the structural properties of the considered graph. In the next subsection, we survey the results showing

that graph properties allow to give upper bounds on the cop-number. Firstly, we describe the lower bounds on the cop-number derived from graph properties. That is, we show how the structure of a graph may make the life of the robber easier.

Recall that the *girth* of a graph is the minimum length of its induced cycles. Clearly, the greater the minimum degree of a graph is, the more ways to escape the robber has. If moreover, there are no short cycles in the graph, the cops cannot 'surround" the robber easily. Hence, minimum degree and girth are natural parameters that determine whether lots of cops are needed to capture the robber. More formally,

Theorem 2 *[AF84] Let G be a graph with minimum degree δ and girth ≥ 5, $cn(G) \geq \delta$.*

We sketch the proof of the above theorem. Consider any graph G with minimum degree δ and girth at least 5. First, the size of any dominating set in graphs with no short cycles is at least its minimum degree. Hence, the robber has a safe initial position v. Now, let v be the current position of the robber at the robber's turn. Then, because G contains no triangles nor squares, $N(v)$ is a stable set and v is the unique common neighbor of any two nodes in $N(v)$. Hence, if strictly less than δ cops are available, a simple calculation implies that there is a neighbor w of v such that no cops occupy a node in $N[w]$: the robber can safely go to w.

The result of Aigner and Fromme has been later generalized by Frankl. His idea is that in a graph with large girth, a ball with large radius around any node v looks like a tree T_v. Roughly, if the robber starts from a node v with a branch of T_v containing no cops, then the robber can reach in a finite number of steps and without being captured another node satisfying the same property. Going on in this way provides an escape strategy for the robber.

Theorem 3 *[Fra87a] Let G be a graph with minimum degree at least δ and with girth at least $8t-3$. Then, $cn(G) > \delta^t$.*

Theorems 2 and 3 are important because they show that the cop-number in not bounded in general. Indeed, for any $\delta \in \mathbb{N}$, there exists a graph with minimum degree δ and girth at least 5. In particular, for arbitrary large n, there exist $\Omega(\sqrt{n})$-regular graphs with girth at least 5. For instance, in [AFLN08], a family of (p^m+2)-regular graphs of girth 5 and of order $2p^{2m}$, $p \geq 5$ prime, is described. Hence,

Corollary 1 *For arbitrary large n, there are n-node graphs G with $cn(G) = \Omega(\sqrt{n})$.*

Other graph classes achieve the same order of magnitude for the cop-number. For instance, the projective plane viewed as the bipartite graph with $2(q^2+q+1)$ vertices (points and lines) and where the edges correspond to the incidence relation, is $q+1$-regular and has girth at least 5, hence it has cop-number at least $q+1$ [BKL13, LP12].

As shown in the next subsection, the cop-number of many classes of graphs appears to be bounded. The case of graphs with small order (at most 10 nodes) has been studied in [BBB+13]. However, bounding the cop-number of any graph still remains a challenging problem. Hence, while the cop-number of any graph G is trivially bounded by the minimum size of a dominating set of G, a general upper bound of the cop-number of G in terms of its order is a thirty years outstanding problem. In 1987, Frankl mentioned the following conjecture of Henri Meyniel:

Conjecture 1 (Meyniel) *[Fra87a] For any n-node graph G, $cn(G) = O(\sqrt{n})$.*

During the last few years, many other variants of Cops and Robber games have been studied, mainly to bring new evidence and techniques to try to prove the Meyniel's conjecture (see Section 1.2 and, e.g., [BB12]).

1.1.2 Beyond Meyniel conjecture: when graph structure helps

In this section, we survey the work showing that the cop-number is bounded in many graph classes. Mainly, the cop-number has been bounded in terms of other graph parameters (e.g., genus, treewidth, chordality, etc.). Therefore, in the class of graphs where the considered graph invariant is bounded, the cop-number is bounded too and, in particular, its computation is polynomial. For instance, in graphs with dominating set of size at most k, the cop-number is trivially at most k.

Maximum degree and diameter don't help

We first recall that the cop-number is not bounded even in graph with maximum degree at most three or with diameter at most two.

Andreae answers a question in [AF84] by showing that the maximum degree is not a general upper bound on the cop number. To do so, Andreae is using the fact that there are regular graphs with arbitrary large girth and then the result holds by Theorem 3.

Theorem 4 *[And84] For any $\Delta, k \geq 3$, there is a Δ-regular graph G with $cn(G) \geq k$.*

Even for graphs with diameter 2, the cop-number cannot be bounded by some constant. This result seems to be well known as folklore. With J. Chalopin, V. Chepoi and Y. Vaxès, we proved it by considering the following family of graphs: Let us consider a set $S = \{1, \cdots, 3k\}$ and let G be the graph with vertex set V, the set of all subsets of S with k elements, and such that $x, y \in V$ are adjacent if and only if $x \cap y = \emptyset$.

Lemma 1 (folklore(?)) *For any $k \geq 1$, there is a $\binom{3k}{k}$-node $\binom{2k}{k}$-regular graph G with diameter 2 and $cn(G) \geq k$.*

While diameter and maximum degree are useless to upper bound the cop-number, there are many rich graph classes in which few cops are required to capture the robber. As already mentioned, the class of bounded size dominating set graphs is an example.

Graphs with bounded genus and graphs excluding some fixed minor

Aigner and Fromme were the first to use several cops to capture the robber. For this purpose, they made cops collaborating as follows: one cop "controling" a subgraph separating the graph, while the other cops can go toward the component occupied by the robber. More formally, some cops *control* a subgraph H of the graph G, if there is a strategy, such that, after a finite number of steps, these cops ensure that the robber cannot go to a vertex of H without being captured immediately at the next cops' move. For instance, one cop can control any arbitrarily large star. The main idea of Aigner and Fromme is that one cop is sufficient to control any shortest path in a graph [AF84].

Controling a shortest path. This result is the cornerstone of most of the capture strategies described in what follows. Therefore, we give more details about it. The idea of the proof is that, given a shortest path P in a graph, if, at some step when the robber occupies a vertex r, the cop is occupying a vertex $c \in V(P)$ that satisfies $d(r, z) \geq d(c, z)$ for all $z \in V(P)$, then, whatever be the next move of the robber to $r' \in N[r]$, there exists a neighbor of $c' \in N[c]$ preserving the property. Such a vertex c of P is called the *shadow* of the position r of the robber on P. In other words, once the cop has reached the shadow of the robber, then it can always follow the shadow of the robber. Clearly, if the robber reaches a vertex of P when the cop is occupying its shadow, then the robber is captured during the next step. Moreover, it is easy to show that the cop can reach the shadow of the robber in a finite number of steps.

Grids. As an example, consider a grid where a cop occupies the same row as the robber, then this cop may remain on the same column (the shortest path) and ensure that if the robber reaches this column it will be captured at the next step. Using this property, it is easy to prove that two cops are enough to capture one robber in any grid: a possible winning strategy is for one cop to control one column while the second cop reaches the next column (in the direction of the robber) until he controls it. Then, the role of the two cops are reversed. Since one cop is obviously not sufficient to capture the robber in any grid, then $cn(G) = 2$ for any grid G.

Bounded genus graphs. The main result of Aigner and Fromme is that three cops are sufficient to capture one robber in any planar graph. This surprising result follows the facts that one cop can control any shortest path and that there always is a separator consisting of three shortest paths in planar graphs [LT80]. Roughly, in any planar graph, two shortest paths forming a cycle (separator) can be controlled by two cops that therefore isolate the robber in a smaller part of the graph. Then a third cop is sent to control a new shortest path to separate the zone accessible by the robber, and so on, recursively. Hence,

Theorem 5 *[AF84] For any planar graph G, $cn(G) \leq 3$.*

Moreover this bound is tight because the dodecahedron is a planar graph with minimum degree 3 and girth at least 5, and therefore 3 cops are necessary [AF84].

Using the fact that removing an uncontractible cycle lets the genus g of a graph decreasing, Quilliot uses at most $2g$ cops to recursively reduce the genus of the component where the robber stands. Then, 3 remaining cops are used to capture the robber in the planar component where it stands. Hence, $2g+3$ cops are sufficient to capture a robber in any graph with genus at most g [Qui85]. By studying more carefully how to deal with shortest path separators to gradually reduce the genus of the zone accessible by the robber, Schröder improved this bound:

Theorem 6 *[Sch01] For any graph G with genus g, $cn(G) \leq \lfloor \frac{3}{2}g \rfloor + 3$.*

The case of non-orientable surfaces has been considered in [CFJT12]. However, the best lower bound for graphs with bounded genus is far from the upper bound. For instance, the projective plane has genus $g = \Omega(n)$ (because of its number of edges) and therefore, its cop-number is $\Theta(\sqrt{g})$. Hence, the following conjecture is still open[3] even for graphs with genus $g = 2$:

Conjecture 2 *For any graph G with genus g, $cn(G) \leq g + 3$.*

Another large class of graphs has bounded cop-number: any graph excluding a minor with bounded number of edges. Recall that a *minor* of a graph G is any subgraph of a graph obtained from G after sequentially contracting some edges. Given a H-minor-free graph G, Andreae uses shortest paths to somehow "represent" the edges of the graph H in G. Since H is not a minor of G, at some point, this set of paths separates the graph, and this can be done recursively.

Theorem 7 *[And86] Let H be any graph and v be any non isolated vertex in H. Then, for any graph G excluding H as a minor, $cn(G) \leq |E(H \setminus v)|$.*

Note that, recently, this result has been used to compute some particular decomposition for Minor-free graphs [AGG+13].

Using similar arguments, Andreae gived explicit graphs H for which any graph G excluding H as minor is such that $cn(G) \leq 3$ [And86]. Moreover, $cn(G) \leq \lceil n/3 \rceil + 1$ for any graph G with no wheel W_n as minor [And86].

Treewidth, chordality, bipartite graphs and others

The cop-numbers of many graph classes have been studied such as the cop-number of Cayley graphs [Fra87b, Ham87, Ham87], of various products of graphs [Tos88, MM87, NN98], of graphs with strong isometric dimension two [FN01]. Also, series-parallel graphs have cop-number at most two [The08]. More generally, it is not

[3] Note that for $g = 1$, above theorem prove the conjecture. However, it is not known whether it is tight: are there toroidal graphs with cop-number 4?

difficult to show that graphs G with bounded treewidth $tw(G)$ have cop-number at most $tw(G)/2 + 1$ [JKT10, BCF$^+$13]. Also, it is at most k in any graph of clique-width at most k [FGK08, j-FGK$^+$10].

More recently, Lu and Peng proved the Meyniel conjecture for graphs with diameter at most 2 and for bipartite graphs with diameter at most 3 [LP12]. More precisely, Lu and Peng proved by induction on $|V(H)|$ that, if the robber is constrained to move on the vertices of a k-degenerate subgraph H of G, then k cops can capture the robber [LP12]. Hence, if there is $S \subseteq V$ such that $G \setminus N(S)$ is k-degenerate, then $cn(G) \leq |S| + k$. The cops are placed uniformly at random on the vertices of the k-core K of G (maximum subgraph of G with all vertices having degree at least $k+1$) until the cops dominate all vertices in K. Then, k extra cops can capture the robber in $G \setminus K$. The expected number of cops to dominate the $\sqrt{n}$-core of G is at most $\sqrt{n}$ [LP12]. Hence:

Theorem 8 *[LP12] For any n-node graph G with diameter at most 2, or which is bipartite with diameter 3, $cn(G) \leq 2\sqrt{n} - 1$.*

While the smallest induced cycle leads to a lower bound on the cop-number, strangely, little is known related to the largest induced cycle (chordality). A graph is *k-chordal* if its greatest induced cycle has length at most k. In [AF84], it is shown that $cn(G) \leq 3$ for any 2-connected 5-chordal graph G. We generalize this result.

[c-KLNS12, j-KLNS14]

Theorem 9 *Let $k \geq 3$. For any k-chordal connected graph G, $cn(G) \leq k - 1$, and there exists a strategy where all $k - 1$ cops always occupy a chordless path.*

Proof. Let $v \in V$ be any vertex and place all cops at it. Then, the robber chooses a vertex. Now, at some step, assume that the cops are occupying $\{v_1, \cdots, v_i\}$ which induce a chordless path, $i \leq k - 1$, and it is the turn of the cops (initially $i = 1$). Let $N = \bigcup_{j \leq i} N[v_j]$, and if the robber occupies a vertex in N, it is captured during the next move. Else, let $R \neq \emptyset$ be the connected component of $G \setminus N$ occupied by the robber. Finally, let S be the set of vertices in N that have some neighbor in R. Clearly, while R is not empty, then so does S.

Now, there are two cases to be considered.

- If $N(v_1) \cap S \subseteq \bigcup_{1 < j \leq i} N[v_j]$. This case may happen only if $i > 1$. Then, "remove" the cop(s) occupying v_1. That is, the cops occupying v_1 go to v_2. Symmetrically, if $N(v_i) \cap S \subseteq \bigcup_{1 \leq j < i} N[v_j]$, then the cops occupying v_i go to v_{i-1}. Then, the cops occupy a shorter chordless path while the robber is still restricted to R.

- Otherwise, there is $u \in (N(v_1) \cap S) \setminus (\bigcup_{1 < j \leq i} N[v_j])$ and $v \in (N(v_i) \cap S) \setminus (\bigcup_{1 \leq j < i} N[v_j])$. First, we show that this case may happen only if $i < k - 1$. Indeed, otherwise, let P be a shortest path between such u and v with all internal vertices in R (possibly, P is reduced to an edge). Such a path exists by definition of S. Then $(v_1, \cdots, v_i, v, P, u)$ is a chordless cycle of length at least $i + 2$. Since G is k-chordal, this implies that $i + 2 \leq k$.

Then one cop goes to $v_{i+1} := v$ while all the vertices in $\{v_1, \cdots, v_i\}$ remain occupied. Since $v \in S$, it has some neighbor in R, and then, the robber is restricted to occupy R', the connected component of $G \setminus (N \cup N[v])$ which is strictly contained in R.

Therefore, proceeding as described above strictly reduces the area of the robber (i.e., R) after $< k$ steps, R decreases progressively and the robber is eventually captured. □

Note that previous Theorem somehow extends the model in [CN05], where the authors consider the game when two cops always remaining at distance at most 2 from each other must capture a robber. It is possible to improve the previous result in the case of 4-chordal graphs, i.e. for $k = 4$. More precisely:

Lemma 2 *For any 4-chordal connected graph G, $cn(G) \leq 2$ and there always exists a winning strategy for the cops such that they are always at distance at most one from each other.* *[c-KLNS12, j-KLNS14]*

Theorem 9 relies on chordless paths P in G such that $N[P]$ is a separator of G, i.e., there exist vertices a and b of G such that all paths between a and b intersect $N[P]$. In Section 2.1, we show how to adapt this to compute particular tree-decompositions.

Random graphs

Recently, the cop-number of random graphs has been intensively investigated. Besides the fact that many real networks share important structural properties of particular classes of random graphs (and therefore, studying the cop-number of such classes gives hints on the cop-number of real networks), some of these studies have provided new techniques that have been used for bounding the cop-number of general graphs (see below).

Erdös-Rényi graphs have been first investigated in [BHW07] (see also [BKP12]). In such a graph $G(n, p)$, each edge has independent and fixed (independent of the number n of nodes) probability $0 < p < 1$ to exist. In that case, it is proved that the cop-number is $\Theta(\log n)$ *asymptotically almost surely* (a.a.s.), i.e., with probability tending to one when n goes to infinity. This result (clearly, the upper bound) is mainly based on similar results on the dominating number of such graphs. More precisely,

Theorem 10 *[BHW07] For any fixed $0 < p < 1$ and for any $\epsilon > 0$, a.a.s:*

$$(1 - \epsilon) \log_{\frac{1}{1-p}} n \leq cn(G(n, p)) \leq (1 + \epsilon) \log_{\frac{1}{1-p}} n.$$

Then, sparse random graphs (where $p = d/n$, $0 < d < 1$) have been studied in [BPW07]. Roughly, such n-node graphs consist of the union of trees or unicyclic graphs with at most $\log \log n$ vertices a.a.s. Since the authors of [BPW07] define the

cop-number of a not connected graph as the sum of the cop-number of its connected components, sparse random graphs have large cop-number a.a.s.

The case of denser graphs, for which $pn = n^\alpha$, $0 < \alpha < 1$, has been first studied in [BKL13]. Then, Luczak and Pralat proved that the cop number of dense random graphs has an intriguing "zig-zag" behavior:

Theorem 11 *[LP10] Let $0 < \alpha < 1$ and $d = n \cdot p = n^{\alpha+o(1)}$, then*

- *if $\frac{1}{2j+1} < \alpha < \frac{1}{2j}$ for some $j \geq 1$, then a.a.s. $cn(G(n,p)) = \Theta(d^j)$.*
- *if $\frac{1}{2j} < \alpha < \frac{1}{2j-1}$ for some $j \geq 1$, then a.a.s. $\Omega(\frac{n}{d^j}) = cn(G(n,p)) = \Theta(\frac{n}{d^j}\log n)$.*

Finally, random graphs with given degree distribution have been investigated in [BPW07]. Given a sequence $w = (w_1, \cdots, w_n)$ where w_i is the expected degree of node v_i:

Lemma 3 *[BPW07] Let G be a random graph with expected degree distribution w. Let $p_0 \geq \frac{\max_i w_i^2}{\sum_i w_i}$. For any $\epsilon > 0$, with probability at least $1 - e^{-\Theta(n^\epsilon)}$:*

$$cn(G) \geq (1-\epsilon)\log_{\frac{1}{1-p_0}} n.$$

The case w being a power law distribution is particularly interesting since many real networks have this property. Because such graphs have many isolated vertices, [BPW07] concludes that their cop-number is $\Theta(n)$, where n is the number of nodes. On the other hand, regular-random graphs are considered in [PVW11]. Recently, it has been proved that Meyniel Conjecture holds for binomial random graphs [PW13]. In particular, it is proved that the conjecture holds for a general class of graphs with some specific expansion-type properties [PW13].

Many questions remain open in this area. Indeed, my point of view is that the cop-number of a graph should be defined as the maximum cop-number of its connected components (not the sum). Following this definition, most of the above questions should be revisited.

General graphs

We showed above that Meyniel conjecture holds in many graph classes. However, the question is still open, in general graphs.

The first upper bound for general graphs has been proposed by Frankl who showed that $cn(G) = O(n\frac{\log\log n}{\log n})$ in any n-node graph G [Fra87a]. This bound has then been improved to $O(\frac{n}{\log n})$ in [Chi08] by recursively finding long minimum distance caterpillar between the root and a leaf of some arbitrary BFS-tree of G. More precisely, a subgraph T of G is a *minimum distance caterpillar* if it consists of a shortest (in G) path P and some vertices adjacent to P. Generalizing the fact that one cop can control one shortest path, Chiniforooshan showed that 5 cops are sufficient to control any minimum distance caterpillar in a graph. Then, using

any BFS-tree, it is possible to compute a minimum distance caterpillar (from the root to a leaf of the BFS-tree) with size $O(\log n)$ in any n-node graph. Going on recursively, any graph can be partitioned into $O(n/\log n)$ minimum distance caterpillars. Hence, $cn(G) = O(\frac{n}{\log n})$ in any n-node graph G [Chi08].

Recently, several groups improved independently this result to obtain the best current bound:

Theorem 12 *[SS11, LP12] For any connected n-node graph G, $cn(G) = O(\frac{n}{2^{(1-o(1))\sqrt{\log n}}})$.*

To conclude this section, we try to summarize the structural properties of a minimal counter example for the Meyniel conjecture. From previous results, it follows that:

Assertion 1 *Any minimal counter example n-node graph G to the Meyniel conjecture would have:*

- *maximum degree $o(\sqrt{n})$;*
- *diameter at least 3 (or at least 4 if G is bipartite) and $o(\sqrt{n})$;*
- *minimum hitting set of the induced cycles of length $\omega(\sqrt{n})$ with size at least $\omega(\sqrt{n})$;*
- *genus, clique-width and treewidth $\omega(\sqrt{n})$, and*
- *any graph with $O(\sqrt{n})$ edges as a minor.*

For completeness, we also like to mention that the cop-number of infinite graphs has also been studied [CLP00, HLSW02, BHW07, BHT10].

1.2 Variants of Cops and Robber games

The previous section shows that new techniques or approaches must be invented to solve the Meyniel conjecture. Therefore, many variants and generalizations of the Cops and Robber game have been proposed to handle this problem.

1.2.1 Fast Cops and Robber.

A first natural generalization of $\mathcal{C}\&\mathcal{R}$ games is the one where the cops and the robber are faster [FGK08]. We say that the cops have speed $s \geq 1$ if, at its turn, a cop can move along a path of length at most s. The speed s' of the robber is defined similarly. Clearly, if one cop is faster than the robber, then it will eventually catch the robber by decreasing their relative distance at each step. Hence, we may assume that $s' \geq s$. Let $cn_{s,s'}(G)$ be the smallest number of cops with speed s required to capture a robber with speed s' in a graph G.

On the one hand, this variant seems to behave as the original one. For instance, the graphs that are cop-win in this variant, i.e., graphs where one cop with speed

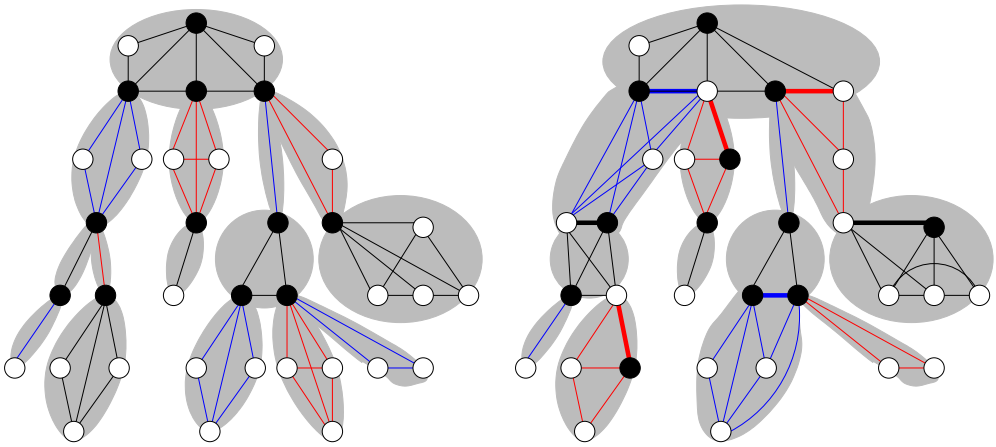


Figure 1.1: (a) A big brother graph. (b) A big two-brother graph.

s can capture a robber with speed s', can also be characterized by a kind of dismantling ordering. For any $i \geq 0$, H subgraph of a graph G and $x \in V(H)$, let $N_i(x, H)$ be the set of nodes at distance at most i from x in H. If $H = G$, we note $N_i(x, H) = N_i(x)$. Finally, $N_i[x] = N_i(x) \cup \{x\}$.

[j-CCNV11] **Theorem 13** *Let $s' \geq s$ and G be a graph. $cn_{s,s'}(G) = 1$ if and only if the vertices of G can be ordered $\{v_1, \cdots, v_n\}$ such that, for any $i < n$, there is $j > i$ with $N_{s'}(v_i, G \setminus v_j) \cap \{v_i, \cdots, v_n\} \subseteq N_s[v_j]$. Moreover, this can be decided in polynomial time.*

When $s = 1$, we achieve a stronger characterization. A graph is *dually chordal* if its line graph is chordal and its clique hypergraph is an hypertree. A graph G is a *big brother graph*, if its block[4]-decomposition can be represented in the form of a rooted tree T in such a way that each block of G has a dominating vertex and for each block B distinct from the root, the articulation point between B and its father-block dominates B (see Figure 1.1(a)).

[j-CCNV11] **Theorem 14** *Let G be a graph and $s' > 0$. $cn_{1,s'}(G) = 1$ if and only if:*

- *Case $s' = 1$: G is dismantable. (see also [NW83, Qui83, AF84])*
- *Case $s' = 2$: G is a* dually-chordal graph
- *Case $s' > 2$: G is a* big brother graph

The speed of the robber being fixed, the speed that one cop must achieve to be able to capture the robber seems related to the hyperbolicity of the graph: the smaller the hyperbolicity is (somehow, the closer to a metric of a tree, the metric

[4] A *block* is a maximal 2-connected component

of the graph is), the easier it is for the cop. A graph G is called a *Helly* graph if its family of balls (1-neighborhood) satisfies the Helly property: any collection of pairwise intersecting balls has a common vertex.

Lemma 4

- *Let $s' \geq 4\delta \geq 0$, s' even. For any δ-hyperbolic graph G, $cn_{\frac{s'}{2}+2\delta,s'}(G) = 1$.* *[j-CCNV11]*
- *If $s' \geq 2s$, then if $cn_{s,s'}(G) = 1$ then the graph G is $(s'-1)$-hyperbolic.*
- *If $s' > s$, then any bridged graph or Helly graph G with $cn_{s,s'}(G) = 1$ is s'^2-hyperbolic.*

The previous lemma suggests that there is a function f such that, for $s' > s$, any graph G such that $cn_{s,s'}(G) = 1$ is $f(s')$-hyperbolic. Such a result would give a new characterization of hyperbolicity of graphs. Chalopin, Chepoi and others recently closed this question [CCPP13].

On the complexity point of view, this variant also behaves similarly as the original one. More precisely, the proof that computing the cop-number of a graph is $W[2]$-hard works for the general case $s' \geq s = 1$ [FGK08, j-FGK$^+$10]. The case of interval graphs has been considered in [FGK08, j-FGK$^+$10] where a polynomial time algorithm is presented that computes $cn_{1,s'}(G)$ for any interval graph G and any $s' \geq 1$ fixed. Gavenciak designed a polynomial time algorithm in the case s' is unbounded [Gav11]. In that case, an upper bound of $O(\sqrt{n})$ has been proved in [Meh10]. The case of unbounded s' has recently been studied in [Mar14].

When $s' \geq s = 1$, it has been shown that the cop number of a connected n-node graph can be as large as $\Omega(n^{\frac{s'}{s'+1}})$ [AM11, Meh11], hence generalizing the Meyniel conjecture for larger speeds. However, even in this case, the conjecture remains open: the best general upper bound on the cop-number of n-node graphs being $\frac{n}{\alpha^{(1-o(1))\sqrt{\log_\alpha n}}}$ where $\alpha = 1 + 1/s'$ [FKL12]. This latter result generalizes the results of [LP12, SS11] in the case $s' = 1$.

On the other hand, we proved that capturing a fast robber may be very different than the original game. Indeed, all results concerning bounded genus graphs in the original variant fail as soon as the speed of the robber increases[5]. Specifically, we proved:

Theorem 15 *For any $n \times n$ square-grid G, $cn_{1,2}(G) \in \Omega(\sqrt{\log(n)})$.* *[c-NS08, j-FGK$^+$10]*

This result must be put in contrast with the fact that $cn(G) \leq 2$ for any square-grid G. Moreover, we did not manage to apply the previous techniques for lower bounds. To prove Theorem 15, we propose an evasion strategy for a robber with

[5]Thanks to F.V. Fomin who asked me, in a forest of Puerto Varas during LAGOS 2007, whether capturing a fast robber could be done with a constant number of cops in a planar graph.

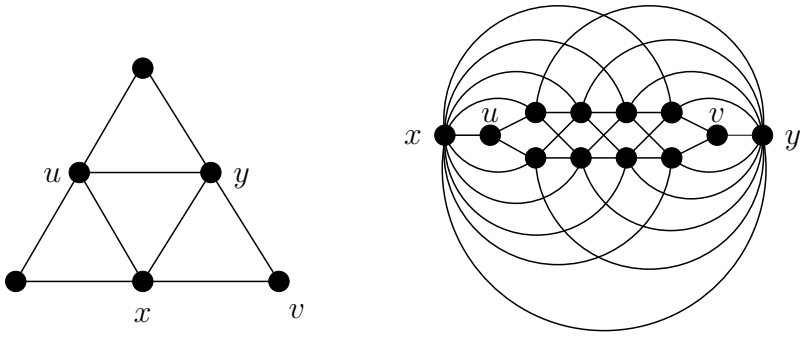


Figure 1.2: (a) a graph G with $cn_k^v(G) = 1 < cn_{1,k}(G)$ for all $k \geq 2$, and (b) a graph G with $cn_4^v(G) = 1 < cn_5^v(G)$.

speed 2 against $k \geq 1$ cops in any $n \times n$ square-grid, where $n \in \Omega(2^{k^2})$. Intuitively, the strategy we designed for the robber is defined recursively. There are $k+1$ levels in our strategy. For any $0 < i \leq k$, the level-i strategy uses the level-$(i-1)$ strategy as a subroutine. The key point is that the level-i strategy only deals with i cops: $cop_1, \ldots, cop_i$, and maintains as an invariant the fact that cop_i remains "far enough" from the robber.

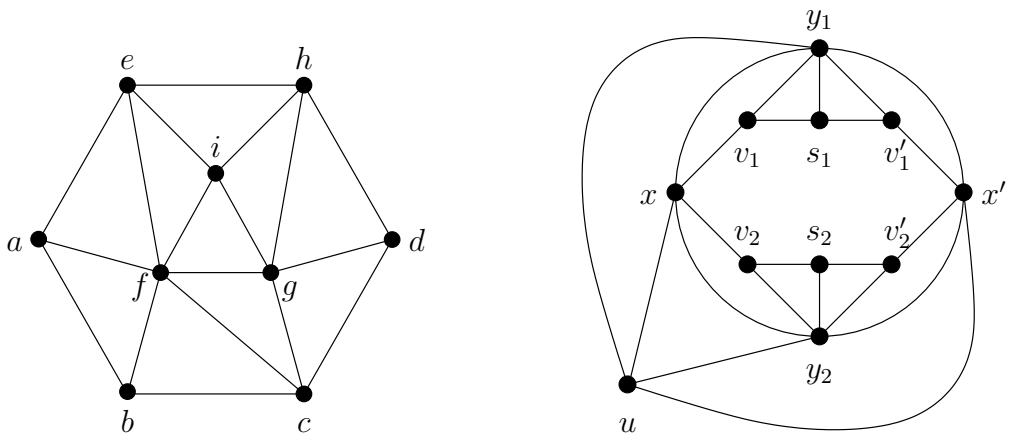
The above result can be generalized to any $s' > s$ where the ratio between the speeds appears in the basis of the logarithm. Moreover,

[c-NS08, j-FGK+ 10]

Theorem 16 *Let H be a planar graph containing an $n \times n$ square-grid G with $n \in \Omega(2^{k^2})$ as an induced subgraph, then $cn_{1,2}(H) \geq k$.*

Surprisingly, the best known upper bound for $cn_{1,2}(G)$ in $n \times n$ square-grid G is $O(n)$. Therefore, capturing a fast robber is not fully understood even in simple topologies. New techniques are required for achieving new progress in this direction. In particular, establishing a link with similar games in other metric spaces (Euclidean or hyperbolic spaces) could be a good approach (e.g., see [BI93, Ben01, BLW09, AR10]).

Recently, the technique of Aigner and Fromme has been applied to polygonal environments [BKIS12]: more precisely, it is shown that 3 cops are sufficient to capture a visible robber in any polygonal environment with arbitrary complexity. In [KS12, KS13], the case of an invisible evader is studied: $O(\sqrt{h} + \log n)$ cops are sufficient when the speed of all players is at least the minimum distance between nodes (n is the number of nodes of the polygon and h the number of holes) [KS12] while $\Omega(n^{2/3})$ may be required for arbitrary speed [KS13]; $\Theta(\sqrt{n})$ cops are necessary and sufficient when $h = 0$ and a general upper bound of $O(n^{5/6})$ is proven [KS13].



(a) (b)

Figure 1.3: (a) 2-bidismantlable graph G with $cn_2^v(G) > 1$ (b) Graph G with $cn_2^v(G) = 1$ that is not strongly 2-bidismantlable.

1.2.2 Visibility, radius of capture and other variants

When the robber can hide

Another natural generalization of $\mathcal{C}\&\mathcal{R}$ games is the one proposed by Clarke [Cla09]. In this variant, the robber is visible only every $w \geq 1$ turns. Let $cn_w^v(G)$ be the smallest number of cops that can capture in G a robber that shows up only every w steps and is hidden the remaining turns. Note that, if a cop reaches the same node as the robber, it is captured even if it is invisible. By definition, $cn_1^v(G) = cn(G)$ is the classical cop-number.

Note that it is not clear that, for a fixed graph G, $cn_w^v(G)$ is a non decreasing function of w. In [j-CCNV11], we proved that, for all $w \geq 2$, there are graphs G such that $1 = cn_w^v(G) < cn_{w+1}^v(G)$ (see Figure 1.2(b)). However, the question to know whether $cn_{w+1}^v(G) = 1$ implies $cn_w^v(G) = 1$ for any G and w is still open.

In the case when the robber can hide ($w > 1$) classical dismantling orders are not sufficient anymore to characterize cop-win graphs. No full characterization has been found until now. However, a characterization of some necessary or sufficient conditions have been proved in some other cases.

A graph G is called a *big two-brother graph* if G can be represented as an ordered union of subgraphs $G_1, \cdots, G_r$ in the form of a tree T rooted at G_1 such that G_1 has a dominating vertex and any G_i , $i > 1$, contains one vertex or two adjacent vertices disconnecting G_i from its father and one of these two vertices dominates G_i (see Figure 1.1(b)).

Theorem 17 *$cn_w^v(G) = 1$ for all $w \geq 1$ if and only if G is a big two-brother graphs.* *[j-CCNV11]*

For particular values of w, we prove several necessary or sufficient conditions, but none of them provides a full characterization. A graph $G = (V, E)$ is *w-bidismantlable* if the vertices of V can be ordered $(v_1, \cdots, v_n)$ in such a way that,

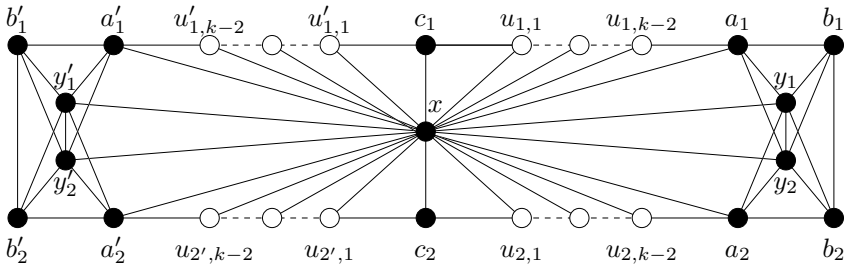


Figure 1.4: A graph G with $cn^v_w(G) = 1$ that is not w-bidismantlable

for any $1 \leq i < n$, there exist two adjacent or coinciding vertices x_i, y_i with $y_i = v_j$,$x_i = v_\ell$ and $j, \ell > i$ such that $N_w(v_i, G \setminus \{x_i, y_i\}) \cap X_i \subseteq N_1(y_i)$, where $X_i = \{v_i, v_{i+1}, \cdots, , v_n\}$. A graph G is *strongly 2-bidismantlable* if G admits a 2-bidismantling order such that, for any vertex v_i, $i < n$, $y_i = x_i$ or $N_2(v_i, G \setminus \{y_i\}) \cap X_i \subseteq N(x_i, G \setminus \{y_i\})$.

[j-CCNV11]

Lemma 5

- *Any graph G with $cn^v_2(G) = 1$ is 2-bidismantable. There are 2-bidismantable graphs G with $cn^v_2(G) > 1$ (see Fig 1.3(a)).*
- *Any strongly 2-bidismantable graph G satisfies $cn^v_2(G) = 1$. There are graphs G with $cn^v_2(G) = 1$ that are not strongly 2-bidismantable (see Fig 1.3(b)).*
- *Let $w > 1$ be an odd integer. Any w-bidismantable graph G satisfies $cn^v_w(G) = 1$. There are graphs G with $cn^v_w(G) = 1$ that are not w-bidismantable (see Fig 1.4).*

A way to obtain a full characterization of cop-win graphs in this variant would probably be to study the case when the robber can remain hidden at most $w - 1$ steps. However, it seems more difficult to analyze.

Numerous results have also been dedicated to the case when all players are always invisible (*hunter-and-rabbit game*). In that case, some work allows the pursuer to be faster than the robber [SD13], or probabilistic strategies for the cops have been proposed [ARS+03, IKK06, IK08, KMP13, DDTY13]. In particular, in the all-invisible case, a single cop can catch the robber in $\Theta(n \log n)$ expected time in any n-node graph [ARS+03]. Some cases when visibility is limited depending on the distance is studied in [IKK06, IK08].

When the robber can shoot

The last variant we would like to insist on is the one where the cops can capture the robber at some distance [BCP10]. In other words, in this variant, the robber is captured as soon as she is at distance at most $r \geq 0$ from a cop. r is called the *radius of capture* of the cops. Let $cn^d_r(G)$ be the smallest number of cops with radius capture r that can capture a robber in G. Note that $cn^d_0(G) = cn(G)$ is the classical cop-number.

Bonato et al. proved that computing $cn_r^d(G)$ is NP-hard in general graphs and gave an algorithm that decides whether $cn_r^d(G) \leq k$ in time $O(n^{2k+3})$ [BCP10]. The analogous of the Meyniel conjecture has also been studied: there are graphs G for which $cn_r^d(G) \geq (\frac{n}{r})^{1/2+o(1)}$ and $cn_r^d(G) = O(\frac{n}{\log \frac{2n}{r+1}} \frac{\log(r+2)}{r+1})$ in any n-node graph G [BCP10].

Surprisingly, the characterization of cop-win graphs in this variant seems much harder. As far as we know, almost nothing is known to characterize cop-win graphs for $r > 0$, but in the following very restricted case:

Theorem 18 *A bipartite graph G is such that $cn_1^d(G) = 1$ if and only if V can be ordered $(v_1, \cdots, v_n)$, with $\{v_{n-1}, v_n\} \in E$ and, for any $i < n-1$, there is $j > i$, $N_1(v_i, X_i) \setminus \{v_i\} \subseteq N_1(v_j)$.* *[j-CCNV11]*

Miscellaneous

To conclude this section, we should mention other variants that have been studied: when the goal of the cops is not to capture the robber but to avoid that she reaches some given subgraph [FGH$^+$08, FGL09, FGH$^+$11, SSV11], when each cop has a limited number of moves [FGL10, FGL12], when cops may be helped by traps or radars [CN00, CN01, CC06], when the robber can attack the cops [BFG$^+$13] etc. Also, other objectives are considered such as the capture time [BGHK09, BGKP13, KW13].

We also like to mention that almost nothing is known about the Cops and Robber games in directed graphs. For instance, a characterization of cop-win graphs in this variant, say when only the robber is constrained to follow the arcs, seems much more difficult to obtain (if any). We started investigated this topic during the internship of Mélanie Ducoffe.

1.3 Web-page prefetching and Surveillance game

The games mentioned above are natural variants of Cops and Robber games and they all are very interesting from the theoretical point of view, in particular because they offer new ideas for solving Meyniel conjecture. In this section, we study a similar game that, for once (?), has been introduced mainly because it models a practical problem. More precisely, we defined the surveillance game to study the prefetching problems [c-FGJM$^+$12, j-FGJM$^+$14].

Prefetching.

Prefetching is a basic technique in computer science. It exploits the parallelism between the execution of one task and the transfer of information necessary to the next task, in order to reduce waiting times. Consider the execution of some program by a CPU, some future instruction of the program makes reference to a memory block. When this instruction is being processed, the CPU must wait until this

memory block is loaded into the registers to be able to proceed. Hence, one way to speed up the execution of such program is to fetch this memory block before said instruction is being processed [God09]. The possibility to preemptively load data necessary for the execution of some task is called prefetching in computer science domains. A modern instance occurs in the Web, where browsers may download documents connected to the currently viewed document (Web page, video, etc.) while it is being read or viewed. If the browser can anticipate which web page the web-surfer would like to read, the web-surfer will be given the impression of a greater download speed. For these reasons link prefetching has been proposed as an Internet Draft standard by Mozzila [FS04, Inc99]. As of this work, several browsers attempt to use some kind of prefetching to lessen the waiting time of its users [wik]. Unfortunately, some difficulties arise when dealing with prefetching. In general, the amount of available space to store the documents (web pages, memory block, instructions, etc.) might be small, making it necessary to choose which documents are going to be prefetched. Moreover, the speed of the prefetching mechanism might not be arbitrarily big, restricting the amount of documents that can be prefetched before some task has to wait. Lastly, the prefetching mechanism might be a resource that we would like to minimize. In the case of web pages, the bottleneck resource is the bandwidth. Hence, every page that is prefetched and never seen by the user is a wasted resource.

The execution digraph of a prefetching problem is a graph in which vertices represents all the tasks involved in the problem and arcs represent dependencies between the prefetching of such tasks, meaning that if there is an arc between two vertices, then one must be prefetched before the other. In prefetching web pages, vertices represent web pages and arcs hyperlinks from one web page to another. Some studies [JG97, GCD02, MJM10], use a markovian model with a transitioning function over the arcs of the execution digraph to model the behaviour of the web surfer. However, the exact solution of such a model requires an exponential time in the number of vertices of the execution digraph. Such a model also constraints the web surfer to access web pages through a predetermined specific random manner.

It worths also mentioning more "technical" studies of prefetching. There are mainly three types of prefetching. In local prefetching [WLZC12], the client has no aid from the server when deciding which documents to prefetch. In the server based hints prefetching [AEFP98, AZN99, Mog96], the server can aid the client to decide which pages to prefetch. Lastly, in the proxy based prefetching [FCLJ99], a proxy that connects its clients with the server decides which pages to prefetch. Moreover, some studies consider that the prefetching mechanism has perfect knowledge of the web-surfer's behaviour [PM96, KLM97]. In these studies, the objective is to minimize the waiting time of the web-surfer with a given bandwidth, by designing good prediction strategies for which pages to prefetch.

We are concerned with *perfect* prefetching, i.e., ensuring that the Web surfer never accesses a document that has not been prefetched yet. In other words, the surfer is "impatient" in the sense that she does not tolerate waiting for information. Due to network's capacity (bandwidth) limitation, it is important to limit the

number of Web pages that can be prefetched at each step: We aim at determining its minimum value. In addition to being simpler than a fully specified optimization problem, this question does not need specific assumptions on the behavior of the Web surfer as in [GCD02, MJM10].

Surveillance game.

We introduced the *surveillance game* which is a model to study a local prefetching scheme to guarantee that a websurfer never has to wait a web-page to be downloaded, whilst minimizing the bandwidth necessary to achieve this [c-FGJM$^+$12, j-FGJM$^+$14]. This two-player game involves one Player moving a mobile agent, called *surfer*, along the edges of a graph, while a second Player, called *observer*, marks the vertices of the graph. The surfer wins if it manages to reach an unmarked vertex. The observer wins otherwise.

More formally, let $G = (V, E)$ be an undirected simple n-node graph, $v_0 \in V$, and $k \in \mathbb{N}^*$. Initially, the surfer stands at v_0 which is marked and all other nodes are not marked. Then, turn-by-turn, the observer first marks k unmarked vertices and then the surfer may move to a neighbor of her current position. Once a node has been marked, it remains marked. The surfer wins if, at some step, she reaches an unmarked vertex; and the observer wins otherwise. Note that the game lasts at most $\lceil \frac{n}{k} \rceil$ turns (after which all nodes are marked). When the game is played on a directed graph, the surfer has to follow arcs when it moves. A *k-strategy for the observer from v_0*, or simply a *k-strategy from v_0*, is a function $\sigma : V \times 2^V \to 2^V$ that assigns the set $\sigma(v, M) \subseteq V$ of vertices, $|\sigma(v, M)| \leq k$, that the observer should mark in the *configuration* (v, M), where $M \subseteq V$, $v_0 \in M$, is the set of already marked vertices and $v \in M$ is the current position of the surfer. We emphasize the fact that σ depends implicitly on the graph G, i.e., it is based on the full knowledge of G (this will always be the case but for Theorem 29 below). A k-strategy from v_0 is *winning* if it allows the observer to win whatever be the sequence of moves of the surfer starting in v_0. The *surveillance number* of a graph G with initial node v_0, denoted by $sn(G, v_0)$, is the smallest k such that there exists a winning k-strategy starting from v_0.

Discussion on the hypothesis.

In the surveillance game, the fugitive plays the role of the Web surfer moving in the execution (di)graph while the observer must prefetch the Web pages before the fugitive reaches them. Before going further, we aim at discussing some hypotheses of our model.

First, in our model, we assume a constant prefetching time for all the Web pages. It is however not a strong assumption since the surveillance game may also model the fact that some Web pages are heavier than others. Indeed, let us assume that each Web page u has a proper size $\mathcal{W}(u)$ and so a proper prefetching time. Consider the graph G^p obtained by replacing any node u of G by a clique K_u of size $\mathcal{W}(u)$

and any edge $\{u, v\}$ by a complete bipartite graph between K_u and K_v. Thus, the surveillance problem for the weighted graph G is equivalent to the problem in G^p.

We also assume that the step duration is the minimum visiting time among all pages. If there exists a perfect prefetching strategy with this constant duration time, then this strategy is also a perfect prefetching strategy with the initial visiting times for all the pages. This hypothesis corresponds to studying the worst case in which the visiting time of all pages is constant (and so corresponds to the minimum visiting time among all the pages).

The strongest assumption is probably the infinite memory, that is when a Web page is prefetched, then it remains prefetched. In other words, a vertex that is marked remains marked for all the following steps of the surveillance game. That is, we assume that the amount of local memory available is, usually, much bigger than what a web surfer can browse in a given time, making the network resources (bandwidth) the principal concern in designing good prefetching strategies. This is motivated by the fact that memory in modern computers is not scarce anymore, which makes network resources the critical ones. However, local storage memory is also a potential issue, and prefetching is classically associated with the question of cache management. We plan to investigate two more realistic models corresponding to two cache management policies. The first variant assumes that a marked vertex becomes unmarked after a constant number of steps. The second model allows the observer to unmark some vertices, respecting the constraint that the total number of nodes that are marked never exceeds a given threshold corresponding to the maximum number of Web pages that can be prefetched simultaneously.

1.3.1 Complexity and algorithms in several graph classes

This section is devoted to the study of the computational complexity of the surveillance game. All these results can be found in [c-FGJM$^+$12, j-FGJM$^+$14].

As a warm-up, let us describe a simple basic strategy σ_B: at each step, the observer simply marks all unmarked neighbors of the current position of the fugitive. Note that, the surfer always arrives to any vertex (but v_0) by an already marked neighbor. This allows us to derive simple general bounds.

[c-FGJM$^+$12, j-FGJM$^+$14]

Assertion 2 *For any graph G with maximum degree Δ and for any v_0 with degree d_0,*

$$d_0 \leq sn(G, v_0) \leq \max\{\Delta - 1, d_0\}.$$

From Assertion 2, it is almost straightforward that the surveillance number of graphs with maximum degree 3 or with a universal vertex can be easily computed.

[c-FGJM$^+$12, j-FGJM$^+$14]

Lemma 6 *Let G be a connected undirected graph with maximum degree at most three and at least one edge. Then, $1 \leq sn(G, v_0) \leq 3$ and*

- *$sn(G, v_0) = 1$ iff G is a path, where v_0 has degree one;*
- *$sn(G, v_0) = 3$ iff v_0 has degree 3.*

[c-FGJM+12, j-FGJM+14]

Lemma 7 *Let G be an undirected graph with a universal vertex. For any $v_0 \in V(G)$ with degree d_0, $sn(G, v_0) = \max\{d_0, \lceil \frac{n-1}{2} \rceil\}$.*

One important and useful result is that we can restrict our study to simpler trajectories of the surfer. That is, when computing the surveillance number of a graph, we don't need to consider all possible moves of the surfer. More precisely, in the *monotone* variant of the surveillance game, the surfer is restricted to move at every step and to follow only induced paths in G. That is, for all $\ell > 0$, after having followed a path $(v_0, \cdots, v_\ell)$, the surfer is not allowed reaching a vertex in $N[\{v_0, \cdots, v_{\ell-1}\}]$. Note that if the surfer cannot move, then she loses. Let $msn(G, v_0)$ be the smallest k such that there is a winning monotone k-strategy in G when the surfer starts from v_0, i.e., the observer can win, marking at most k vertices at each step, against a surfer constrained to follow induced paths.

Theorem 19 *For any (di)graph G and $v_0 \in V(G)$, $sn(G, v_0) = msn(G, v_0)$.* [c-FGJM+12, j-FGJM+14]

We give the intuition of the proof. Assume that the observer may win the monotone game in G by marking at most k nodes at each step. Then, a strategy for the non-monotone game can be defined as follows. As long as the fugitive follows an induced path, the observer just plays according to its monotone strategy. It is "clear" that if the fugitive does not move in some step, that does not hurt the observer at all. Now consider the first time that the robber breaks the rule of moving along an induced path. Say her trajectory is $v_0, v_1, \cdots, v_m$ and then she moves to v_{m+1} which is a neighbour of v_j for $0 \leq j \leq m-1$. Then the observer can imagine that time has gone backwards, and consider the time j, when the fugitive has moved along the trajectory $v_0, v_1, \cdots, v_j$, and then she is moving to v_{m+1} in the next step. The observer just plays according to the strategy that he would have played if the robber had taken the trajectory $v_0, v_1, \cdots, v_j, v_{m+1}$. Intuitively, the only difference with the real game is that some additional vertices are marked in the real game, which does not hurt the observer. Hence the observer can still win by marking k vertices in each step in the original variant.

Previous result means that we can always consider that the fugitive follows induced paths, and so that the fugitive has to move at every step because an induced path is necessarily a simple path. This assumption is very important in the proofs of next results since it simplifies them a lot.

Complexity

We have shown that the problem of computing the surveillance number is actually very difficult in "simple" graph classes. By reducing the 3-Hitting Set Problem [GJ90], we proved the following theorems. The intuition is that when marking the nodes, the observer has to choose few nodes that cover all possible neighborhood of the next position of the surfer.

In particular, the problem is not FPT when parameterized by its solution:

Theorem 20 *Deciding if $sn(G, v_0) \leq 2$ is NP-hard in chordal graphs.* [c-FGJM+12, j-FGJM+14]

The game is also difficult, when the number of turns is bounded. In particular, in split graphs, the game is limited to two turns since there are no induced paths of length > 2.

[c-FGJM+12, j-FGJM+14]

Theorem 21 *Deciding whether $sn(G, v_0) \leq k$ is NP-hard in split graphs.*

Next, we show that the problem of deciding whether $sn(G, v_0) \leq 4$ is PSPACE-complete in DAGs. Following the ideas in [FGL10], it is done by reducing the 3-QSAT problem [GJ90]. Intuitively, the choice of which nodes must be marked by the observer at each step corresponds to the existential quantifiers, while each move of the surfer corresponds to a universal quantifier.

[c-FGJM+12, j-FGJM+14]

Theorem 22 *Deciding whether $sn(G, v_0) \leq 4$ is PSPACE-complete in DAGs.*

An interesting question is to know if the problem remains PSPACE-hard in undirected graphs. In comparison, Mamino recently proved that the Cops and Robber game is PSPACE-hard in undirected graphs [Mam13].

Algorithms

Besides the bad news above, we obtained some positive results by designing a general exponential algorithm for computing the surveillance number and polynomial time algorithms in the case of trees and interval graphs.

The general algorithm consists of a classical backward labeling of the configuration digraph, a.k.a. arena digraph (e.g., see [ACP87, HM06, j-FFN09]).

[c-FGJM+12, j-FGJM+14]

Theorem 23 *$sn(G, v_0)$ can be computed in time $O^*(4^n)$ on n-node graphs.*

The case of trees can be dealt with by using a slight generalization of the game. In this generalization, v_0 is initially marked and the observer can mark at most $k_0 \geq 0$ other vertices before the fugitive's first step. Let $k \geq 0$ and T be a rooted tree. We define the function $f_k : V(T) \to \mathbb{N}$ in the following recursive way:

- $f_k(v) = 0$ for any leaf v of T;
- for any $v \in V(T)$ with d children, $f_k(v) = \max\{0, d + \sum_{w \in C} f_k(w) - k\}$, where C is the set of children of v.

We prove that, in any tree T rooted at v_0, $f_k(v_0) = 0$ if and only if $sn(T, v_0) \leq k$. Hence, by a simple dynamic programming algorithm:

[c-FGJM+12, j-FGJM+14]

Theorem 24 *For any tree T and any v_0, $sn(T, v_0)$ can be computed in time $O(n \cdot \log n)$.*

We moreover give a combinatorial characterization of the surveillance number of trees.

[c-FGJM+12, j-FGJM+14]

Theorem 25 *For any graph G and $v_0 \in V(G)$, $sn(G, v_0) \geq \max \left\lceil \frac{|N[S]|-1}{|S|} \right\rceil$, where the maximum is taken over all $S \subseteq V(G)$ inducing a connected subgraph of G containing v_0. Moreover, there is equality in the case of trees.*

In [c-GMN+13], we showed that the inequality may be strict.

Finally, we prove that, in interval graphs, the observer wins if it is able to protect some particular paths, whose number is polynomial in the size of the graph. Intuitively, the hardest case for the observer is when the surfer first chooses a direction, left or right (with respect to the realization of the interval graph), and then always goes to its neighbor with leftest (or rightest) end. Hence, to compute the surveillance number of an interval graph it is sufficient to check, for all integers k, if the observer can protect these paths.

[c-FGJM+12, j-FGJM+14]

Theorem 26 *$sn(G, v_0)$ can be computed in time $O(n \cdot \Delta^3)$ in the class of n-node interval graphs with maximum degree Δ.*

Regarding the prefetching motivation, it would be interesting to study the behavior of the surveillance number in the Web graph. That is, we would like to study this problem in random graphs and more specifically in such a graph with power law degree distribution.

1.3.2 Connected and online Surveillance Game

By nature of the web-page prefetching problem, in particular because of the huge size of the web digraph, it is not realistic to assume that a strategy may mark any node of the network, even nodes that are "far" from the current position of the surfer. For this reason, in [c-FGJM+12, j-FGJM+14] we also introduced the *connected* variant of the surveillance game.

A strategy σ is said *connected* if $\sigma(v, M) \cup M$ induces a connected subgraph of G for any M, $v_0 \in M \subseteq V(G)$. In other words, a node may be marked only if it has an already marked neighbor. Note that the basic strategy $\sigma_{\mathcal{B}}$ is connected. The *connected surveillance number* of a graph G with initial node v_0, denoted by $csn(G, v_0)$, is the smallest k such that there exists a winning connected k-strategy starting from v_0.

All the results of the previous section extend to the connected variant of the surveillance game [c-FGJM+12, j-FGJM+14]. In particular, the surveillance number and its connected counterpart are equal in the class of trees and of interval graphs. Therefore, a natural question is to know how both parameters may differ one from the other. By definition, $csn(G, v_0) \geq sn(G, v_0)$ for any graph G and $v_0 \in V(G)$. Moreover, it is easy to find a graph where the two parameters differ (see Figure 1.5). More generally,

[c-FGJM+12, j-FGJM+14]

Lemma 8 *Let $k \geq 2$. There exist a graph G and a vertex $v_0 \in V(G)$ such that $k + 1 = csn(G, v_0) > sn(G, v_0) = k$.*

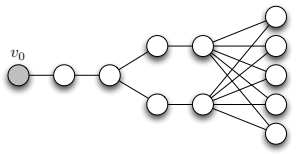


Figure 1.5: A graph G and $v_0 \in V(G)$ such that $csn(G, v_0) = sn(G, v_0) + 1$.

However, finding a larger difference has been much more difficult. In [c-GMN$^+$13] we proved the following result which is currently the best known lower bound. Note that the smallest graph in the family mentioned below has around 1.650.000 vertices.

[c-FGJM$^+$12, j-FGJM$^+$14]

Theorem 27 *There are graphs G, $v_0 \in V(G)$ such that $csn(G, v_0) = sn(G, v_0)+2$.*

On the other hand, except the trivial upper bound $csn(G, v_0) \leq \Delta \cdot sn(G, v_0)$, the best upper bound we have been able to prove is:

[c-FGJM$^+$12, j-FGJM$^+$14]

Theorem 28 *Let G be any connected n-node graph and $v_0 \in V(G)$, then*

$$csn(G, v_0) \leq \sqrt{sn(G, v_0) \cdot n}.$$

To try to get deeper understanding of the cost of connectivity, we defined an even more constrained variant of the surveillance game. We also studied this variant for its own interest since it is a much more realistic model for the Webpage prefetching.

Indeed, the connected surveillance game still seems unrealistic since the web-browser cannot be asked to have the full knowledge of the web digraph. Therefore, we define the *online surveillance game*. In this game, the observer discovers the considered graph while marking its nodes. That is, initially, the observer only knows the starting node v_0 and its neighbors. After the observer has marked the subset M of nodes, it knows M and the vertices that have a neighbor in M. The next set of vertices to be marked depends only on this knowledge, i.e., the nodes at distance at least two from M are unknown. In other words, an *online strategy* is based on the current position of the surfer, the set of already marked nodes and knowing only the subgraph H of the marked nodes and their neighbors (see [c-GMN$^+$13] for more details). By definition, the next nodes marked by such a strategy must be known, i.e., adjacent to an already marked vertex. Therefore, an online strategy is connected.

We are interested in the competitive ratio of winning online strategies. The competitive ratio $\rho(\mathcal{S})$ of a winning online strategy $\mathcal{S}$ is defined as

$$\rho(\mathcal{S}) = \max_{G, v_0 \in V(G)} \frac{\mathcal{S}(G, v_0)}{sn(G, v_0)},$$

where $\mathcal{S}(G, v_0)$ denotes the maximum number of vertices marked by $\mathcal{S}$ in G at each turn, when the surfer starts in v_0. Note that, because any online winning strategy $\mathcal{S}$ is connected, $csn(G, v_0) \leq \rho(\mathcal{S})sn(G, v_0)$ for any graph G and $v_0 \in V(G)$. Unfortunately, we show that

Theorem 29 *The best competitive ratio of online winning strategies is $\Theta(\Delta)$, with Δ the maximum degree.* *[c-FGJM+12, j-FGJM+14]*

Hence, the gap between the surveillance number and its online counterpart is not bounded. Moreover, the previous result holds even when restricted to trees. Since, for any tree T and for any $v_0 \in V(T)$, $csn(T, v_0) = sn(T, v_0)$, there might be an arbitrary gap between the connected surveillance number and the online surveillance number.

The question of bounding the gap between connected and not connected surveillance numbers remains open.

1.4 Fractional Combinatorial games

Motivated by the numerous remaining open questions related to both the Cops and Robber games and the Surveillance game, we propose a new way to investigate them, namely through linear programming techniques [s-GNPS13]. Roughly, we aim at designing Integer Linear Programs (ILP) to describe the games. Then, one possible direction is to relax the integrality constraint and try to derive approximation algorithms. In this section, we present a formal definition of fractional games and we give the preliminary results we obtained in this direction.

1.4.1 Fractional Cops and Robber and Surveillance Games

Cops and Robber game

Let us first consider the classical Cops and Robber game where we relax the "integrality" of the cops. That is, we assume that each cop can divide itself into arbitrary small pieces at each step[6]. Formally, the moves of the cops can be defined as follows: at some turn when an amount $\alpha \in \mathbb{R}^+$ of cops is occupying a node v of a graph, then during its move, an amount of α_w cop can move to w, for any $w \in N[v]$, such that $\sum_{w \in N[v]} \alpha_w = \alpha$. The robber is captured if an amount of at least 1 cop is occupying the same vertex as it.

As an example, let us consider the cycle C_4 (with classical cop-number $cn(C_4) = 2$) with 4 vertices $\{a, b, c, d\}$. A fractional strategy may be to place one-half of cop on each of the vertices a, b and c (then using $\frac{3}{2}$ cops in total). Then, wherever be the node x chosen by the robber, during the next turn, two halves of cops can reach x and then capture the robber. Hence, this example shows that, in the fractional variant, it is possible to capture a robber in a graph G with strictly less than $cn(G)$ cops. We actually proved a more disappointing result:

Theorem 30 *For any graph G and any $\epsilon > 0$, in the fractional variant, $1 + \epsilon$ cops can capture any robber in G and in a linear (in $|V(G)|$) number of turns.* *[s-GNPS13]*

[6] Informally, if one cop is occupying a node then, during its turn, it can move its left leg to one neighbor, its right leg and arm to another neighbor, and keep the remaining of its body on its current position.

Indeed, consider $1+\epsilon$ cops used to capture a robber in a n-node graph. First, place $\frac{1+\epsilon}{n}$ cops in each node. Wherever be the position of the robber, $\frac{1+\epsilon}{n}$ cops are at the same node and can follow the robber at each step of the game. At some step, assume that $0 < \beta < 1$ cop is "following" the robber. Then, the remaining $1+\epsilon-\beta$ cops can spread in the graph such that $\frac{1+\epsilon-\beta}{n}$ cop occupy each node. Then, $\beta + \frac{1+\epsilon-\beta}{n}$ can now "follow" the robber, and the strategy goes on like this. Eventually, at least one cop is following the robber and captures it.

We said that the above result is disappointing because it implies that fractional variant of Cops and Robber game cannot provide an interesting approximation of the classical variant. While we did not get any result in that direction yet, the fractional variant might give some hint to study other "integral" variants of Cops and Robber games such as the variant with fast robber.

Surveillance game

In the fractional surveillance game, the observer is allowed to partially mark the nodes. That is, at each step, the observer marks an amount $m_v > 0$ of any node $v \in V(G)$, such that $\sum_{v \in V(G)} m_v \leq k$.

Recall that the classical surveillance game is NP-hard even when the number of turns equals two [c-FGJM$^+$12, j-FGJM$^+$14]. This is mainly because the surveillance game is close to hitting set problems. For the same reason, it is possible to formulate a strategy for the observer as a linear program. Indeed, consider the game limited to two turns in a graph G and starting from $v_0 \in V(G)$. That is, the fugitive first moves to a neighbor w of v_0 then to a neighbor of w. To decide whether the observer using k marks per turn can win, it is sufficient to find the nodes that must be marked during the first turn: during the second turn, it is necessary and sufficient for the observer to mark the unmarked neighbors of w. Therefore, we use one variable m_v per node $v \in V(G) \setminus N[v_0]$ that is set to 1 if v must be marked during the first turn and to 0 otherwise. The surveillance number of G starting in v_0 and limiting the number of turn to two is then the solution of the following linear program.

$$\begin{array}{lrcll} \text{Minimize} & k & & & \\ \text{Subject to:} & \sum_{v \in V(G) \setminus N[v_0]} m_v & \leq & k - |N[V_0]| & \\ & |N[w] \setminus N[v_0]| & \leq & k + \sum_{v \in N[w] \setminus N[v_0]} m_v & \forall w \in N(v_0) \\ & m_v & \in & \{0,1\} & \forall v \in V(G) \setminus N[v_0] \end{array} \tag{1.1}$$

Clearly, the fractional relaxation of the above program provides a polynomial-
[s-GNPS13] time algorithm to compute the solution for the fractional surveillance game. Moreover, set-cover admits a polynomial-time approximation scheme with logarithmic ratio via LP-rounding [Vaz01]. Therefore, we hope proving that fractional surveillance game also provides such approximation scheme.

In next section, we show how we can extend these ideas to more general games.

1.4.2 General fractional Games

[s-GNPS13]

We conclude this Chapter on Combinatorial games by presenting some on-going work that we expect to have promising consequences. We don't give here all technical details but focus on the main ideas of our new approach.

Not fully formal description of the game

We define a general game played by two players, denoted by $\mathcal{C}$ and $\mathcal{R}$, on some subset of $\mathbb{R}^{2n}$. A *configuration* of the game consists of two vectors: the vector $c \in \mathbb{R}^n_+$ of $\mathcal{C}$ and the one $r \in \mathbb{R}^n_+$ of $\mathcal{R}$. For instance, in Cops and Robber or in Surveillance games played in a n-node graph, r_i represents the amount of the robber at node i, and c_i represents the amount of cops (or of marks) at node i, for any $i \leq n$.

The game is then defined thanks to various polytopes of $\mathbb{R}^{2n}$. The set $\mathcal{V} \subseteq \mathbb{R}^{2n}$ is a polytope defining the valid configurations for Player $\mathcal{C}$, i.e., if Player $\mathcal{R}$ wins if it manages to reach a vector in $\mathbb{R}^{2n} \setminus \mathcal{V}$. The set $\mathcal{I} \subseteq \mathcal{V}$ is the set of initial configurations, and $\mathcal{W} \subseteq \mathcal{V}$ is the polytope of winning configurations for $\mathcal{C}$. For instance, $\mathcal{W} = \{(c,r) \in \mathbb{R}^{2n} \mid c_i \geq r_i,\ \forall i \leq n\}$ is the set of winning set for $\mathcal{C}$ in Cops and Robber game and it is the set of valid configurations in the surveillance game. The set of winning configurations is $\mathcal{W} = \{(c,r) \in \mathbb{R}^{2n} \mid c_i \geq 1,\ \forall i \leq n\}$ in the surveillance game.

Starting from any configuration in $\mathcal{I}$ and turn-by-turn, each player can modify its own vector $v \in \mathbb{R}^n_+$ by applying it some linear transformation: the player may go to a new vector $\delta \cdot v + x$, where x belong to a given convex $\mathcal{X}$ set containing 0, and $\delta \in \Delta_E$ where

$$\Delta_E = \left\{ [\alpha_{i,j}]_{1 \leq i,j \leq n} \;\middle|\; \begin{array}{r} \forall 1 \leq i,j \leq n, \alpha_{i,j} \geq 0, \text{ and} \\ \forall j \leq n, \sum_{1 \leq i \leq n} \alpha_{i,j} = 1, \text{ and} \\ \text{if } \{i,j\} \notin E \text{ then } \alpha_{i,j} = 0 \end{array} \right\}$$

Note that the set Δ_E is a set of stochastic matrices that can be defined through a graph $G = (V, E)$. Intuitively, for any $1 \leq i, j \leq n$, $\delta_{i,j}$ represents the fraction of tokens initially present in $j \in V$ that moved along $\{j,i\} \in E$ to reach $i \in V$. The set $\mathcal{X}$ represents some amount of tokens that may be added at the player's turn.

The goal of $\mathcal{C}$ is to eventually reach a configuration in $\mathcal{W}$, and the goal of $\mathcal{R}$ is either to perpetually avoid it, or to eventually reach a configuration in $\mathbb{R}^{2n} \setminus \mathcal{V}$.

Preliminary results and on-going work

[s-GNPS13]

Without entering more into the details, mainly based on the convexity of the winning states and of the allowed moves, we design an algorithm that computes the set of valid configurations from which $\mathcal{C}$ can win after some number of turns. Roughly, we start by the set of winning configurations which can be described by a polytope. Then, by a backward induction, from the set of configurations from which Player $\mathcal{C}$ wins after i turns, we compute the polytope describing the set of configurations

from which Player $\mathcal{C}$ wins after $i+1$ turns. Unfortunately, our generic algorithm takes an exponential-time even when the number of turns is fixed. The complexity of our algorithm only comes from some projection-step that exponentially increases the number of constraints defining the polytope at each step. However, many such constraints seem redundant and there is still some hope to reduce their number. The time-complexity of deciding which Player may win is still open.

Another important result we obtained is that, under some extra hypothesis on the convex sets defining the game, Player $\mathcal{R}$ can be restricted to play integrally, i.e., we can impose that the vector $r \in \mathbb{N}^n$ always remains integral without making $\mathcal{C}$ more powerful (if $\mathcal{R}$ wins in $\mathbb{R}^n$ then it wins also in $\mathbb{N}^n$). In particular, this implies that our game provides a fractional relaxation of the Cops and Robber games and of the Surveillance game.

One advantage of this new formalization is that it allows us to model various games, not restricted to the Cops and Robber and of the Surveillance games. Another example of game that fits into our framework is the famous *Angels and Devils problem* defined in [BCG82, Con96, BL06, Klo07]. We are also currently investigating the *Eternal games*. In the *Eternal Dominating Set game*, Player $\mathcal{C}$ first places its $k \geq 1$ tokens on vertices of a graph G. Then, at each turn, $\mathcal{R}$ chooses one node and $\mathcal{C}$ may move one or (depending on the variants) several tokens such that at least one token goes to the chosen node [BCG$^+$04, KM09, GHH05]. The *Eternal Vertex Cover game* is defined similarly but Player $\mathcal{R}$ attacks an edge along which at least one token of $\mathcal{C}$ must slide (e.g., [ABB$^+$07, FGG$^+$10]).

Chapter 2

Tree Decomposition, Graph Searching and Applications

Content

One important aspect of pursuit-evasion games is the equivalence between a variant of these games, namely graph searching games, and tree-decompositions. In this chapter, we present our contributions related to this equivalence and to some applications of it.

In Section 2.1, we first present several algorithmic applications of tree-decompositions. More generally, this section is devoted to the description of classical or more recent techniques of parameterized complexity and their applications. Among other, it is question of color coding, cut and count method, meta-kernelization, graph minor theory, bidimensionality, etc. Very recent results (based both on kernelization and dynamic programming) show that many problems can be dealt with using these approaches. We then focus on the problem of computing the chordality of a graph (that have further applications in Section 4.2.2) and describe a particular kind of tree-decomposition, related to treelength (and somehow to our study of Cops and Robber games), and its application to compute chordality of graphs [c-KLNS12, j-KLNS14].

Then, we present our contributions on graph searching. Section 2.2 is first devoted to the study of *non-deterministic graph searching* (defined in [t-Nis07, j-FFN09]) which provides a general approach for graph decompositions [s-ACN07]. Through the notions of *partitioning-trees* and *partition functions*, we obtain general duality results [j-AMNT09] and a unified algorithm to compute decompositions [s-BBM$^+$13]. Section 2.2 is also devoted to survey directed graph decompositions which, unfortunately, seem not fall into our general framework.

On the other hand, a nice directed graph searching variant, namely the *processing game*, has been defined to model the *routing reconfiguration problem* [CPPS05]. In Section 2.3, we present our contributions on this variant [c-CCM$^+$10, j-CCM$^+$11, c-NS13] and give some evidences that it might lead to a "good" definition of directed tree-decomposition. We finally study the routing reconfiguration problem itself by considering various "real-networks" constrains [c-CHM$^+$09, c-CMN09, c-BCM$^+$12].

2.1 Algorithmic applications of tree-decompositions

This section is devoted to present some related work on treewidth and tree-decompositions. In particular, we focus on their applications for computing properties of graphs, via fixed parameterized tractable (FPT) algorithms.

2.1.1 Treewidth

Our aim here is not to survey the huge amount of previous work on treewidth but to mention some important results that are closely related to our main topic, the computation of graph properties. For nice surveys on treewidth, see e.g. [Bod98, Bod07].

Given a graph $G = (V, E)$, a *tree-decomposition* $(T, \mathcal{X})$ of G consists of a tree T and a family $\mathcal{X} = (X_t)_{t \in V(T)}$ of subsets of V such that

- $\cup_{t \in V(T)} X_t = V$;
- for any $e \in E$, there is $t \in V(T)$ such that the *bag* X_t contains both ends of e;
- for any $v \in V$, $\{t \in V(T) \mid v \in X_t\}$ induces a subtree of T.

The width of $(T, \mathcal{X})$ is $\max_{t \in V(T)} |X_t| - 1$, the maximum size of a bag minus one. The *treewidth*[1] of G, denoted by $tw(G)$, is the minimum width among all tree-decompositions of G [RS86a]. If T is restricted to be a path, $(T, \mathcal{X})$ is a *path-decomposition* of G, and the pathwidth $pw(G)$ of G is the minimum width among all path-decompositions of G [RS83].

A fundamental result related to treewidth is that many NP-complete problems can be solved "efficiently" in the class of graphs with bounded treewidth. Indeed, given a tree-decomposition of a graph, many problems can be solved by dynamic programming on the decomposition in a time (mainly) depending on the width of this decomposition. In particular, the famous Courcelle's theorem states that all problems expressible in monadic second order logic can be solved in linear-time in the class of graphs with bounded treewidth [Cou90]. That is, the algorithm is linear in the size of the input graph G but may be exponential in $tw(G)$.

The second interest of treewidth that is worth to be mentioned here is its implication in the Graph Minors theory of Robertson and Seymour. In particular, they proved that the set of bounded treewidth graphs is well-quasi-ordered [RS90] and that the class of graphs excluding a fixed planar graph as a minor has bounded treewidth [RS84, RS86b] which is an important step in their proof of the Wagner conjecture [RS04]. The fact that excluding a planar graph as minor implies having bounded treewidth mainly relates on the so called grid-theorem saying that any graph either has bounded treewidth or admits a large grid as minor [RST94]. We will see later the role played by the following recent improvement of Robertson-Seymour and Thomas theorem in the bidimensionality theory.

[1] Note that, while the term treewidth has been defined by Robertson and Seymour in their work on Graph Minors, similar notions already appeared in previous work, e.g., [Ros74].

Theorem 31 *Let $r > 0$. Let G be a graph with no $r \times r$-grid as minor, then*

- $tw(G) \leq 2^{O(r^2 \log r)}$ *[KK12]*[2]
- *if G is H-minor free, then $\exists c_H > 0$ such that $tw(G) \leq c_H r$ [DH08b, KK12]*

There are graphs with treewidth $\Omega(r^2 \log r)$ and no $\omega(r) \times \omega(r)$-grid as minor [RST94].

Deciding if the treewidth, respectively the pathwidth, of a graph is at most k is NP-complete when k is part of the input [ACP87, MHG+88]. However, since in most applications, computing a tree-decomposition with "small" width is desired, many work has been devoted to this task. For k fixed, several polynomial-time algorithms (exponential in k) have been proposed that either decide if the input graph has treewidth $> k$ or compute a tree-decomposition of width at most $O(k)$ (e.g., [MT91, Ree92, Lag96] that are based on finding "good" balanced separators). Bodlaender and Kloks designed an algorithm that, given a tree-decomposition with width $O(k)$ of a graph G, either decides that $tw(G) > k$ or computes a tree-decomposition (resp., path-decomposition) with width $\leq k$ for G in time $k^{O(k^3)}n$ [BK96]. Bodlaender and Thilikos used similar approach for graph searching [BT97a]. Using the Bodlaender-Kloks algorithm and improving the computation of an approximate decomposition (based on a pre-processing considering vertices with "low" degree), Bodlaender designed a linear-time algorithm that decides if $tw(G) \leq k$ in time $k^{O(k^3)}n$ in any n-node graph G [Bod96]. Very recently, Bodlaender *et al.* designed an algorithm that, for any input n-node graph G and an integer $k > 0$ being fixed, either outputs that $tw(G) > k$, or gives a tree decomposition of G of width at most $5k + 4$, in time $2^{O(k)}n$ [BDD+13]. To conclude this paragraph on treewidth computation, let us mention that the best (as far as we know) known polynomial-time approximation algorithm for computing the treewidth has approximation ratio $\sqrt{\log OPT}$ [FHL05a] (based on Semidefinite Programming), that treewidth can be approximated in polynomial-time up to a constant ratio in planar graphs [ST94] (algorithm for computing the branch-decomposition) and that several heuristics that compute lower or upper bounds on treewidth have been proposed, e.g., in [BB05, BK07, BK10, BK11]. Note that the complexity of computing the treewidth of planar graphs is a longstanding open problem.

Other aspects on tree-decompositions (duality results, case of directed graphs) are postponed to Section 2.2.

2.1.2 Applications to parameterized complexity

Since many interesting problems are known to be NP-complete in general, a huge amount of research is devoted to obtain efficient algorithms in particular graph classes. For instance, for any fixed integer $k > 0$, there is an algorithm that, given an input n-node graph G, decides if $tw(G) \leq k$ in time $O(n^{k+2})$ [ACP87]. Following this result, computing the treewidth is polynomial in the class of graphs

[2] A very recent unpublished result improves this bound to a polynomial in r [CC13].

with bounded treewidth. However, it is desired to optimize the dependence on the parameter k. For this purpose, the $k^{O(k^3)}n$-time Bodlaender algorithm mentioned above [BK96] is better in the sense that it allows to understand that the complexity of the problem does not come from the size of the graph but from its structure. Parameterized complexity theory provides a framework for such a refined analysis of hard algorithmic problems [DF99, FG06, Nie06].

A *parameterized problem* Π takes an input instance I and a fixed non-negative integer k (the *parameter*). Π is Fixed Parameter Tractable (FPT) if it can be solved by an algorithm with running time $f(k)n_I^{O(1)}$, where n_I is the size of the instance and f is a function depending only on k. The typical parameter is the size of the solution, but when graph problems are concerned, using the treewidth of the input graph as parameter has allowed to achieve numerous interesting results.

In particular, as already mentioned, many graph problems can be solved, by "simple" dynamic programming approach, in time $c^{O(tw)}n^{O(1)}$ in the class of n-node graphs, when a tree-decomposition of width tw is given, for some constant c. For instance, vertex cover and independent set can be solved in time $2^{O(tw(G))}n^{O(1)}$ in n-node graphs G [Nie06], dominating set can be solved in time $4^{O(tw(G))}n^{O(1)}$ [Nie06], q-coloring and odd cycle traversal (minimum number of nodes to be removed to obtain a bipartite graph) can be solved in time $c^{O(tw(G))}n^{O(1)}$ for some constants c [FG06, Nie06]. Moreover, such running times are essentially optimal unless the Exponential Time Hypothesis fails [LMS11]. The fact that, for these problems, the dependence in tw is single exponential comes from the fact that solutions are "locally checkable". Roughly, for each bag of the tree-decomposition, the dynamic programming process can only keep the partial solutions (sets of $2^{O(tw)}$ nodes) that can be checked using a constant number of bits by testing only neighborhoods.

On the other hand, for many problems involving more "global constraints" such as connectivity, a similar approach leads to running time $tw^{O(tw)}n^{O(1)}$. For instance, in problems like connected vertex cover, connected dominating set or hamiltonian cycle, the dynamic programming process have to keep track of all the ways (orderings, connected components, etc.) in which the (partial) solution can traverse the corresponding bag of the tree-decomposition. Finding algorithms for such problems with running time depending only (single) exponentially in tw has been a long-standing challenge. Dorn *et al.* first solved it for Hamiltonian cycle, TSP and longest cycle in planar graphs [DPBF10]. Their approach is based on the sphere-cuts of Seymour and Thomas [ST94]: basically, in planar graph a branch-decomposition can be found such that all separators correspond to "cycle" in the graph. Therefore, the number of ways the solution can traverse the bags is related to Catalan number which gives a single exponential in the size of the bag (see also [ST10]). Later on, Dorn, Fomin and Thilikos generalized these results to bounded-genus graphs [DFT06] and then to H-minor free graphs [DFT12]. Very recently, Bodlaender *et al.* proved that similar results can be obtained in general graphs by using clever techniques to keep track of the connected components of the partial solutions [BCKN13]. The latter result improves (by de-randomizing it

and extending it to counting and weighted problems) the *cut-and-count* method proposed in [CNP+11].

The *bidimensionality theory* takes use of several algorithms described above to obtain FPT sub-exponential time algorithms for many problems (see [DH08a] and references therein). Consider the problem of deciding if some graph parameter p is at most k^2. Moreover, assume that (1) p is large in a $k \times k$-grid $G_{k\times k}$, say $p(G_{k\times k}) > k^2$, and (2) p is closed under taking minor. Then an algorithm consists of the following: first run on the input n-node graph G a polynomial time algorithm that either computes a tree-decomposition of width $O(k)$ or states $tw(G) > k$; in the first case, use above dynamic programming algorithms to decide in time $2^k n^{O(1)}$ (i.e., sub-exponential in the parameter); in the latter case, if G admits a "good" structure (e.g., exclude a given minor), by Theorem 31, G contains $G_{k\times k}$ as minor and so $p(G) > k^2$ by the properties of p [DHT05, DH08a]. This technique has then been extended for parameter with less constrained properties (e.g., contraction-closed and large in "grid-like" graphs) but reducing a bit the considered classes of graphs: planar, bounded genus, excluding a fixed graph as minor [DFHT05, DH08a].

Another aspect of parameterized complexity is that it provides a framework to study the pre-processing of instances of hard problems [GN07, Bod09]. Given a problem Π with parameter k, a kernelization algorithm transforms inputs (I, k) of Π to other instances (I', k') of Π in time polynomial in $|I| + k$ and such that $(I, k) \in \Pi$ if and only if $(I', k') \in \Pi$ (i.e., without modifying the value of the solution), and $k' \leq k$ and $|I'| \leq f(k)$ where f is any function depending on k. $f(k)$ is the *size* of the kernel. Clearly, if Π is decidable and admits such a kernelization algorithm, then it is FPT since, after having computed the kernel, an exhaustive search can be done in time depending only in $f(k)$. It is part of the folklore that:

Theorem 32 *[Bod09] A parameterized problem Π is FPT if and only if Π is decidable and Π has a kernelization algorithm.*

One critical question is then to reduce the size of the kernel of FPT problems. For instance, a linear kernel (i.e., of size $O(k)$) leads to FPT algorithms that are "only" single exponential in k. Recent results provide techniques to show that some problems have no polynomial kernel (unless $NP \subseteq coNP/poly$, or other properties making the complexity hierarchy collapse) [BDFH09, FS11]. On the other hand, powerful techniques have been introduced that provide polynomial kernel for large classes of parameterized problems [BFL+09, FLMS12, KLP+13]. Once again, treewidth plays an important role: these results are mainly based on *protusion decompositions* that roughly consist in partitioning a graph: one part having bounded (in the parameter) treewidth and the other part being composed with bounded number of components of bounded diameter and with bounded border. Moreover, because of properties of the considered problems, the connected components of the second part can be replaced (without modifying the solution) by bounded size gadgets.

This section has presented several powerful applications of treewidth and tree-decompositions. However, computing "good" tree-decompositions in an efficient

way (especially, with efficient implementations) remains a challenge. A way to bypass it would be to propose other structures easier to compute and that provide nice algorithmic applications. In the next section, we present some preliminary research in this direction.

2.1.3 Computing chordality and "caterpillar" tree-decompositions

We focus on the particular problem of computing the chordality of graphs. The *chordality* of a graph is the length of its largest induced (chordless) cycle. A graph is *k-chordal* if its chordality is at most k. In Section 4.2.2, we will present some applications of the results we obtained on chordality for compact routing.

Computing longest cycles or paths in graphs are natural problems that have been studied a lot. As a generalization of the Hamiltonian cycle/path problems, these problems are NP-complete [GJ90]. FPT-algorithms have been proposed in various graph classes (planar, bounded genus, H minor free for fixed H, etc.). Maybe the most famous one is the *color coding* method by Alon *et al.* [AYZ95]. Roughly, given a graph G excluding some fixed H as minor, randomly color the nodes with k colors and give acyclic orientation with bounded out-degree to the edges (this is possible since G excludes H fixed and then has bounded degeneracy). Then, by dynamic programming, it is possible in "FPT-time" to search for a directed path of length k (number of vertices) and using k distinct colors. If G has a path of length k, then the algorithm finds it with constant probability. Using hash functions, it is finally possible to de-randomized the algorithm [AYZ95]. Other FPT algorithms have been designed to decide if a graph contains some graph with k nodes as subgraph. Among other, let us mention the linear algorithm of Eppstein in bounded genus graphs [Epp99] which, as far as I understand, is a kind of ancestor of the kernelization algorithms based on protusion (see also [Bak94]).

The problem of deciding whether the chordality of a graph G is at most k is NP-complete if k is as part of the input. Indeed, if G' is obtained by subdividing all the edges in G once, then there is an induced cycle of length at least $2|V(G)|$ in G' if and only if G has a Hamilton cycle. It is coNP-hard to decide whether an n-node graph G is k-chordal for $k = \Theta(n)$ [Ueh99]. Several problems related to chordality have been considered. Finding the longest induced path is $W[2]$-complete [CF07]. In [KK09], the problem of deciding whether there is an induced cycle passing through k given nodes is studied. This problem is NP-Complete in planar graphs when k is part of the input and in general graphs even for $k = 2$. However, a FPT algorithm based on the Graph Minors machinery exists in planar graphs [KK09]. Finding an induced cycle of size exactly k in d-degenerate graph is FPT if k and d are fixed parameters [CCC06]. On the other hand, the chordality is closed under contraction and large in planar triangulated grids and, therefore, it can take advantage of the bi-dimensionality framework: Sau and Thilikos designed a FPT-algorithm for chordality in planar graphs [ST10].

Our approach has been different. Our study of Cops and Robber games in k-chordal graphs (see Theorem 9 in Section 1.1.2) led us to the following algorithm.

Let us define a k-caterpillar as a graph that has a dominating set which induces a chordless path of order at most $k-1$.

Theorem 33 *There is an algorithm that takes a m-edge-graph G and $k \geq 3$ as input and, in time $O(m^2)$ (independent on k),* *[c-KLNS12, j-KLNS14]*

- *either returns an induced cycle of length at least $k+1$;*
- *or returns a tree-decomposition of G in which each bag induces a k-caterpillar.*

By definition of k-chordal graph and Theorem 33, any k-chordal graph admits a *caterpillar tree-decomposition*, i.e., a tree-decomposition in which each bag induces a k-caterpillar. The next corollary follows easily, improving exponentially the upper bound on treewidth of k-chordal graphs in [BT97b].

Corollary 2 *Any k-chordal graph G with maximum degree Δ has treewidth at most $(k-1)(\Delta-1)+2$, tree-length at most k and hyperbolicity at most $\lfloor \frac{3}{2}k \rfloor$.*

There is an algorithm that, given a m-edge graph G and $k \geq 3$, states that either G has chordality at least $k+1$ or G has hyperbolicity at most $\lfloor \frac{3}{2}k \rfloor$, in time $O(m^2)$.

Our caterpillar-decomposition seems to be a promising restriction of *treelength* (see [DG07, Lok10]). We are currently investigating algorithmic applications of it.

2.2 From pursuit-evasion games' point of view

Graph searching games provide a powerful tool for studying graph decompositions (e.g., see [KP86, Bie91, Bod98, Bod07]). In [t-Nis07], we defined a non-deterministic graph searching that provides a generalized approach for both path-decompositions and tree-decompositions. In this section, we present the recent results we obtained on this variant and their applications to the understanding of graph decompositions.

2.2.1 Non-deterministic Graph Searching, branched decompositions

In graph searching[3], a team of *searchers* aims at capturing a *fugitive* running in a connected undirected simple graph G [KP86]. The searchers and the fugitive occupy the nodes of G. The fugitive is arbitrary fast and can goes through the paths of G as long as it does not meet a searcher. It is captured if a searcher occupies the same vertex as it and if the fugitive cannot escape, i.e., all neighbors of its position are also occupied. Searchers can be placed at or removed from the nodes of G. A *strategy* for the searchers is a sequence of *steps* (placement or removal) that results in capturing the fugitive whatever it does[4]. The number of searchers *used* by a

[3]The variant presented in this section is referred to as *node graph searching* [KP86].

[4]Note that an important difference with the games presented in Chapter 1 is that, in graph searching, both players play simultaneously, not turn-by-turn.

strategy is the maximum number of searchers simultaneously occupying nodes of the graph.

The corresponding optimization problem is to minimize the number of searchers needed to capture a fugitive in a graph G. If the fugitive is invisible, this number is the *node search number* of G, denoted by $ns(G)$. For visible fugitive (i.e., the strategy of the searchers may be guided by the current position of the fugitive), this number is the *visible node search number* of G, denoted by $vns(G)$.

A strategy is said *monotone* if each node is occupied at most once by a searcher. A crucial property is that both variants visible and invisible are *monotone*. That is, in any graph G, there is a monotone strategy capturing the invisible, resp. the invisible, fugitive using $ns(G)$, resp., $vns(G)$, searchers [BS91, LaP93, ST93]. The main consequences of the monotonicity property is, first, that the problems of computing the search numbers are in NP and, second, that these parameters are equivalent to pathwidth and treewidth respectively because monotone strategies are equivalent to path- and tree-decompositions. More precisely:

Theorem 34 *[KP86, Bie91, ST93, Bod98] For any graph G, $ns(G) = pw(G) + 1$ and $vns(G) = tw(G) + 1$.*

From above theorem, it follows that computing visible or invisible search numbers are NP-hard problems. However, they can be solved in polynomial-time in several graph classes among which the class of trees. Precisely, the search number (and corresponding strategy) of trees can be computed in linear-time [EST94, PHsH$^+$00, Sko03, CHM12], while visible search number trivially equals 2 in trees. For more related work on graph searching, we refer to [t-Nis07, FT08, BY11] and to Chapter 3 of this thesis.

In [c-FFN05, j-FFN09, t-Nis07], we introduced non-deterministic graph searching as a unified version of visible and invisible graph searching games. In non-deterministic graph searching with parameter $q \in \mathbb{N} \cup \{\infty\}$, the fugitive is *a priori* invisible, but the searchers have the ability to see it a limited number q of times. More precisely, in a non-deterministic strategy in a graph $G = (V, E)$, the allowed steps are: to place a searcher at a node, to remove a searcher from a node or to perform a *query*. When occupying the nodes in $X \subseteq V$ and performing a query, the searchers learn the "position" of the fugitive, i.e., the connected component of $V \setminus X$ occupied by the fugitive. The *q-limited search number* of a graph G, denoted by $ns_q(G)$, is the smallest number of searchers required to capture a fugitive in G in this setting (i.e., using at most q query-steps) [c-FFN05, j-FFN09, t-Nis07]. In particular, $ns_0(G) = ns(G)$ and $ns_\infty(G) = vns(G)$ for any graph G.

In [j-MN08, c-MN07], we proved that non-deterministic graph searching is monotone. In [c-FFN05, j-FFN09], we have shown that computing the q-limited search number is NP-hard in star-like graphs, for any fixed $q \in \mathbb{N} \cup \{\infty\}$, and we designed exact exponential algorithm for computing it. The monotonicity result once again allows us to prove the equivalence between non-deterministic graph searching and a particular tree-decomposition, namely the *q-branched decomposition*.

Given a rooted tree T, with root r, a *branching node* of T is a node with at least two children. Let $q \geq 0$. A tree T is *q-branched* if it can be rooted in a node r and every path in T from (root) r to a leaf contains at most q branching nodes. A tree-decomposition $(T, \mathcal{X})$ of a graph G is q-branched if T is q-branched. The *q-branched treewidth* of a graph G, denoted by $tw_q(G)$, is the minimum width of any q-branched tree decomposition of G.

Theorem 35 *Let $q \geq 0$, and G a graph, $tw_q(G) = ns_q(G) - 1$.* *[c-FFN05, j-FFN09, j-MN08, c-MN07]*

Motivated by the fact that pathwidth of trees are well characterized, we investigated non-deterministic graph searching in trees to get a better understanding of q-branched decompositions. First let us recall the known results:

Theorem 36 *[Sko03, MHG+88] For any tree T on n vertices, $ns_0(T) \leq 1 + \log_3(n-1)$ and $ns_0(T)$ can be computed in time linear in n.*

We design a polynomial-time two-approximation algorithm to compute $sn_q(T)$ in any tree T and for any $q \in \mathbb{N} \cup \{\infty\}$. Our algorithm is exact for $q \in \{0, 1\}$.

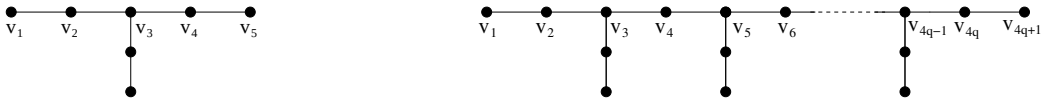
Theorem 37 *There is an algorithm that, for any tree T and any integer $q > 0$, returns an integer $k \leq 2 \cdot ns_q(T) + 1$ and a corresponding q-limited search strategy using k searchers, in time polynomial in $|V(T)|$ (independent of q).* *[s-ACN07]*

Moreover, if $q \in \{0, 1\}$, $k = ns_q(T)$.

Our algorithm proceeds by intricate dynamic programming on the tree and is based on two main technical results. First, we show how to compute in polynomial-time a node strategy in a graph G when the initial positions of some searchers are imposed, and using the smallest number of searchers needed in this setting. In other words, given a graph G and $X \subseteq V(G)$, our subroutine computes a path-decomposition with first bag X and with minimum width among all the path-decompositions starting from X. Second, we define a restricted version of non-deterministic graph searching where, once a searcher is placed on some contaminated vertex, it cannot be removed as long as the next query has not been performed (unless no query remains). We show that this constraint requires at most twice the number of searchers of the initial variant and we prove that the corresponding search-number can be computed in polynomial time using the subrouting.

After considering the minimization of the number of searchers given a fixed amount of queries, we study the "dual" problem of minimizing the number of queries that are required for a fixed number of searchers to clear a tree. More precisely, for any tree T and $k \geq 2$, let $q_k(T)$ be the smallest number of queries required to clear T using at most k searchers. First, we design an algorithm that computes $q_2(T)$ for any tree T in time quadratic in the size of T. Then, we derive some lower and upper bounds:

Theorem 38 *For any tree T, $q_2(T) \leq \lceil |V(T)|/8 \rceil$, and this bound is tight.* *[s-ACN07]*



(a) T_1, $q_2(T_1) = 1$ (b) T_q, $q_2(T_q) = \lceil |V(T_q)|/8 \rceil$

Figure 2.1: Tree T_q, $q \geq 1$, with $8q - 1$ nodes such that $q_2(T_q) = q = \lceil |V(T_q)|/8 \rceil$.

The lower bound of the previous theorem is achieved by the trees depicted in Figure 2.1.

[s-ACN07]

Theorem 39 *For any tree T on n nodes, and for any $k \geq 3$, $q_k(T) \leq 2\lceil \log_2 n \rceil$. Moreover, for any fixed k the bound is asymptotically tight: for any n_0, there exists $n \geq n_0$ and a tree T_n on n nodes such that $q_k(T_n) = \Omega(\log_2 n)$.*

Our main result concerning trees is a polynomial-time 2-approximation algorithm for computing $s_q(T)$ for any tree T and $q \geq 1$. However, for any fixed $q > 1$, the complexity of the problem to decide whether $ns_q(T) \leq k$ ($k > 2$ being part of the input) remains open. The fact that the parameter ns_q is minor-closed implies that the problem is Fixed Parameter Tractable, i.e., there is an algorithm which decides in time $f(k) \cdot \text{poly}(|V(T)|)$ whether $ns_q(T) \leq k$, where f is some function depending only on k. Since, $ns_q(T) \leq ns_0(T) = O(\log |V(T)|)$ [MHG+88], $f(k) = c^{O(k)}$ for some constant c would lead to a polynomial-time algorithm.

In next section, we present general results on graph decompositions that somehow follow from our techniques to prove monotonicity of non-deterministic graph searching.

2.2.2 Partitioning trees and general set decompositions

Duality treewidth/bramble

The monotonicity result that we provide in [j-MN08, c-MN07] is actually a unification of the different techniques given in [BRST91, BS91, RS91, ST93]. One important consequence of these techniques is the characterization of "dual" structures for various decompositions: *blockages* for path-decompositions [BRST91], *tangles* for branch-decompositions [RS91], or *brambles* for tree-decompositions [ST93].

As an example, a *bramble* in a graph $G = (V, E)$ is a set $\mathcal{B}$ of pairwise touching[5] subsets of V. The *order* of $\mathcal{B}$ is the minimum size of a hitting set, i.e., the smallest number of vertices in V that intersect each set in $\mathcal{B}$.

The min-max theorem for treewidth of Seymour and Thomas can be stated as:

Theorem 40 *[ST93] In any graph G, the maximum order of a bramble in G equals $tw(G) + 1$.*

[5]Two sets $X, Y \subseteq V(G)$ are touching if either they intersect or if there is an edge $xy \in E(G)$ with $x \in X$ and $y \in Y$.

Intuitively, a bramble of order k provides an escape strategy for a visible fugitive against $k-1$ searchers. Indeed, let $\{B_1, \cdots, B_r\}$ be a bramble of order k in a graph G. It is proved in [ST93] that it can be taken such that each B_i induces a connected subgraph of G. Using at most $k-1$ searchers, there is always a subgraph $G[B_i]$ for some $i \leq r$ that is free of searcher. Because the elements of the bramble are connected and are pairwise touching, the fugitive can always safely reach the element of the bramble that is free of searcher.

Therefore, the maximum order of a bramble in G equals the visible search number of G. On the other hand, the monotone visible search number of G equals its treewidth plus one. Hence, we get directly that the maximum order of a bramble in G is at most $tw(G)+1$. Seymour and Thomas proved the other inequality by a technical work on tree-decomposition and derived the monotonicity of visible graph searching from this. On the opposite, our proof of the monotonicity [j-MN08, c-MN07] allows us to derive their result.

In what follows, we explained how we generalized our technique to other decompositions, not restricted to graphs.

Note, however, that other duality results related to graph searching game seem not to be related to our approach, namely the LIFO-search (searchers have labels and a searcher can be removed only if it is the smallest searcher present at a some node) is monotone and leads to a min-max theorem between tree-depth and cycle-rank in (di)graphs [GHT12].

Partitioning-trees and Partition functions

A *partition function* of a set E is a function from $\mathcal{P}(E)$, the set of partitions of E[6], to $\mathbb{R} \cup \{\infty\}$. A *partitioning-tree* (T, σ) of E is a tree T together with a one-to-one mapping σ from E to the set of leaves of T. Hence, every internal node v or edge e of T corresponds to the partition T_v (or to the bipartition T_e) of E whose parts are the set of leaves of the subtrees obtained by deleting v (or e). Given a partition function Φ of E, the Φ-width of a partitioning-tree (T, σ) of E is the maximum of $\Phi(T_v)$, $\Phi(T_e)$ over all internal nodes v of T and edges e of T. The Φ-width of E is the minimum width of the partitioning-trees of E [j-AMNT09].

Using the appropriated partition functions, partitioning-trees generalize most of the graph-decompositions. For instance, in a graph $G = (V, E)$, for any partition $\mathcal{P} = (E_1, \cdots, E_r)$ of E, let $\delta(\mathcal{P})$ be the number of vertices that are adjacent to edges in at least two parts of $\mathcal{P}$. The δ-width of E is equivalent to the treewidth of G, and any partitioning-tree of E corresponds to a tree-decomposition of G [j-AMNT09]. Some graph decompositions also require to add restrictions on the tree-structure: any partitioning-tree (T, σ) of E (resp., of V) where internal nodes of T have degree three is a branch-decomposition (resp., a carving decomposition) of G.

We also abstract the notion of brambles for any partition function Φ. A *k-bramble* $\mathcal{B}$ of E is a set of pairwise intersecting subsets of E such that for any

[6] A partition of E is a set $(P_1, \cdots, P_r)$ of pairwise disjoint non-empty subsets of E with $\cup_{i \leq r} P_i = E$.

partition $\mathcal{P} = (E_1, \cdots, E_r)$ of E with $\Phi(\mathcal{P}) \leq k$, there is $i \leq r$ such that $E_i \in \mathcal{B}$. In [j-AMNT09], we give a min-max theorem linking the Φ-width of a set with the existence of particular brambles, when Φ satisfies some properties.

Let $\mathcal{X} = \{X_1, \ldots, X_n\}$ be a partition of E and let $F \subseteq E$ and $F^c = E \setminus F$. Let us define the following notation. $\mathcal{X}_{X_i \to F^c} := \{X_1 \cap F, \ldots, X_{i-1} \cap F, X_i \cup F^c, X_{i+1} \cap F, \ldots, X_n \cap F\}$.A partition function Φ is *weakly submodular* if for every pair of partitions $\mathcal{X} = \{X_1, \ldots, X_n\}$ and $\mathcal{Y} = \{Y_1, \ldots, Y_l\}$, and for all $i \neq j$, at least one of the following holds:

1. There exists F such that $X_i \subseteq F \subseteq (Y_j \setminus X_i)^c$ and $\Phi(\mathcal{X}) > \Phi(\mathcal{X}_{X_i \to F})$;
2. $\Phi(\mathcal{Y}) \geq \Phi(\mathcal{Y}_{Y_j \to X_i^c})$.

Using similar techniques as in [j-MN08] to transform partitioning-trees, we get:

[j-AMNT09]

Theorem 41 *Let Φ be a weakly submodular partition function on a set E and $k \in \mathbb{R}^+$. All k-brambles contain a singleton if and only if E has Φ-width at most k.*

The above theorem actually generalizes all previous duality results and extends it to new decompositions such as the treewidth of matroids [j-AMNT09]. Later on, this result has been improved in [LMT10] where the shown duality result does not rely on the submodularity of partition functions but on a combinatorial property of the partitions. Moreover, based on the results of [LMT10], an exponential-time algorithm to compute brambles has been designed in [CMT09].

[s-BBM+13]

In [s-BBM+13], we investigate sufficient conditions for a partition function Φ such that there exists a polynomial time algorithm that decides if a set has Φ-width at most k, k being a fixed parameter. Basically, these properties abstract the fact that corresponding parameters are close under taking minor and that they can be computed by dynamic programming only remembering some bounded information concerning the "border" of the subgraphs. Our algorithm is based on dynamic programming and generalize all algorithms in [BK96, BT97a, TSB00, BT04, TSB05a, TSB05b]. Moreover, our algorithm allows to take into account the structure of the partitioning trees and, therefore, provides the first FPT algorithm for computing the q-branched treewidth or the special treewidth [Cou10b].

Hence, the notions of partition function and partitioning tree allow us to unify and generalize several results on graph decompositions. While our FPT algorithm has some theoretical interest (in particular, it is sufficient to check that a partition function satisfies some properties to directly get an explicit FPT algorithm), it is far to be practical. Therefore, designing efficient FPT algorithms (e.g., that are "only" exponential in the parameter) for computing graph decompositions is still a challenge, even in very restricted graph classes. For instance, the best known algorithm that computes optimal path-decompositions of outerplanar graphs has complexity $O(n^{11})$ [CHS07, FT07].

Moreover, our duality result does not cope with rooted tree-decompositions such as branched tree-decompositions and all decompositions of directed graphs. In next section, we briefly discuss about the case of decompositions of directed graphs.

2.2.3 Related work on decompositions of directed graphs

During the last decade, several digraph decompositions have been proposed in order to try to bring to directed graphs the same algorithmic power as tree-decompositions provide in undirected graphs. Interestingly, most of these attempts have been defined through graph searching games. An important difference between directed graph searching games and undirected ones arises via the notion of monotonicity. In the directed case, there are two distinct definitions of monotonicity: a game is *cop-monotone* if each node is occupied at most once by a searcher, it is *robber-monotone* if the area reachable by the robber never increases. Clearly a cop-monotone game is robber-monotone. However, as shown below, the converse is not always true.

Johnson *et al.* first defined the *directed tree-decomposition* which roughly "translates" the connectivity properties of tree-decomposition into strong connectivity properties in directed graphs [JRST01]. Their variant is closely related to the graph searching game where the visible fugitive has the extra constraint that it can move only in strongly connected components free of searchers. That is, the fugitive can go from nodes u to v if there is a directed path from u to v free of searchers and a directed path from v to u free of searchers. It has been shown that, in this game, the non-monotone, the cop-monotone and the robber-monotone variants may differ [JRST01, Adl07]. Adler also show that directed treewidth is not closed under taking butterfly minors [Adl07]. Because of the non-monotonicity result, no min-max theorem can be expected via graph searching. However, [JRST01] proved a weaker result: if k searchers have a winning strategy in a digraph D, then $3k-1$ searchers have a robber-monotone winning strategy in D, which leads to a min-max theorem up to a constant ratio between directed treewidth and *havens* [JRST01]. In [EHS13], it is proved that the cop-monotone version of this game is actually equivalent to the D-width defined by Safari [Saf05] leading to an exact algorithm for computing this variant. Moreover, they show that D-width and directed treewidth are actually equivalent (in the sense that one is bounded if and only if the other is bounded) [EHS13]. On the algorithmic applications side, Johnson *et al.* proved that hamiltonian path, hamiltonian circuit and k-disjoint paths (k fixed) problems can be solved in polynomial time for digraphs with bounded directed treewidth [JRST01].

For tackling other problems, other digraph decompositions have been proposed. The DAG-decomposition is weaker than directed tree-decomposition (bounded DAG-width implies bounded directed treewidth) [BDH$^+$12]. It corresponds to the cop-monotone version of the game where the visible fugitive is constraint to follow the direction of arcs. While robber-monotone and cop-monotone variants coincide [BDH$^+$12], this game is not monotone [KO11]. The winner of parity game is polynomially solvable in digraphs of bounded DAG-width [BDH$^+$12]. However, a drawback of DAG-decomposition is that the best known upper bound of the size of such decomposition with width k in a n-node digraph is $O(n^k)$ (it is not known whether deciding if a digraph has DAG-width at most k is in NP). Another decomposition weaker than directed tree-decomposition is the Kelly-decomposition that

corresponds to the robber-monotone variant of the the game where an inert[7] fugitive forced to follow the arcs [HK08]. Again, this game is not monotone [KO11]. A polynomial-time algorithm to recognize digraphs with Kelly-width at most 2 is given in [MTV10]. Approximation algorithms for computing directed treewidth, Dag- and Kelly-width, with approximation ratio $O(\log^{3/2} n)$ have been designed in [KKK11].

Until now, no powerful enough digraph decomposition has been found. Indeed, many NP-hard problems remain untractable in class of graphs with bounded Dag-,Kelly- or directed tree-width, e.g., feedback arc/vertex set or digraph grundy coloring [KO11]. Unfortunately, Ganian *et al.* give evidence that such a "good" digraph decomposition is unexpected to be substantially different than the tree-decomposition of the underlying undirected graph [GHK$^+$10]. A width digraph measure δ is *powerful* if any MSO_1 definable decision problem can be solved in polynomial time in digraphs with bounded δ-width. More precisely, Ganian *et al.* show that, if δ is a digraph width measure that is (1) not treewidth-bounding (there is no computable function b such that $\delta(D) \leq k$ implies that $tw(UD) \leq b(k)$ where UD is the underlying undirected graph of digraph D), (2) monotone under taking directed topological minors and (3) efficiently orientable (roughly, the δ-widths of two digraphs with same underlying graph cannot differ too much), then $P = NP$ or δ is not powerful [GHK$^+$10]. They also show that relaxing the second condition by "monotone under taking subdigraphs" makes possible the existence of powerful widths.

To conclude, let us mention that several directed path-decompositions and directed invisible graph searching have also been proposed [Bar06, Yan07, YC07, YC08b, YC08a]. Contrary to their visible counterparts, all are monotone. In next section, we study another such a variant (coming from routing problems), prove its monotonicity and its equivalence with a digraph decomposition. It would be interesting to study the algorithmic applications of such a decomposition. Moreover, the visible variant of this game could provide a "powerful" decomposition.

2.3 Application to Routing Reconfiguration: Processing game

In previous section, we have seen that no "good" decomposition of directed graphs have been defined yet. In this section, we study a directed graph searching game, namely the *processing game*, introduced by Coudert *et al.* to model routing reconfiguration problem [CPPS05]. While our study mainly focus on applications to routing reconfiguration, we also give some hints that pocessing game could lead to a "good" decomposition of directed graphs.

[7] A fugitive is inert if it is invisible but can move only when a searcher is going to land on the same node. This version has been introduced for undirected graphs in [ST99].

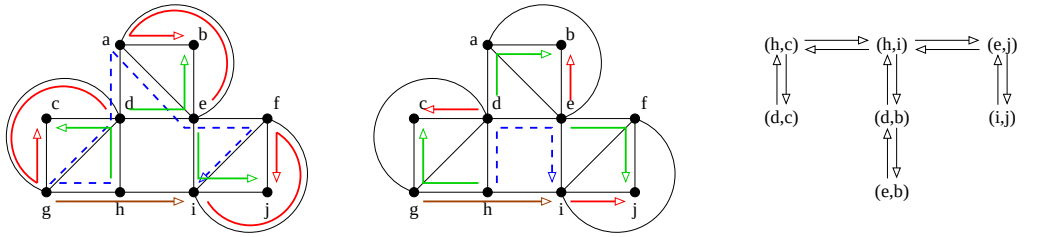
2.3.1 Routing reconfiguration in WDM networks

In connection oriented networks such as Wavelength Division Multiplexing (WDM), SONET (Synchronous optical networking), or MPLS (Multiprotocol Label Switching) networks, each connection request $d \in \Pi$ is assigned a *lightpath* under the *wavelength continuity constraint*, that is to say a path in the topology and an end-to-end wavelength. The *Routing and wavelength assignment* (RWA) problem consists in assigning such a lightpath for each request under the constraint that two optical paths sharing a fiber have distinct wavelengths [Muk92, JMT07] (see also Chapter 5 in [Cou10a] and references therein). The set of lightpaths, or routes, obtained in that way is called a *configuration*.

Current such networks are becoming more flexible, offering new on-demand services (on-demand TV, mobile Internet, peer-to-peer) which leads to a constant evolution of the traffic pattern. Moreover, these networks also allow better management of maintenance operations (requiring to switch off equipments) and equipments failures – due to earthquake, tsunami, or a backhoe unfortunately breaking a pipe containing some fibers. A building block for flexibility and reliability is the possibility to *reconfigure* the routing, that is to compute new optical paths for some connection requests and then to switch the traffic from former to new optical paths. Such process may however affect the quality of service by inducing potential traffic disruptions. Moreover, due to physical layer impairments [SS09], reconfiguring the routing – setting up the new lightpaths – induces some cost for the network operator, in particular if service level agreements are not fulfilled. Thus, the routing reconfiguration process must be carefully optimized.

A classical approach for reconfiguring the routing is based on the *Move-to-Vacant* (MTV) scheme [LL96, MM99, CBL07]. Basically, the MTV scheme consists in sequentially choosing a lightpath, computing a new route using available resources for the corresponding request, and then switching the request from its current lightpath to its new route in a *make-before-break* fashion as standardized for MPLS networks [MP06]. This process is repeated sequentially until the reached configuration achieves the desired constraints (e.g., overall usage of resources, availability of a desired route). However, such a greedy policy leads to a poor usage of resources and so to higher blocking probability: new connection requests might be rejected although the network has enough resources to serve all the traffic, up to the rerouting of some existing connections. The main issues when using the MTV approach are to guarantee the convergence of such a process and to control the number of route changes. Moreover, such solutions are not sufficient and interruptions may be necessary [JS03].

To ensure both a fast termination and that the final configuration satisfies the desired performance criteria, another approach consists in pre-computing the target configuration and then to focus on the reconfiguration itself, that is deciding in which ordering the existing lightpaths should be switched to achieve the final configuration. More formally, the *reconfiguration problem* is to determine the "best" sequence (order) of connections rerouting to move from the current configuration



(a) Initial set of routes $\mathcal{R}_1$. (b) Final set of routes $\mathcal{R}_2$. (c) Dependency digraph D from S_1 to S_2.

Figure 2.2: Instance of the reconfiguration problem consisting of a network with 10 nodes and symmetric arcs, 8 connections $(h,i),(h,c),(d,c),(d,b),(e,b),(e,j),(i,j),(g,i)$ to be reestablished. Figure 2.2(a) depicts the initial set of routes $\mathcal{R}_1$, Figure 2.2(b) the final set $\mathcal{R}_2$, and Figure 2.2(c) the dependency digraph from $\mathcal{R}_1$ to $\mathcal{R}_2$.

to a predetermined target configuration, under the constraint that the connections are moved one by one [JS03]. However, the final lightpath of a request $d \in \Pi$ may use resources (e.g., a wavelength on a fiber) that are used by the initial lightpath of another request $d' \in \Pi$. This latter request d' must then be moved before the connection d. The difficulty of the reconfiguration problem lies mainly in the existence of dependency cycles that require to temporarily suspend some connections to allow switching the other requests. Such an interruption corresponds to the concept of *break-before-make* standardized for MPLS networks [MP06]. A break-before-make starts by interrupting the lightpath of a request before establishing the new route. Several recent studies have considered the framework proposed by Jose and Somani [JS03] to minimize either the total number of interruptions, or the maximum number of requests that are simultaneously interrupted, during a reconfiguration [CPPS05, Sol09, SP10].

Formal description

More formally, the physical network is modeled by a digraph $G = (V, E)$ whose arcs have capacity one. Given an instance $\mathcal{I} \subseteq 2^{V \times V}$ of connection requests, a *routing* $\mathcal{R}$ is a set of arc-disjoint directed paths associated to the requests, one directed path from x to y in G for each request $r = (x, y) \in \mathcal{I}$. So $\mathcal{R}(r)$ is the route of request r in G. The *routing reconfiguration problem* consists in switching connection requests one after the other from an initial routing $\mathcal{R}_1$ to a precomputed routing $\mathcal{R}_2$ in such a way that the smallest number of requests are simultaneously interrupted. An important assumption is that, once a request has been (re)routed, it cannot be moved anymore. In this model, each link of G has capacity one, i.e., each directed link may be used by at most one request. In [CS11], the dependency digraph $D = (W, A)$ of $\mathcal{R}_1$ and $\mathcal{R}_2$ is defined as follows. W is the set of requests

$r \in \mathcal{I}$ with different routes in $\mathcal{R}_1$ and $\mathcal{R}_2$. Moreover, there is an arc from request u to request v if there exists a link e of G that belongs to both paths $\mathcal{R}_2(u)$ and $\mathcal{R}_1(v)$.

As an example, Figure 2.2 presents a physical network with 10 nodes and 8 connection requests. Figures 2.2(a) and 2.2(b) describe the initial and final routing respectively. Finally, the corresponding dependency digraph is depicted in Figure 2.2(c). A way to reconfigure the instance depicted in Figure 2.2 may be to interrupt connections $(h, c), (d, b), (e, j)$, then set up the new paths of all other connections, tear down their old routes, and finally, set up the new paths of connections $(h, c), (d, b), (e, j)$. Such a strategy interrupts a total of 3 connections and these ones are interrupted simultaneously. Another strategy may consist of interrupting the connection (h, i), then sequentially: interrupt connection (h, c), reconfigure (d, c) without interruption for it, set up the new route of (h, c), then reconfigure in the same way first (d, b) and (e, b) without interruption for these two requests, and then (e, j) and (i, j). Finally, set up the new route of (h, i). The second strategy implies the interruption of 4 connections, but at most 2 connections are interrupted simultaneously.

The *process number* of D, denoted by $pn(D)$, is a digraph invariant that equals the smallest number of requests that have to be simultaneously interrupted during the reconfiguration phase. In particular, $pn(D) = 0$ if and only if D is a DAG. Using this formulation, polynomial-time algorithms have been designed to decide whether a digraph has process number 1 or 2 [CS11]. A polynomial-time and distributed algorithm that computes the process number of symmetric trees is given in [CHM12]. However, the problem of computing the process number of a digraph has been shown NP-complete and difficult to approximate [CS11]. This is a bad news concerning the routing reconfiguration problem because there is no hope that the real instances of this problem lead to restricted classes of dependency digraph. A hope would have been to show that dependency digraphs of real instances would belong to particular digraph classes where the problem would be polynomial. However, we proved:

Theorem 42 *Any digraph D is the dependency digraph of an instance of the routing reconfiguration problem where the physical network is a grid.* *[j-CCM+11, c-CCM+10]*

Actually, the routing reconfiguration problem can be reformulated as a graph searching problem in the dependency digraph. In what follows, we describe the results that we obtained on the reconfiguration problem and on this graph searching game.

2.3.2 Processing Game and digraph decomposition

The *processing game* has been defined in [CPPS05] as it models the routing reconfiguration problem. Recall that, in a dependency digraph of an instance of the reconfiguration problem, the nodes represents to the requests and the arcs model dependency between ressources.

In the *processing game*, the searchers aim at *processing* all nodes of a digraph. A node is said *safe* if all its out-neighbors are either occupied or already processed. Given $D = (V, A)$ where all nodes are initially unoccupied and not processed, a *process strategy* is a sequence $(s_1, \ldots, s_n)$ of steps that results in processing all nodes of D. Each step s_i is one of the following moves: place a searcher at a node, process a safe *unoccupied* node, or process a safe *occupied* vertex and remove the searcher from it. The *process number* of D, denoted by $pn(D)$, is the minimum number of searchers that are needed to process D.

In this game, the placement of a searcher at a node models the interruption of the corresponding request. The processing of a safe node represents the re-routing of the corresponding request because the resources it requires are available (because the node is safe, all requests corresponding to its out-neighbors have already been rerouted or interrupted). Hence, minimizing the number of searchers is then equivalent to minimizing the number of simultaneous interruptions in the reconfiguration problem. For instance, in the dependency digraph of Figure 2.2(c), the two strategies corresponding to the two reconfiguration processes described in the example above are the following. A first strategy consists of first placing searchers at the nodes in $X = \{(h, c), (d, b), (e, j)\}$, then processing all other nodes, and finally processing the nodes in X. Another strategy would be to place one searcher at (h, i) and sequentially, place a second searcher at an unprocessed neighbor x of (h, i), process the unoccupied neighbor of x, then process x and finally remove the searcher at x. After all nodes but (h, i) have been processed, (h, i) can be processed which concludes the strategy.

This initial variant of processing game is inherently monotone. Indeed, once they have been rerouted, the requests are not considered anymore, i.e., processed nodes are not recontaminated.

A new monotone digraph searching game

In [c-NS13], we extend the definition of the Processing game to allow recontamination: a processed node becomes unprocessed as soon as it has an unoccupied and unprocessed out-neighbor. Our main result is that recontamination does not help, i.e., $pn(D)$ searchers are still sufficient to process any digraph D in this variant.

[c-NS13] **Theorem 43** *The processing game is monotone.*

This result has interesting side-effects. It allows us to prove that processing strategies are the algorithmic counter-part of a particular digraph decomposition and that the process number is invariant when reversing all arcs of a digraph [c-NS13].

All together, these results show that the processing game is the following graph searching game. In a digraph D, searchers may be placed at or removed from the nodes of D. An invisible fugitive is running with arbitrary speed along the arcs of D while it does not meet any searcher. The fugitive has the additional constraint that it must always move. The fugitive is captured if a searcher lands at the same

node and the fugitive cannot escape, or if the fugitive runs into a searcher (because of the constraint of always moving), or if it cannot move anymore. The process number of D is the minimum number of searchers that are needed to capture the fugitive.

Note that, in symmetric directed graphs, the process number differs by at most one from the pathwidth of the underlying undirected graph [CPPS05]. It would be very interesting to study the variant of this game when the fugitive is visible. Indeed, it is a variant of the game defined by Johnson *et al.* (the fugitive is more constrained) corresponding to the directed tree-decompositions [JRST01]. Hence, as pointed out in previous section, it might offer new algorithmic perspectives.

Tradeoff: from Minimum Feedback Vertex Set to Process number

A trivial upper bound for the process number of a digraph D is the minimum size of a feedback vertex set (FVS) of D[8], denoted by $mfvs(D)$. Indeed, placing searchers at each node of a feedback vertex set of D allows to capture the fugitive. While such a strategy may use a lot of searchers (compared with the process number of D), such a strategy using $mfvs(D)$ searchers will capture the fugitive by occupying the smallest number of nodes.

We then tried to establish tradeoffs between these two objectives: minimizing the number of vertices that are occupied during a process strategy and minimizing the number of searchers used by a strategy. In terms of the routing reconfiguration problem, these objectives correspond respectively (1) to minimize the total number of disrupted connections [JS03], and (2) to minimize the maximum number of concurrent interruptions [CPPS05, Sol09, SP10].

A (p, q)-*process strategy* denotes a process strategy for D using at most p searchers and covering at most q vertices. Given an integer $q \geq mfvs(D)$, we denote by $pn_q(D)$ the minimum p such that a (p, q)-process strategy for D exists. On the other hand, for any $p \geq pn(D)$, let $mfvs_p(D)$ be the minimum q such that a (p, q)-process strategy for D exists. These parameters are illustrated in Figure 2.3.

We first give some complexity results.

Theorem 44 *Let $p \geq pn(D)$, $q \geq mfvs(D)$ be fixed integers, and D a digraph. Computing $pn_q(D)$, $\frac{pn_q(D)}{pn(D)}$ is NP-complete and cannot be approximated up to an additive constant. Computing $mfvs_p(D)$, $\frac{mfvs_p(D)}{mfvs(D)}$ is not in APX.* *[j-CCM+11, c-CCM+10]*

Then, we show that no good tradeoffs can be expected in general. That is, restricting the number of occupied vertices may arbitrarily increase the number of searchers:

Theorem 45 *For any $C > 0$, $q \in \mathbb{N}$, there exists a digraph D s.t. $\frac{pn_{mfvs+q}(D)}{pn(D)} > C$.* *[j-CCM+11, c-CCM+10]*

[8] $F \subseteq V(D)$ is a FVS of a digraph D if the digraph induced by $V(D) \setminus F$ is acyclic.

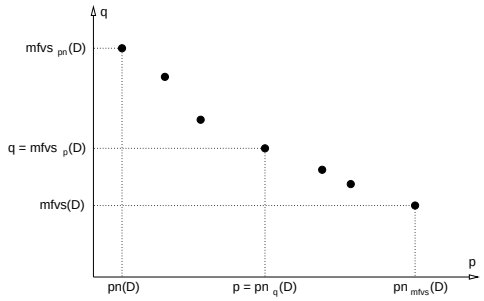


Figure 2.3: Representation of the tradeoff's parameters. In abscissa, p denotes the number of searchers. In ordinate, q denotes the number of occupied vertices.

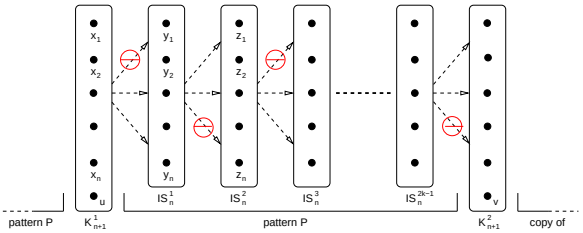


Figure 2.4: Digraph D used in proof of Theorem 46

Similarly, restricting the number of searchers may arbitrarily increase the number of occupied vertices. To prove the next Theorem (setting $n = p + 2$ and $k > \frac{4C-3}{2}$), we define the digraph D of Fig. 2.4. K^1_{n+1} is a symmetric clique of $n+1$ nodes $x_1, \ldots, x_n, u$. IS^1_n and IS^2_n are two independent sets of n nodes each: respectively $y_1, \ldots, y_n$ and $z_1, \ldots, z_n$. In D, there is an arc from x_i to y_j, $i = 1, \ldots, n$, $j = 1, \ldots, n$, if and only if $j \geq i$. There is an arc from y_i to z_j, $i = 1, \ldots, n$, $j = 1, \ldots, n$, if and only if $i \geq j$. The other arcs of D are built in such a way for other independent sets $IS^3_n, \ldots, IS^{2k-1}_n$ and the symmetric clique K^2_{n+1}. These arcs and the independent sets form the pattern P (see Fig. 2.4). Between K^2_{n+1} and K^1_{n+1}, the same pattern is built.

[j-CCM+ 11, c-CCM+ 10]

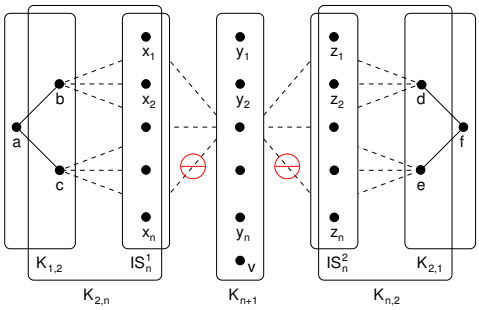
Theorem 46 *For any $C > 0$ and any integer $p \geq 0$, there exists a digraph D such that $\frac{mfvs_{pn+p}(D)}{mfvs(D)} > C$.*

For $p = 0$, the digraphs D of previous theorem have process number 3 (independently of C). We then addressed the behaviour of $\frac{mfvs_{pn}(D)}{mfvs(D)}$ for symmetric digraphs D. In contrast, the ratio is not arbitrary large anymore when the process number is bounded.

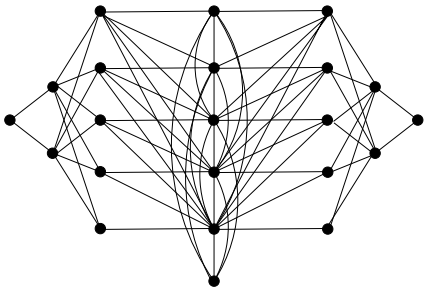
[j-CCM+ 11, c-CCM+ 10]

Lemma 9 *For any symmetric digraph D, $\frac{mfvs_{pn}(D)}{mfvs(D)} \leq pn(D)$.*

It would be interesting to know if the ratio $\frac{mfvs_{pn}(D)}{mfvs(D)}$ is bounded by a constant in symmetric digraphs D. Such a constant would be at least 3 as shown in next



(a) Symmetric digraph D of Lemma 10



(b) Same digraph D when $n = 5$.

Figure 2.5: Symmetric digraph D of Lemma 10 (a) and D when $n = 5$ (b).

lemma. The digraph used in this lemma can be built as follows. Let D be the symmetric digraph of Fig. 2.5(a). Let IS_n^1 and IS_n^2 be two independent sets of n nodes each: respectively $x_1, \ldots, x_n$ and $z_1, \ldots, z_n$. Let K_{n+1} be a symmetric clique of $n+1$ nodes $y_1, \ldots, y_n, v$. In D, there are two symmetric arcs between x_i and y_j, and between z_i and y_j, $i = 1, \ldots, n$, $j = 1, \ldots, n$, if and only if $j \geq i$. Furthermore the two right nodes of $K_{1,2}$ and nodes of IS_n^1 form a complete symmetric bipartite subgraph (the same construction for $K_{2,1}$ and IS_n^2). The symmetric digraph of Fig. 2.5(b) represents D when $n = 5$.

Lemma 10 *$\forall \varepsilon > 0$, there exists a symmetric digraph D s.t. $\frac{mfvs_{pn}(D)}{mfvs(D)} \geq 3\text{-}\epsilon$.* *[j-CCM+11, c-CCM+10]*

2.3.3 Adding constraints from telecommunication networks

In previous subsection, we have presented our study of routing reconfiguration with the graph searching point of view. On the other hand, we have also investigated this problem by introducing constraints arising in real networks. Not surprisingly, the problems become much more difficult. Moreover, it appears that, unfortunately, the graph searching perspective is not adapted anymore to study several of these problems (at least, we did not manage to use graph searching paradigm for obtaining interesting results).

Priority connexions

[c-CHM+09]

In [c-CHM+09], we propose a model to handle several classes of service. During the reconfiguration process as described above, any connection may be forced to be interrupted, and so this induces some traffic perturbations that clients may not accept. Moreover, some clients may sign a specific contract forbidding interruptions, and so the operator offers two classes of services. To cope with these two classes, we introduce the new constraint that imposed some particular connections, called the *priority connections*, not to be interrupted. In the process number game formulation, this constraint is modeled by a specific subset (given as input of the problem)

of the nodes of the dependency digraph that cannot host searchers. That is, the single way to process these *priority nodes* is to deal with all their outneighbors first. Given a set $Q \subseteq V(D)$ of priority nodes, let $pn_Q(D)$ be the minimum number of searchers required to process the digraph D without placing any searchers at the nodes in Q. Note that a direct cycle of priority nodes in the dependency digraph makes the reconfiguration impossible, in which case, we set $pn_Q(D) = \infty$.

We show that this problem is actually equivalent to the problem without priority connections. More precisely, when reconfiguration is possible (if $pn_Q(D) < \infty$), we present a simple transformation of any digraph D with a set $Q \subseteq V(D)$ of priority nodes into another digraph D^* (with size polynomial in the size of D) without priority nodes such that $pn(D^*) = pn_Q(D)$ [c-CHM$^+$09]. Moreover, we proved that, for every digraph D and set $Q \subseteq V(D)$ of priority nodes, if $pn_Q(D) < \infty$, then $pn_Q(D) \leq pn(D) + |N^+(Q)|$. Moreover, this bound is asymptotically tight [c-CHM$^+$09]. We also gave an heuristic, based on random walks (similar to Pagerank), that computes an upper bound of the process number. Simulations show that our heuristic is efficient in some graph classes (circular arc graphs and digraphs with process number 2) but far to be optimal in graphs with a lot of symmetries (grids) [c-CHM$^+$09].

Shared bandwidth

[c-CMN09]

One clear limitation of the model studied in previous subsection is the fact that links are supposed to have capacity 1. That is, this model considers that a connection request uses all the bandwidth of a link (e.g. wavelength) and it is too limited to handle cases in which a request uses only a fraction of the bandwidth of a link (e.g. MPLS, SONET and so traffic grooming).

In [c-CMN09], we proposed a model generalizing [CS11, JS03] to handle cases in which a request uses only a fraction of the bandwidth of a link. The main difficulty of the reconfiguration problem comes from the fact that the final routing only gives the links used by the requests but not the unit of capacity. Therefore, we have some choice when performing the scheduling of the reconfiguration phase. We have shown that the routing reconfiguration problem becomes much more difficult. Precisely, we proved that, when the requests have bandwidth requirement 1 and that links have capacity at least 3, the problem of deciding whether there exists a scheduling of the rerouting without traffic interruption is NP-complete [c-CMN09]. In contrast, when arc's capacities are equal to 1, this problem corresponds to decide whether the dependency digraph is acyclic.

Physical cost

[c-BCM$^+$12]

Finally, we addressed the problem of reconfiguration in a different way, which is the inclusion of physical constraints to the establishment of an optical path [c-BCM$^+$12]. We aim at optimizing the reconfiguration cost which is induced by physical layer impairments. Indeed, the transmission of an optical signal in a fiber is subject to

many parameters: bandwidth, transmission power, signal attenuation requiring the use of amplifiers every 50-80 km, phase shifts associated the imperfection of the laser and the distortion of the fiber, and various electro-magnetic effects (see [SS09] for more details). This requires extremely fine adjustments to ensure good transmission quality, but everything has to be redone (or adapted) when a new wavelength is used in the fiber. Thus, setting up a new lightpath in a network has a cost (energy, time and/or man power) due to the recalibration on all fibers used by the path which depends (non-linearly) on the number of wavelengths already present. In addition, these changes can affect the entire network by spreading corrections to be made. See [MCK+09] for an example of the inclusion of these effects in the computation of optical routing.

We presente a first theoretical study of the routing reconfiguration problem including physical layer impairments. We model the cost of rerouting a request in a configuration as a simple non linear function depending on the *load* of the links used by the lightpath to be established, where the load of a link is the number of lightpaths crossing it in the current configuration. Our cost function also depends on a tunable parameter $\alpha \geq 0$ which is an exponent, e.g., $\alpha = 1$ corresponds to a linear cost function. While simple, the cost function we proposed allows to capture several constraints due to physical layer impairments as the non-linearity.

Assuming that the reconfiguration is possible without interruptions, our objective is to compute a scheduling of the requests that achieves a minimum global cost. We prove that this problem is NP-complete even in a physical network with two nodes (and parallel links) and when $\alpha = 0$ [c-BCM+12]. We then characterize a particular class of instances where the problem can be solved in linear time, for any $\alpha \geq 0$. In the particular case of a ring topology and $\alpha = 1$, we give a simple linear algorithm for solving the problem. Finally, we design several heuristics and report on numerical simulations [c-BCM+12].

It is a bit disappointing to see that adding more realistic constraints makes the problems become very difficult even when restricted to simple topologies. One hope here is therefore to use our study of graph structures in order to propose efficient pre-processing to reduce the size of the instances, or to design better heuristics.

Chapter 3

Graph Searching: Connectivity, Exclusivity, Distributed settings

---

Content

As mentioned in previous chapters, graph searching has been studied because it provides an alternative definition of graph decompositions and also because it allows to model and study telecommunication networks' problems such as routing reconfiguration. In this chapter, we investigate graph searching from another point of view. Namely, the clearing of graphs is a natural application to study the computational power of mobile agents. We study the minimum abilities that mobile agents must be endowed for clearing a graph in a distributed way.

In Section 3.1, we recall the definitions of the different "classical" variants of graph searching and their relationship with graph decompositions studied in previous chapters.

Section 3.2 focuses on *connected graph searching* where the strategies must ensure that the clear part of the graph must always induce a connected subgraph. This property has been initially defined because it ensures safe communications between the searchers in a distributed environment. We first survey the recent results on this area, the main of which being that the connected search number of a graph is at most twice its pathwidth [Der12b], answering one of the main open questions of my Ph.D. thesis. Then we present our contributions on the cost of the monotonicity in a distributed setting where searchers explicitly communicate using whiteboards on the nodes [c-INS07, j-INS09]. This contribution concludes some work in [t-Nis07].

In Section 3.3, we present our contributions on *exclusive graph searching*. We have introduced this variant because it solves two serious limitations of classical graph searching as far as practical applications are concerned: the fact that a node can be simultaneously occupied by several searchers and the fact that searchers can jump from one node to any other one. On the other hand, the exclusivity property (that stated that two searchers can never occupy the same node simultaneously) allows to study the graph searching problem in the distributed *Look-Compute-Move* model which somehow provides an approach of fault-tolerant graph searching. We start by presenting the general centralized results we obtained on exclusive graph searching [c-BBN12, c-BBN13, s-MNP14], then we describe algorithms to solve this problem in the Look-Compute-Move model in various graph topologies [c-BBN13, c-DDN$^+$13b, j-DDN$^+$14b, c-DNN14].

3.1 Quick reminder on Graph Searching

In this section, we recall the basic definitions of *search strategies* and *search numbers* and the main known results. Our goal is not to be exhaustive but rather to focus on the new results that have been developed since my Ph.D. thesis. For a more complete overviews on graph searching, we refer to [t-Nis07, FT08, BY11] and the recent book [Bre12].

Graph searching and search numbers

Graph searching [Par78a] aims at clearing the nodes and edges of a graph using a team of mobile agents, called *searchers*. Actually, this is equivalent to the capture of an invisible fugitive in G as considered in Section 2.2. For instance, Breish initially introduced this problem to model the rescue of a lost person in the network of caves [Bre67]. In this chapter, we refer to the clearing semantic because it fits better with the considered applications.

Initially all nodes and edges of a graph $G = (V, E)$ are *contaminated*. A node is *cleared* when it is occupied by a searcher. An edge $e \in E$ is *cleared* if either searchers occupy simultaneously both its ends, or a searcher slides along e. However, an unoccupied clear node is *recontaminated* as soon as there is a path free of searchers from it to a contaminated node. Similarly, an edge is recontaminated if one of its ends is recontaminated.

A *strategy* consists of a finite sequence of *steps*, or *moves*, that results in a state where all nodes and edges are clear, where each step consists in either sliding a searcher along an edge, or placing a searcher at some node of the graph, or removing a searcher from a node of G. The number of searchers *used* by a strategy is the maximum number of searchers present in the graph among all its steps. The *search number*, denoted by $s(G)$, of a graph G is the smallest integer k such that there is a strategy that clears G using k searchers.

The variant of graph searching defined above is usually refer to as *mixed graph searching* [BS91]. Two other equivalent variants have also been defined. In *node graph searching* [Bie91] (see also Section 2.2), a strategy is defined as above but the only possible moves are the placement and the removal of a searcher (the searchers don't slide along edges). On the other hand, in the *edge search* variant [Par78a], the strategies are defined similarly as mixed strategies (allowed moves are place, remove and slide) but the edges can be cleared only by sliding a searcher along it. The corresponding graph invariants are respectively the *node search number*, denoted by ns, and the *edge search number*, denoted by es.

These three variants are very close one from each other (note that both edge and node strategies are mixed strategies). In particular, for any graph G, $ns(G) - 1 \leq es(G) \leq ns(G) + 1$ and $s(G) \leq ns(G) \leq s(G) + 1$ (all inequalities are tight) [KP86]. Simple (and polynomial-time) transformations allow to go from one of these variants to another one [KP86]. For instance, Kirousis and Papdimitriou proved that $s(G^+) = es(G)$ for any graph G where G^+ is obtained from G by subdividing each

edge once [KP86]. Since the problem of computing the edge-search number is NP-hard in general graphs [MHG+88], this result also holds for other variants. Note however that the complexity of computing the difference between two variants of search numbers in a graph is still unknown.

Link with graph decompositions and complexity

An important property of graph searching is the *monotonicity*. Namely, for any graph G, if there is a node strategy that clears G using k searchers, then there is a node strategy that clears G using k searchers and without any recontamination. A strategy is said *monotone* if no recontamination occurs during it. The monotonicity property can be reformulated as follows:

Theorem 47 *[BS91] For any graph G, there is a monotone mixed (resp., node, resp., edge) strategy that clears G using $s(G)$ (resp., $ns(G)$, resp., $s(G)$) searchers.*

The above property is very important. First, it ensures the polynomial length of optimal strategies and therefore, it shows that the corresponding decision problems are in NP. Second, it is the corner stone of the following theorem that establishes a link between the node search number of a graph and its *pathwidth* $pw(G)$.

Theorem 48 *[Bie91] For any graph G, $ns(G) = pw(G) + 1$.*

Because path decompositions offer nice algorithmic tools (see Chapter 2) and because of the above equivalence, many work has been devoted to compute optimal search strategies in various graph classes. Let us mention some results about complexity of graph searching.

A graph is *cubic* if its maximum degree is at most 3. The problem of computing the edge search number has been shown to be NP-hard in the class of cubic planar graphs [MS88]. Moreover, the reduction from edge search to mixed search of Kirousis and Papadimitriou preserves planarity and maximum degree [KP86]. More precisely, the transformation in [KP86] consists in subdividing once each edge, therefore, in the resulting graph, the set of vertices with degree exactly three induces an independent set. All together, it gives the following result that we will use in Section 3.3:

Theorem 49 *[MS88, KP86] The problem of computing the (mixed) search number is NP-hard in the class of cubic planar graphs where the set of vertices with degree exactly three induces an independent set.*

On the other hand, search problems can be solved in various graph classes in polynomial time. Among other, the mixed search number can be computed in polynomial time in interval and split graphs [FHM10a, FHM10b] and permutation graphs [HM08]; a polynomial time algorithm has been designed for the edge search number in cographs [HM09, GHM12]; the pathwidth is polynomial-time computable in circular arc graphs [ST07] unicyclic graphs [EM04], Biconvex Bipartite Graphs [PY07] some subclasses of chordal graphs [PTK+00] etc.

As discussed in Section 3.2.2, trees are very important structures when graph searching is concerned, hence, the case of trees has been particularly studied [Par78a, MHG+88, EST94, PHsH+00]. In particular, Skodinis designed a linear-time algorithm for computing the pathwidth of trees [CHM12]. A generic and distributed algorithm for computing in time $O(n \log n)$ any of the search numbers in n-node trees (only the initial setting of the algorithm differ) has been designed in [CHM12], where the interesting notion of hierarchical decomposition of trees is introduced.

3.2 Distributed connected Graph Searching

In this section, we focus on *connected graph searching* in a distributed setting. That is we focus on edge graph searching with the additional constraint of connectivity: an edge search strategy is said *connected* if the clear area (the subset of clear nodes) always induces a connected subgraph. First, we recall what connected graph searching is and we survey the recent advances in this area.

3.2.1 Recent progress on Connected Graph Searching

The connectivity constraint has been introduced in [BFFS02, BFST03].

A connected search strategy $\mathcal{S}$ in a graph $G = (V, E)$ and using $k \geq 1$ searchers can be defined as follows. First, a node $v_0 \in V$ is chosen and all the k searchers are placed at it. Then, $\mathcal{S}$ is a sequence of moves, where each move consists in sliding a searcher at $u \in V$ along an edge $e = \{u, v\} \in E$ and such a move is allowed only if, after the sliding, there is path of clear edges from v_0 to v. In other words, either the edge e is cleared by the move (and remains clear) or node v was already clear before this move. The *connected search number* of a graph G, denoted by $cs(G)$, is the smallest k such that there exists a connected search strategy that clears G using k searchers.

The previous definition clearly allows recontamination. Monotone connected search strategies are defined in a similar way: first, a node $v_0 \in V$ is chosen and all the k searchers are placed at it, then, the strategy consists of a sequence of moves, where each move consists in sliding a searcher at $u \in V$ along an edge $e = \{u, v\} \in E$ only if either u is still occupied by a searcher after the move, or all incident edges of u but possibly e were already clear before the move. One important and surprising result is that, contrary to the classical graph searching, in the connected variant, recontamination may help [YDA09]. Hence, the *monotone connected search number* of a graph G, denoted by $mcs(G)$, may be strictly larger than its connected search number $cs(G)$. A consequence of this result is that it is not known whether the problem of computing the connected search number of a graph is in NP. Another difference between the search number and its (monotone or not) connected counter part is that it is not minor-closed. Hence, it is not clear whether the problem of computing the (monotone) connected search number of graphs admits a fixed parameter tractable algorithm. However, both parameters are closed under taking contraction (folklore?).

Recontamination does not help for connected graph searching in trees, i.e., $mcs(T) = cs(T)$ in any tree T [BFFS02]. In the same paper, it is shown that computing the connected search number can be done in polynomial time in trees. Recently, it has been shown that recontamination does not help in the class of graphs with connected search number at most two[1]. That is, for any graph G, $mcs(G) = 2$ if and only if $cs(G) = 2$. The result follows the characterization of this class of graphs by exhibiting the family of 177 minimal-contraction obstructions.

On the other hand, connectivity has some price: in [BFST03], it has been proved that $cs(T) \leq 2s(T) - 2$ in any tree T and this bound is tight in trees. Therefore, the question of the cost of connectivity then arises naturally: how far the connected search number of a graph is from its pathwidth? In other words, does there exist a constant $c \geq 2$ bounding the ratio between connected search number and search number in any graph? During my Ph.D., we investigated further this question, proving that $c \leq \log n$ in any n-node graphs [c-FN06a, j-BFF$^+$12] and that c is bounded in the class of graphs with bounded maximum degree and chordality [j-Nis09]. We also proved that, in n-node graphs, the ratio is $\Theta(\log n)$ in the variant that aims at capturing a visible fugitive in a connected way [j-FN08, c-FN06b].

Recently, Dereniowski closed the question by showing that $mcs(G) \leq 2s(G)$ in any graph G [Der12b]. In particular, he designed a polynomial time algorithm that transforms any monotone search strategy using k searchers into a connected one using at most $2k$ searchers. His result also shows that the ratio between monotone connected search number and connected search number is bounded by 2.

To conclude this section, let us mention that Dereniowski also investigated the weighted variant of connected graph searching. In this setting the weight $w(v)$ of a vertex v determines the number of searchers needed to guard it, i.e. if less than $w(v)$ searchers occupy v, then recontamination may spread through this node. Similarly, the weight of an edge e determines the minimum number of searchers that have to simultaneously slide along e to clear it. Following the work of Mihai and Todinca who proved that weighted graph searching is NP-hard in trees [MT09], Dereniowski proved that weighted connected graph searching is also NP-hard in trees [Der11]. On the positive side, a polynomial time approximation with approximation ratio 3 is designed for this problem in [Der12a]. Similar results were proved while with different terminology (speaking about edge-width instead of weight) [BTK11].

3.2.2 Distributed Graph Searching in unknown graphs

A major reason for which the connectivity constraint has been introduced is that it ensures safe communications between the searchers during the execution of the strategy. For instance, when the searchers have to coordinate themselves but have no way to communicate when they are far from each others, possible solutions would be either to leave some messages on the nodes or to use a searcher for carrying

[1] On-going work of Micah J. Best, Arvind Gupta, Dimitrios M. Thilikos and Dimitris Zoros. Personal communication of D. Thilikos.

instructions between other searchers. In both cases, the connectivity constraint helps since it allows to avoid that messages are let on contaminated nodes that the searcher crosses when moving in the contaminated area to transfer instructions.

In this section, we study the clearing of graphs in such environment where the searchers have only a local vision of their environment and must communicate to coordinate the clearing.

Distributed model.

More precisely, the k searchers are modeled by synchronous autonomous mobile computing entities (automata) with distinct IDs from 1 to k. Otherwise searchers are all identical, run the same program and use at most $O(\log n)$ bits of memory, where n is the number of nodes of the network. A network is modeled by an undirected connected graph G . A priori, the network is asynchronous and anonymous, that is, the nodes are not labelled. The edges incident to any node u are labelled from 1 to its degree, so that the searchers can distinguish the different edges incident to a node. Every node of the network has a zone of local memory, the *whiteboard* in which searchers can read, erase, and write symbols. It is moreover assumed that searchers can access these whiteboards in fair mutual exclusion. The goal is then to design an algorithm, called *search protocol*, such that the fewest number of searchers running this algorithm achieves the clearing of the graph. More precisely, the team of searchers must execute a connected edge search strategy for G. In these settings, the searchers do not know in advance in which graph they are launched. That is, when occupying some node u, a searcher executes the algorithm only based on its current state (the *memory* of the searcher), on the content of the whiteboard at u and on the degree of u.

Contributions.

Distributed algorithms for achieving connected search strategies have been designed for several specific graph classes such as trees [BFFS02], grids [FLS05], hypercubes [FHL05b, FHL08], etc. (see also [FHL06, FHL07, Luc07, Luc09]).

During my Ph.D., we attack this problem for general graphs. We first designed a general algorithm allowing $mcs(G)+1$ to connectedly clear any graph G [j-BFNV08, c-BFNV06]. Since, the extra searcher (compared to the centralized case) cannot be avoided due to the asynchronicity of the network, this is optimal. On the positive side, the required memory for the searchers is only logarithmic in the number of searchers. On the other hand, the size of the whiteboards must be polynomial in the size of the graph and, more importantly, the executed strategy is not monotone: the execution time may be exponential.

To face this problems, we investigated the minimum amount of global information on the graph G that the searchers must have in such a way that $mcs(G) + 1$ searchers can clear G in a monotone (and connected) way. We show that, using searchers with memory of $O(\log n)$ bits and whiteboards of size $O(\log n)$, $\Theta(n \log n)$

bits of information are necessary and sufficient in any n-node graphs [j-NS09, c-NS07]. For doing this, we used the framework of computation with *advice* as defined in [FIP10]: searchers are given a priori some string of bits that can be used by the algorithm. Intuitively, this extra information encodes a spanning tree of G "along" which the connected search is done.

After my Ph.D., we consider the same question on another angle: what is the smallest number of searchers that are required to achieve a monotone connected search strategy without any global information. Again, searchers have memory of $O(\log n)$ bits and whiteboards are of size $O(\log n)$. We then show that monotonicity has an important cost (in terms of number of searchers) in a distributed setting [j-INS09, c-INS07].

More precisely, the *cost* of a search protocol $\mathcal{P}$ in a graph G with homebase v_0 is measured by the ratio between the number of searchers it uses to clear G and the search number $mcs(G, v_0)$ of G. This ratio, maximized over all graphs and all starting nodes, is called the *competitive ratio* $r(\mathcal{P})$ of the protocol $\mathcal{P}$. We first showed that:

Theorem 50 *Any search protocol for clearing n-node graphs has competitive ratio $\Omega(\frac{n}{\log n})$.* *[j-INS09, c-INS07]*

Actually, we prove that the above result holds even restricted to the class of trees with maximum degree 3. The intuition is that, since the tree is discover online, we may ensure that, whatever be the protocol, the searchers progress in the "wrong" direction. That is, they clear first a long path, of length $\Omega(n)$, while all nodes of it must be protected by one searcher to avoid recontamination. Hence, the cost of any protocol will be $\Omega(n)$ in an n-node tree while it is known that $mcs(T, v_0) = O(\log n)$ for any tree T and any homebase $v_0 \in V(T)$ [BFFS02, BFST03].

Then, we showed that this lower bound is actually tight.

Theorem 51 *There is a search protocol $\mathcal{P}$ such that, for any connected n-node graph G and any $v_0 \in V(G)$, Protocol $\mathcal{P}$ enables $O(\frac{n}{\log n}\, mcs(G, v_0))$ searchers to clear G in a monotone connected way, starting from v_0.* *[j-INS09, c-INS07]*

In other words, $\mathcal{P}$ has competitive ratio $O(\frac{n}{\log n})$.

This protocol $\mathcal{P}$ maintains a subtree of the (connected) clear part, spanning the clear node that are incident to contaminated edges (in particular, such nodes are occupied). The first goal of T is that it is used for the searchers to communicate and its root is used to host all the currently "unused" searchers. The main goal of T is to help the protocol to determine the next edge to be cleared at each phase. For this purpose, the protocol also maintains a minor S of T (obtained roughly by contracting the unoccupied node with degree two in T). The choice of the next edge to be cleared is done by maintaining S as close as possible to a complete ternary tree. Moreover, the new clear edge increases the depth of S only if no other choice is available. The intuitive reason of this choice is that the complete ternary tree is the tree requiring the (asymptotic) largest number of searchers compared to the size

of the tree, even for a centralized algorithm. Therefore, looking for such a minor in T, i.e., in G, allows to keep some control on the search number of G and, hence, it avoids to use too many searchers.

Our technique let think that it would be interesting to compute the pathwidth of a graph using some substructure (for which path decomposition can be easily computed) of this graph. We discuss this idea further to conclude this section.

Further questions.

Besides its practical motivation, this study of graph searching with limited knowledge has been useful to identify some structures that are simple enough to be computed locally and that allow the clearing of the graph. More precisely, in both works where we impose the monotonicity, the clearing is done using a particular spanning tree of the graph G. The case with advice shows that there exists a spanning tree T such that the clearing of the graph can be done optimally by "following" T [j-NS09, c-NS07]. Clearly, the complexity of the computation of such a tree is hidden in the advice used in [j-NS09, c-NS07]. On the other hand, in [j-INS09, c-INS07], we show that we can somehow greedily (and, a priori, without the knowledge of the whole graph) compute a spanning tree S such that clearing the graph using S gives an approximation algorithm for clearing G using $o(n)s(G)$ searchers. I think it would be particular interesting to identify other structures (or "better" spanning trees) that could be polynomially computable and would allow an efficient clearing of the graph.

In other words, I would like to investigate the design of approximation algorithm for computing the pathwidth[2] of a graph using path-decompositions of simpler substructures of the graph (e.g., spanning trees).

It is also worth to mention that recent work investigates new variants of (centralized or distributed) graph searching. For instance, natural variants are the ones where recontamination is not instantaneous but takes some fixed amount of steps [FMS08, DJS13] or when more than one contaminated neighbor is required to recontaminate nodes [LPS07]. Also, the length of the strategy has been considered [BH06, BTK11, DKZ13]. In next section, we define and study another variant where no two robots can meet in a node or in an edge.

3.3 Exclusive and Perpetual graph Searching

The distributed algorithms described in previous section have several drawbacks. First, the searchers are endowed powerful abilities such as no constant memory. Moreover, the algorithms work in static networks when no faults occur. Hence, we

[2]Note that pathwidth is not approximable up to an additive constant in general [DKL87]. In planar n-node graphs, it can be approximate up to ratio $O(\log n)$, using the polynomial algorithm for branchwidth [ST94]. I am not aware of any other result concerning approximation for pathwidth.

are interested in investigating graph searching in faulty environment and to propose self-stabilizing algorithms for it. We should mention that some self-stabilizing algorithms for graph searching have been proposed in [MM09, BMM10]. However, in this work, it is assume that the searchers can detect which nodes and edges are dirty. To overpass this assumption, we proposed a model for *perpetual* graph searching where searchers have to clear a network infinitely often. For this purpose, we investigated graph searching in the CORDA model where searchers have very weak capabilities but are able to always "see" the current positions of all searchers. This model naturally led us to define *exclusive* graph searching where no two searchers can share a node.

3.3.1 Exclusive graph searching

In this section, we define the *exclusive* variant of graph searching and present the results we obtained about it. The *exclusivity constraint* states that each node can be occupied by at most one searcher at a time. Also, an edge cannot be simultaneously crossed by several searchers, even in different directions. *Exclusive graph searching* is defined as mixed graph searching with extra exclusivity constraint and such that searchers cannot jump from one node to another one, i.e., searchers can only slide along edges [c-BBN12, c-BBN13].

More formally, given a connected n-node graph G, an *exclusive search strategy* in G, using $k \leq n$ searchers consists in (1) placing the k searchers at k different nodes of G, and (2) performing a sequence of *moves*. A move consists in sliding one searcher from one extremity u of an edge $e = \{u, v\}$ to its other extremity v. Such a move can be performed only if v is free of searchers. That is, exclusive-search limits the strategy to place at most 1 searcher at each node, at any point in time. The edges of graph G are supposed to be initially contaminated. An edge becomes clear whenever either a searcher slides along it, or one searcher is placed at each of its extremities. An edge becomes recontaminated whenever there is a path free of searchers from that edge to a contaminated edge. A search strategy is *winning* if its execution results in all edges of the graph G being simultaneously clear. The exclusive-search number of G, denoted by $xs(G)$ is the smallest k for which there exists a winning search strategy in G.

Exclusive graph searching addresses two serious limitations of classical variants (node-, edge- and mixed-graph searching) as far as practical applications are concerned.

- First, classical variants assume that any node can be simultaneously occupied by several searchers. This assumption may be unrealistic in several contexts. Typically, placing several searchers at the same node may simply be impossible in a physical environment in which, e.g., the searchers are modeling physical searchers moving in a network of pipes. In the case of software agents deployed in a computer network, maintaining several searchers at the same node may consume local resources (e.g., memory, computation cycles, etc.). The exclusivity constraint aims at dealing with this problem.

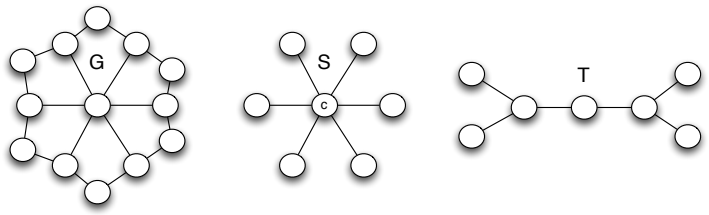


Figure 3.1: Graph S (center) is a star with maximum degree $\Delta = 6$ which is an induced subgraph of G (right) and such that $s(S) = 2 < s(G) = xs(G) = 3 < xs(S) = 5 = \Delta - 1$. Tree T (left) has $xs(T) = 2$ but any monotone exclusive strategy requires at least 3 searchers.

- Second, most variants of graph searching also suffer from another unrealistic assumption: searcher are enabled to "jump" from one node of the graph, to another, potentially far away, node (e.g., see the classical mixed-search, defined above). We restrict ourselves to the more realistic *internal* search strategies [BFFS02, BFST03], in which searchers are limited to move along the edges of the graph, that is, restricted to satisfy the *internality constraint*.

Contrary to classical variants that are somehow equivalent since the corresponding search numbers differ by at most 2, exclusive graph searching behaves differently from previous graph searching models. These differences are mainly due to the combination of the two restrictions introduced in exclusive search: two searchers cannot occupy the same node (exclusivity) and a searcher cannot "jump" (internality). Therefore, intuitively, the difficulty occurs when a searcher has to go from one node u to a far away node v, and all paths from u to v contain an already occupied node. Figure 3.1 illustrates these differences that are formally stated below.

Consider a simple example of a star with central node c and n leaves. In the classical graph searching, one searcher can occupy c, while a second searcher will sequentially clear all leaves, either by jumping from one leaf to another, or by sliding from one leaf to another, and therefore occupying several times the already occupied node c. In exclusive graph searching, such strategies are not allowed. Intuitively, if a searcher r_1 has to cross a node v that is already occupied by another searcher r_2, the latter should step aside for letting r_1 pass. However, r_2 may occupy v to preserve the graph from recontamination, and moving away from v could lead to recontaminate the whole graph. To avoid this, it may be necessary to use extra searchers (compared to the classical graph searching) that will guard several neighbors of v to prevent from recontamination when r_2 gives way to r_1. It follows that, as opposed to all classical search numbers, which differ by at most some constant multiplicative factor, the exclusive search number may be arbitrary large compared to the mixed-search number, even in trees. For instance, it is easy to check that $xs(S_n) = n - 2$ for any n-node star S_n, $n \geq 3$. More generally,

[c-BBN13] **Fact 1** *For any tree T with maximum degree $\Delta \geq 2$, $xs(T) > \Delta - 2$.*

This result shows an exponential increase in the number of searchers used to clear a graph since the mixed-search number of n-node trees is at most $O(\log n)$ [Par78b].

We now turn our attention to the monotonicity property. Indeed, another important difference of exclusive search compared to classical graph searching is that it is not monotone. As explained in the example of a star, when a searcher needs to cross another one, letting the former searcher pass may lead to recontaminate some edges. In spite of that, the goal of the winning strategy is to prevent an "uncontrolled" recontamination (e.g., see tree T in Figure 3.1).

Fact 2 *Exclusive graph searching is not monotone, even in trees or in 2-star-like graphs*[3]. *[c-BBN13, s-MNP14]*

Last, but not least, contrary to classical graph searching, exclusive graph searching is not closed under minor. Indeed, even taking a subgraph can decrease the connectivity which, surprisingly, may not help the searchers (due to the exclusivity constraint). That is, there exist a graph G and a subgraph H of G such that $xs(H) > xs(G)$ (cf. Figure 3.1). Nevertheless, exclusive-search is closed under subgraph in trees:

Lemma 11 *For any tree T and any subtree T' of T, $xs(T') \leq xs(T)$.* *[c-BBN13]*

Contrary to classical graph searching, the proof of this result is not trivial because of the exclusivity property. To prove it, we have to transform an exclusive strategy $\mathcal{S}$ for T into a strategy $\mathcal{S}'$ for T' using the same number of searchers, and without violating the exclusivity property. The fact that $\mathcal{S}$ may be not monotone (i.e., some recontamination may occur during $\mathcal{S}$) makes the proof technical, because one has to "control" the recontamination of T' in $\mathcal{S}'$.

On the positive side, we show that,

Theorem 52 *For any graph G with maximum degree Δ,* *[c-BBN13]*

$$\mathtt{s}(G) \leq xs(G) \leq (\Delta - 1)(\mathtt{s}(G) + 1).$$

To prove it, we consider a node strategy $\mathcal{S}$ for G using $\mathtt{s}(G)$ searchers. To build an exclusive strategy for G, we mimic $\mathcal{S}$ using a team of $\Delta - 1$ searchers to "simulate" each searcher in $\mathcal{S}$. By Fact 1, this upper bound is asymptotically tight.

The above theorem let us some hope to obtain approximation algorithms for the pathwidth of graphs. Unfortunately, we proved that

Theorem 53 *The problem of computing the exclusive search number is NP-hard in the class of cubic planar graphs.* *[s-MNP14]*

[3] A connected graph $G = (V, E)$ is a *star-like graph* if V can be covered by cliques $C_0, C_1, \cdots, C_r$ such that, for any $i, j \leq r$ with $i \neq j$, $C_i \cap C_j \subseteq C_0$. Said differently, a graph is a star-like graph if it is chordal and its clique-tree is a star. A graph is *k-star-like* if $c_i = |C_i \setminus C_0| \leq k$ for any $1 \leq i \leq r$.

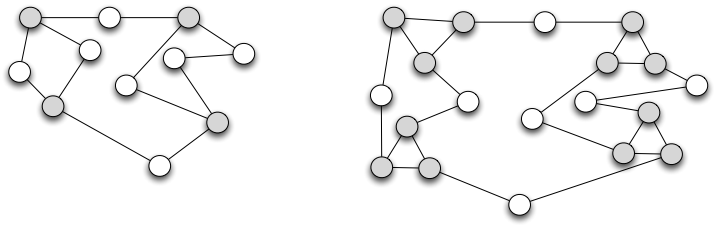


Figure 3.2: Planar cubic graph G where degree-3 nodes (in grey) induce an independent set (left) and Graph $G^{\triangle}$ where nodes with degree 3 of G are "replaced" by triangles (right).

Intuitively, exclusive search differs from mixed search because searchers can only slide and therefore, because of the exclusivity property, the searchers have to avoid to met other searchers at the same node. The reduction consists in proving that, for any planar graph G with maximum degree at most 3 (*cubic*) and such that no two nodes with degree three are adjacent, $s(G) = xs(G^{\triangle})$ where $G^{\triangle}$ is obtained by replacing any node with degree three by a triangle (see Fig. 3.2). Intuitively, the triangles allow the searchers to bypass each other.

While computing the exclusive search number is NP-hard, designing a polynomial approximation algorithm for this parameter would imply a polynomial-time approximation of the pathwidth in bounded degree graphs.

Naturally, we turn our attention toward particular graph classes. It is very surprising that computational complexities of monotone Exclusive Graph Searching
[s-MNP14] and Pathwidth cannot be compared. Indeed, we show that monotone Exclusive Graph Searching is NP-complete in split graphs where Pathwidth is known to be solvable in polynomial time [Gus93]. On the other hand, we prove that monotone Exclusive Graph Searching is in P in a subclass of star-like graphs where Pathwidth is known to be NP-hard [Gus93]. Let us focus now in trees:

[c-BBN13] **Theorem 54** *The problem of computing the exclusive search number $xs(T)$, and corresponding exclusive strategy, is polynomially computable in class of trees T.*

The above theorem is proved by designing a dynamic programming algorithm based on the following characterization that is similar to the one of Parsons in the case of classical graph searching [Par78b]. Given a node v in a tree T, a connected component of $T \setminus \{v\}$ is called a *branch* at v. Our characterization establishes a relationship between the exclusive-search number of T and the exclusive-search number of some of the branches adjacent to any node in T. More precisely, we prove that:

Theorem 55 *Let $k \geq 1$. For any tree T, $xs(T) \leq k$ if and only if, for any node v,*
[c-BBN13] *the following three properties hold:*

1. *v has degree at most $k+1$;*
2. *for any branch B at v, $xs(B) \leq k$;*

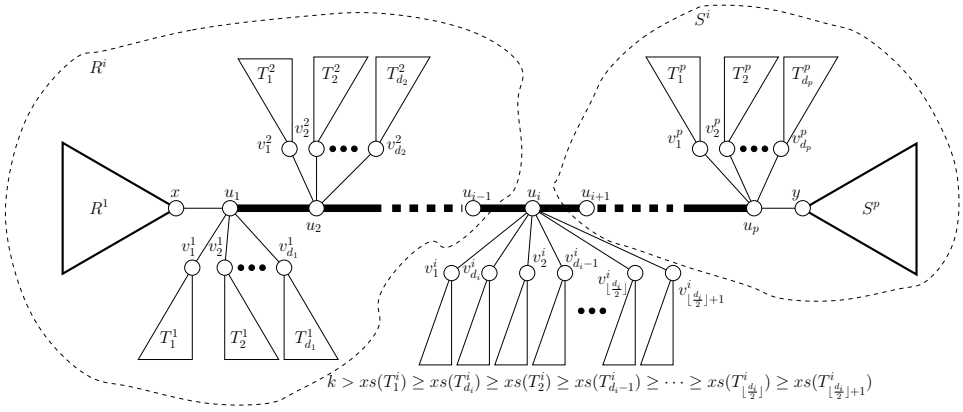


Figure 3.3: A tree T with avenue $A = (u_1, \cdots, u_p)$. For any subtree X of $T \setminus A$, $xs(X) < k$.

3. for any even $i > 1$, at most i branches B at v have $xs(B) \geq k - i/2 + 1$.

The fact that the first property is necessary directly follows from Fact 1. The second property is necessary by Lemma 11. For proving that the third property is necessary, we first have to prove that, for any tree T, any branch B of T, and any exclusive strategy for T, there is a step of the strategy where at least $xs(B)$ searchers are occupying the nodes of B. While such a result is trivial in classical graph searching, it is not the case anymore subject to exclusivity and internality properties. In particular, in classical graph searching, the result is true for any subtree (not necessarily a branch) while it is not the case for the exclusive variant (see [c-BBN13]). Indeed, let us consider a sub-tree T' of tree T. If T' is given independently of T, the movements of searchers are more constrained because the searchers have less "space" in T'. On the contrary, when T' is inside the tree T, the searchers can use the "extra space" provides by T to clear T'. Therefore, we have to prove that, for any tree T, if there exist $v \in V(T)$ and an even integer $i > 1$ such that there is a set $B = \{T_j : xs(T_j) \geq k - i/2 + 1\}$ of branches at v and $|B| > i$, then $xs(T) > k$.

On the other hand, we show that any tree satisfying the conditions of above theorem can be decomposed in a particular way, depending on k (see Fig. 3.3). Following [MHG+88], we prove that there is a unique path $A = (u_1, \cdots, u_p)$ in T called *avenue* such that $p \geq 1$ and, for any component (subtree) T' of $T \setminus A$, there is an exclusive strategy that clears T' using $< k$ searchers, i.e., $xs(T') < k$ (see bold line in Fig. 3.3). Next, we describe an exclusive search strategy using at most k searchers, that clears any tree decomposed in such a way. The strategy consists in clearing the subtrees of $T \setminus A$, starting with the subtrees that are adjacent to u_1, then the ones adjacent to u_2 and so on, finishing in u_p. To clear a subtree T' of $T \setminus A$, we proceed in a recursive way. That is, we recursively use our algorithm on T' using $k' < k$ searchers. The first difficulty is to ensure that no subtrees that have been cleared are recontaminated. For this purpose, when clearing T', the remaining $k - k'$ searchers that are not needed to clear it are used to prevent

recontamination. The second difficulty is to ensure exclusivity: while these $k - k'$ searchers are protecting from recontamination, k' searchers should be able to enter T' to clear it.

To conclude this section, let us mention that the exclusivity constraint may be generalized as follows. Consider a graph with weighted nodes. The weight k of node v means that v can be occupied by at most k searchers simultaneously. Clearly, all nodes having weight 1 corresponds to the exclusive case. On the other hand, if all nodes have weight 2, then the number of searchers needed to clear the graph is the classical mixed search number. It would be interesting to study the variant in which a subset of nodes have weight 1 while others have weight 2.

Finally, since the complexities of Pathwidth and monotone Exclusive Graph Searching cannot be compared and since they are equivalent in bounded degree graphs, it would be very interesting whether there would exist a graph class in which one variant is polynomially computable while the other one is NP-hard.

3.3.2 Perpetual graph searching in CORDA model

In this section, we present our study of exclusive graph searching in the CORDA model and in various network topologies. When the considered network is a ring, our algorithms are more general since they also address other coordination problems such as perpetual exploration and gathering.

CORDA model

In the field of searcher-based computing systems, we consider $k \geq 1$ searchers placed on the nodes of an input graph. searchers are equipped with visibility sensors and motion actuators, and operate in *Look-Compute-Move* cycles in order to achieve a common task (see [FPS12]).

The Look-Compute-Move model considers that in each cycle a searcher takes a snapshot of the current global configuration (Look), then, based on the perceived configuration, takes a decision (deterministically) to stay idle or to move to one of its adjacent nodes (Compute), and in the latter case it moves to this neighbor (Move). Cycles are performed asynchronously, i.e., the time between Look, Compute, and Move operations is finite but unbounded, and it is decided by the adversary for each searcher. Hence, searchers that cannot communicate may move based on outdated perceptions. From the practical viewpoint, the Look-Compute-Move model faithfully describes the behavior of some real searchers.

In the continuous plane, this model is referred in the literature also as the *CORDA* model [Pre07]. The inaccuracy of the sensors used by searchers to scan the surrounding environment motivates its discretization. Moreover, searchers can model software agents moving on a computer network. Various problems have been studied in this setting and several algorithms have been proposed for particular topologies such as lines, rings, trees and grids.

We consider a minimalist variant of the Look-Compute-Move model which has

very weak hypothesis. Neither nodes nor edges of the graph are labeled and no local memory is available on nodes. Searchers are *anonymous*, *asynchronous*, *uniform* (i.e. they all execute the same algorithm), *oblivious* (memoryless) and have no common sense of orientation. Apart for the gathering problem, guided by physical constraints, the searchers may also satisfy the *exclusivity property*, according to which at most a node can be occupied by at most one searcher [BBMR08a]. In contrast to the CORDA model in the continuous plane, we assume that moves are instantaneous, and hence any searcher performing a Look operation sees all other searchers at nodes and not on edges. Note that, in a discrete asynchronous environment this does not constitute a limitation to the model. In fact, an algorithm cannot take advantages from seeing searchers on the edges as the adversary can decide to perform the Look operations only when the searchers are on the nodes. On the other hand, if an algorithm takes advantage from the assumption that the searchers always occupy nodes, the same algorithm can be applied by adding the rule that if a searcher sees another searcher on an edge, it just don't move (i.e. it waits until all the searchers occupy only nodes). In the following, we denote such model as the *discrete CORDA model*.

Related work.

The discrete CORDA model received a lot of attention in the recent years. Most of the proposed algorithms consider that the starting configuration is *exclusive*, i.e., any node is occupied by at most one searcher. Moreover, in case of symmetric topologies (e.g., paths, rings and grids), the starting configuration is sometimes assumed to be *rigid*, i.e., asymmetric and aperiodic w.r.t. the positions of the searchers.

In the following, we review the literature concerning the CORDA model on various graph topologies. Two main problems have been considered: exploration (e.g., see [CFI$^+$08, AGP$^+$11]) and gathering (e.g., see [CGP09, BCG$^+$10, Pel12]). Main results are summarized in Table 3.1.

In the problem of *graph exploration with terminaison* problem, a team of mobile agents is spread in a graph and each node must be visited at least once by an agent, then agents must be able to decide that all vertices have been visited at least once [FIPS07, DPT09, FIPS10, CFMS10, FIPS11, FIPS13]. On the other hand, the *exclusive perpetual graph exploration* requires that each searcher visits each node of the graph infinitely many times. Moreover, it adds the exclusivity constraint [BBMR08a, BBMR08b, BMPBT10, BMPBT11] . In [BMPBT10], first results on n-node rings are given. In detail, the paper gives algorithms for $k = 3$ and $n \geq 10$, for $k = n-5$ (if $n \bmod k \neq 0$), and shows that the problem is infeasible for $k = 3$ and $n \leq 9$, and for some symmetric configurations where $k \geq n - 4$.

The gathering problem consists in moving all the searchers in the same node and remain there. In [DDKN12] and [DDN13a], a full characterization of the gathering on grid and tree topologies, respectively, without any multiplicity detection is given. On rings, it has been proven that the gathering is unsolvable if the searchers

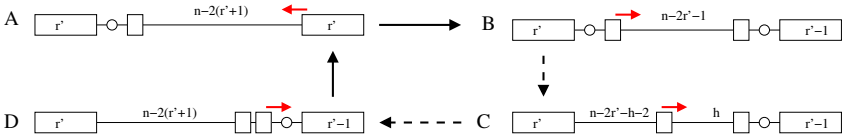
Table 3.1: Non exhaustive survey on coordination problems in CORDA model

	Paths	Trees	Rings	Grids	Arbitrary graphs
Exploration with stop	[FIPS11]	[FIPS10]	[FIPS07] + multiplicity detection		[CFMS10] + local edge labeling
Exclusive Perpetual Exploration			[BMPBT10] [c-DDN+13b] [j-DDN+14b]	[BBMR08a] + orientation [BMPBT11] no orientation	
Gathering +global mult. detection +local mult. detection			[DDN12] [IIKO10, KLOT11] [KLOT12] [c-DDN+13b] [j-DDN+14b]		[DN13]
Exclusive Perpetual Graph Searching	[c-BBN12] [c-BBN13]	[c-BBN12] [c-BBN13]	[c-DDN+13b] [j-DDN+14b] [c-DNN14]		

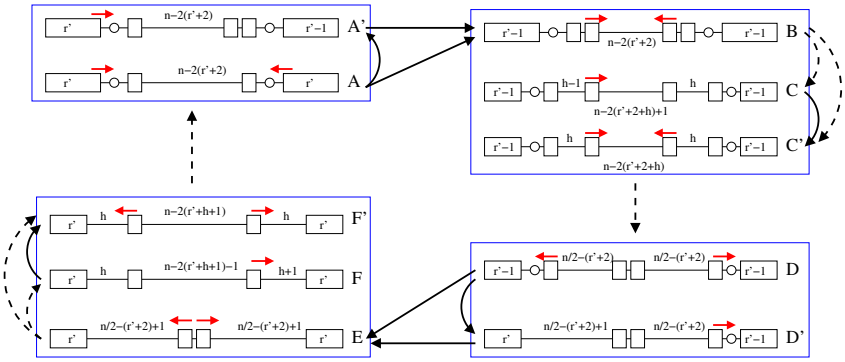
are not empowered by the so-called *multiplicity detection* capability [KMP08], either in its *global* or *local* version. In the former type, a searcher is able to perceive whether any node of the graph is occupied by a single searcher or more than one (i.e., a *multiplicity* occurs). In the latter type, a searcher is able to perceive the multiplicity only if it is part of it. Using the global multiplicity detection capability, in [KMP08], some impossibility results have been proven. Then, several algorithms have been proposed for different kinds of exclusive initial configurations in [DDN11, KKN10, KMP08]. These papers left open some cases which have been closed in [DDN12] where a unified strategy for all the gatherable configurations has been provided. With local multiplicity detection capability, an algorithm starting from rigid configurations where the number of searchers k is strictly smaller than $\lfloor \frac{n}{2} \rfloor$ has been designed in [IIKO10]. In [KLOT11], the case where k is odd and strictly smaller than $n-3$ has been solved. In [KLOT12], the authors provide an algorithm for the case where n is odd, k is even, and $10 \leq k \leq n-5$. Papers [KLOT11] and [KLOT12] do not assume that the initial configuration is rigid. The remaining cases with local multiplicity detection are left open and a the design of a unified algorithm for all the cases is still not known.

Exclusive graph searching in CORDA model in trees

We study (perpetual) exclusive graph searching in discrete CORDA in trees. We denote by $\texttt{fs}(G)$ the set of integers k such that k searchers can clear a graph G in this way. That is, $k \in \texttt{fs}(G)$ if there is a distributed protocol allowing k searchers to clear G in discrete CORDA whatever be the starting exclusive positions of the k searchers in G. By definition, if $k \in \texttt{fs}(G)$ then $xs(G) \leq k \leq |V(G)|$.



(a) Perpetual strategy for $2r'+1$ agents in a n-node path, $n > 2r'+3$.



(b) Perpetual strategy for $2(r'+1)$ agents in a n-node path, with n even and $n > 2(r'+2)$.

Figure 3.4: An empty square represents one agent, an empty circle represents one node without agent, a rectangle with label ℓ represents ℓ searchers on consecutive vertices and a line with label h represents h consecutive vertices without searchers. In configurations C and C', $h \geq 1$, and in Configurations F and F', $h \geq 2$.

As a warm-up, we fully characterize $\texttt{fs}(P_n)$ for any n-node path P_n. The distributed protocols are described in Figures 3.4(a) and 3.4(b). To give a flavor of the difficulties that arise in this model, we give the following result.

Assertion 3 *Let $P = (v_1, \cdots, v_{2p+1})$ be an n-node path for some odd $2p+1 = n \geq 3$ and let $2 \leq 2r < n$. $2r$ searchers cannot perpetually clear P.*

Indeed, if the searchers are initially placed at $\{v_1, \cdots, v_r, v_{n-r+1}, \cdots, v_n\}$, then, v_{p+1} can never be occupied by a searcher. Indeed, the adversary can schedule the moves such that, at any step when v_i is occupied for $i \leq p$, then v_{n-i+1} is also occupied. In particular, if a searcher occupying v_p moves to v_{p+1}, then the searcher occupying v_{p+2} will do the same and two searchers will occupy simultaneously v_{p+1}, a contradiction. So, $2r \notin ps(P)$ for any $2r < n$ in any n-node path P with n odd.

Theorem 56 *Let P_n be an anonymous n-node path.* *[c-BBN12, c-BBN13]*

- $\texttt{fs}(P_1) = \{1\}$ *and* $\texttt{fs}(P_2) = \{1, 2\}$*;*
- $\texttt{fs}(P_n) = \{3, \cdots, n\}$ *for any $n \geq 3$ even;*
- $\texttt{fs}(P_n) = \{3, 5, .., 2k+1, .., n\}$*, $n \geq 3$ odd.*

Our main result consists in a characterization of $\texttt{fs}(T)$, for any asymmetric tree T. We prove that, if T has no symmetries (i.e., has no non-trivial automorphisms), then $\texttt{fs}(T) = \{xs(T), \cdots, |V(T)|\}$. In words, in the case of asymmetric trees, we surprisingly prove that the minimum number of searchers needed in discrete CORDA does not increase in comparison with the one needed in a powerful centralized setting. This result is based on the explicit design of a distributed protocol enabling perpetual graph searching by k searchers, for any $k \geq xs(T)$. This protocol widely uses the structure of the strategy given for centralized setting described in Theorem 55. The main idea is to adapt the centralized strategy such that all the obtainable positions of searchers are pairwise distinct and that all the moves are uniquely defined. This classical method being powerful still demands a technically difficult work.

We strongly use the fact that, in any asymmetric tree, each searcher can assign distinct labels to the nodes such that each node of the tree is always given the same label, by all searchers and at any Compute action. Therefore, in discrete CORDA, an anonymous asymmetric tree can be seen as a *uniquely labeled tree* (whose nodes are given distinct identifiers).

[c-BBN12, c-BBN13]

Theorem 57 *For any anonymous asymmetric tree or any uniquely labeled tree T, there exists an algorithm to clear T in the discrete CORDA model, either using $k \geq xs(T)$ searchers, or $k \geq 2$ if T is a labeled line with ≥ 3 nodes.*

When T possesses symmetries (i.e., has non-trivial automorphisms), the computation of $\texttt{fs}(T)$ becomes more complex. Following [FIPS10], two distinct nodes u and v are said to be *isomorphic* if there exists an automorphism[4] of G which carries u to v. We show that then $\texttt{fs}(T)$ depends on the set $\mathcal{S}_T$ of isomorphic nodes separated by a path of even length in T. Our impossibility results are mainly due to the exclusivity property and symmetries (rather than due to the perpetual nature of the problem).

[c-BBN12, c-BBN13]

Lemma 12 *For any tree T,*

$$\texttt{fs}(T) \cap \{i \in \mathbb{N} \ : \ |\mathcal{S}_T| \leq i < |\mathcal{S}_T| + xs(T \setminus \mathcal{S}_T)\} = \emptyset.$$

A set $S \subseteq \mathcal{S}_T$ is *fully-isomorphic* in the tree T, if there is a non-trivial automorphism f of T such that $f(S) = S$, and, for any $u \in S$, the distance between u and $f(u)$ is even.

[c-BBN12, c-BBN13]

Lemma 13 *Let T be a tree and $S \subset V(T)$ be fully-isomorphic in T. Then $|S| \notin \texttt{fs}(T)$.*

A tree T is *very symmetric* if there is an edge $e = \{u, v\} \in E(T)$ such that there is an automorphism from one subtree of $T \setminus e$ to the other, carrying u to v. In that

[4] An automorphism of G is a one-to-one mapping $f : V(G) \to V(G)$ s.t. $\{u, v\} \in E(G)$ if and only if $\{f(u), f(v)\} \in E(G)$.

case, each such a subtree is called an *half* of T. To conclude this section, we extend our results in asymmetric trees and in paths to a larger class of trees. Our results are constructive since we design the corresponding algorithms.

Theorem 58 *Let T be any anonymous tree. Either nodes of $T_0 = T \setminus S_T$ can be assigned distinct labels, or T_0 is very symmetric with half T'_0. Let $x = |\mathcal{S}_T| + xs(T_0)$ in the former case, and $x = |\mathcal{S}_T| + 2 \cdot xs(T'_0)$ otherwise. There is an algorithm in the discrete CORDA model that allows to clear T perpetually using k searchers, for any k, $x \leq k \leq |V(T)|$.* *[c-BBN12, c-BBN13]*

Coordinating searchers in Rings for various tasks in CORDA model

Here, we propose a unified approach to solve the exclusive perpetual exploration, the exclusive perpetual graph searching, and the gathering problems on rings. The relevance of the ring topology is motivated by its completely symmetrical structure. It means that algorithms for rings are more difficult to be devised as they cannot exploit any topological structure, as all nodes look the same. In particular, periodicity and symmetry arguments must be carefully handled. In fact, our algorithms are only based on searchers' disposal and not on topology.

Note that, contrary to our study on trees, here we are interested in the initial configurations of the searchers for which the tasks are feasible (that is, we do not look for the number of searchers required to accomplish the task whatever be the initial configuration).

For presenting our positive results, we need the following notations. We consider a ring with $n \geq 3$ nodes $\{v_0, \cdots, v_{n-1}\}$, where v_i is connected to $v_{i+1 \bmod n}$ for any $0 \leq i < n$. Moreover, let $k \geq 1$ searchers occupy k distinct nodes of the ring. Given a configuration $\mathcal{C}$, and for any $i \leq n$, let $\mathcal{S}_i = (r_0^i, \cdots, r_{n-1}^i) \in \{0,1\}^n$ be the sequence such that $r_j^i = 0$ if $v_{i+j \bmod n}$ is occupied in $\mathcal{C}$ and $r_j^i = 1$ otherwise, $1 \leq j \leq n$. Intuitively, $\mathcal{S}_i$ represents the positions of searchers, starting at v_i. A *supermin* of $\mathcal{C}$ is any representation of $\mathcal{C}$ that is minimum in the lexicographical order (among all representations of $\mathcal{C}$, i.e., starting from any node and either clockwise or anti-clockwise).

An exclusive configuration is called *symmetric* if the ring admits a geometrical *axis of symmetry* with respect to the positions of the searchers. An exclusive configuration is called *periodic* if it is invariable under non-trivial (i.e., non-complete) rotations. A configuration which is aperiodic and asymmetric is called *rigid*. A key property that allows us to deal with aperiodic configurations is that:

Lemma 14 *[DDN12] Given a configuration $\mathcal{C}$,*

- *$\mathcal{C}$ has a unique supermin representation if and only if $\mathcal{C}$ is either rigid or it admits only one axis of symmetry passing through the supermin;*
- *$\mathcal{C}$ has two supermin representations iff $\mathcal{C}$ is either aperiodic and symmetric with the axis not passing through any supermin or it is periodic with period $\frac{n}{2}$;*

- *$\mathcal{C}$ has at least 3 supermin representations iff $\mathcal{C}$ is periodic, with period at most $\frac{n}{3}$.*

Impossibility results. First, we gave some impossibility results for the exclusive perpetual graph searching in rings. The proofs are complicated cases analysis where we explicitly design the behavior of an adversary (the asynchronous scheduler of the searchers) in all cases, ensuring that the ring cannot be cleared whatever be the algorithm. Note these impossibility results hold even if a particular initial configuration is specified. More precisely, we proved that:

[c-DDN+13b, j-DDN+14b]

Theorem 59 *For any initial configuration $\mathcal{C}$, there is no algorithm, in the CORDA model, that solves the exclusive perpetual clearing problem in a n-node ring using k searchers starting from $\mathcal{C}$, if*

- *$n > 2$ and $k \leq 2$ or $k = n - 2$ or $k = n - 1$, or*
- *$n > 3$ and $k \leq 3$, or*
- *$2 \leq k < n \leq 9$.*

It is easy to see that graph searching is also not feasible for any k even starting from any configuration with an axis of symmetry passing through an empty node. Indeed, any synchronous execution of any algorithm would either cause a collision in the node lying on the axis or does not allow to search the edges incident to such node.

For the gathering problem, it was known that the problem is not feasible for $k < 3$ or $k \geq n - 4$ or starting from a symmetric configuration with one axis of symmetry passing through two edges.

In the following, a configuration is *allowed* if it is rigid or if it is aperiodic and its symmetry is not the ones excluded by above impossibility results. That is, for graph searching, the symmetric allowed configurations are all aperiodic ones with k odd and those with k even where the axis does not pass through an empty node, provided that $3 < k < n - 2$ and $n > 9$. For gathering, the symmetric allowed configurations are all aperiodic ones with the axis of symmetry not passing through two edges and $3 \leq k < n - 4$.

Algorithms for graph searching, perpetual exploration and gathering.

We now briefly present the algorithms we design to solve the above-mentioned problems. Our algorithms consist of two phases. The first phase is common to all problems and allows $k > 2$ searchers to achieve a particular exclusive configuration in an n-node ring, $k < n - 2$. The second phase depends on the task: we give an algorithms for solving the exclusive perpetual graph searching problem that, in addition, solves the perpetual exploration problem, then another algorithms is designed for solving the gathering problem. In a first work, we dealt only with rigid configurations [c-DDN$^+$13b, j-DDN$^+$14b], then we managed to extent this work to aperiodic configurations [c-DNN14]. Our results can be summarized in the following two theorems.

Theorem 60 *There is an algorithm solves both the exclusive perpetual clearing and exploration problems using k robots in any n-node ring, starting from any exclusive and allowed configuration, $n > 9$ and $4 < k < n-3$ (but for $(n,k) \in \{(10,5);(10,6)\}$ when the configuration is symmetric).* *[c-DDN$^+$13b, j-DDN$^+$14b, c-DNN14]*

Theorem 61 *There exists an algorithm performing the gathering of $k > 2$ robots on rings of $n > k+2$ nodes starting from any exclusive and allowed configuration, when the robots are empowered with the local multiplicity detection.* *[c-DDN$^+$13b, j-DDN$^+$14b, c-DNN14]*

The core of the algorithms in [c-DDN$^+$13b, j-DDN$^+$14b] is a procedure, called Asym, allowing to achieve some particular configuration. Let $4 \leq k < n-2$ and let $\mathcal{C}^a = (0^{k-1}, 1, 0, 1^{n-k-1})$ be the configuration with $k-1$ consecutive searchers, one empty node and one searcher.

Lemma 15 *Let $4 \leq k < n-2$ searchers standing in an n-node ring and forming a rigid exclusive configuration, Algorithm* Asym *eventually terminates achieving configuration $\mathcal{C}^a$ and all intermediate configurations obtained are exclusive and rigid.* *[c-DDN$^+$13b, j-DDN$^+$14b]*

Basically, Algorithm Asym ensures that, from any rigid exclusive configuration, one searcher, that can be uniquely distinguished, moves to an unoccupied neighbor, achieving another rigid configuration while strictly decreasing the supermin. Then, in [c-DNN14], we designed Algorithm Align, generalizing Asym, that handles all allowed configurations (not only rigid). Difficulties are multiple.

First, in allowed symmetric configurations, we cannot ensure that a unique searcher will move. In such a case, the algorithm may allow a searcher r to move, while r is reflected by the axis of symmetry to another searcher r'. Since r and r' are indistinguishable and execute the same algorithm, r' should perform the same (symmetric) move. Hence, r may move while the corresponding move of r' is postponed because of asynchronicity (i.e. r' has performed the Look phase but not yet the Move phase). The configuration reached after the move of r has a potential so-called *pending* move (the one of r' that will be executed eventually). To deal with this problem, our algorithm ensures that reached configurations that might have a pending move are asymmetric, distinguishable and the pending move is unique. Therefore, in such a case, our algorithm forces the pending move. That is, contrary to [c-DDN$^+$13b, j-DDN$^+$14b] where Algorithm Asym ensures to only go through rigid configurations, the subtlety is here consists in possibly going from an asymmetric configuration to a symmetric one. To distinguish such a configuration, we define the notion of adjacent configurations. An asymmetric configuration $\mathcal{C}$ is *adjacent* to a symmetric configuration $\mathcal{C}'$ with respect to a move M (one searcher moving to one of its unoccupied adjacent neighbor) if $\mathcal{C}$ can be obtained from $\mathcal{C}'$ by applying M to only one of the searchers permitted to move by M. Another difficulty is to ensure that all met configurations are allowed for the considered problem.

Further work

Continuing our study of what is possible in the discrete CORDA model using robots with weak capabilities, we currently investigate the exclusive graph searching prob-

lem in grids with Euripides Markou and Christos papageorgakis. It would be interesting to address this problem in general graphs.

Other problems are of interest: perpetual exploration seems surprisingly difficult in trees, we are working on it with Gianlorenzo d'Angelo, Lélia Blin and Janna Burman. Motion planning of robots that must go from several initial positions to personalized destinations without colliding is another practical problem we address with Gianlorenzo d'Angelo and Xavier Defago. For addressing this problem, we first dealt with exchange of information between robots.

[s-DDN14a]

Last but not least, I would be interested by considering this class of problem in more realistic setting. For instance, real robots are not so stupid as the extreme cases we are considering. Therefore, using the techniques we developed above but with stronger robots should lead to efficient practical solutions.

Chapter 4

Complexity of Locality: Routing and Graph Properties

Content

This chapter deals with routing, broadcasting and connectivity[1] in various environments. As communication networks are concerned, the different constraints that appear in wireless networks, Internet networks, Peer-to-peer networks, etc., imply various difficulties and techniques to address these problems. The common point of all the considered environments is that the current networks are becoming always larger and, therefore, distributed and even local algorithms are required. The difference between distributed and local is mainly related to the time that is allowed to the nodes to exchange messages. Indeed, the less exchanges are allowed and the less the information can spread through the network, and the more the knowledge the nodes have about the network is partial. Most of our studies focus on distributed setting where network's nodes have knowledge with various levels of locality. On the other hand, in a distributed setting, it is helpful to have more global knowledge about the structural properties of the considered network. We both use such a structural knowledge for efficient routing distributed algorithms and provide distributed algorithms to compute such properties.

Section 4.1 consists of a melting-pot on several routing problems we studied in wireless and peer-to-peer networks. Since I am far to be an expert on these kind of networks, I focus on the models and techniques we used in these specific problems. First, the personalized broadcasting and, equivalently, the gathering (many sources, one sink) problems are considered in grid radio networks in presence of interferences (Section 4.1.1). Here, the optimization problems consists in finding paths and schedules (sequences of matchings, induced matchings, etc.) to route messages as fast as possible and avoiding interference [c-BNRR09, s-BLN$^+$13]. Second, we study the problem of routing packets between undifferentiated sources and sinks in a network (Section 4.1.2). We provide a purely local algorithm that transmits packets hop by hop in the network and study its behavior [c-CHN$^+$10]. Finally, motivated by video streaming problems, we investigate broadcasting (one to all) in Peer-to-peer networks (Section 4.1.3). Here, the objective is to maintain an efficient broadcasting tree, both minimizing the delay and maximizing the bandwidth, in a purely local model [c-GMNP13].

[1] Contrary to previous chapters, this chapter does not deal at all with pursuit-evasion games.

Section 4.2 focuses on *compact routing* motivated by the increase of Internet network's size. The main question is the design of routing tables that are zone of local memory allowing to guide messages routing. We first survey the literature on compact routing in various graph classes. Then, we present our results obtained in the case of graphs with bounded chordality or hyperbolicity [c-NRS09, j-NRS12, c-KLNS12, j-KLNS14]. These results rely on structural characterizations presented in previous chapters. We also proposed a randomized fault-tolerant routing algorithm. We prove it achieves good performances in grids and expanders [c-HIKN10].

Finally, Section 4.3 presents a new model of distributed computing that allows nodes of a graph to gather only local knowledge [c-BMN$^+$11, c-BKN$^+$12, j-BKM$^+$14]. We investigate the computation of various graph structural properties in this model, providing several impossibility results such as algorithms that actually compute some non-trivial properties. This study mainly aims at deciding which connectivity properties (connectivity, spanning trees, BFS trees, etc.) may be computed with local knowledge.

4.1 Routing in various telecommunication networks

This eclectic section presents three different problems, related to routing, that we studied in wireless and peer-to-peer networks. The first one is an optimization problem that consists in finding routes and schedules of messages in a centralized setting [c-BNRR09, s-BLN+13]. The next two works are the design of purely local algorithms and the study of their convergence and performances: one for multicast [c-CHN+10] and the other one for maintaing an efficient diffusion tree in dynamic networks [c-GMNP13]. Since the kind of networks studied in this section is a bit far from the main topic of this thesis, the related work is minimalist and the interested reader can see the references therein for more information.

4.1.1 Data Gathering in Wireless Grids with Interferences

We first study a problem that was motivated by designing efficient strategies to provide internet access using wireless devices [BBS05]. Typically, several houses in a village need access to a gateway (for example a satellite antenna) to transmit and receive data over the Internet. To reduce the cost of the transceivers, multi-hop wireless relay routing is used. We formulate this problem as *gathering* information in a Base Station of a wireless multi-hop network when interferences constraints are present. This problem is also known as *data collection* and is particularly important in sensor networks and access networks.

Model of communication

We adopt the network model considered in [BGK+06, BGR10, BKK+10, FFM04, RS08]. The network is represented by a node-weighted symmetric digraph $G = (V, E)$. More specifically, each node in V represents a *device* (sensor, station, ...) that can transmit and receive data. The network is assumed to be synchronous. We suppose that each device is equipped with an half duplex interface: a node cannot both receive and transmit during a step. This models the near-far effect of antennas: when one is transmitting, it's own power prevents any other signal to be properly received. There is a special node called the *Base Station (BS)*, which is the final destination of all data possessed by the nodes of the network. Each node may have any number of pieces of information, or *messages*, to transmit, including none. There is an arc from u to v if u can transmit a message to v. We suppose that the digraph is symmetric; so if u can transmit a message to v, then v can also transmit a message to u. Therefore G represents the graph of possible communications. *Calls* (transmissions) are directed: a call (s, r) is defined as the transmission of one message from the node s, the *sender*, to node r, the *receiver*, where $(s, r) \in E$.

We use a binary asymmetric model of interference based on the distance in the communication graph. Let $d(u, v)$ denote the distance between u and v in G and d_I be an nonnegative integer. We assume that when a node u transmits, all nodes v such that $d(u, v) \leq d_I$ are subject to the interference from u's transmission. We assume that all nodes of G have the same interference range d_I. Two calls (s, r)

and (s', r') do not interfere if and only if $d(s, r') > d_I$ and $d(s', r) > d_I$. Otherwise calls interfere (or there is a collision). These hypotheses are strong and under the assumption of a centralized view. Moreover the binary interference model is a simplified version of the reality, where the Signal-to-Noise-and-Interferences Ratio has to be above a given threshold for a transmission to be successful. However, our results will give upper bounds on the maximum possible number of users in the network.

Following [FFM04, GR09, RS07, RS08, ZTG05] we assume that no buffering is done at intermediate nodes and each node forwards a message as soon as it receives it. One of the rationales behind this assumption is that it frees intermediate nodes from the need to maintain costly state information and message storage.

Gathering and Personalized broadcasting

We focus on grids as they model well both access networks and also random networks [KLNP09]. The *gathering problem* consists in computing a multi-hop schedule for each message to reach the BS under the constraint that during any step any two calls do not interfere within the interference range d_I. Actually, we will describe the gathering schedule by illustrating the schedule for the equivalent *personalized broadcasting problem* since this formulation allows us to use a simpler notation and simplify the proofs. In this problem, the base station BS has initially a set of *personalized* messages and they must be sent to their destinations, i.e., each message has a personalized destination in V, and possibly several messages may have the same destination. The problem is to find a multi-hop schedule for each message to reach its corresponding destination node under the same constraints as the gathering problem. The completion time or *makespan* of the schedule is the number of steps used for all messages to reach their destination and the problem aims at computing a schedule with minimum makespan.

Gathering problems have received much recent attention. The papers most closely related to our work are [BGP$^+$11, BGR10, FFM04, GR09, RS07]. [FFM04] firstly introduced the data gathering problem in a model for sensor networks similar to the one adopted in this paper. It deals with $d_I = 0$ and gives optimal gathering schedules for trees. Under the same hypothesis, an optimal efficient algorithm for general networks is presented in [GR09] in the case each node has exactly one message to deliver. In [BGR10] (resp [BGP$^+$11]) optimal gathering algorithms for tree networks in the same model considered in the present paper, are given when $d_I = 1$ (resp.,$d_I \geq 2$). In [BGP$^+$11] it is also shown that the Gathering Problem is NP-complete if the process must be performed along the edges of a *routing tree* for $d_I \geq 2$ (otherwise the complexity is not known). Furthermore, for $d_I \geq 1$ a $(1 + \frac{2}{d_I})$ factor approximation algorithm is given for general networks. The case of *open-grid* where BS stands at a corner and no messages have destinations in the first row or first column, called axis in the following, is considered in [RS07], where a 1.5-approximation algorithm is presented. Other related results can be found in [BCY09, BGK$^+$06, BP12, BY10](see [BKK$^+$10] for a survey). There are

Interference	Additional hypothesis	Performances	
		without buffering	with buffering
$d_I = 0$		+2-approximation	
	open grid	+1-approximation	
$d_I = 1$	1-open grid	+3-approx.	×1.5-approx.
	2-open grid	×1.5-approximation	
$d_I = 2$	2-open grid	+4-approx.	×2-approx.

Table 4.1: Performances of our gathering and personalized broadcasting algorithms.

articles in which data buffering is allowed at intermediate nodes, achieving a smaller makespan. In [BGK+06], a 4-approximation algorithm is given for any graph. In particular the case of grids is considered in [BP12], but with exactly one message per node.

Contributions

We have proposed algorithms to solve the personalized broadcasting problem (and so the equivalent gathering problem) in a grid with the model described above (synchronous, no buffering, one message transmitted by step and binary asymmetric model interference with a parameter d_I). Initially all messages stand at the base station BS and each message has a particular destination node (possibly several messages may be sent to the same node). Our algorithms compute in linear time (in the number of messages) schedules with no calls interfering, with a makespan differing from the lower bound by a small additive constant. The main results are summarized in Table 4.1.

More precisely, our results vary a little bit depending on the positions of the destinations with respect to the base station. The grid is endowed natural coordinates with the base station as origin, i.e., with coordinates $(0, 0)$. A *corner* of the grid is a node with degree 2. The row and column of the base station are called the *axis* of the grid. A grid with arbitrary base station is said to be an *open-grid* if no destination nodes are on the axes. A grid with arbitrary base station is said to be an 2-*open-grid* if no destination nodes are at distance at most 1 from any axis. Finally, an instance is called *1-open grid* if the base station is in a corner and at least one of the following conditions is satisfied: (1) All messages have destination nodes in the set $\{(x, y) : x \geq 2 \text{ and } y \geq 1\}$; (2) All messages have destination nodes in the set $\{(x, y) : x \geq 1 \text{ and } y \geq 2\}$.

Theorem 62 *There are linear-time (in the number of messages) algorithms that solve the gathering and the personalized broadcasting problems in any grid, achieving an optimal makespan up to an additive constant c where:* *[c-BNRR09, s-BLN+13]*

$c = 2$ *when* $d_I = 0$*;* $c = 1$ *in open-grid when* $d_I = 0$*;*

$c = 3$ *in* 1*-open-grid when* $d_I = 1$*;* $c = 4$ *in* 2*-open-grid when* $d_I = 2$*.*

Our algorithms are based on greedy ordering of the messages (by non-increasing distance to BS) sent along simple (with at most one "turn") shortest paths, which

gives a lower bound on the makespan but may induce collisions. Then, we prove that, by dynamic programming, we can slightly modify the position of the messages (by local exchanges) in this ordering and add some small detour to achieve a valid schedule (without collision) and obtaining a makespan close to the optimal one.

Note that the complexity of computing an optimal schedule is not known in our setting.

4.1.2 Stability of a local and greedy routing algorithm

In this short section, we study the problem of routing packets between undifferentiated sources and sinks in a network modeled by a multigraph [c-CHN+10]. We consider a distributed and local algorithm that transmits packets hop by hop in the network and study its behavior. At each step, a node transmits its queued packets to its neighbors in order to optimize a local gradient. This protocol is greedy since it does not require to record the history about the past actions, and local since nodes only need informations about their neighborhood. We study the protocol *stability region*, i.e., the packet arrival rates such that the number of packets stored at the nodes of the network keeps bounded.

In previous work, Srikant et al.[WS05] studied distributed and local algorithms to transmit packets in a network (see also [TE92]). In this work, packets are injected into the network following a stochastic process that respects a *strict* feasibility constraint, meaning that the number of added packets at each time is always strictly lower than the value of the maximum flow. In case of a single destination, other work has considered processes in which packets are generated by an adversary who wants to make the protocol fail [Tsa97, ABBS01]. In this work, we consider the case of multicast (many-to-many).

Let $G = (V, E)$ be a n-node multigraph modeling the considered network. $\mathcal{S}$ is the set of sources and $\mathcal{D}$ the one of destinations. To each vertex is associated a queue length which represents the number of packets waiting to be transmitted at this node. We represent this queue length by $q_t(v)$ for $v \in V$ and a given time step t. The network is synchronous and at each time step: (1) each source $s \in \mathcal{S}$ injects $in(s)$ packets in its queue, (2) each link can transmit at most 1 packet, and this packet can be lost without any notification, and (3) each sink $d \in \mathcal{D}$ extracts $\min\{out(d), q_t(d)\}$ packets of its queue. All links can transmit at the same time, so we do not consider interference constraints.

We are interested in the total number of stored packets at a given time in the network. We propose and study the very simple following algorithm called Local greedy gradient (LGG): each node u transmits 1 packet through each of its outgoing arcs uv if v as a smaller queue length, as long as u still has packets in its queue. In particular, if u has more than $q_t(u)$ neighbors with smaller queue length, then it chooses to send to its $q_t(u)$ neighbors of smallest queue length.

Our study can be related to the distributed algorithm for the maximum flow problem proposed by Goldberg and Tarjan [GT88]. Let G^* be the multigraph obtained from G by adding a virtual source s^* and a virtual sink d^*, with a link

of infinite capacity between s^* and s for all $s \in \mathcal{S}$, and a link of capacity $out(d)$ between d and d^* for all $d \in \mathcal{D}$. The network G is *unsaturated* if it exists a fractional s^*-d^*-flow Φ in G^* such that $in(s) < \Phi(s^*, s)$ for each source $s \in \mathcal{S}$. We show that,

Theorem 63 *In unsaturated networks, LGG is stable.* *[c-CHN+ 10]*

For this purpose, we show that, for all t, $P_t = \sum_{v \in V} q_t^2(v)$ is bounded by some polynomial function of n, the maximum degree of G and the value of a maximum flow from s^* to d^*. The case of *saturated* networks, i.e., if it exists a fractional s^*-d^*-flow Φ in G^* such that $in(s) \leq \Phi(s^*, s)$ for each source $s \in \mathcal{S}$, is still open.

4.1.3 Maintaining efficient diffusion trees for streaming systems

In this section, we propose and analyze a simple local algorithm to balance a tree [c-GMNP13]. The motivation comes from live distributed streaming systems in which a source diffuses a content to peers via a tree, a node forwarding the data to its children. Such systems are subject to a high churn, peers frequently joining and leaving the system. It is thus crucial to be able to repair the diffusion tree to allow an efficient data distribution. In particular, due to bandwidth limitations, an efficient diffusion tree must ensure that node degrees are bounded. Moreover, to minimize the delay of the streaming, the depth of the diffusion tree must also be controlled.

Context

Trees are inherent structures for data dissemination in general and particularly in peer-to-peer live streaming networks. Indeed, the dissemination of a single piece of information generally defines a tree structure. Even in *unstructured* networks, whose main characteristic is lack of defined structure, many systems look into perpetuating such underlying trees, e.g. the second incarnation of Coolstreaming [LQK+08] or PRIME [MR09]. Unsurprisingly, early efforts into designing peer-to-peer video streaming concentrated on defining tree-based structures for data dissemination. However, such structures suffer two drawbacks. First, they lead to unused bandwidth at the leaves of the tree. To fix this problem, it has been proposed to maintain multiple concurrent trees to tolerate failures, and internal nodes in a tree are leaf nodes in all other trees to optimize bandwidth [CDK+03, VYF06]. Second, these structures are fragile because of the important node churn in live streaming networks [LQK+08]. To ensure churn-resiliency, several approaches have been proposed such as maintaining redundancy within the network structure [PTT09, LXHL11]. However, proposed solutions to maintain balanced trees are generally to periodical rebuild the whole tree from scratch. The analysis of these systems focus on the feasibility, construction time and properties of the established overlay network, see for example [CDK+03, VYF06] and [DFC07] for a theoretical analysis. But this work usually abstracts over the issue of tree maintenance. Generally, in this work, when some elements (nodes or links) of the networks fail, the nodes disconnected

from the root execute the same procedure as for initial connection. To the best of our knowledge, there are no theoretical analysis on the efficiency of tree maintenance in streaming systems, reliability is estimated by simulations or experiments as in [CDK+03].

Balancing a tree

The problem setting is as follows. A single source provides live media to some nodes in the network. This source is the single reliable node of the network, all other peers may be subject to failure. Each node may relay the content to further nodes. Due to limited bandwidth, both source and any other node can provide media to a limited number $k \geq 2$ of nodes. The network is organized into a logical tree, rooted at the source of media. That is, we assume that the overlay is already a spanning tree. However, it may have an arbitrary shape, e.g. be a path or a star (all nodes connected directly to the root). Note that the delay between broadcasting a piece of media by the source and receiving by a peer is given by its distance from the root in the logical tree. Hence our goal is to minimize the tree depth, while following degree constraints. It is easy to see that such a tree T is any *k-balanced* tree, i.e., any k-ary tree[2] such that, for each node $v \in V(T)$, the sizes of the subtrees rooted in the children of v differ by at most one.

In this work (following e.g., [HLP+07, CDPT08, BCH+09]), we assume that nodes can reliably communicate, form connections and detect failures. We assume a *reconnection process*: when a node leaves, its children reattach to its parent. This can be done locally if each node stores the address of its grandfather in the tree. Note that this process is performed independently of the bandwidth constraint, hence after multiple failures, a node may become the parent of many nodes. The case of concurrent failures of father and grandfather can be handled by reattaching to the root of the tree. Other more sophisticated reconnection processes have been proposed, see for example [HLP+07].

This process can leave the tree in a state where either the bandwidth constraints are violated (the degree of a node is larger than k) or the tree depth is not optimal. Thus, we propose a distributed *balancing process*, where based on information about its degree and the subtree sizes of its children, a node may perform a local operation at each turn. We show that this balancing process, starting from any tree, converges to a balanced tree and we evaluate the convergence time.

Distributed model, algorithm and analysis

Nodes are autonomous entities running the same algorithm. Each node v has a local memory where it stores the size n_v of its subtree, the size of the subtrees of its children and the size of the subtrees of its grand-children, i.e., for any child x of v and for any child y of x, v knows n_x and n_y. Computations performed by the nodes are based only on the local knowledge, i.e., the information present in the local memory and that concerns only nodes at distance at most 2. We consider a

[2] A rooted tree is a k-ary tree if each node has k children or less if its children are all leaves.

synchronous setting. That is, the time is slotted in turns. At each turn, any node may run the algorithm based on its knowledge and, depending on the computation, may do one of the following *operations*. In the algorithm we present below, each operation done by a node v consists of rewiring at most two edges at distance at most 2 from v. Moreover, each operation may be done using a constant number of messages of size $O(\log n)$.

In this setting, at every turn, all nodes sequentially run the algorithm. In order to consider the worst case scenario, the order in which all nodes are scheduled during one turn is given by an adversary. The algorithm must ensure that after a finite number of turns, the resulting tree is k-balanced. We are interested in time complexity of the worst case scenario of the repair. That is, the performance of the algorithm is measured by the maximum number of turns after which the tree becomes k-balanced, starting from any n-node tree.

At each turn, a node v executes the algorithm described on Figure 4.1. It is important to emphasis that the balancing process requires no memory of the past operations.

Algorithm executed by a node v in a tree T. If v is not a leaf, let $(v_1, v_2, \cdots, v_d)$ be the $d \geq 1$ children of v ordered by subtree-size, i.e., $n_{v_1} \geq n_{v_2} \geq \cdots \geq n_{v_d}$.

1. **If** v is underloaded (i.e., $d < k$), let a be a child of v_1 with biggest subtree size. **Then** a becomes a child of v.
2. **Else if** v is overloaded (i.e., $d > k$), **then** v_{k+1} becomes a child of v_k.
3. **Else if** v is imbalanced (i.e., $d = k$ and v has two children x and y of v such that $|n_x - n_y| > 1$) **and if** v_1 and v_k are not overloaded, let a and b be two children of v_1 and v_k respectively such that $|n_{v_1} - n_a + n_b - (n_{v_k} - n_b + n_a)|$ is minimum.

   Then a and b exchange their parent.

Figure 4.1: Balancing Process

Theorem 64 *Starting from any n-node rooted tree, the balancing process ($k = 2$) reaches a 2-balanced tree in $O(n^2)$ turns.* *[c-GMNP13]*

To prove the above theorem, we define a (non-trivial) potential function and show that: (1) the initial value of the potential function is bounded; (2) its value may raise due to pull operations (operation 1), but in a limited number of turns and by a bounded amount; (3) a swap operation (operation 3) may not increase its value; (4) if no push nor pull operations (operations 2 or 3) are done, there exists at least one node doing a swap operation, strictly decreasing the potential function.

Unfortunately, we don't know whether this analyze is tight since the best lower bound we found is the following. If the initial tree is a n-node star, then at least $\Omega(n)$ turns are needed before the resulting tree is k-balanced [c-GMNP13].

To get a better understanding, we assume an extra global knowledge: each node knows whether it has a descendant that is not balanced. This extra information

is updated after each operation. Then, our algorithm is modified by adding the condition that any node v executing the balancing process can do a pull or swap operation only if all its descendants are balanced. Adding this property allows to prove better upper bounds on the number of steps, by avoiding conflict between an operation performed by a node and an operation performed by one of its not balanced descendant. We moreover prove that this upper bound for our algorithm is asymptotically tight, reached when input tree is a path.

[c-GMNP13]

Theorem 65 *Starting from any n-node rooted tree, the balancing process with global knowledge reaches a 2-balanced tree in $O(n \log n)$ turns. This bound is tight.*

Because of the significant gap in our worst case analysis of the Balancing process, we also investigated the performance of the algorithm running an implementation under a discrete event simulation. From these simulations, the number of turns spent to balance a random tree progresses logarithmically in regard to the tree size. An interesting continuation of our work would be the examination of our algorithm's average behavior, which as hinted by simulations should yield $O(\log n)$ bound on balancing time.

4.2 Compact routing

In previous Section, we have considered two extreme models for routing messages in a network. In Section 4.1.1, the computation was fully centralized and the objective was to achieve an optimum delay, i.e., mainly routing via shortest paths. On the other hand, in Section 4.1.2, nodes had no knowledge at all about the network and the objective was to ensure some throughput without any guaranties on messages delivery and delay. Moreover, in both cases, there were either a single or undistinguished destinations. Such approaches cannot be used in networks such as the Internet where a certain level of guaranty is required while nodes cannot have a full knowledge of the topology.

In this section, we consider distributed communication networks where messages must be efficiently delivered between pairs of processors. Any source or intermediary node must be able to fast route messages to their destination nodes with only limited knowledge. More precisely, each node is an autonomous computing entity. When a message arrives at some node, the node can read the *header* of the message that contains information including the message's destination. Then, using this information and its local information, stored in its *routing table*, the node decides the out-port by which the message must be transferred to eventually reach its destination. The distributed algorithm executed by the node to direct the traffic is called *routing scheme*. If the designer of the routing scheme is allowed to define the identifiers of the nodes (then, the name of the destination may bring some information for the routing) then the scheme is said *labelled*. Otherwise, i.e., if the routing scheme must work whatever be the identifiers, it is said *name independent*.

When investigating routing schemes, several complexity measures arise. On the one hand, it is desirable to use as short paths as possible for routing messages. Efficiency of a routing scheme is measured in terms of its *multiplicative stretch factor* (resp., *additive stretch factor*), i.e., the maximum ratio (resp., difference) between the length of a route computed by the scheme and that of a shortest path connecting the same pair of nodes. On the other hand, as the amount of storage at each processor is limited, the routing information stored in the processors' local memory (the *routing tables*) must not require too much space with respect to the size of the network. As it is shown below, acceptable tradeoff can only been achieved when considering networks with specific structural properties. A routing scheme that works in any graph is said *universal*.

Last but not least, designing a routing algorithm requires not only to define the routing scheme but also to give an algorithm for computing and updating the routing tables. Because of the dynamicity of networks, it is important to be able to compute the routing information in an efficient distributed way. While many works propose good tradeoffs between the stretch and the size of routing tables, the algorithms that compute those tables are often impracticable because they are centralized or because of their time-complexity (e.g., require to compute some good decomposition or parameter of the graph). However, in the context of large scale networks like social networks or Internet, even polynomial time algorithms are inefficient. Below, we insist on the tradeoff between the length of the computed routes and the time complexity of computing "compact"[3] routing tables.

4.2.1 Tradeoff knowledge/performance/graph structures

The problem of designing routing schemes with small routing tables (RT) size as been introduced at the very beginning of the Internet [KK77, PU89]. It is well-known that no *universal routing scheme* can achieve shortest paths with compact routing tables. More formally, for any such scheme achieving routes with optimal length, there are graphs where routing tables with $\Omega(n \log n)$ bits per node are required [FG97]. Following the work in [TZ01, AGM$^+$04, TZ05, AGM06b], a universal name-independent routing scheme with stretch linear in k and space $n^{1/k} polylog(n)$ space is provided [AGM$^+$08]. Surprisingly, this achieves the same performances as the ones of labelled schemes [TZ01, TZ05]. Lower bounds are provided in [AGM06a] where Abraham et al. prove that there are weighted trees for which every name-independent routing scheme with space less than $n^{1/k}$ requires stretch at least $2k + 1$ and average stretch at least $k/4$.

Subsequently, the interest of the scientific community was turned toward specific properties of graphs. Dedicated routing scheme have been proposed for trees [FG01] and planar graphs [Tho04], and more generally for graphs excluding some fixed minor [AGM05, AG06]. Several universal routing schemes have been proposed that have good performance when the graph has some bounded parameter.

[3] By compact, people generally mean that routing tables have size logarithmic in the network's size.

Bounded doubling dimension.

The *doubling dimension* of a graph G, and more generally of a metric space (X, d), is the least k such that any ball of radius R can be covered by 2^k balls of radius $R/2$ [GKL03]. The problem of determining the doubling dimension of a graph is NP-complete. Fraigniaud et al. [FLV08] studied the doubling dimension of the Internet and they verified that the doubling dimension of the Internet is large. It has also been considered in the context of small world and augmented graphs [FLL06, FLL10]. Efficient compact routing schemes have been proposed for static graphs with bounded doubling dimension, both in the labelled and in the name-independent models. In the labelled model, routing schemes with multiplicative stretch depending on the doubling dimension and using logarithmic routing tables have been proposed in [CGMZ05, AGGM06]. More precisely, [CGMZ05] shows how to perform $(1 + x)$-stretch routing on metrics for any $0 < x < 1$ with routing tables of size at most $(d/x)^{O(d)} \log^2(D)$ bits with at most $(d/x)^{O(d)} \log(D)$ entries in any graph with doubling dimension d and diameter D. The tables can be computed in a distributed way using a distributed breadth-first-search algorithm; this result has been extended in [Sli07]. A name-independent routing scheme for graphs with doubling dimension d is provided in [AGGM06, KRX06]. It achieves constant multiplicative stretch and requires tables of size $2^{O(d)} \log n$ bits. Moreover, it has been proven that any name-independent routing scheme using $o(dn)$ bits has stretch at least 3.

Bounded growth.

A weighted undirected network is Δ *growth-bounded* if the number of nodes at distance $2r$ around any given node is at most Δ times the number of nodes at distance r around the node. G has growth dimension s if G is 2^s growth-bounded. An alternative definition is the following: an undirected graph $G = (V, E)$ is called growth-bounded if there exists a polynomial bounding function $f(r)$ such that for every $v \in V$ and $r \geq 0$, the size of any maximal independent set in the r-neighborhood $\Gamma_r(v)$ is at most $f(r)$. The problem of determining the value Δ bounding the growth of a graph is NP-complete. In [FLV08] it is shown that the ball growth of the Internet is large. We remark that any metric with constant growth-bound has constant doubling dimension, while the opposite is not always true [GKL03]. Abraham and Mahlki [AM05] proposed a name-independent compact routing scheme for graphs with bounded growth. Given a weighted undirected network and $\epsilon > 0$, their routing scheme allows routing along paths of stretch $1 + \epsilon$ and uses routing tables of only $O(1/\epsilon^{O(\log \Delta)} \log^5 n)$ bits per node with high probability. Duchon et al. [DHLS06] proved that if a graph has *moderate* growth, i.e., the size of the balls of radius r is bounded by $r^{O(1/(r \log r))}$, then it can be turned into a navigable small world (augmented by the addition of some "long links").

General approach.

There is a general approach where actually all known graph-specific name-independent routing scheme belongs to. A (s, δ)-*sparse cover* [AP90] for a graph G is a set of clusters $\mathcal{C} \subset 2^{V(G)}$ such that, for any $r \geq 0$: (1) $\forall u \in V(G)$, $\exists C \in \mathcal{C}$ such that $B(u, r) \subseteq C$, (2) $\forall C \in \mathcal{C}$, the diameter of $G[C]$ is $\leq sr$, and (3) $\forall u \in V(G)$, $|\{C \in \mathcal{C} \ : \ u \in C\}| \leq \delta$. If graph G with diameter D has a (s, δ)-sparse cover, then it also admits a name-independent routing scheme with space $\delta \log D \cdot polylog(n)$ and stretch $O(s)$ [TZ05, AGMW10]. The sparse cover technique applies on unweighted minor-free graphs (see also [AGG$^+$13]), but also on weighted bounded growth/doubling dimension graphs. What is interesting in this approach is that the corresponding routing schemes are generic, i.e., universal, and they do not need the pre-computation of some parameter of graphs. However, as a drawback, some sparse cover must be either known in advance or discovered.

Internet-like topologies.

Recent studies have applied compact routing schemes on Internet-like topologies by taking advantage of their topological properties (network diameter growing logarithmically in the number of nodes and node degree distribution following power law). In [KFY04], Krioukov *et al.* showed that the average performance of the stretch-3 compact routing scheme of Thorup on Internet-like topologies is much better than its worst case, it achieves an average stretch = 1.1 (up to 70% of all pair-wise paths being stretch-1 shortest paths). Recently, it has been shown that the general name-dependant routing scheme of Thorup and Zwick performs quite well for scale-free networks [CSTW12]: RT size are proved to be in $O(n^t)$ where t is a small constant less than 1/3 depending on the power-low coefficient of the node degree distribution.

Challenging issues.

The above mentioned techniques have two main drawbacks when large-scale and dynamic networks are concerned. The first is that large scale networks like the Internet do not behave well with respect to the above parameters (doubling dimension, bounded growth, etc.). For instance, the AS network is far from having a small treewidth since it is highly connected and, hence, exhibits a high clustering coefficient. The second challenge stems from that the efficiency of these routing schemes mainly follows from the fact that they use some global view/representation/decomposition of graphs that are difficult to compute even in a static and centralized setting. In particular, drawbacks resulting from the name-dependence of the routing scheme remain unaddressed and limit their applicability to static topologies (thus inapplicable for dynamic and evolutive topologies such as the Internet).

4.2.2 Routing in k-chordal graphs

In order to deal with this problems, we have focused on another structural property of graphs, namely the *chordality*. Our main objective is to provide universal routing scheme for which routing tables are easy to compute and that achieve interesting performances in graphs with small chordality [c-NRS09, j-NRS12, c-KLNS12, j-KLNS14].

Large scale networks are known to have low (logarithmic) diameter and to have high clustering coefficient. Therefore, their chordality (the length of the longest induced cycle) is expected to be somehow limited (e.g., see [Fra05]). Hence, we focus on the class of k-chordal graphs. A graph G is called *k-chordal* if it does not contain induced cycles longer than k. The *chordality* of a graph is the smallest k such that the graph is k-chordal. A 3-chordal graph is simply called *chordal*. This class of graphs received particular interest in the context of compact routing. Dourisboure and Gavoille proposed routing tables of at most $\log^3 n / \log\log n$ bits per node, computable in time $O(m+n\log^2 n)$, that give a routing scheme with additive stretch $2\lfloor k/2 \rfloor$ in the class of k-chordal graphs [DG02, DG07]. Also, Dourisboure proposed routing tables computable in polynomial time, of at most $\log^2 n$ bits, but that give an additive stretch $k+1$ [Dou05]. Using a Lexicographic Breadth-First Search (Lex-BFS) ordering (resp., BFS-ordering) of the vertices, Dragan designed a $O(n^2)$-time algorithm to approximate the distance between all pairs of nodes up to an additive constant of 1 in n-node chordal graphs, and up to $k-1$ in, more general, k-chordal graphs [Dra05]. All these time complexity results consider the centralized model of computation.

Contributions

In [c-NRS09, j-NRS12], we present a simple universal routing scheme based on spanning trees of graphs. Once computing a rooted spanning tree of the graph, the messages have just to be rooted in the tree unless a neighbor w (not necessarily in the tree) of their current position is an ancestor of the destination. In the latter case, the message is routed to w. Our main contribution is to show that, using particular spanning trees (using BFS-orderings of the vertices of the graph), this routing scheme achieves good performances in k-chordal graphs.

[c-NRS09, j-NRS12]

In order to compute the routing tables of any n-node graph with diameter D we propose a distributed algorithm which uses messages of size $O(\log n)$ and takes $O(D)$ time. The corresponding routing scheme achieves the stretch of $k-1$ on k-chordal graphs [c-NRS09, j-NRS12]. We then propose a routing scheme that achieves a better additive stretch of 1 in chordal graphs. In this case, the distributed computation of the routing tables takes $O(\min\{\Delta D, n\})$ time, where Δ is the maximum degree of the graph. Our routing schemes use addresses of size $\log n$ bits and local memory of size $2(d-1)\log n$ bits per node of degree d. The main advantage of our routing scheme is that the routing tables of this scheme can be (re)built from scratch in a efficient way, i.e., spreading a small amount of messages through the network.

We then continued investigating compact routing schemes in chordal graphs to achieve better tradeoffs [c-KLNS12, j-KLNS14]. Our idea has been to take use of the caterpillar tree-decomposition introduced in Chapter 2. Indeed, in such tree-decomposition, it is not the size of the "bags" that are important but the structure of the subgraphs induced by the bags (caterpillars). Such a structure is actually very useful for routing. Moreover, recall that the algorithm we presented (see Section 2.1) to compute such decompositions is greedy which makes it amenable to a distributed implementation. Using caterpillar tree-decompositions, we obtained the following result.

Theorem 66 *For any n-node m-edge graph G with maximum degree Δ and with a k-good tree-decomposition, there is a labelled routing scheme $\mathcal{R}$ with the following properties. $\mathcal{R}$ uses addresses of size $O(\log n)$ bits, port-numbers of size $O(\log \Delta)$ bits and routing tables of size $O(k \cdot \log \Delta + \log n)$ bits. The routing tables, addresses and port-numbers can be computed in time $O(m^2)$. Except the address of the destination (not modifiable), the header of a message contains $O(k \cdot \log \Delta)$ modifiable bits. The header and next hop are computed in time $O(1)$ at each step of the routing. Finally, the additive stretch is $\leq 2k(\lceil \log \Delta \rceil + \frac{5}{2}) - 2\lceil \log \Delta \rceil - 4$.* *[c-KLNS12, j-KLNS14]*

Since the Internet is thought to have *few* long induced cycles (because of its high clustering), this approach seems promising and it would be nice to point out other properties satisfied by the Internet and that could fit in this framework (dealing with the structure of the bags). A possible direction is to introduce some graph embeddings into metric spaces and see if that would provide more tools or structures (like e.g., the triangular inequality) to improve the behaviour and the analysis of our scheme. In particular, introducing the hyperbolicity in this framework is our next step. Since computing the hyperbolicity in huge networks is already a challenging issue, we have started working on attempts to provide a new characterization of this property. Thanks to the Cops and Robber games, we have established a link between such an ordering of the vertices and the hyperbolicity of the graph. This gives an algorithmic approach and a new (partial) characterization for the graph with bounded hyperbolicity (see Section 1.2).

4.2.3 Fault tolerant routing

Compact routing has also been studied in dynamic environment. More precisely, Courcelle and Twigg introduced compact routing with forbidden sets where the routing tables must ensure efficient routing even if any bounded number of nodes fail [CT07] (see also [ACGP10, ACG12]). In [c-HIKN10], we study the problem of finding a destination node t by a mobile agent in an unreliable network where a bounded number of unknown nodes have a byzantine behavior.

We consider the problem of locating an item hosted by some node of the network when each of the nodes maintains a database storing the first edge on a shortest path to the node hosting the desired item (the *destination*). The search is performed by a mobile agent with a limited perception of the environment and with little memory

which starts from some initial node, the *source*. When occupying a node, the mobile agent can perform a query to the node's database that reveals to it an edge that is the beginning of a shortest path from the current node to the item. We assume, however, that some nodes may provide wrong information, that is, a node v may be a *liar* and indicate an edge that does not belong to any shortest path from v to the destination. This is motivated by the fact that inaccuracies occur in the nodes'databases because nodes may malfunction or be malicious, or may store out-of-date information due to the movement of items or the dynamicity of the network.

The problem is then to deal with the potentially incorrect information and to find the desired item. That is, the mobile agent can decide to follow the edge pointed by its current node's database or not. The performance of the search is measured by comparing the *searching time* (a.k.a. *hitting time*), i.e., the length of the walk followed by the agent from the source to the destination, with the length d of a shortest path between these nodes.

Related Work

The search problem in the presence of liars was first investigated by Kranakis and Krizanc [KK99]. In this seminal work, they designed algorithms for searching in distributed networks with ring or torus topology, when a node has a constant probability of being a liar [KK99]. The case when the number k of liars is bounded was first considered in [HKK04], where deterministic algorithms were designed for particular topologies like the complete graph, ring, torus, hypercube, and bounded degree trees. In particular, in bounded degree trees, it is proved that the search time is lower-bounded by $\Omega(d+2^{\min\{k,d\}})$ [HKK04]. Simple randomized and memoryless algorithms are designed in [HKKK08] for the case of bounded degree graphs, where the mobile agent follows the advice with some fixed probability $p > 1/2$. In this class of graphs, the authors showed that the expected distance covered before reaching the destination is upper-bounded by $O(d+r^k)$, where $r = \frac{q}{1-q}$ [HKKK08]. Moreover, this bound is tight since they proved a lower bound of $\Omega(d + r^k)$ in the torus [HKKK08]. While this bound is a bit disappointing, it can be improved for particular graph classes.

Roughly speaking, the algorithms we present in this paper consist of alternation of phases of given duration: either the agent keeps on following the advice provided by the nodes or it walks choosing the next visited node uniformly at random in the neighborhood of the current position. This is closely related to biased random walks (BRW) which are random walks in which nodes have a statistical preference to shift the walker towards the target, or more generally, prevent the walker from staying too long in one vicinity [ABK$^+$96]. More formally, biased random walks are used in network exploration in order to speed up the time required to visit the whole network without an a-priori knowledge of the topology and without an edge/node labeling requirement. For instance, Ikeda *et al.* proved in [IKOY03] that, assuming the knowledge of the degrees of the neighbors, a biased random walk can explore any graph within $O(n^2 \log n)$ edge traversals whereas a uniform random walk takes

Table 4.2: Searching in a path with liars. The listed strategies have no knowledge of inbound ports used by the agent.

Strategy	Expected searching time	Memory	Reference
BRW $[p < 1/2]$	$2^{\Omega(d)}$	–	[Lov93]
BRW $[p = 1/2]$	$\Omega(dn)$	–	
BRW $[p > 1/2]$	$\Omega(d + 2^{\Omega(k)})$	–	[HKKK08]
R/A	$2d + k^{\Theta(1)} + o(d)$	timer	[c-HIKN10]

Table 4.3: Searching with liars in random Δ-regular graphs. Results marked with (*) allow the agent to make use of labels of inbound ports and to have knowledge of n and k.

Strategy	Expected searching time (a.a.s.)	Memory	Reference
* lower bound	$\Omega(\min\{(\Delta-1)^k, (\Delta-1)^d, \log_d n\})$		[c-HIKN10]
BRW $[p > 1/2]$	$\Theta(\min\{(\Delta-1)^k, n^{\Theta(1)}\})$	–	[CF05]
BRW $[p = 1/2]$	$\Omega(\log_d^2 n)$	–	
BRW $[p < 1/2]$	$n^{\Theta(1)}$	–	
* R/A/E	$O(k \log n)$	$\Theta(\log k + \log\log_d n)$	[c-HIKN10]
R/A	$O(k^3 \log^3 n)$	timer	[c-HIKN10]

$\Theta(n^3)$ steps for some graphs. In our context, however, the bias may be erroneous due to the presence of liars.

In our work, we focus on some particular and widely used topologies. Expanders and random regular graphs have been extensively studied for the design of optimal networks and algorithms for routing [NPSY94, KR96, Fri00]. Because of their low diameter and high connectivity, random regular graphs are also of interest in Peer-to-Peer networks (e.g., see [GHW08]). More generally, it can be observed that many interaction networks like peer to peer overlay networks, small worlds and scale-free networks are expanders despite this is not proved in the original papers. For instance, Bourassa and Holt [BH03] proposed a fully decentralized protocol based on random walks for the nodes to join and leave the network. They conjectured that their protocol produces random regular graphs, which was proved formally in [CDG07]. On the other hand, Cooper [CF05] *et al.* show that random regular graphs are expanders.

Contributions

[c-HIKN10]

In [c-HIKN10], we investigate the search problem in the class of regular graphs in the presence of a bounded number of liars. Our main contribution is the design and study of a randomized algorithm, called R/A that alternates phases of pure random walk (R) with phases in which the agent follows the advice (A). We show

that Algorithm R/A improves upon previous algorithms for the search problem in paths and random Δ-regular graphs. In particular, in these classes, we prove that the Algorithm R/A achieves searching time much smaller than $\Omega(d + 2^k)$, which is the lower bound for general regular graphs [HKKK08]. Note that the graph classes we consider capture the two extreme types of behavior in terms of expansion, since random Δ-regular graphs are good expanders, while the other classes are highly symmetric graphs with poor expansion. Note, however, that Algorithm R/A is generic and works for any topology. We also introduce Algorithm $R/A/E$ in which, in addition of the phases random (R) and advice (A), there is a phase where the agent explores (E) some bounded ball around its current position. This latter algorithm requires more memory for the agent.

Tables 4.2 and 4.3 establish a comparison between the performances of Algorithm R/A, and the performances of the *Biased Random Walk* (BRW).

4.3 Local models and Property testing

We conclude this chapter by discussing an important issue that is related to both graph property measurements and the potential exploitation of these measurements by routing algorithms. Centralized solutions for property testing/measurements do not allow efficiently facing the dynamicity of the Internet networks (variations of both the topology and the traffic). Hence, it is natural to propose distributed or even local solutions. That is, solutions where the nodes have *bounded memory* and *only local knowledge* of the network topology. The difference between *distributed* and *local* algorithms is mainly relative to the time that is allowed to the nodes to exchange messages. Indeed, the less exchanges are allowed, the less the information can spread through the network and the more partial the knowledge the nodes have about the network is. However, the notion of distributed or local algorithms is not well formalized since numerous models exist. Among other difficult points, one of our objectives is to determine relevant distributed models to be considered. That is, most of the properties that are relevant for routing schemes have very little chance to be computable with a pure local view of the topology. On the other hand, gathering all the information in one node and then spreading the result would not be realistic in the Internet (it would imply too much control traffic).

Some theoretical and over-simplified models were conceived for studying particular aspects of distributed protocols such as fault-tolerance, synchronism, locality, congestion, etc. In the model CONGEST (see the book of Peleg [Pel00]) a network is represented by a graph whose nodes correspond to network processors and edges to inter-processor links. The communication is synchronous and occurs in discrete rounds. In each round every processor can send a message of size $O(\log n)$ through each of its outgoing links (where n is the number of nodes).

Some variations to the CONGEST model have been proposed. The general idea is to remove some restrictions (making it more powerful) in order to focus on some particular issues. In that spirit, Linial [Lin92] introduced in a seminal

paper the free model (also called LOCAL [Pel00]). The only difference with the CONGEST model is that the restriction on the size of the messages is removed so that every vertex is allowed to send unbounded size messages in every round. By relaxing the constraint on the size of the messages, this model focuses on the issue of locality in distributed systems. For that reason, the answer to the question *What cannot be computed locally?* [KMW04] (in the LOCAL model) appears to be a crucial one. In this context, Kuhn et al. [KMW04] show that difficult problems like minimum vertex cover and minimum dominating set cannot be approximated well when processors can exchange arbitrary long messages during a bounded number of rounds. Frischknecht *et al.* prove that $\Omega(n^2)$ bits must be exchanged to determine the girth of a graph or to decide if the graph has diameter at most 4 [FHW12]. Grumbach and Wu [GW09] also use the CONGEST model. They are interested in *frugal computations*, i.e., computations where the total amount of information traversing each edge is $O(\log n)$. Their motivation was to understand the impact of locality. In fact, they show that for planar networks or networks of bounded degree, any first order logic formula can be evaluated frugally (in their setting). More recently, an important research effort is done to formalize local models of computation [Fra10, FRT11].

Since we need to face both locality and dynamicity issues, we are developing new techniques allowing to obtain global structural information from local (partial) views of the network. We have investigated the question of determining which graph properties can or cannot be computed using only local information. The distributed model we consider is a restriction of the classical CONGEST (distributed) model and it is close to the simultaneous message model defined by Babai, Kimmel and Lokam [BKL95]. More precisely, each of these n nodes only knows its own ID and the IDs of its neighbours and is allowed to send a message of $O(\log n)$ bits to some central entity, called the referee. We then investigate whether the referee is able to decide some basic structural properties of the network topology G or not. More formally:

Definition 1 *A* one-round protocol Γ *is a family* $(\Gamma^l_n, \Gamma^g_n)_{n\in\mathbb{N}}$*, where:*

- $\Gamma^l_n : \{1,\dots,n\} \times \mathcal{P}(\{1,\dots,n\}) \to \{0,1\}^*$ *is the* local function *of* Γ *for graphs of size* n*,*
- $\Gamma^g_n : (\{0,1\}^*)^n \to \{0,1\}^*$ *is the* global function *of* Γ *for graphs of size* n*.*

Given $G = (V = \{1,\dots,n\}, E)$*, the message vector of* Γ *on* G *is:*

$$\Gamma^l(G) = \Big(\Gamma^l_n(1, N_G(1)), \dots, \Gamma^l_n(n, N_G(n))\Big). \tag{4.1a}$$

The output of Γ *on* G *is:*

$$\Gamma(G) = \Gamma^g_n(\Gamma^l(G)). \tag{4.1b}$$

We define

$$|\Gamma^l(G)| = \max_{1\le i\le n} |\Gamma^l_n(i, N_G(i))|. \tag{4.1c}$$

Γ *is said to be* frugal *if:*

$$\max_{G \text{ graph of } n \text{ nodes}} |\Gamma^l(G)| = O(\log n). \tag{4.1d}$$

Notice that function Γ^l_n can be evaluated in any pair (i, N) where $i \in \{1, \ldots, n\}$ and $N \subseteq \{1, \ldots, n\}$. Then, $\Gamma^l_n(i, N)$ corresponds to the message sent to the referee by node i in a graph of n nodes when its neighborhood is N.

In [c-BMN$^+$11, j-BKM$^+$14], we show that even simple questions cannot be solved in general.

[c-BMN$^+$11, j-BKM$^+$14]

Theorem 67 *There is no one-round frugal protocol allowing the referee to decide whether an arbitrary graph G contains a square or a triangle as a subgraph, or has diameter at most* 3*.*

On the other hand, the referee can decode the messages in order to have full knowledge of G when G belongs to graph classes like planar graphs, bounded-treewidth graphs and, more generally, bounded-degeneracy graphs.

[c-BMN$^+$11]

Theorem 68 *There exists a one-round frugal protocol allowing the referee to reconstruct graphs of bounded degeneracy.*

Questions related to the connectivity of arbitrary graphs are still open. We have continued our investigation by proposing variations of our distributed model. Following our framework, we have exhibited a hierarchy of problems and distributed models of computation. Without elaborating on the details, we have proposed four models of distributed computation, called SimAsync, SimSync, FreeAsync and FreeSync. SimAsync is exactly the model described in Definition 1. The remaining three models are more powerful since nodes are allowed to send sequentially their message to the referee and may have access to previous messages. In particular, FreeAsync allows us to simulate asynchronicity of the network. For any model X, we denote by $X[f(n)]$ the set of problems that can be solved in n-node graphs in the model X when the nodes are allowed to send messages of size $O(f(n))$ bits.

[c-BKN$^+$12, j-BKM$^+$14]

Theorem 69 *For any $f(n) = o(n)$ we have that*
SimAsync$[f(n)] \subsetneq$ SimSync$[f(n)] \subsetneq$ FreeAsync$[f(n)] \subseteq$ FreeSync$[f(n)]$.

In the following table, we summarize our results. no (respectively, yes) in row i and column j means that property i cannot (respectively, can) be computed in Model j.

The SimAsync Model is very constrained and so offers very limited power of computation to nodes. On the other hand, the FreeSync Model is very powerful but it requires too many computational resources and so is not realistic for practical applications. Hence, one crucial question for the EULER project is to design an intermediate local model that would be realistic enough while allowing to compute basic properties such as connectivity, and possibly more elaborate properties.

	SimAsync	SimSync	FreeAsync	FreeSync
Build k-degenerate	yes	yes	yes	yes
rooted MIS	no	yes	yes	yes
Square	no	no	?	?
EOB-BFS	no	no	yes	yes
BFS	?	?	?	yes

Table 4.4: Classification of communication models. Build k-degenerate is the problem of deciding whether a graph is k-degenerate and to build its adjacency matrix if it is the case. Rooted MIS is the problem to compute a maximal independent set containing a predetermined node. Square is the problem of deciding whether a graph contains a square. EOB-BFS aims at computing a BFS-tree in particular bipartite labelled graphs where the bipartition is defined by the parity of the labels of the nodes.

Up to now, our study has focused on the case of static networks and it should be extended in order to include dynamicity. This requires the understanding of this dynamics and the introduction of suitable models of dynamicity. For this purpose, new algorithmic tools that take timing into account should be developed.

Appendix: Other contributions on complexity and graph structures

Content

In the last chapter of this thesis, we present our study of two optimization problems in graphs. These studies are in marge of previous chapters of this thesis, on the one hand because they are mainly centralized, on the other hand because their motivation is mainly theoretical. However, the general guideline is again to study the complexity of these problems depending of the structural properties of the considered graphs.

First, we study the complexity of computing the *hull number* of graphs in various graph classes [j-ACG$^+$13]. This parameter somehow extends the notion of classical convexity to graphs.

Second, we prove that the *max-coloring problem* cannot be solved in sub-exponential time in trees unless the Exponential Time Hypothesis (ETH) is wrong [c-ANP14].

A.1 Convexity: hull number of some graph classes

We study the geodetic convexity of graphs focusing on the problem of the complexity to compute inclusion-minimum hull set of a graph in several graph classes [j-ACG$^+$13]. For any two vertices $u, v \in V$ of a connected graph $G = (V, E)$, the *closed interval* $I[u, v]$ of u and v is the the set of vertices that belong to some shortest (u, v)-path. For any $S \subseteq V$, let $I[S] = \bigcup_{u,v \in S} I[u, v]$. A subset $S \subseteq V$ is *geodesically convex* if $I[S] = S$. In other words, a subset S is convex if, for any $u, v \in S$ and for any shortest (u, v)-path P, $V(P) \subseteq S$. This clearly generalizes Euclidean convexity to graphs. Given a subset $S \subseteq V$, the *convex hull* $I_h[S]$ of S is the smallest convex set that contains S. We say that S is a *hull set* of G if $I_h[S] = V$. The size of a minimum hull set of G is the *hull number* of G, denoted by $hn(G)$. The Hull Number problem is to decide if $hn(G) \leq k$, for a given graph G and an integer k.

In the seminal work [ES85], the authors present some upper and lower bounds on the hull number of general graphs and characterize the hull number of some particular graphs. The corresponding minimization problem has been shown to be NP-complete [DGK$^+$09]. Dourado *et al.* also proved that the hull number of unit interval graphs, cographs and split graphs can be computed in polynomial time [DGK$^+$09]. Bounds on the hull number of triangle-free graphs are shown in [DPRS10]. The hull number of the cartesian and the strong product of two connected graphs is studied in [CCJ04, CHM$^+$10]. Recently, the case of chordal graphs have been considered in [KN13]. In [HJM$^+$05], the authors have studied the relationship between the *Steiner number* and the hull number of a given graph. An oriented version of the Hull Number problem is studied in [CFZ03, Far05]. Other parameters related to the geodetic convexity have been studied in [CHZ02, CWZ02]. For work in other graph convexities than the geodetic convexity, the reader may refer to [CMS05, DPS10, FJ86, CM99] and to Chapter 5 in [Soa13].

[j-ACG$^+$13]

Contributions. We first answer an open question in [DGK$^+$09] by showing that the Hull Number problem is NP-hard even when restricted to the class of bipartite graphs (reduction of 3-SAT). Then, we design polynomial time algorithms to solve the Hull Number problem in several graphs' classes: the class of complements of bipartite graphs, the class of $(q, q-4)$-graphs[4]. The latter result, generalizing some results in [ACG$^+$11], consists into a FPT algorithm parametrized by q and obtained by using Primeval Decomposition [BO99a] (generalization of modular decomposition). Finally, we prove tight upper bounds on the hull number of graphs. In particular, we show that the hull number of an n-node graph G without simplicial vertices is at most $\lceil \frac{3(n-1)}{5} \rceil$ in general, at most $\lceil \frac{n-1}{2} \rceil$ if G is regular or has no triangle, and at most $\lceil \frac{n-1}{3} \rceil$ if G has girth at least 6.

Among the remaining interesting questions, I should mention the complexity of computing the hull number in planar graphs (in contrast, for any set S of a d-dimensional euclidean space, the *gift wrapping algorithm* computes the convex hull and a minimum-inclusion hull set of S in polynomial-time in $|S|$, d being fixed).

[4] A graph $G = (V, E)$ is a $(q, q-4)$-graph, for a fixed $q \geq 4$, if for any $S \subseteq V$, $|S| \leq q$, S induces at most $q - 4$ paths on 4 vertices [BO95].

A.2 Weighted Coloring in trees

In [GZ97], Guan and Zhu generalized Graph Coloring to vertex-weighted graphs. A *(vertex) weighted graph* (G, w) consists of a loop-less graph $G = (V, E)$ and a weight function $w : V \to \mathbb{R}_+$ over the vertices of G. Given a proper k-coloring $c = (S_1, \ldots, S_k)$[5] of a weighted graph (G, w), the *weight of color* i $(1 \leq i \leq k)$ is defined by $w(i) = \max_{v \in S_i} w(v)$. The *weight of coloring* c is $w(c) = \sum_{i=1}^{k} w(i)$. The *weighted chromatic number* of (G, w), denoted by $\chi_w(G)$, is the minimum weight of a proper coloring of (G, w). The Weighted Coloring Problem (also known as Max-coloring [PRV11, PPR05, PR05, PRV04, PPR04]) takes a weighted graph (G, w) as input and looks for $\chi_w(G)$ [GZ97].

As it generalizes Graph Coloring to weighted graphs, this problem is NP-hard in general graphs. Moreover, it has been shown NP-hard in bipartite graphs [DdWMP02], where Graph Coloring is trivial. Weighted Coloring has been shown to be NP-hard in the classes of split graphs, interval graphs and triangle-free planar graphs with bounded degree [DdWMP02, PRV04, dWDE$^+$09, EMP06, PRV11]. On the other hand, the weighted chromatic number of cographs and of some subclasses of bipartite graphs can be found in polynomial-time [DdWMP02, dWDE$^+$09]. Constant-factor approximation algorithms have been designed for various graph classes such as interval graphs, perfect graphs, etc. [PRV04, PPR04, PPR05, PR05, EL12]. In particular, it is known that Weighted Coloring can be approximated by a factor $\frac{8}{7}$ in bipartite graphs and cannot be approximated by a factor $\frac{8}{7} - \epsilon$ for any $\epsilon > 0$ in this graph class unless $P = NP$ [PR05].

Guan and Zhu showed that, given a fixed parameter $r \in \mathbb{N}$, the minimum weight of a coloring using at most r colors can be computed in polynomial-time (exponential in r) in the class of bounded treewidth graphs (a.k.a. partial k-trees) [GZ97]. They left open the question of the time-complexity of the Weighted Coloring Problem in this class (partial k-trees) and, in particular, in trees. In [PR05], a sub-exponential algorithm and a polynomial-time approximation scheme to compute the weighted chromatic number of trees are presented. Later on, Escoffier et al. proposed a polynomial-time approximation scheme to compute the weighted chromatic number of bounded treewidth graphs [EMP06]. Kavitha and Mestre recently presented polynomial-time algorithms for subclasses of trees [KM12]. They show that computing the weighted chromatic number can be done in linear time in the class of trees where nodes with degree at least three induce a stable set [KM12].

In the last years, many studies have tackled the Weighted Coloring Problem, however its complexity was still unknown on trees. We answer this question.

Theorem 70 *If ETH is true, then the best algorithm to compute the weighted chromatic number of an n-node tree T has time-complexity $n^{\Theta(\log n)}$.* *[c-ANP14]*

The reduction we found is quite intricate. In particular, as an intermediary step, we define and use a variant of 3-SAT where we introduce some dependencies between valid assignments of subgroup of variables.

[5] A proper k-coloring of G is a partition $c = (S_1, \ldots, S_k)$ of V such that, for any $1 \leq i \leq k$, S_i is a non-empty independent set of vertices that have the same color i.

Conclusion and Perspectives

Recap of open problems. Rather than repeating once again our contributions, we prefer to list here some of the open problems mentioned throughout this manuscript. We don't give the exhaustive list of all questions already mentioned but focus on the problems that seem difficult (i.e., open for a long time) and on some intriguing questions that may be easier.

Chapter 1. We naturally start with the Meyniel's conjecture (1985) that asks whether the cop-number of any n-node graph is $O(\sqrt{n})$ (page 21). A possible next step toward its solution would be to prove the conjecture for new graph classes, in order to refine Assertion 1. More anecdotic but still very intriguing is the question of determining the minimum number of cops with speed one that are needed to capture a robber with speed two in a $n \times n$ grid (page 29). A bit further from Cops and Robber games, many problems remain open in the Surveillance game. In particular, what is the cost of connectivity? In other words, does there a constant $c > 0$ exist such that, for any graph G, the connected surveillance number of G is at most c times (or differs of at most c from) the surveillance number of G (page 39)? Finally, concerning turn-by-turn games, what is the complexity of the fractional game and does it provide an approximation for known integral games (page 43)?

Chapter 2. Even though we did not work on it, it is worth mentioning here the question of whether computing the treewidth of planar graphs can be solved in polynomial time (page 47). Also, while the treewidth problem has no polynomial kernel (unless $NP \subseteq coNP/poly$), is there a better (single exponential?) FPT algorithm for the treewidth problem? Since non-deterministic graph searching allowed us to get new results on tree-decomposition, further understanding of this graph searching variant is expected. Here are some questions. What is the complexity of non-deterministic graph searching in trees (page 53)? Moreover, is there a dual structure for branched treewidth, similar to brambles for treewidth or blockage for pathwidth (page 54)? More generally, can some digraph decomposition be defined that fit in our partion function framework (page 54)? In other words, can we extend our duality result (Th. 41) and/or our FPT algorithm (page 56) to some digraph decomposition? More generally, is there a "good" digraph decomposition (Section 2.2.3)?

Chapter 3. Here, the remaining questions are interesting for the new techniques that are required to answer them. In particular, the classical graph searching problem belongs to NP because it satisfies the monotonicity property (page 71). What about non-monotone variants such as connected or exclusive graph searching? Also, the study of graph searching in a distributed setting

led to an interesting perspective on how to compute polynomial-time approximation for computing the pathwidth of graphs (page 76). Can this problem be approximated up to a constant ratio? In the context of mobile agents coordination, many open problems remain, such as tasks that can be executed by a team of mobile robots. Since we mainly consider the feasibility of few tasks (Section 3.3.2), the question of achieving them while optimizing some objective function (for instance, minimizing the maximum number of moves of each robot for saving energy) remains open.

Chapter 4. For once, let us start with an anecdotic open problem: what is the complexity of computing the optimal scheduling (with minimum makespan) for the information gathering problem in grids (page 95)? More importantly/interestingly, most clever compact routing schemes rely on centralized algorithms that compute routing tables with a global knowledge of the graphs (Section 4.2.1). Is it possible to achieve the same performance as existing algorithms by manipulating only local structural knowledge? In other words, can we design local algorithms for computing routing tables that allow efficient compact routing schemes in some particular graph classes? The last very intriguing question we would like to mention is: is there a one-round frugal protocol (Definition 1) that decides if an input graph is connected? Finally, pursuing the study of the hierarchy of distributed models as in Theorem 69 seems interesting for designing efficient local algorithms that compute graph properties.

Some perspectives. The question of designing efficient distributed algorithms to cope with the increasing size of large-scale communication networks remains widely open. On the centralized point of view, recent results brought new breakthrough for the design of "efficient" (i.e. single exponential) FPT algorithms (see, e.g., Section 2.1). However, most (all?) of the proposed solutions rely on highly centralized methods such as the pre-computation of tree-decompositions (e.g., bidimensionality) or the use of hash functions for de-randomization (e.g., color coding). Moreover, such algorithms are difficult to implement efficiently, not only because of their exponential (in the parameter) components (unavoidable because of the NP-hardness of the problems), but also because of some omitted polynomial (in the size of the instance) parts. Hence, pursuing theoretical studies on the design of efficient algorithms for NP-complete problems is a necessary direction of research.

For this purpose, I will continue my studies of new algorithms to compute tree- and path-decompositions of graphs. As already mentioned, few heuristics and approximation algorithms exist and this is an approach that I want to investigate. In our study of routing reconfiguration, we have designed an heuristic algorithm to compute the process number of dependency digraphs (page 66). We currently continue in this direction by adding pre-processing steps and this gives promising results for new heuristic for computing the pathwidth of graphs. Extending our approach to treewidth seems quite challenging.

[c-CMN14]

A different approach is to focus on different graph decompositions. For instance, it is interesting to look at tree-decompositions whose bags satisfy particular structures instead of bounding their size. We have started this study by proposing a polynomial-time algorithm that computes "caterpillar decompositions" (Theorem 33) and we expect this kind of tree-decomposition to have nice algorithmic applications.

Of course, it would be ideal to discover a particular structure that would be computable efficiently in a distributed (local?) setting and that could help for algorithmic purpose as well. I aim at continuing my study on the hierarchy of distributed models in order to understand what can be expected.

Finally, I would also be interested in other applications than telecommunication networks. Recently, some members of COATI have used their expertise in optimization in the domain of bio-informatics. It is clearly an area where my study of efficient algorithms in large-scale graphs has applications.

To conclude, Pursuit-evasion games remain one of my favorite topics which is an inexhaustible (?) source of questions.

Bibliography

My bibliography

— International journals —

[j-ACG+13] Júlio Araújo, Victor Campos, Frédéric Giroire, Nicolas Nisse, Leonardo Sampaio, and R. Soares. On the hull number of some graph classes. *Theoretical Computer Science*, 475:1–12, 2013.

[j-AMNT09] Omid Amini, Frédéric Mazoit, Nicolas Nisse, and Stéphan Thomassé. Submodular partition functions. *Discrete Mathematics*, 309(20):6000–6008, 2009.

[j-BFF+12] Lali Barrière, Paola Flocchini, Fedor V. Fomin, Pierre Fraigniaud, Nicolas Nisse, Nicola Santoro, and Dimitrios M. Thilikos. Connected graph searching. *Information and Computation*, 219:1–16, 2012.

[j-BFNV08] Lélia Blin, Pierre Fraigniaud, Nicolas Nisse, and Sandrine Vial. Distributed chasing of network intruders. *Theoretical Computer Science*, 399(1-2):12–37, 2008.

[j-BKM+14] Florent Becker, Adrian Kosowski, Martin Matamala, Nicolas Nisse, Ivan Rapaport, Karol Suchan, and Ioan Todinca. Allowing each node to communicate only once in a distributed system: shared whiteboard models. *Distributed Computing*, 2014. to appear.

[j-CCM+11] Nathann Cohen, David Coudert, Dorian Mazauric, Napoleão Nepomuceno, and Nicolas Nisse. Tradeoffs in process strategy games with application in the WDM reconfiguration problem. *Theoretical Computer Science*, 412(35):4675–4687, 2011.

[j-CCNV11] Jérémie Chalopin, Victor Chepoi, Nicolas Nisse, and Yann Vaxès. Cop and robber games when the robber can hide and ride. *SIAM J. Discrete Math.*, 25(1):333–359, 2011.

[j-DDN+14b] Gianlorenzo D'Angelo, Gabriele Di Stefano, Alfredo Navarra, Nicolas Nisse, and Karol Suchan. Computing on rings by oblivious robots: a unified approach for different tasks. *Algorithmica*, 2014. to appear.

[j-FFN09] Fedor V. Fomin, Pierre Fraigniaud, and Nicolas Nisse. Nondeterministic graph searching: From pathwidth to treewidth. *Algorithmica*, 53(3):358–373, 2009.

[j-FGJM+14] Fedor V. Fomin, Frédéric Giroire, Alain Jean-Marie, Dorian Mazauric, and Nicolas Nisse. To satisfy impatient web surfers is hard. *Theoretical Computer Science*, 526:1–17, 2014.

[j-FGK+10] Fedor V. Fomin, Petr A. Golovach, Jan Kratochvíl, Nicolas Nisse, and Karol Suchan. Pursuing a fast robber on a graph. *Theoretical Computer Science*, 411(7-9):1167–1181, 2010.

[j-FN08] Pierre Fraigniaud and Nicolas Nisse. Monotony properties of connected visible graph searching. *Information and Computation*, 206(12):1383–1393, 2008.

[j-INS09] David Ilcinkas, Nicolas Nisse, and David Soguet. The cost of monotonicity in distributed graph searching. *Distributed Computing*, 22(2):117–127, 2009.

[j-KLNS14] Adrian Kosowski, Bi Li, Nicolas Nisse, and Karol Suchan. k-chordal graphs: From cops and robber to compact routing via treewidth. *Algorithmica*, 2014. to appear.

[j-MN08] Frédéric Mazoit and Nicolas Nisse. Monotonicity of non-deterministic graph searching. *Theoretical Computer Science*, 399(3):169–178, 2008.

[j-Nis09] Nicolas Nisse. Connected graph searching in chordal graphs. *Discrete Applied Mathematics*, 157(12):2603–2610, 2009.

[j-NRS12] Nicolas Nisse, Ivan Rapaport, and Karol Suchan. Distributed computing of efficient routing schemes in generalized chordal graphs. *Theoretical Computer Science*, 444:17–27, 2012.

[j-NS09] Nicolas Nisse and David Soguet. Graph searching with advice. *Theoretical Computer Science*, 410(14):1307–1318, 2009.

— International conferences —

[c-ANP14] Júlio Araújo, Nicolas Nisse, and Stéphane Pérennes. Weighted Coloring in Trees. In *31st Symposium on Theoretical Aspects of Computer Science (STACS)*, Leibniz International Proceedings in Informatics (LIPIcs) series. Schloss Dagstuhl, 2014. to appear.

[c-BBN12] Lélia Blin, Janna Burman, and Nicolas Nisse. Brief announcement: Distributed exclusive and perpetual tree searching. In *26th International Symposium on Distributed Computing (DISC)*, volume 7611 of *Lecture Notes in Computer Science*, pages 403–404. Springer, 2012.

[c-BBN13] Lélia Blin, Janna Burman, and Nicolas Nisse. Exclusive graph searching. In *21st European Symposium on Algorithms (ESA)*, volume 8125 of *Lecture Notes in Computer Science*, pages 181–192. Springer, 2013.

[c-BCM+12] Sonia Belhareth, David Coudert, Dorian Mazauric, Nicolas Nisse, and Issam Tahiri. Reconfiguration with physical constraints in WDM networks. In *Workshop on New Trends in Optical Networks Survivability (Netsurviv), Proceedings of IEEE Int. Conf. on Communications (ICC Workshop)*, pages 6257–6261. IEEE, 2012.

[c-BFNV06] Lélia Blin, Pierre Fraigniaud, Nicolas Nisse, and Sandrine Vial. Distributed chasing of network intruders. In *13th International Colloquium on Structural Information and Communication Complexity (SIROCCO)*, volume 4056 of *Lecture Notes in Computer Science*, pages 70–84. Springer, 2006.

[c-BKN+12] Florent Becker, Adrian Kosowski, Nicolas Nisse, Ivan Rapaport, and Karol Suchan. Allowing each node to communicate only once in a distributed system: shared whiteboard models. In *24th ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 11–17. ACM, 2012.

[c-BMN+11] Florent Becker, Martín Matamala, Nicolas Nisse, Ivan Rapaport, Karol Suchan, and Ioan Todinca. Adding a referee to an interconnection network: What can(not) be computed in one round. In *25th IEEE International Symposium on Parallel and Distributed Processing (IPDPS)*, pages 508–514. IEEE, 2011.

[c-BNRR09] Jean-Claude Bermond, Nicolas Nisse, Patricio Reyes, and Hervé Rivano. Minimum delay data gathering in radio networks. In *8th International Conference on Ad-Hoc, Mobile and Wireless Networks (ADHOC-NOW)*, volume 5793 of *Lecture Notes in Computer Science*, pages 69–82. Springer, 2009.

[c-CCM+10] Nathann Cohen, David Coudert, Dorian Mazauric, Napoleão Nepomuceno, and Nicolas Nisse. Tradeoffs in process strategy games with application in the WDM reconfiguration problem. In *5th Int. Conf. on Fun with Algorithms (FUN)*, volume 6099 of *Lecture Notes in Computer Science*, pages 121–132. Springer, 2010.

[c-CHM+09] David Coudert, Florian Huc, Dorian Mazauric, Nicolas Nisse, and Jean-Sébastien Sereni. Routing reconfiguration/process number: Coping with two classes of services. In *13th Conference on Optical Network Design and Modeling (ONDM)*. IEEE, 2009.

[c-CHN+10] Christelle Caillouet, Florian Huc, Nicolas Nisse, Stéphane Pérennes, and Hervé Rivano. Stability of a localized and greedy routing algorithm. In *12th Workshop on Advances in Parallel and Distributed Computational Models (APDCM), IPDPS Workshop*, pages 1–8. IEEE, 2010.

[c-CMN09] David Coudert, Dorian Mazauric, and Nicolas Nisse. On rerouting connection requests in networks with shared bandwidth. *DIMAP workshop on Algorithmic Graph Theory (AGT), Electronic Notes in Discrete Maths*, 32:109–116, 2009.

[c-CMN14] David Coudert, Dorian Mazauric, and Nicolas Nisse. Experimental evaluation of a branch and bound algorithm for computing pathwidth. In *13th International Symposium on Experimental Algorithms (SEA)*, Lecture Notes in Computer Science. Springer, 2014. to appear.

[c-DDN+13b] Gianlorenzo D'Angelo, Gabriele Di Stefano, Alfredo Navarra, Nicolas Nisse, and Karol Suchan. A unified approach for different tasks on rings in robot-based computing systems. In *15th Workshop on Advances in Parallel and Distributed Computational Models (APDCM), IPDPS Workshop*, pages 667–676. IEEE, 2013.

[c-DNN14] Gianlorenzo D'Angelo, Alfredo Navarra, and Nicolas Nisse. Robot searching and gathering on rings under minimal assumptions. In *15th International Conference on Distributed Computing and Networking (ICDCN)*, Lecture Notes in Computer Science, pages 149–164, 2014.

[c-FFN05] Fedor V. Fomin, Pierre Fraigniaud, and Nicolas Nisse. Nondeterministic graph searching: From pathwidth to treewidth. In *30th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 3618 of *Lecture Notes in Computer Science*, pages 364–375. Springer, 2005.

[c-FGJM+12] Fedor V. Fomin, Frédéric Giroire, Alain Jean-Marie, Dorian Mazauric, and Nicolas Nisse. To satisfy impatient web surfers is hard. In *6th International Conference on Fun with Algorithms (FUN)*, volume 7288 of *Lecture Notes in Computer Science*, pages 166–176. Springer, 2012.

[c-FN06a] Pierre Fraigniaud and Nicolas Nisse. Connected treewidth and connected graph searching. In *7th Latin American Symp. on Theoretical Informatics (LATIN)*, volume 3887 of *Lecture Notes in Computer Sc.*, pages 479–490. Springer, 2006.

[c-FN06b] Pierre Fraigniaud and Nicolas Nisse. Monotony properties of connected visible graph searching. In *32nd International Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, volume 4271 of *Lecture Notes in Computer Science*, pages 229–240. Springer, 2006.

[c-GMN+13] Frédéric Giroire, Dorian Mazauric, Nicolas Nisse, Stéphane Pérennes, and Ronan P. Soares. Connected surveillance game. In *20th International Colloquium on Structural Information and Communication Complexity (SIROCCO)*, volume 8179 of *Lecture Notes in Computer Science*, pages 68–79. Springer, 2013.

[c-GMNP13] Frédéric Giroire, Remigiusz Modrzejewski, Nicolas Nisse, and Stéphane Pérennes. Maintaining balanced trees for structured distributed streaming systems. In *20th International Colloquium on Structural Information and Communication Complexity (SIROCCO)*, volume 8179 of *Lecture Notes in Computer Science*, pages 177–188. Springer, 2013.

[c-HIKN10] Nicolas Hanusse, David Ilcinkas, Adrian Kosowski, and Nicolas Nisse. Locating a target with an agent guided by unreliable local advice: how to beat the random walk when you have a clock? In *29th Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pages 355–364. ACM, 2010.

[c-INS07] David Ilcinkas, Nicolas Nisse, and David Soguet. The cost of monotonicity in distributed graph searching. In *11th International Conference on Principles of Distributed Systems (OPODIS)*, volume 4878 of *Lecture Notes in Computer Science*, pages 415–428. Springer, 2007.

[c-KLNS12] Adrian Kosowski, Bi Li, Nicolas Nisse, and Karol Suchan. k-chordal graphs: From cops and robber to compact routing via treewidth. In *39th International Colloquium on Automata, Languages, and Programming (ICALP (2))*, volume 7392 of *Lecture Notes in Computer Science*, pages 610–622. Springer, 2012.

[c-MN07] Frédéric Mazoit and Nicolas Nisse. Monotonicity of non-deterministic graph searching. In *33rd International Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, volume 4769 of *Lecture Notes in Computer Science*, pages 33–44. Springer, 2007.

[c-NRS09] Nicolas Nisse, Ivan Rapaport, and Karol Suchan. Distributed computing of efficient routing schemes in generalized chordal graphs. In *16th International Colloquium on Structural Information and Communication Complexity (SIROCCO)*, volume 5869 of *Lecture Notes in Computer Science*, pages 252–265. Springer, 2009.

[c-NS07] Nicolas Nisse and David Soguet. Graph searching with advice. In *14th International Colloquium on Structural Information and Communication Complexity (SIROCCO)*, volume 4474 of *Lecture Notes in Computer Science*, pages 51–65. Springer, 2007.

[c-NS08] Nicolas Nisse and Karol Suchan. Fast robber in planar graphs. In *34th International Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, volume 5344 of *Lecture Notes in Computer Science*, pages 312–323, 2008.

[c-NS13] Nicolas Nisse and Ronan P. Soares. On the monotonicity of process number. In *7th Latin-American Algorithms, Graphs and Optimization Symposium (LAGOS)*, volume 44 of *Electronic Note Discrete Maths*, pages 141–147. Elsevier, 2013.

— Submitted papers —

[s-ACN07] Omid Amini, David Coudert, and Nicolas Nisse. Some results on nondeterministic graph searching in trees. Technical Report INRIA-00174965, INRIA, 2007.

[s-BBM+13] Pascal Berthomé, Tom Bouvier, Frédéric Mazoit, Nicolas Nisse, and Ronan P. Soares. A unified FPT algorithm for width of partition functions. Technical Report RR-8372, INRIA, 2013.

[s-BLN+13] Jean-Claude Bermond, Bi Li, Nicolas Nisse, Hervé Rivano, and Min-Li Yu. Data Gathering and Personalized Broadcasting in Radio Grids with Interferences. Technical Report RR-8218, INRIA, 2013.

[s-DDN14a] Gianlorenzo D'Angelo, Xavier Défago, and Nicolas Nisse. Stigmergy of anonymous agents in discrete environments. Technical report, 2014.

[s-GNPS13] Frédéric Giroire, Nicolas Nisse, Stéphane Pérennes, and Ronan P. Soares. Fractional combinatorial games. Technical Report RR-8371, INRIA, 2013.

[s-MNP14] Euripides Markou, Nicolas Nisse, and Stéphane Pérennes. Exclusive graph searching vs. pathwidth. Technical Report RR-8523, INRIA, 2014.

— Ph.D. Thesis —

[t-Nis07] Nicolas Nisse. *Les jeux des gendarmes et du voleur dans les graphes. Mineurs de graphes, stratégies connexes et approches distribuées.* PhD thesis, Université Paris-Sud 11, France, July 2007. URL: `http://hal.inria.fr/tel-00168818/`.

References

[ABB+07] Mark Anderson, Christian Barrientos, Robert C. Brigham, Julie R. Carrington, Richard P. Vitray, and Jay Yellen. Maximum demand graphs for eternal security. *J. Combin.Math.Combin.Comput.*, 61, 2007.

[ABBS01] Baruch Awerbuch, Petra Berenbrink, Andre Brinkmann, and Christian Scheideler. Simple routing strategies for adversarial systems. In *42th Annual Symposium on Foundations of Computer Science (FOCS)*, 2001.

[ABK+96] Yossi Azar, Andrei Z. Broder, Anna R. Karlin, Nathan Linial, and Steven Phillips. Biased random walks. *Combinatorica*, 16(1):1–18, 1996.

[ACG+11] Julio Araujo, Victor Campos, Frédéric Giroire, Leonardo Sampaio, and Ronan P. Soares. On the hull number of some graph classes. In *European Conference on Combinatorics, Graph Theory and Applications*, EuroComb'11, 2011.

[ACG12] Ittai Abraham, Shiri Chechik, and Cyril Gavoille. Fully dynamic approximate distance oracles for planar graphs via forbidden-set distance labels. In *44th Symposium on Theory of Computing Conference (STOC)*, pages 1199–1218. ACM, 2012.

[ACGP10] Ittai Abraham, Shiri Chechik, Cyril Gavoille, and David Peleg. Forbidden-set distance labels for graphs of bounded doubling dimension. In *29th Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pages 192–200. ACM, 2010.

[ACP87] Stefan Arnborg, Derek G. Corneil, and Andrzej Proskurowski. Complexity of finding embeddings in a k-tree. *SIAM J. Algebraic Discrete Methods*, 8(2):277–284, 1987.

[Adl07] Isolde Adler. Directed tree-width examples. *J. Comb. Theory, Ser. B*, 97(5):718–725, 2007.

[AEFP98] Yonatan Aumann, Oren Etzioni, Ronen Feldman, and Mike Perkowitz. Predicting event sequences: Data mining for prefetching web-pages, 1998.

[AF84] Martin Aigner and Michael Fromme. A game of cops and robbers. *Discrete Applied Mathematics*, 8:1–12, 1984.

[AF88] Richard P. Anstee and Martin Farber. On bridged graphs and cop-win graphs. *J. Comb. Theory, Ser. B*, 44(1):22–28, 1988.

[AFLN08] Marien Abreu, Martin Funk, Domenico Labbate, and Vito Napolitano. A family of regular graphs of girth 5. *Discrete Mathematics*, 308(10):1810–1815, 2008.

[AG06] Ittai Abraham and Cyril Gavoille. Object location using path separators. In *25th Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pages 188–197. ACM, 2006.

[AGG+13] Ittai Abraham, Cyril Gavoille, Anupam Gupta, Ofer Neiman, and Kunal Talwar. Cops, robbers, and threatening skeletons: Padded decomposition for minor-free graphs. Research report, 2013. URL: `http://arxiv.org/abs/1311.3048`.

[AGGM06] Ittai Abraham, Cyril Gavoille, Andrew V. Goldberg, and Dahlia Malkhi. Routing in networks with low doubling dimension. In *26th IEEE International Conference on Distributed Computing Systems (ICDCS)*, page 75. IEEE Computer Society, 2006.

[AGM+04] Ittai Abraham, Cyril Gavoille, Dahlia Malkhi, Noam Nisan, and Mikkel Thorup. Compact name-independent routing with minimum stretch. In *16th Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 20–24. ACM, 2004.

[AGM05] Ittai Abraham, Cyril Gavoille, and Dahlia Malkhi. Compact routing for graphs excluding a fixed minor. In *19th International Conference on Distributed Computing (DISC)*, volume 3724 of *Lecture Notes in Computer Science*, pages 442–456. Springer, 2005.

[AGM06a] Ittai Abraham, Cyril Gavoille, and Dahlia Malkhi. On space-stretch tradeoffs: lower bounds. In *18th Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 207–216. ACM, 2006.

[AGM06b] Ittai Abraham, Cyril Gavoille, and Dahlia Malkhi. On space-stretch tradeoffs: upper bounds. In *18th Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 217–224. ACM, 2006.

[AGM+08] Ittai Abraham, Cyril Gavoille, Dahlia Malkhi, Noam Nisan, and Mikkel Thorup. Compact name-independent routing with minimum stretch. *ACM Transactions on Algorithms*, 4(3), 2008.

[AGMW10] Ittai Abraham, Cyril Gavoille, Dahlia Malkhi, and Udi Wieder. Strong-diameter decompositions of minor free graphs. *Theory of Computing Systems*, 47(4):837–855, 2010.

[AGP+11] Christoph Ambühl, Leszek Gasieniec, Andrzej Pelc, Tomasz Radzik, and Xiaohui Zhang. Tree exploration with logarithmic memory. *ACM Trans. Algorithms*, 7(2):17:1–17:21, 2011.

[AIK84] Akeo Adachi, Shigeki Iwata, and Takumi Kasai. Some combinatorial game problems require omega(n^k) time. *J. ACM*, 31(2):361–376, 1984.

[AJB99] Réka Albert, Hawoong Jeong, and Albert-László Barabási. Internet: Diameter of the world-wide web. *Nature*, 401:130–131, 1999.

[Als04] Brian Alspach. Searching and sweeping graphs: a brief survey. In *Le Matematiche*, pages 5–37, 2004.

[AM05] Ittai Abraham and Dahlia Malkhi. Name independent routing for growth bounded networks. In *17th Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 49–55. ACM, 2005.

[AM11] Noga Alon and Abbas Mehrabian. On a generalization of meyniel's conjecture on the cops and robbers game. *Electr. J. Comb.*, 18(1), 2011.

[And84] Thomas Andreae. Note on a pursuit game played on graphs. *Discrete Applied Mathematics*, 9:111–115, 1984.

[And86] Thomas Andreae. On a pursuit game played on graphs for which a minor is excluded. *J. Comb. Theory, Ser. B*, 41(1):37–47, 1986.

[AP90] Baruch Awerbuch and David Peleg. Sparse partitions (extended abstract). In *31st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 503–513. IEEE Computer Society, 1990.

[AR10] Laurent Alonso and Edward M. Reingold. Bounds for cops and robber pursuit. *Comput. Geom.*, 43(9):749–766, 2010.

[ARS+03] Micah Adler, Harald Räcke, Naveen Sivadasan, Christian Sohler, and Berthold Vöcking. Randomized pursuit-evasion in graphs. *Combinatorics, Probability & Computing*, 12(3), 2003.

[AYZ95] Noga Alon, Raphael Yuster, and Uri Zwick. Color-coding. *J. ACM*, 42(4):844–856, 1995.

[AZN99] David Albrecht, Ingrid Zukerman, and Ann Nicholson. Pre-sending documents on the www: a comparative study. In *16th Int. Joint Conf. on Artificial Intelligence*, pages 1274–1279, 1999.

[BA99] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.

[Bak94] Brenda S. Baker. Approximation algorithms for NP-complete problems on planar graphs. *J. ACM*, 41(1):153–180, 1994.

[Bar06] János Barát. Directed path-width and monotonicity in digraph searching. *Graphs and Combinatorics*, 22(2):161–172, 2006.

[BB05] Emgad H. Bachoore and Hans L. Bodlaender. New upper bound heuristics for treewidth. In *4th InternationalWorkshop on Experimental and Efficient Algorithms (WEA)*, volume 3503 of *Lecture Notes in Computer Science*, pages 216–227. Springer, 2005.

[BB12] William Baird and Anthony Bonato. Meyniel's conjecture on the cop number: a survey. *Journal of Combinatorics*, 3:225–238, 2012.

[BBB+13] William Baird, Andrew Beveridge, Anthony Bonato, Paolo Codenotti, Aaron Maurer, John McCauley, and Silviya Valeva. On the minimum order of k-cop-win graphs. Research report, 2013. URL: `http://arxiv.org/abs/1308.2841`.

[BBH08] Shaunak Dattaprasad Bopardikar, Francesco Bullo, and João Pedro Hespanha. A pursuit game with range-only measurements. In *47th IEEE Conference on Decision and Control (CDC)*, pages 4233–4238. IEEE, 2008.

[BBMR08a] Roberto Baldoni, François Bonnet, Alessia Milani, and Michel Raynal. Anonymous graph exploration without collision by mobile robots. *Inf. Process. Lett.*, 109(2):98–103, 2008.

[BBMR08b] Roberto Baldoni, François Bonnet, Alessia Milani, and Michel Raynal. On the solvability of anonymous partial grids exploration by mobile robots. In *12th Int. Conf. On Principles Of Distributed Systems (OPODIS)*, volume 5401 of *Lecture Notes in Computer Science*, pages 428–445. Springer-Verlag, 2008.

[BBS05] P. Bertin, J-F. Bresse, and B. Le Sage. Accès haut débit en zone rurale: une solution "ad hoc". *France Telecom R&D*, 22:16–18, 2005.

[BCF+13] Anthony Bonato, Nancy E. Clarke, Stephen Finbow, Shannon Fitzpatrick, and Margaret-Ellen Messinger. A note on bounds for the cop number using tree decompositions. Research report, 2013. URL: `http://arxiv.org/abs/1308.2839`.

[BCG82] Elwyn R. Berlekamp, John H. Conway, and Richard K. Guy. Winning ways for your mathematical plays. *Games in Particular*, 2, 1982.

[BCG+04] A.P. Burger, E.J. Cockayne, W.R. Grundlingh, C.M. Mynhardt, J.H.van Vuuren, and W. Winterbach. Infinite order domination in graphs. *J. Combin.Math.Combin.Comput.*, 50, 2004.

[BCG+10] Evangelos Bampas, Jurek Czyzowicz, Leszek Gasieniec, David Ilcinkas, and Arnaud Labourel. Almost optimal asynchronous rendezvous in infinite multidimensional grids. In *24th Int. Symp. on Distributed Computing (DISC)*, volume 6343 of *Lecture Notes in Computer Science*, pages 297–311, 2010.

[BCH+09] G. Bosilca, C. Coti, T. Herault, P. Lemarinier, and J. Dongarra. Constructing resiliant communication infrastructure for runtime environments. In *International Conference in Parallel Computing*, 2009.

[BCKN13] Hans L. Bodlaender, Marek Cygan, Stefan Kratsch, and Jesper Nederlof. Deterministic single exponential time algorithms for connectivity problems parameterized by treewidth. In *40th International Colloquium on Automata, Languages, and Programming (ICALP (1))*, volume 7965 of *Lecture Notes in Computer Science*, pages 196–207. Springer, 2013.

[BCP10] Anthony Bonato, Ehsan Chiniforooshan, and Pawel Pralat. Cops and robbers from a distance. *Theoretical Computer Science*, 411(43):3834–3844, 2010.

[BCY09] Jean-Claude Bermond, Ricardo Correa, and Min-Li Yu. Optimal gathering protocols on paths under interference constraints. *Discrete Mathematics*, 309(18):5574–5587, 2009.

[BDD+13] Hans L. Bodlaender, Pål G. Drange, Markus S. Dregi, Fedor V. Fomin, Daniel Lokshtanov, and Michal Pilipczuk. A $o(c^k n)$ 5-approximation algorithm for treewidth. In *54th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2013.

[BDFH09] Hans L. Bodlaender, Rodney G. Downey, Michael R. Fellows, and Danny Hermelin. On problems without polynomial kernels. *J. Comput. Syst. Sci.*, 75(8):423–434, 2009.

[BDH+12] Dietmar Berwanger, Anuj Dawar, Paul Hunter, Stephan Kreutzer, and Jan Obdrzálek. The dag-width of directed graphs. *J. Comb. Theory, Ser. B*, 102(4):900–923, 2012.

[Bea04] Laurent Beaudou. *Le gendarme et le voleur, recherche d'intrus dans un graphe*. master thesis (in french), ENS Lyon, France, 2004.

[Ben01] Itai Benjamini. Survival of the weak in hyperbolic spaces, a remark on competition and geometry. *American Mathematical Society*, 130(3):723–726, 2001.

[BFFS02] Lali Barrière, Paola Flocchini, Pierre Fraigniaud, and Nicola Santoro. Capture of an intruder by mobile agents. In *14th ACM Symposium on Parallel Algorithms and Architectures (SPAA)*, pages 200–209, 2002.

[BFG+13] Anthony Bonato, Stephen Finbow, Przemyslaw Gordinowicz, Ali Haidar, William Kinnersley, Dieter Mitsche, Pawel Pralat, and Ladislav Stacho. The robber strikes back. Research report, 2013. URL: `http://arxiv.org/abs/1308.2843`.

[BFL+09] Hans L. Bodlaender, Fedor V. Fomin, Daniel Lokshtanov, Eelko Penninkx, Saket Saurabh, and Dimitrios M. Thilikos. (meta) kernelization. In *50th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 629–638. IEEE Computer Society, 2009.

[BFST03] Lali Barrière, Pierre Fraigniaud, Nicola Santoro, and Dimitrios M. Thilikos. Searching is not jumping. In *29th International Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, pages 34–45, 2003.

[BGHK09] Anthony Bonato, Petr A. Golovach, Gena Hahn, and Jan Kratochvíl. The capture time of a graph. *Discrete Mathematics*, 309(18):5588–5595, 2009.

[BGK+06] Jean-Claude Bermond, Jérome Galtier, Ralf Klasing, Nelson Morales, and Stéphane Pérennes. Hardness and approximation of gathering in static radio networks. *Parallel Processing Letters*, 16(2):165–183, 2006.

[BGKP13] Anthony Bonato, Przemyslaw Gordinowicz, Bill Kinnersley, and Pawel Pralat. The capture time of the hypercube. *Electr. J. Comb.*, 20(2):P24, 2013.

[BGP+11] Jean-Claude Bermond, Luisa Gargano, Stéphane Pérennes, Adele A. Rescigno, and Ugo Vaccaro. Optimal time data gathering in wireless networks with omni-directional antennas. In *18th International Colloquium on Structural Information and Communication Complexity (SIROCCO)*, volume 6796 of *Lecture Notes in Computer Science*, pages 306–317. Springer-Verlag, 2011.

[BGR10] Jean-Claude Bermond, Luisa Gargano, and Adele A. Rescigno. Gathering with minimum completion time in sensor tree networks. *Journal of interconnection networks*, 11(1-2):1–33, 2010.

[BH03] Virgil Bourassa and Fred B. Holt. Swan: Small-world wide area networks. In *International Conference on Advances in Infrastructure*, 2003.

[BH06] Franz-Josef Brandenburg and Stephanie Herrmann. Graph searching and search time. In *32nd Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM)*, volume 3831 of *Lecture Notes in Computer Science*, pages 197–206. Springer, 2006.

[BHT10] Anthony Bonato, Gena Hahn, and Claude Tardif. Large classes of infinite k-cop-win graphs. *Journal of Graph Theory*, 65(4):334–342, 2010.

[BHW07] Anthony Bonato, Gena Hahn, and Changping Wang. The cop density of a graph. *Contributions to Discrete Mathematics*, 2(2), 2007.

[BI93] Alessandro Berarducci and Benedetto Intrigila. On the cop number of a graph. *Adv. in Applied Math.*, 14:389–403, 1993.

[Bie91] Daniel Bienstock. Graph searching, path-width, tree-width and related problems (a survey). *DIMACS Ser. in Discrete Mathematics and Theoretical Computer Science*, 5:33–49, 1991.

[BK96] Hans L. Bodlaender and Ton Kloks. Efficient and constructive algorithms for the pathwidth and treewidth of graphs. *J. Algorithms*, 21(2):358–402, 1996.

[BK07] Hans L. Bodlaender and Arie M. C. A. Koster. On the maximum cardinality search lower bound for treewidth. *Discrete Applied Mathematics*, 155(11):1348–1372, 2007.

[BK10] Hans L. Bodlaender and Arie M. C. A. Koster. Treewidth computations i. upper bounds. *Information and Computation*, 208(3):259–275, 2010.

[BK11] Hans L. Bodlaender and Arie M. C. A. Koster. Treewidth computations ii. lower bounds. *Information and Computation*, 209(7):1103–1119, 2011.

[BKIS12] Deepak Bhadauria, Kyle Klein, Volkan Isler, and Subhash Suri. Capturing an evader in polygonal environments with obstacles: The full visibility case. *I. J. Robotic Res.*, 31(10):1176–1189, 2012.

[BKK+10] Vincenzo Bonifaci, Ralf Klasing, Peter Korteweg, Alberto Marchetti-Spaccamela, and Leen Stougie. Data gathering in wireless networks. In A.Koster and X. Munoz, editors, *Graphs and Algorithms in Communication Networks*, pages 357–377. Springer Monograph, 2010.

[BKL95] László Babai, Peter G. Kimmel, and Satyanarayana V. Lokam. Simultaneous messages vs. communication. In *12th International Symposium on Theoretical Aspects of Computer Science (STACS)*, pages 361–372, 1995.

[BKL13] Béla Bollobás, Gábor Kun, and Imre Leader. Cops and robbers in a random graph. *J. Comb. Theory, Ser. B*, 103(2):226–236, 2013.

[BKP12] Anthony Bonato, Graeme Kemkes, and Pawel Pralat. Almost all cop-win graphs contain a universal vertex. *Discrete Mathematics*, 312(10):1652–1657, 2012.

[BL06] Béla Bollobás and Imre Leader. The angel and the devil in three dimensions. *J. Comb. Theory, Ser. A*, 113(1):176–184, 2006.

[BLW09] Béla Bollobas, Imre Leader, and M. Walters. Lion and man – can both win? Research report, 2009. URL: `http://arxiv.org/abs/0909.2524`.

[BM08] Adrian Bondy and U.S.R. Murty. *Graph Theory*. Springer, 2008.

[BMM10] J. R. S. Blair, Frederik Manne, and Rodica Mihai. Efficient self-stabilizing graph searching in tree networks. In *12th Int. Symp. on Stabilization, Safety, and Security of Distributed Systems (SSS)*, volume 6366 of *Lecture Notes in Computer Science*, pages 111–125. Springer, 2010.

[BMPBT10] Lélia Blin, Alessia Milani, Maria Potop-Butucaru, and Sébastien Tixeuil. Exclusive perpetual ring exploration without chirality. In *24th Int. Symp. on Distributed Computing (DISC)*, volume 6343 of *Lecture Notes in Computer Science*, pages 312–327, 2010.

[BMPBT11] François Bonnet, Alessia Milani, Maria Potop-Butucaru, and Sébastien Tixeuil. Asynchronous exclusive perpetual grid exploration without sense of direction. In *15th Int. Conf. On Principles Of Distributed Systems (OPODIS)*, volume 7109 of *Lecture Notes in Computer Science*, pages 251–265. Springer, 2011.

[BN11] Anthony Bonato and Richard J. Nowakowski. *The game of Cops and Robber on Graphs*. American Math. Soc., 2011.

[BO95] Luitpold Babel and Stephan Olariu. On the isomorphism of graphs with few p4s. In *21st International Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, pages 24–36. Springer-Verlag, 1995.

[BO99a] Luitpold Babel and Stephan Olariu. On the p-connectedness of graphs - a survey. *Discrete Applied Mathematics*, 95(1-3):11 – 33, 1999.

[BO99b] Tamer Basar and Geert Jan Olsder. *Dynamic Noncooperative Game Theory*. Society for Industrial and Applied Mathematics, 1999.

[Bod96] Hans L. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM J. Comput.*, 25(6):1305–1317, 1996.

[Bod98] Hans L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theoretical Computer Science*, 209(1-2):1–45, 1998.

[Bod07] Hans L. Bodlaender. Treewidth: Structure and algorithms. In *14th International Colloquium on Structural Information and Communication Complexity (SIROCCO)*, volume 4474 of *Lecture Notes in Computer Science*, pages 11–25. Springer, 2007.

[Bod09] Hans L. Bodlaender. Kernelization: New upper and lower bound techniques. In *4th International Workshop on Parameterized and Exact Computation (IWPEC)*, volume 5917 of *Lecture Notes in Computer Science*, pages 17–37. Springer, 2009.

[BP12] Jean-Claude Bermond and Joseph Peters. Optimal gathering in radio grids with interference. *Theoretical Computer Science*, 457:10–26, 2012.

[BPW07] Anthony Bonato, Pawel Pralat, and Changping Wang. Pursuit-evasion in models of complex networks. *Internet Mathematics*, 4(4):419–436, 2007.

[Bre67] Richard L. Breish. An intuitive approach to speleotopology. *Southwestern Cavers*, 6:72–78, 1967.

[Bre12] Richard L. Breish. *Lost in a Cave: applying graph theory to cave exploration.* Greyhound press, 2012.

[BRST91] Daniel Bienstock, Neil Robertson, Paul D. Seymour, and Robin Thomas. Quickly excluding a forest. *J. Comb. Theory, Ser. B*, 52(2):274–283, 1991.

[BS91] Daniel Bienstock and Paul D. Seymour. Monotonicity in graph searching. *J. Algorithms*, 12(2):239–245, 1991.

[BT97a] Hans L. Bodlaender and Dimitrios M. Thilikos. Constructive linear time algorithms for branchwidth. In *24th Int. Coll. on Automata, Languages and Programming (ICALP)*, volume 1256 of *Lecture Notes in Computer Science*, pages 627–637. Springer, 1997.

[BT97b] Hans L. Bodlaender and Dimitrios M. Thilikos. Treewidth for graphs with small chordality. *Discrete Applied Mathematics*, 79(1-3):45–61, 1997.

[BT04] Hans L. Bodlaender and Dimitrios M. Thilikos. Computing small search numbers in linear time. In *1st International Workshop on Parameterized and Exact Computation (IWPEC)*, volume 3162 of *Lecture Notes in Computer Science*, pages 37–48, 2004.

[BTK11] Richard B. Borie, Craig A. Tovey, and Sven Koenig. Algorithms and complexity results for graph-based pursuit evasion. *Auton. Robots*, 31(4):317–332, 2011.

[BY10] Jean-Claude Bermond and Min-Li Yu. Optimal gathering algorithms in multi-hop radio tree networks with interferences. *Ad Hoc and Sensor Wireless Networks*, 9(1-2):109–128, 2010.

[BY11] Anthony Bonato and Boting Yang. Graph searching and related problems. *invited book chapter in: Handbook of Combinatorial Optimization*, 2011.

[CBL07] Xiaowen Chu, Tianming Bu, and X.-Y. Li. A study of lightpath rerouting schemes in wavelength-routed WDM networks. In *IEEE Intl. Conference on Communications (ICC)*, pages 2400–2405. IEEE, 2007.

[CC06] Nancy E. Clarke and Emma L. Connon. Cops, robber, and alarms. *Ars Comb.*, 81, 2006.

[CC13] Chandra Chekuri and Julia Chuzhoy. Polynomial bounds for the grid-minor theorem. Research report, 2013. URL: `http://arxiv.org/abs/1305.6577`.

[CCC06] Leizhen Cai, Siu Man Chan, and Siu On Chan. Random separation: A new method for solving fixed-cardinality optimization problems. In *Second International Workshop on Parameterized and Exact Computation (IWPEC)*, volume 4169 of *Lecture Notes in Computer Science*, pages 239–250. Springer, 2006.

[CCJ04] Gilbert B. Cagaanan, Sergio R. Canoy, and Jr. On the hull sets and hull number of the cartesian product of graphs. *Discrete Mathematics*, 287(1-3):141 – 144, 2004.

[CCPP13] Jérémie Chalopin, Victor Chepoi, Panos Papasoglu, and Timothée Pecatte. Cop and robber game and hyperbolicity. Research report, 2013. URL: `http://arxiv.org/abs/1308.3987`.

[CDG07] Colin Cooper, Martin E. Dyer, and Catherine S. Greenhill. Sampling regular graphs and a peer-to-peer network. *Combinatorics, Probability & Computing*, 16(4):557–593, 2007.

[CDK+03] Miguel Castro, Peter Druschel, Anne-Marie Kermarrec, Animesh Nandi, Antony Rowstron, and Atul Singh. SplitStream: high-bandwidth multicast in cooperative environments. In *nineteenth ACM symposium on Operating systems principles*, page 313, 2003.

[CDPT08] Eddy Caron, Ajoy K. Datta, Franck Petit, and Cédric Tedeschi. Self-stabilization in tree-structured peer-to-peer service discovery systems. In *IEEE Symposium on Reliable Distributed Systems*, pages 207–216, 2008.

[CF05] Colin Cooper and Alan Frieze. The cover time of random regular graphs. *SIAM J. Discret. Math.*, 18(4):728–740, 2005.

[CF07] Yijia Chen and Jörg Flum. On parameterized path and chordless path problems. In *CCC*, pages 250–263, 2007.

[CFI+08] Reuven Cohen, Pierre Fraigniaud, David Ilcinkas, Amos Korman, and David Peleg. Label-guided graph exploration by a finite automaton. *ACM Trans. Algorithms*, 4(4):42:1–42:18, 2008.

[CFJT12] Nancy E. Clarke, S. Fiorini, G. Joret, and Dirk O. Theis. A little note on the cops and robber game on graphs embedded in non-orientable surfaces. *Graphs and Combinatorics*, pages 1–6, 2012. URL: `http://aps.arxiv.org/abs/math/0803.0538v2`.

[CFMS10] Jérémie Chalopin, Paola Flocchini, Bernard Mans, and Nicola Santoro. Network exploration by silent and oblivious robots. In *36th International Workshop on Graph Theoretic Concepts in Computer Science (WG)*, volume 6410 of *Lecture Notes in Computer Science*, pages 208–219, 2010.

[CFZ03] Gary Chartrand, John Frederick Fink, and Ping Zhang. The hull number of an oriented graph. *International Journal of Mathematics and Mathematical Sciences*, 2003(36):2265–2275, 2003.

[CGMZ05] Hubert T.-H. Chan, Anupam Gupta, Bruce M. Maggs, and Shuheng Zhou. On hierarchical routing in doubling metrics. In *16th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 762–771. SIAM, 2005.

[CGP09] Jurek Czyzowicz, Leszek Gasieniec, and Andrzej Pelc. Gathering few fat mobile robots in the plane. *Theoretical Computer Science*, 410(6–7):481 – 499, 2009.

[Che97] Victor Chepoi. Bridged graphs are cop-win graphs: An algorithmic proof. *J. Comb. Theory, Ser. B*, 69(1):97–100, 1997.

[Chi08] Ehsan Chiniforooshan. A better bound for the cop number of general graphs. *Journal of Graph Theory*, 58(1):45–48, 2008.

[CHI11] Timothy H. Chung, Geoffrey A. Hollinger, and Volkan Isler. Search and pursuit-evasion in mobile robotics - a survey. *Auton. Robots*, 31(4):299–316, 2011.

[CHM+10] Jose Cáceres, Carmen Hernando, Merce Mora, Ignacio M. Pelayo, and Maria L. Puertas. On the geodetic and the hull numbers in strong product graphs. *Computers & Mathematics with Applications*, 60(11):3020 – 3031, 2010.

[CHM12] David Coudert, Florian Huc, and Dorian Mazauric. A distributed algorithm for computing the node search number in trees. *Algorithmica*, 63(1-2):158–190, 2012.

[CHS07] David Coudert, Florian Huc, and Jean-Sébastien Sereni. Pathwidth of outerplanar graphs. *Journal of Graph Theory*, 55(1):27–41, 2007.

[CHZ02] Gary Chartrand, Frank Harary, and Ping Zhang. On the geodetic number of a graph. *Networks*, 39(1):1–6, 2002.

[cis12] Networking solutions white paper, 2012. URL: `http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481374_ns827_Networking_Solutions_White_Paper.html`.

[CKS81] Ashok K. Chandra, Dexter Kozen, and Larry J. Stockmeyer. Alternation. *J. ACM*, 28(1):114–133, 1981.

[Cla09] Nancy E. Clarke. A witness version of the cops and robber game. *Discrete Mathematics*, 309(10):3292–3298, 2009.

[CLP00] Marc Chastand, François Laviolette, and Norbert Polat. On constructible graphs, infinite bridged graphs and weakly cop-win graphs. *Discrete Mathematics*, 224(1-3):61–78, 2000.

[CLRS01] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to algorithms*. MIT press, Cambridge, 2001.

[CM99] Manoj Changat and Joseph Mathew. On triangle path convexity in graphs. *Discrete Mathematics*, 206(1-3):91 – 95, 1999.

[CM12] Nancy E. Clarke and Gary MacGillivray. Characterizations of k-copwin graphs. *Discrete Mathematics*, 312(8):1421–1425, 2012.

[CMS05] Manoj Changat, Henry Martyn Mulder, and Gerard Sierksma. Convexities related to path properties on graphs. *Discrete Mathematics*, 290(2-3):117 – 131, 2005.

[CMT09] Mathieu Chapelle, Frédéric Mazoit, and Ioan Todinca. Constructing brambles. In *34th International Symposium on Mathematical Foundations of Computer Science (MFCS)*, volume 5734 of *Lecture Notes in Computer Science*, pages 223–234. Springer, 2009.

[CN00] Nancy E. Clarke and Richard J. Nowakowski. Cops, robber, and photo radar. *Ars Comb.*, 56, 2000.

[CN01] Nancy E. Clarke and Richard J. Nowakowski. Cops, robber, and traps. *Utilitas Mathematica*, 60:91–98, 2001.

[CN05] Nancy E. Clarke and Richard J. Nowakowski. Tandem-win graphs. *Discrete Mathematics*, 299(1-3):56–64, 2005.

[CNP+11] Marek Cygan, Jesper Nederlof, Marcin Pilipczuk, Michal Pilipczuk, Johan M. M. van Rooij, and Jakub Onufry Wojtaszczyk. Solving connectivity problems parameterized by treewidth in single exponential time. In *IEEE 52nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 150–159. IEEE, 2011.

[Con96] John H. Conway. The angel problem. *Games of No Chance*, 29:3–12, 1996.

[Cou90] Bruno Courcelle. The monadic second-order logic of graphs. i. recognizable sets of finite graphs. *Information and Computation*, 85(1):12–75, 1990.

[Cou10a] David Coudert. *Algorithmique et optimisation dans les réseaux de télécommunications*. Habilitation à diriger des recherches, Université Nice Sophia-Antipolis (UNS), March 2010. URL: `http://tel.archives-ouvertes.fr/tel-00466400/`.

[Cou10b] Bruno Courcelle. Special tree-width and the verification of monadic second-order graph properties. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, volume 8 of *LIPIcs*, pages 13–29, 2010.

[CPPS05] David Coudert, Stéphane Pérennes, Quang-Cuong Pham, and Jean-Sébastien Sereni. Rerouting requests in WDM networks. In *7me Rencontres Francophones sur les Aspects Algorithmiques des Télécommunications (AlgoTel)*, pages 17–20, 2005.

[CS76] Ashok K. Chandra and Larry J. Stockmeyer. Alternation. In *17th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 98–108. IEEE, 1976.

[CS11] David Coudert and Jean-Sébastien Sereni. Characterization of graphs and digraphs with small process numbers. *Discrete Applied Mathematics*, 159(11):1094–1109, 2011.

[CSTW12] Wei Chen, Christian Sommer, Shang-Hua Teng, and Yajun Wang. A compact routing scheme and approximate distance oracle for power-law graphs. *ACM Transactions on Algorithms*, 9(1):4, 2012.

[CT07] Bruno Courcelle and Andrew Twigg. Compact forbidden-set routing. In *24th Annual Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 4393 of *Lecture Notes in Computer Science*, pages 37–48. Springer, 2007.

[CWZ02] Gary Chartrand, Curtiss E. Wall, and Ping Zhang. The convexity number of a graph. *Graphs and Combinatorics*, 18:209–217, 2002.

[DDKN12] Gianlorenzo D'Angelo, Gabriele Di Stefano, Ralf Klasing, and Alfredo Navarra. Gathering of robots on anonymous grids without multiplicity detection. In *19th Int. Coll. on Structural Information and Communication Complexity (SIROCCO)*, volume 7355 of *Lecture Notes in Computer Science*, pages 327–338, 2012.

[DDN11] Gianlorenzo D'Angelo, Gabriele Di Stefano, and Alfredo Navarra. Gathering of six robots on anonymous symmetric rings. In *18th Int. Coll. on Structural Inf. and Com. Complexity (SIROCCO)*, volume 6796 of *Lecture Notes in Computer Science*, pages 174–185, 2011.

[DDN12] Gianlorenzo D'Angelo, Gabriele Di Stefano, and Alfredo Navarra. How to gather asynchronous oblivious robots on anonymous rings. In *26th Int. Symp. on Distributed Computing (DISC)*, volume 7611 of *Lecture Notes in Computer Science*, pages 330–344, 2012.

[DDN13a] Gianlorenzo D'Angelo, Gabriele Di Stefano, and Alfredo Navarra. Gathering asynchronous and oblivious robots on basic graph topologies under the look-compute-move model. In S. Alpern, R. Fokkink, L. Gasieniec, R. Lindelauf, and V.S. Subrahmanian, editors, *Search Theory: A Game Theoretic Perspective*, pages 197–222. Springer, 2013.

[DDTY13] Dariusz Dereniowski, Danny Dyer, Ryan M. Tifenbach, and Boting Yang. Zero-visibility cops and robber game on a graph. In *Third Joint International Conference on Frontiers in Algorithmics and Algorithmic Aspects in Information and Management (FAW-AAIM)*, volume 7924 of *Lecture Notes in Computer Science*, pages 175–186. Springer, 2013.

[DdWMP02] Marc Demange, Dominique de Werra, Jérôme Monnot, and Vangelis Th. Paschos. Weighted node coloring: When stable sets are expensive. In *28th Int. Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, volume 2573 of *Lecture Notes in Computer Science*, pages 114–125. Springer, 2002.

[Der11] Dariusz Dereniowski. Connected searching of weighted trees. *Theoretical Computer Science*, 412(41):5700–5713, 2011.

[Der12a] Dariusz Dereniowski. Approximate search strategies for weighted trees. *Theoretical Computer Science*, 463:96–113, 2012.

[Der12b] Dariusz Dereniowski. From pathwidth to connected pathwidth. *SIAM J. Discrete Math.*, 26(4):1709–1732, 2012.

[DF99] Rod G. Downey and Michael R. Fellows. *Parameterized Complexity.* Springer Verlag, 1999.

[DFC07] György Dan, Viktória Fodor, and Ilias Chatzidrossos. On the performance of multiple-tree-based peer-to-peer live streaming. In *26th IEEE International Conference on Computer Communications*, pages 2556–2560, 2007.

[DFHT05] Erik D. Demaine, Fedor V. Fomin, Mohammad Taghi Hajiaghayi, and Dimitrios M. Thilikos. Subexponential parameterized algorithms on bounded-genus graphs and h-minor-free graphs. *J. ACM*, 52(6):866–893, 2005.

[DFT06] Frederic Dorn, Fedor V. Fomin, and Dimitrios M. Thilikos. Fast subexponential algorithm for non-local problems on graphs of bounded genus. In *10th ScandinavianWorkshop on Algorithm Theory (SWAT)*, volume 4059 of *Lecture Notes in Computer Science*, pages 172–183. Springer, 2006.

[DFT12] Frederic Dorn, Fedor V. Fomin, and Dimitrios M. Thilikos. Catalan structures and dynamic programming in H-minor-free graphs. *J. Comput. Syst. Sci.*, 78(5):1606–1622, 2012.

[DG02] Yon Dourisboure and Cyril Gavoille. Improved compact routing scheme for chordal graphs. In *16th International Conference on Distributed Computing (DISC)*, volume 2508 of *Lecture Notes in Computer Science*, pages 252–264. Springer, 2002.

[DG07] Yon Dourisboure and Cyril Gavoille. Tree-decompositions with bags of small diameter. *Discrete Mathematics*, 307(16):2008–2029, 2007.

[DGK+09] Mitre C. Dourado, John G. Gimbel, Jan Kratochvíl, Fábio Protti, and Jayme L. Szwarcfiter. On the computation of the hull number of a graph. *Discrete Mathematics*, 309(18):5668 – 5674, 2009. Combinatorics 2006, A Meeting in Celebration of Pavol Hell's 60th Birthday (May 1-5, 2006).

[DH08a] Erik D. Demaine and MohammadTaghi Hajiaghayi. The bidimensionality theory and its algorithmic applications. *Comput. J.*, 51(3):292–302, 2008.

[DH08b] Erik D. Demaine and MohammadTaghi Hajiaghayi. Linearity of grid minors in treewidth with applications through bidimensionality. *Combinatorica*, 28(1):19–36, 2008.

[DHLS06] Philippe Duchon, Nicolas Hanusse, Emmanuelle Lebhar, and Nicolas Schabanel. Could any graph be turned into a small-world? *Theoretical Computer Science*, 355(1):96–103, 2006.

[DHT05] Erik D. Demaine, Mohammad Taghi Hajiaghayi, and Dimitrios M. Thilikos. Exponential speedup of fixed-parameter algorithms for classes of graphs excluding single-crossing graphs as minors. *Algorithmica*, 41(4):245–267, 2005.

[DJS13] Yessine Daadaa, Asif Jamshed, and Mudassir Shabbir. Network decontamination with a single agent. Research report, 2013. URL: `http://arxiv.org/abs/1307.7307`.

[DKL87] Narsingh Deo, Mukkai S. Krishnamoorthy, and Michael A. Langston. Exact and approximate solutions for the gate matrix layout problem. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 6(1):79–84, 1987.

[DKZ13] Dariusz Dereniowski, Wieslaw Kubiak, and Yori Zwols. Minimum length path decompositions. Technical report, 2013. URL: `abs/1302.2788`.

[DN13] Gabriele Di Stefano and Alfredo Navarra. Optimal gathering of oblivious robots in anonymous graphs. In *20th International Colloquium on Structural Information and Communication Complexity (SIROCCO)*, Lecture Notes in Computer Science. Springer, 2013.

[Dou05] Yon Dourisboure. Compact routing schemes for generalised chordal graphs. *Journal of Graph Algorithms and Applications*, 9(2):277–297, 2005.

[DPBF10] Frederic Dorn, Eelko Penninkx, Hans L. Bodlaender, and Fedor V. Fomin. Efficient exact algorithms on planar graphs: Exploiting sphere cut decompositions. *Algorithmica*, 58(3):790–810, 2010.

[DPRS10] Mitre C. Dourado, Fábio Protti, Dieter Rautenbach, and Jayme L. Szwarcfiter. On the hull number of triangle-free graphs. *SIAM J. Discret. Math.*, 23:2163–2172, 2010.

[DPS10] Mitre C. Dourado, Fábio Protti, and Jayme L. Szwarcfiter. Complexity results related to monophonic convexity. *Discrete Applied Mathematics*, 158(12):1268 – 1274, 2010.

[DPT09] Stéphane Devismes, Franck Petit, and Sébastien Tixeuil. Optimal probabilistic ring exploration by semi-synchronous oblivious robots. In *16th Int. Coll. on Structural Information and Communication Complexity (SIROCCO)*, volume 5869 of *Lecture Notes in Computer Science*, pages 195–208, 2009.

[Dra05] Feodor F. Dragan. Estimating all pairs shortest paths in restricted graph families: a unified approach. *Journal of Algorithms*, 57(1):1–21, 2005.

[dWDE+09] Dominique de Werra, Marc Demange, Bruno Escoffier, Jérôme Monnot, and Vangelis Th. Paschos. Weighted coloring on planar, bipartite and split graphs: Complexity and approximation. *Discrete Applied Mathematics*, 157(4):819–832, 2009.

[EHS13] William Evans, Paul Hunter, and Mohammad Ali Safari. D-width and cops and robbers. Research report, 2013. unpublished.

[EL12] Leah Epstein and Asaf Levin. On the max coloring problem. *Theoretical Computer Science*, 462:23–38, 2012.

[EM04] John A. Ellis and Minko Markov. Computing the vertex separation of unicyclic graphs. *Information and Computation*, 192(2):123–161, 2004.

[EMP06] Bruno Escoffier, Jérôme Monnot, and Vangelis Th. Paschos. Weighted coloring: further complexity and approximability results. *Inf. Process. Lett.*, 97(3):98–103, 2006.

[Epp99] David Eppstein. Subgraph isomorphism in planar graphs and related problems. *J. Graph Algorithms Appl.*, 3(3), 1999.

[ES85] Martin G. Everett and Stephen B. Seidman. The hull number of a graph. *Discrete Mathematics*, 57(3):217 – 223, 1985.

[EST94] John A. Ellis, Ivan H. Sudborough, and Jonathan S. Turner. The vertex separation and search number of a graph. *Information and Computation*, 113(1):50–79, 1994.

[Far87] Martin Farber. Bridged graphs and geodesic convexity. *Discrete Applied Mathematics*, 66:249–257, 1987.

[Far05] Alastair Farrugia. Orientable convexity, geodetic and hull numbers in graphs. *Discrete Appl. Math.*, 148:256–262, 2005.

[FCLJ99] Li Fan, Pei Cao, Wei Lin, and Quinn Jacobson. Web prefetching between low-bandwidth clients and proxies: potential and performance. In *ACM SIGMETRICS Int. Conf. on Measurement and Modeling of Computer Systems*, pages 178–187, 1999.

[FFF99] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. On power-law relationships of the internet topology. In *annual conference of the Special Interest Group on Data Communication (SIGCOMM)*, pages 251–262. ACM, 1999.

[FFM04] Cédric Florens, Massimo Franceschetti, and Robert J. McEliece. Lower bounds on data collection time in sensory networks. *IEEE Journal on Selected Areas in Communications*, 22(6):1110–1120, 2004.

[FG87] Aviezri S. Fraenkel and Elisheva Goldschmidt. Pspace-hardness of some combinatorial games. *J. Comb. Theory, Ser. A*, 46(1):21–38, 1987.

[FG97] Pierre Fraigniaud and Cyril Gavoille. Universal routing schemes. *Distributed Computing*, 10(2):65–78, 1997.

[FG01] Pierre Fraigniaud and Cyril Gavoille. Routing in trees. In *28th International Colloquium on Automata, Languages and Programming (ICALP)*, volume 2076 of *Lecture Notes in Computer Science*, pages 757–772. Springer, 2001.

[FG06] Jörg Flum and Martin Grohe. *Parameterized Complexity Theory*. Springer, 2006.

[FG10] Pierre Fraigniaud and George Giakkoupis. On the searchability of small-world networks with arbitrary underlying structure. In *42nd ACM Symposium on Theory of Computing (STOC)*, pages 389–398. ACM, 2010.

[FGG+10] Fedor V. Fomin, Serge Gaspers, Petr A. Golovach, Dieter Kratsch, and Saket Saurabh. Parameterized algorithm for eternal vertex cover. *Inf. Process. Lett.*, 110(16):702–706, 2010.

[FGH+08] Fedor V. Fomin, Petr A. Golovach, Alexander Hall, Matús Mihalák, Elias Vicari, and Peter Widmayer. How to guard a graph? In *19th International Symposium on Algorithms and Computation (ISAAC)*, volume 5369 of *Lecture Notes in Computer Science*, pages 318–329. Springer, 2008.

[FGH+11] Fedor V. Fomin, Petr A. Golovach, Alexander Hall, Matús Mihalák, Elias Vicari, and Peter Widmayer. How to guard a graph? *Algorithmica*, 61(4):839–856, 2011.

[FGK08] Fedor V. Fomin, Petr A. Golovach, and Jan Kratochvíl. On tractability of cops and robbers game. In *Fifth IFIP International Conference On Theoretical Computer Science (TCS)*, volume 273 of *IFIP*, pages 171–185. Springer, 2008.

[FGL09] Fedor V. Fomin, Petr A. Golovach, and Daniel Lokshtanov. Guard games on graphs: Keep the intruder out! In *7th International Workshop on Approximation and Online Algorithms (WAOA)*, volume 5893 of *Lecture Notes in Computer Science*, pages 147–158. Springer, 2009.

[FGL10] Fedor V. Fomin, Petr A. Golovach, and Daniel Lokshtanov. Cops and robber game without recharging. In *12th Scandinavian Symposium and Workshops on Algorithm Theory (SWAT)*, volume 6139 of *Lecture Notes in Computer Science*, pages 273–284. Springer, 2010.

[FGL12] Fedor V. Fomin, Petr A. Golovach, and Daniel Lokshtanov. Cops and robber game without recharging. *Theory Comput. Syst.*, 50(4):611–620, 2012.

[FHL05a] Uriel Feige, Mohammad Taghi Hajiaghayi, and James R. Lee. Improved approximation algorithms for minimum-weight vertex separators. In *37th Annual ACM Symposium on Theory of Computing (STOC)*, pages 563–572. ACM, 2005.

[FHL05b] Paola Flocchini, Miao Jun Huang, and Flaminia L. Luccio. Contiguous search in the hypercube for capturing an intruder. In *19th IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 2005.

[FHL06] Paola Flocchini, Miao Jun Huang, and Flaminia L. Luccio. Decontamination of chordal rings and tori. In *20th IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, 2006.

[FHL07] Paola Flocchini, Miao Jun Huang, and Flaminia L. Luccio. Decontaminating chordal rings and tori using mobile agents. *Int. J. Found. Comput. Sci.*, 18(3):547–563, 2007.

[FHL08] Paola Flocchini, Miao Jun Huang, and Flaminia L. Luccio. Decontamination of hypercubes by mobile agents. *Networks*, 52(3):167–178, 2008.

[FHM10a] Fedor V. Fomin, Pinar Heggernes, and Rodica Mihai. Mixed search number and linear-width of interval and split graphs. *Networks*, 56(3):207–214, 2010.

[FHM10b] Fedor V. Fomin, Pinar Heggernes, and Rodica Mihai. Mixed search number and linear-width of interval and split graphs. *Networks*, 56(3):207–214, 2010.

[FHW12] Silvio Frischknecht, Stephan Holzer, and Roger Wattenhofer. Networks cannot compute their diameter in sublinear time. In *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1150–1162. SIAM, 2012.

[fIDAC] The Cooperative Association for Internet Data Analysis (CAIDA). The CAIDA AS relationships dataset. URL: `http://www.caida.org/data/active/as-relationships/`.

[FIP10] Pierre Fraigniaud, David Ilcinkas, and Andrzej Pelc. Communication algorithms with advice. *J. Comput. Syst. Sci.*, 76(3-4):222–232, 2010.

[FIPS07] Paola Flocchini, David Ilcinkas, Andrzej Pelc, and Nicola Santoro. Computing without communicating: Ring exploration by asynchronous oblivious robots. In *11th Int. Conf. on Princ. of Dist. Syst. (OPODIS)*, volume 4878 of *Lecture Notes in Computer Science*, pages 105–118. Springer, 2007.

[FIPS10] Paola Flocchini, David Ilcinkas, Andrzej Pelc, and Nicola Santoro. Remembering without memory: Tree exploration by asynchronous oblivious robots. *Theoretical Computer Science*, 411(14-15):1583–1598, 2010.

[FIPS11] Paola Flocchini, David Ilcinkas, Andrzej Pelc, and Nicola Santoro. How many oblivious robots can explore a line. *Inf. Process. Lett.*, 111(20):1027–1031, 2011.

[FIPS13] Paola Flocchini, David Ilcinkas, Andrzej Pelc, and Nicola Santoro. Computing without communicating: Ring exploration by asynchronous oblivious robots. *Algorithmica*, 65:562–583, 2013.

[FJ86] Martin Farber and Robert E Jamison. Convexity in graphs and hypergraphs. *SIAM J. Algebraic Discrete Methods*, 7:433–444, 1986.

[FKL12] Alan M. Frieze, Michael Krivelevich, and Po-Shen Loh. Variations on cops and robbers. *Journal of Graph Theory*, 69(4):383–402, 2012.

[FLL06] Pierre Fraigniaud, Emmanuelle Lebhar, and Zvi Lotker. Brief announcement: On augmented graph navigability. In *20th International Symposium on Distributed Computing (DISC)*, volume 4167 of *Lecture Notes in Computer Science*, pages 551–553. Springer, 2006.

[FLL10] Pierre Fraigniaud, Emmanuelle Lebhar, and Zvi Lotker. Recovering the long-range links in augmented graphs. *Theoretical Computer Science*, 411(14-15):1613–1625, 2010.

[FLMS12] Fedor V. Fomin, Daniel Lokshtanov, Neeldhara Misra, and Saket Saurabh. Planar f-deletion: Approximation, kernelization and optimal fpt algorithms. In *53rd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 470–479. IEEE Computer Society, 2012.

[FLS05] Paola Flocchini, Flaminia L. Luccio, and Lisa Xiuli Song. Size optimal strategies for capturing an intruder in mesh networks. In *2005 International Conference on Communications in Computing (CIC)*, pages 200–206, 2005.

[FLV08] Pierre Fraigniaud, Emmanuelle Lebhar, and Laurent Viennot. The inframetric model for the internet. In *27th IEEE International Conference on Computer Communications (INFOCOM)*, pages 1085–1093. IEEE, 2008.

[FMS08] Paola Flocchini, Bernard Mans, and Nicola Santoro. Tree decontamination with temporary immunity. In *19th International Symposium on Algorithms and Computation (ISAAC)*, volume 5369 of *Lecture Notes in Computer Science*, pages 330–341. Springer, 2008.

[FN01] Shannon L. Fitzpatrick and Richard J. Nowakowski. Copnumber of graphs with strong isometric dimension two. *Ars Comb.*, 59:65–73, 2001.

[FPS12] Paola Flocchini, Giuseppe Prencipe, and Nicola Santoro. *Distributed Computing by oblivious mobile robots*. Morgan and Claypool, 2012.

[Fra87a] Peter Frankl. Cops and robbers in graphs with large girth and cayley graphs. *Discrete Applied Mathematics*, 17:301–305, 1987.

[Fra87b] Peter Frankl. On a pursuit game on cayley graphs. *Combinatorica*, 7(1):67–70, 1987.

[Fra05] Pierre Fraigniaud. Greedy routing in tree-decomposed graphs. In *13th Annual European Symposium on Algorithms (ESA)*, volume 3669 of *Lecture Notes in Computer Science*, pages 791–802. Springer, 2005.

[Fra10] Pierre Fraigniaud. Distributed computational complexities: are you volvo-addicted or nascar-obsessed? In *29th Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pages 171–172. ACM, 2010.

[Fri00] Alan M. Frieze. Edge-disjoint paths in expander graphs. *SIAM J. Comput.*, 30(6):1790–1801, 2000.

[FRT11] Pierre Fraigniaud, Sergio Rajsbaum, and Corentin Travers. Locality and checkability in wait-free computing. In *25th International Symposium on Distributed Computing (DISC)*, volume 6950 of *Lecture Notes in Computer Science*, pages 333–347. Springer, 2011.

[FS04] Darin Fisher and Gagan Saksena. Web content caching and distribution. In *Web content caching and distribution*, chapter Link prefetching in Mozilla: a server-driven approach, pages 283–291. Kluwer Academic Publishers, 2004.

[FS11] Lance Fortnow and Rahul Santhanam. Infeasibility of instance compression and succinct PCPs for NP. *J. Comput. Syst. Sci.*, 77(1):91–106, 2011.

[FT07] Fedor V. Fomin and Dimitrios M. Thilikos. On self duality of pathwidth in polyhedral graph embeddings. *Journal of Graph Theory*, 55(1):42–54, 2007.

[FT08] Fedor V. Fomin and Dimitrios M. Thilikos. An annotated bibliography on guaranteed graph searching. *Theoretical Computer Science*, 399(3):236–245, 2008.

[Gav11] Tomás Gavenciak. Catching a fast robber on interval graphs. In *8th Annual Conference on Theory and Applications of Models of Computation (TAMC)*, volume 6648 of *Lecture Notes in Computer Science*, pages 353–364. Springer, 2011.

[GCD02] Romulus Grigoras, Vincent Charvillat, and Matthijs Douze. Optimizing hypervideo navigation using a Markov decision process approach. In *ACM Multimedia*, pages 39–48, 2002.

[GHH05] Wayne Goddard, Sandra M. Hedetniemi, and Stephen T. Hedetniemi. Eternal security in graphs. *J. Combin.Math.Combin.Comput.*, 52, 2005.

[GHK+10] Robert Ganian, Petr Hlinený, Joachim Kneis, Daniel Meister, Jan Obdrzálek, Peter Rossmanith, and Somnath Sikdar. Are there any good digraph width measures? In *5th International Symposium on Parameterized and Exact Computation (IPEC)*, volume 6478 of *Lecture Notes in Computer Science*, pages 135–146. Springer, 2010.

[GHM12] Petr A. Golovach, Pinar Heggernes, and Rodica Mihai. Edge search number of cographs. *Discrete Applied Mathematics*, 160(6):734–743, 2012.

[GHT12] Archontia C. Giannopoulou, Paul Hunter, and Dimitrios M. Thilikos. Lifo-search: A min-max theorem and a searching game for cycle-rank and tree-depth. *Discrete Applied Mathematics*, 160(15):2089–2097, 2012.

[GHW08] Catherine S. Greenhill, Fred B. Holt, and Nicholas C. Wormald. Expansion properties of a random regular graph after random vertex deletions. *Eur. J. Comb.*, 29(5):1139–1150, 2008.

[GJ90] Michael R. Garey and David S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA, 1990.

[GKL03] Anupam Gupta, Robert Krauthgamer, and James R. Lee. Bounded geometries, fractals, and low-distortion embeddings. In *44th Symposium on Foundations of Computer Science (FOCS)*, pages 534–543. IEEE Computer Society, 2003.

[GLL+99] Leonidas J. Guibas, Jean-Claude Latombe, Steven M. LaValle, David Lin, and Rajeev Motwani. A visibility-based pursuit-evasion problem. *Int. J. Comput. Geometry Appl.*, 9(4/5):471–494, 1999.

[GN07] Jiong Guo and Rolf Niedermeier. Invitation to data reduction and problem kernelization. *SIGACT News*, 38(1):31–45, 2007.

[God09] D.A.G.A.P. Godse. *Microprocessors And Interfacing*. Technical Publications, 2009.

[GR95] Arthur S. Goldstein and Edward M. Reingold. The complexity of pursuit on a graph. *Theoretical Computer Science*, 143(1):93–112, 1995.

[GR09] Luisa Gargano and Adele A. Rescigno. Optimally fast data gathering in sensor networks. *Discrete Applied Mathematics*, 157:1858–1872, 2009.

[GS11] George Giakkoupis and Nicolas Schabanel. Optimal path search in small worlds: dimension matters. In *43rd ACM Symposium on Theory of Computing (STOC)*, pages 393–402. ACM, 2011.

[GT88] Andrew V. Goldberg and Robert E. Tarjan. A new approach to the maximum-flow problem. *Journal of the Association for Computing Machinery*, 35(4):921–940, October 1988.

[Gus93] J. Gustedt. On the pathwidth of chordal graphs. *Discrete Applied Mathematics*, 45(3):233–248, 1993.

[GW09] Stéphane Grumbach and Zhilin Wu. Logical locality entails frugal distributed computation over graphs (extended abstract). In *35th International Workshop on Graph-Theoretic Concepts in Computer Science (WG)*, volume 5911 of *Lecture Notes in Computer Science*, pages 154–165, 2009.

[GZ97] D. J. Guan and Xuding Zhu. A coloring problem for weighted graphs. *Inf. Process. Lett.*, 61(2):77–81, 1997.

[Hah07] Gena Hahn. Cops, robbers, and graphs. *Tatra Mt. Math. Publ.*, 36:163–176, 2007.

[Ham87] Yahya O. Hamidoune. On a pursuit game on Cayley digraphs. *European J. Combin.*, 8:289–295, 1987.

[HJM+05] Carmen Hernando, Tao Jiang, Mercè Mora, Ignacio M. Pelayo, and Carlos Seara. On the steiner, geodetic and hull numbers of graphs. *Discrete Mathematics*, 293(1-3):139 – 154, 2005. 19th British Combinatorial Conference.

[HK08] Paul Hunter and Stephan Kreutzer. Digraph measures: Kelly decompositions, games, and orderings. *Theoretical Computer Science*, 399(3):206–219, 2008.

[HKK04] Nicolas Hanusse, Evangelos Kranakis, and Danny Krizanc. Searching with mobile agents in networks with liars. *Discrete Applied Mathematics*, 137:69–85, 2004.

[HKKK08] Nicolas Hanusse, Dimitris J. Kavvadias, Evangelos Kranakis, and Danny Krizanc. Memoryless search algorithms in a network with faulty advice. *Theoretical Computer Science*, 402(2-3):190–198, 2008.

[HLP+07] Thomas Hérault, Pierre Lemarinier, Olivier Peres, Laurence Pilard, and Joffroy Beauquier. A model for large scale self-stabilization. In *21th International Parallel and Distributed Processing Symposium (IPDPS)*, pages 1–10. IEEE, 2007.

[HLSW02] Gena Hahn, François Laviolette, Norbert Sauer, and Robert E. Woodrow. On cop-win graphs. *Discrete Mathematics*, 258(1-3):27–41, 2002.

[HM06] Gena Hahn and Gary MacGillivray. A note on k-cop, l-robber games on graphs. *Discrete Math.*, 306:2492–2497, 2006.

[HM08] Pinar Heggernes and Rodica Mihai. Mixed search number of permutation graphs. In *2nd Int. Workshop on Frontiers in Algorithmics (FAW)*, volume 5059 of *Lecture Notes in Computer Science*, pages 196–207. Springer, 2008.

[HM09] Pinar Heggernes and Rodica Mihai. Edge search number of cographs in linear time. In *3rd Int. Workshop on Frontiers in Algorithmics (FAW)*, volume 5598 of *Lecture Notes in Computer Science*, pages 16–26. Springer, 2009.

[IIKO10] Taisuke Izumi, Tomoko Izumi, Sayaka Kamei, and Fukuhito Ooshita. Mobile robots gathering algorithm with local weak multiplicity in rings. In *17th Int. Coll. on Structural Information and Communication Complexity (SIROCCO)*, volume 6058 of *Lecture Notes in Computer Science*, pages 101–113, 2010.

[IK08] Volkan Isler and Nikhil Karnad. The role of information in the cop-robber game. *Theoretical Computer Science*, 399(3):179–190, 2008.

[IKK06] Volkan Isler, Sampath Kannan, and Sanjeev Khanna. Randomized pursuit-evasion with local visibility. *SIAM J. Discrete Math.*, 20(1):26–41, 2006.

[IKOY03] Satoshi Ikeda, Izumi Kubo, Norihiro Okumoto, and Masafumi Yamashita. Impact of local topological information on random walks on finite graphs. In *30th International Colloquium on Automata, Languages and Programming (ICALP)*, pages 1054–1067, 2003.

[Inc99] Zona Research Inc. The economic impacts of unacceptable web-site download speeds. White paper, Redwood City, CA, April 1999. URL: `www.webperf.net/info/wp_downloadspeed.pdf`.

[JG97] Doug Joseph and Dirk Grunwald. Prefetching using Markov predictors. In *24th annual international symposium on Computer architecture*, ISCA '97, pages 252–263. ACM, 1997.

[JKT10] Gwenaël Joret, Marcin Kaminski, and Dirk Oliver Theis. The cops and robber game on graphs with forbidden (induced) subgraphs. *Contributions to Discrete Mathematics*, 5(2), 2010.

[JMT07] Brigitte Jaumard, Christophe Meyer, and Babacar Thiongane. Comparison of ILP formulations for the RWA problem. *Optical Switching and Networking*, 4(3-4):157–172, 2007.

[Joh83] David S. Johnson. The NP-completeness column: An ongoing guide. *J. Algorithms*, 4(4):397–411, 1983.

[JRST01] Thor Johnson, Neil Robertson, Paul D. Seymour, and Robin Thomas. Directed tree-width. *J. Comb. Theory, Ser. B*, 82(1):138–154, 2001.

[JS03] N. Jose and A.K. Somani. Connection rerouting/network reconfiguration. In *Design of Reliable Communication Networks*. IEEE, 2003.

[KAI79] Takumi Kasai, Akeo Adachi, and Shigeki Iwata. Classes of pebble games and complete problems. *SIAM J. Comput.*, 8(4):574–586, 1979.

[KFY04] Dmitri V. Krioukov, Kevin R. Fall, and Xiaowei Yang. Compact routing on internet-like graphs. In *23th IEEE International Conference on Computer Communications (INFOCOM)*, 2004.

[KI09] Nikhil Karnad and Volkan Isler. Lion and man game in the presence of a circular obstacle. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5045–5050. IEEE, 2009.

[Kin13] William B. Kinnersley. Cops and robbers is exptime-complete. Research report, 2013. URL: `http://arxiv.org/abs/1309.5405`.

[KK77] Leonard Kleinrock and Farouk Kamoun. Hierarchical routing for large networks; performance evaluation and optimization. *Computer Networks*, 1:155–174, 1977.

[KK99] Evangelos Kranakis and Danny Krizanc. Searching with uncertainty. In *6th International Colloquium on Structural Information and Communication Complexity (SIROCCO)*, pages 194–203, 1999.

[KK09] Y. Kobayashi and K. Kawarabayashi. Algorithms for finding an induced cycle in planar graphs and bounded genus graphs. In *20th Annual ACM-SIAM Symp. on Discrete Alg. (SODA)*, pages 1146–1155. SIAM, 2009.

[KK12] Ken-ichi Kawarabayashi and Yusuke Kobayashi. Linear min-max relation between the treewidth of H-minor-free graphs and its largest grid. In *29th International Symposium on Theoretical Aspects of Computer Science (STACS)*, volume 14 of *LIPIcs*, pages 278–289, 2012.

[KKK11] Shiva Kintali, Nishad Kothari, and Akash Kumar. Approximation algorithms for directed width parameters. Research report, 2011. URL: `http://arxiv.org/abs/1107.4824`.

[KKN10] Ralf Klasing, Adrian Kosowski, and Alfredo Navarra. Taking advantage of symmetries: Gathering of many asynchronous oblivious robots on a ring. *Theoretical Computer Science*, 411:3235–3246, 2010.

[Kle00a] Jon M. Kleinberg. Navigation in a small world. *Nature*, 406:845, 2000.

[Kle00b] Jon M. Kleinberg. The small-world phenomenon: an algorithm perspective. In *31st ACM Symposium on Theory of Computing (STOC)*, pages 163–170, 2000.

[KLM97] Thomas M. Kroeger, Darrell D. E. Long, and Jeffrey C. Mogul. Exploring the bounds of web latency reduction from caching and prefetching. In *USENIX Symposium on Internet Technologies and Systems*, pages 2–2, 1997.

[KLNP09] Ralf Klasing, Zwi Lotker, Alfredo Navarra, and Stéphane Pérennes. From balls and bins to points and vertices. *Algorithmic Operations Research (AlgOR)*, 4(2):133–143, 2009.

[Klo07] Oddvar Kloster. A solution to the angel problem. *Theoretical Computer Science*, 389(1-2):152–161, 2007.

[KLOT11] Sayaka Kamei, Anissa Lamani, Fukuhito Ooshita, and Sebastien Tixeuil. Asynchronous mobile robot gathering from symmetric configurations. In *18th Int. Coll. on Structural Information and Communication Complexity (SIROCCO)*, volume 6796 of *Lecture Notes in Computer Science*, pages 150–161, 2011.

[KLOT12] Sayaka Kamei, Anissa Lamani, Fukuhito Ooshita, and Sebastien Tixeuil. Gathering an even number of robots in an odd ring without global multiplicity detection. In *37th Int. Symp. on Math. Foundations of Computer Science (MFCS)*, volume 7464 of *Lecture Notes in Computer Science*, pages 542–553, 2012.

[KLP+13] Eun Jung Kim, Alexander Langer, Christophe Paul, Felix Reidl, Peter Rossmanith, Ignasi Sau, and Somnath Sikdar. Linear kernels and single-exponential algorithms via protrusion decompositions. In *40th International Colloquium on Automata, Languages, and Programming (ICALP (1))*, volume 7965 of *Lecture Notes in Computer Science*, pages 613–624. Springer, 2013.

[KM09] William F. Klostermeyer and Gary MacGillivray. Eternal dominating sets in graphs. *J. Combin.Math.Combin.Comput.*, 68, 2009.

[KM12] Telikepalli Kavitha and Julián Mestre. Max-coloring paths: tight bounds and extensions. *J. Comb. Optim.*, 24(1):1–14, 2012.

[KMP08] Ralf Klasing, Euripides Markou, and Andrzej Pelc. Gathering asynchronous oblivious mobile robots in a ring. *Theoretical Computer Science*, 390:27–39, 2008.

[KMP13] Athanasios Kehagias, Dieter Mitsche, and Pawel Pralat. Cops and invisible robbers: The cost of drunkenness. *Theoretical Computer Science*, 481:100–120, 2013.

[KMW04] Fabian Kuhn, Thomas Moscibroda, and Roger Wattenhofer. What cannot be computed locally! In *33rd Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pages 300–309. ACM, 2004.

[KN13] Mamadou Moustapha Kanté and Lhouari Nourine. Polynomial time algorithms for computing a minimum hull set in distance-hereditary and chordal graphs. In *39th International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM)*, volume 7741 of *Lecture Notes in Computer Science*, pages 268–279. Springer, 2013.

[KO11] Stephan Kreutzer and Sebastian Ordyniak. Digraph decompositions and monotonicity in digraph searching. *Theoretical Computer Science*, 412(35):4688–4703, 2011.

[KP86] Lefteris M. Kirousis and Christos H. Papadimitriou. Searching and pebbling. *Theoretical Computer Science*, 47(2):205–218, 1986.

[KR96] Jon M. Kleinberg and Ronitt Rubinfeld. Short paths in expander graphs. In *37th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 86–95, 1996.

[KR05] Swastik Kopparty and Chinya V. Ravishankar. A framework for pursuit evasion games in r^n. *Inf. Process. Lett.*, 96(3):114–122, 2005.

[KRX06] Goran Konjevod, Andréa W. Richa, and Donglin Xia. Optimal-stretch name-independent compact routing in doubling metrics. In *25th Annual ACM Symposium on Principles of Distributed Computing (PODC)*, pages 198–207. ACM, 2006.

[KS12] Kyle Klein and Subhash Suri. Catch me if you can: Pursuit and capture in polygonal environments with obstacles. In *26th AAAI Conference on Artificial Intelligence (AAAI)*. AAAI Press, 2012.

[KS13] Kyle Klein and Subhash Suri. Capture bounds for visibility-based pursuit evasion. In *29th Annual Symposium on Computational Geometry (SoCG)*. Springer, 2013.

[KW13] Natasha Komarov and Peter Winkler. Cops vs. gambler. Research report, 2013. URL: `http://arxiv.org/abs/1308.4715`.

[Lag96] Jens Lagergren. Efficient parallel algorithms for graphs of bounded tree-width. *J. Algorithms*, 20(1):20–44, 1996.

[LaP93] Andrea S. LaPaugh. Recontamination does not help to search a graph. *J. ACM*, 40(2):224–245, 1993.

[Lin92] Nathan Linial. Locality in distributed graph algorithms. *SIAM J. Comput.*, 21(1):193–201, 1992.

[Lit53] John E. Littlewood. A mathematicians miscellany. *Mathuen*, VII, 1953.

[LL96] Kuo-Chun Lee and Victor O.K. Li. A wavelength rerouting algorithm in wide-area all-optical networks. *IEEE/OSA Journal of Lightwave Technology*, 1996.

[LMS11] Daniel Lokshtanov, Dániel Marx, and Saket Saurabh. Lower bounds based on the exponential time hypothesis. *Bulletin of the EATCS*, 105:41–72, 2011.

[LMT10] Laurent Lyaudet, Frédéric Mazoit, and Stéphan Thomassé. Partitions versus sets: A case of duality. *Eur. J. Comb.*, 31(3):681–687, 2010.

[Lok10] Daniel Lokshtanov. On the complexity of computing treelength. *Discrete Applied Mathematics*, 158(7):820–827, 2010.

[Lov93] László Lovász. Random walks on graphs: A survey. *Combinatorics*, pages 1–46, 1993.

[LP10] Tomasz Luczak and Pawel Pralat. Chasing robbers on random graphs: Zigzag theorem. *Random Struct. Algorithms*, 37(4):516–524, 2010.

[LP12] Linyuan Lu and Xing Peng. On meyniel's conjecture of the cop number. *Journal of Graph Theory*, 71(2):192–205, 2012.

[LPS07] Fabrizio Luccio, Linda Pagli, and Nicola Santoro. Network decontamination in presence of local immunity. *Int. J. Found. Comput. Sci.*, 18(3):457–474, 2007.

[LQK+08] Bo Li, Yang Qu, Gabriel Y. Keung, Susu Xie, Chuang Lin, Jiangchuan Liu, and Xinyan Zhang. Inside the new coolstreaming: Principles, measurements and performance implications. In *27th IEEE International Conference on Computer Communications*, 2008.

[LS04] Van Bang Le and Jeremy Spinrad. Consequences of an algorithm for bridged graphs. *Discrete Mathematics*, 280(1-3):271–274, 2004.

[LT80] Richard J. Lipton and Robert Endre Tarjan. Applications of a planar separator theorem. *SIAM J. Comput.*, 9(3):615–627, 1980.

[Luc07] Flaminia L. Luccio. Intruder capture in sierpinski graphs. In *4th International Conference on Fun with algorithms (FUN)*, pages 249–261, 2007.

[Luc09] Flaminia L. Luccio. Contiguous search problem in sierpinski graphs. *Theory Comput. Syst.*, 44(2):186–204, 2009.

[LXHL11] Zhenyu Li, Gaogang Xie, Kai Hwang, and Zhongcheng Li. Churn-resilient protocol for massive data dissemination in p2p networks. *IEEE Parallel and Distributed Systems*, 22(8):1342–1349, 2011.

[Mam13] Marcello Mamino. On the computational complexity of a game of cops and robbers. *Theoretical Computer Science*, 477:48–56, 2013.

[Mar14] Héli Marcoux. *Jeux de poursuite policier-voleur sur un graphe - Le cas du voleur rapide.* master thesis (in french), Univ. Laval, Canada, 2014. URL: `http://theses.ulaval.ca/archimede/meta/30386`.

[MCK+09] Konstantinos Manousakis, Konstantinos Christodoulopoulos, Euaggelos Kamitsas, Ioannis Tomkos, and Emmanouel Varvarigos. Offline impairment-aware routing and wavelength assignment algorithms in translucent WDM optical networks. *IEEE/OSA Journal of Lightwave Technology*, 27(12):1856–1877, 2009.

[Meh10] Abbas Mehrabian. Pursuing a superfast robber in an interval graph. *CoRR*, abs/1008.4210, 2010.

[Meh11] Abbas Mehrabian. Lower bounds for the cop number when the robber is fast. *Combinatorics, Probability & Computing*, 20(4):617–621, 2011.

[MHG+88] Nimrod Megiddo, S. Louis Hakimi, Michael R. Garey, David S. Johnson, and Christos H. Papadimitriou. The complexity of searching a graph. *J. Assoc. Comput. Mach.*, 35(1):18–44, 1988.

[MJM10] Olivia Morad and Alain Jean-Marie. Optimisation en temps-réel du téléchargement de vidéos. In *11th Congress of the French Operations Research Soc.*, 2010.

[MM87] M. Maamoun and Henry Meyniel. On a game of policemen and robber. *Discrete Appl. Math.*, 17:307–309, 1987.

[MM99] Gurusamy Mohan and Chebiyyam S.R. Murthy. A time optimal wavelength rerouting algorithm for dynamic traffic in WDM networks. *IEEE/OSA Journal of Lightwave Technology*, 17(3):406–417, 1999.

[MM09] Rodica Mihai and M. Mjelde. A self-stabilizing algorithm for graph searching in trees. In *11th Int. Symp. on Stabilization, Safety, and Security of Distributed Systems (SSS)*, volume 5873 of *Lecture Notes in Computer Science*, pages 563–577. Springer, 2009.

[Mog96] Jeffrey C. Mogul. Hinted caching in the web. In *7th workshop on ACM SIGOPS European workshop: Systems support for worldwide applications*, pages 103–108, 1996.

[MP06] Eric Mannie and Dimitri Papadimitriou. Recovery (protection and restoration) terminology for generalized multi-protocol label switching (GMPLS). RFC 4427, IETF, 2006.

[MR09] Nazanin Magharei and Reza Rejaie. Prime: Peer-to-peer receiver-driven mesh-based streaming. *IEEE/ACM Transactions on Networking*, 17(4):1052–1065, 2009.

[MS88] B. Monien and I. H. Sudborough. Min cut is NP-complete for edge weighted trees. *Theoretical Computer Science*, 58:209–229, 1988.

[MT91] Jirí Matousek and Robin Thomas. Algorithms finding tree-decompositions of graphs. *J. Algorithms*, 12(1):1–22, 1991.

[MT09] Rodica Mihai and Ioan Todinca. Pathwidth is NP-hard for weighted trees. In *Third International Workshop on Frontiers in Algorithmics (FAW)*, volume 5598 of *Lecture Notes in Computer Science*, pages 181–195. Springer, 2009.

[MTV10] Daniel Meister, Jan Arne Telle, and Martin Vatshelle. Recognizing digraphs of kelly-width 2. *Discrete Applied Mathematics*, 158(7):741–746, 2010.

[Muk92] Biswanath Mukherjee. WDM-based local lightwave networks – Part II: Multi-hop systems. *IEEE Network*, 6(4):20–32, 1992.

[Nie06] Rolf Niedermeier. *Invitation to Fixed Parameter Algorithms.* Oxford Lecture Series in Mathematics and Its Applications, 2006.

[NN98] Stewart W. Neufeld and Richard J. Nowakowski. A game of cops and robbers played on products of graphs. *Discrete Mathematics*, 186(1-3):253–268, 1998.

[NPSY94] Sotiris E. Nikoletseas, Krishna V. Palem, Paul G. Spirakis, and Moti Yung. Short vertex disjoint paths and multiconnectivity in random graphs: Reliable network computing. In *21st International Colloquium on Automata, Languages and Programming (ICALP)*, pages 508–519, 1994.

[NW83] Richard J. Nowakowski and Peter Winkler. Vertex-to-vertex pursuit in a graph. *Discrete Mathematics*, 43:235–239, 1983.

[Par78a] Torrence D. Parsons. Pursuit-evasion in a graph. In *International Conference on Theory and applications of graphs*, pages 426–441. Lecture Notes in Math., Vol. 642. Springer, Berlin, 1978.

[Par78b] Torrence D. Parsons. The search number of a connected graph. In *9th Southeastern Conf. on Combinatorics, Graph Theory, and Computing*, Congress. Numer., XXI, pages 549–554. Utilitas Math., 1978.

[Pel00] David Peleg. *Distributed computing: a locality-sensitive approach.* SIAM Monographs on Discrete Mathematics and Applications, 2000.

[Pel12] Andrzej Pelc. Deterministic rendezvous in networks: A comprehensive survey. *Networks*, 59(3):331–347, 2012.

[PHsH+00] Sheng-Lung Peng, Chin-Wen Ho, Tsan sheng Hsu, Ming-Tat Ko, and Chuan Yi Tang. Edge and node searching problems on trees. *Theoretical Computer Science*, 240(2):429–446, 2000.

[PM96] Venkata N. Padmanabhan and Jeffrey C. Mogul. Using predictive prefetching to improve world wide web latency. *SIGCOMM Comput. Commun. Rev.*, 26(3):22–36, 1996.

[PPR04] Sriram V. Pemmaraju, Sriram Penumatcha, and Rajiv Raman. Approximating interval coloring and max-coloring in chordal graphs. In *3rd International Workshop on Experimental and Efficient Algorithms (WEA)*, volume 3059 of *Lecture Notes in Computer Science*, pages 399–416. Springer, 2004.

[PPR05] Sriram V. Pemmaraju, Sriram Penumatcha, and Rajiv Raman. Approximating interval coloring and max-coloring in chordal graphs. *ACM J. of Exp. Algorithmics*, 10, 2005.

[PR05] Sriram V. Pemmaraju and Rajiv Raman. Approximation algorithms for the max-coloring problem. In *32nd International Colloquium on Automata, Languages and Programming (ICALP)*, volume 3580 of *Lecture Notes in Computer Science*, pages 1064–1075. Springer, 2005.

[Pre07] Giuseppe Prencipe. Impossibility of gathering by a set of autonomous mobile robots. *Theoretical Computer Science*, 384:222–231, 2007.

[PRV04] Sriram V. Pemmaraju, Rajiv Raman, and Kasturi R. Varadarajan. Buffer minimization using max-coloring. In *15th ACM-SIAM Symp. on Discrete Alg. (SODA)*, pages 562–571, 2004.

[PRV11] Sriram V. Pemmaraju, Rajiv Raman, and Kasturi R. Varadarajan. Max-coloring and online coloring with bandwidths on interval graphs. *ACM Trans. on Algorithms*, 7(3):35, 2011.

[PTK+00] Sheng-Lung Peng, Chuan Yi Tang, Ming-Tat Ko, Chin-Wen Ho, and Tsansheng Hsu. Graph searching on some subclasses of chordal graphs. *Algorithmica*, 27(3):395–426, 2000.

[PTT09] Meng-Shiuan Pan, Chia-Hung Tsai, and Yu-Chee Tseng. The orphan problem in zigbee wireless networks. *IEEE Transactions on Mobile Computing*, 8(11):1573–1584, 2009.

[PU89] David Peleg and Eli Upfal. A trade-off between space and efficiency for routing tables. *J. ACM*, 36(3):510–530, 1989.

[PVW11] Pawel Pralat, Jacques Verstraëte, and Nicholas C. Wormald. On the threshold for k-regular subgraphs of random graphs. *Combinatorica*, 31(5):565–581, 2011.

[PW13] Pawel Pralat and Nicholas C. Wormald. Meyniel's conjecture holds for random graphs. Research report, 2013. URL: `http://arxiv.org/abs/1301.2841`.

[PY07] Sheng-Lung Peng and Yi-Chuan Yang. On the treewidth and pathwidth of biconvex bipartite graphs. In *4th Int. Conf. on Theory and Applications of Models of Computation (TAMC)*, volume 4484 of *Lecture Notes in Computer Science*, pages 244–255. Springer, 2007.

[Qui83] Alain Quilliot. *Problèmes de jeux, de point fixe, de connectivité et de représentation sur des graphes, des ensembles ordonnés et des hypergraphes.* Thèse de doctorat d'état, Université de Paris VI, France, 1983.

[Qui85] Alain Quilliot. A short note about pursuit games played on a graph with a given genus. *J. Comb. Theory, Ser. B*, 38(1):89–92, 1985.

[Qui86] Alain Quilliot. Some results about pursuit games on metric spaces obtained through graph theory techniques. *European J. Combin.*, 7:55–66, 1986.

[Qui93] Alain Quilliot. Une extension du problème du point fixe pour des graphes simples. *Discrete Mathematics*, 111(1-3):435–445, 1993.

[Ree92] Bruce A. Reed. Finding approximate separators and computing tree width quickly. In *24th Annual ACM Symposium on Theory of Computing (STOC)*, pages 221–228. ACM, 1992.

[Rei79] John H. Reif. Universal games of incomplete information. In *Eleventh Annual ACM Symposium on Theory of Computing (STOC)*, pages 288–308. ACM, 1979.

[Rei84] John H. Reif. The complexity of two-player games of incomplete information. *J. Comput. Syst. Sci.*, 29(2):274–301, 1984.

[Ros74] Donald J. Rose. On simple characterizations of k-trees. *Discrete Mathematics*, 7(3-4):317 – 322, 1974.

[RS83] Neil Robertson and Paul D. Seymour. Graph minors. i. excluding a forest. *J. Comb. Theory, Ser. B*, 35(1):39–61, 1983.

[RS84] Neil Robertson and Paul D. Seymour. Graph minors. iii. planar tree-width. *J. Comb. Theory, Ser. B*, 36(1):49–64, 1984.

[RS86a] Neil Robertson and Paul D. Seymour. Graph minors. ii. algorithmic aspects of tree-width. *J. Algorithms*, 7(3):309–322, 1986.

[RS86b] Neil Robertson and Paul D. Seymour. Graph minors. v. excluding a planar graph. *J. Comb. Theory, Ser. B*, 41(1):92–114, 1986.

[RS90] Neil Robertson and Paul D. Seymour. Graph minors. iv. tree-width and well-quasi-ordering. *J. Comb. Theory, Ser. B*, 48(2):227–254, 1990.

[RS91] Neil Robertson and Paul D. Seymour. Graph minors. x. obstructions to tree-decomposition. *J. Comb. Theory, Ser. B*, 52(2):153–190, 1991.

[RS03a] Neil Robertson and Paul D. Seymour. Graph minors. xvi. excluding a non-planar graph. *J. Comb. Theory, Ser. B*, 89(1):43–76, 2003.

[RS03b] Neil Robertson and Paul D. Seymour. Graph minors. xviii. tree-decompositions and well-quasi-ordering. *J. Comb. Theory, Ser. B*, 89(1):77–108, 2003.

[RS04] Neil Robertson and Paul D. Seymour. Graph minors. xx. wagner's conjecture. *J. Comb. Theory, Ser. B*, 92(2):325–357, 2004.

[RS07] Yoram Revah and Michael Segal. Improved algorithms for data-gathering time in sensor networks ii: Ring, tree and grid topologies. In *Networking and Services, 2007. ICNS. Third International Conference on*, page 46, 2007.

[RS08] Yoram Revah and Michael Segal. Improved bounds for data-gathering time in sensor networks. *Computer Communications*, 31(17):4026–4034, 2008.

[RST94] Neil Robertson, Paul D. Seymour, and Robin Thomas. Quickly excluding a planar graph. *J. Comb. Theory, Ser. B*, 62(2):323–348, 1994.

[Saf05] Mohammad Ali Safari. D-width: A more natural measure for directed tree width. In *MFCS*, volume 3618 of *Lecture Notes in Computer Science*, pages 745–756. Springer, 30th International Symposium on Mathematical Foundations of Computer Science (2005).

[Sch01] Bernd S. W. Schröder. The copnumber of a graph is bounded by $\lfloor \frac{3}{2} genus(g) \rfloor + 3$. *Categorical perspectives, Trends in Mathematics*, pages 243–263, 2001.

[SD13] Florian Shkurti and Gregory Dudek. On the complexity of searching for an evader with a faster pursuer. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4047–4052, 2013.

[SFFF03] Georgos Siganos, Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. Power laws and the as-level internet topology. *IEEE/ACM Trans. Netw.*, 11(4):514–524, 2003.

[Sga01] Jiri Sgall. Solution of david gale's lion and man problem. *Theoretical Computer Science*, 259(1-2):663–670, 2001.

[Sko03] Konstantin Skodinis. Computing optimal linear layouts of trees in linear time. *J. Algorithms*, 47(1):40–59, 2003.

[Sli07] Aleksandrs Slivkins. Distance estimation and object location via rings of neighbors. *Distributed Computing*, 19(4):313–333, 2007.

[Soa13] Ronan Pardo Soares. *Pursuit-Evasion, Decompositions and Convexity on Graphs*. PhD thesis, Université Nice-Sophia Antipolis, France, November 2013.

[Sol09] Fernando Solano. Analyzing two different objectives of the WDM network reconfiguration problem. In *IEEE Global Communications Conference (Globecom)*, 2009.

[SP10] Fernando Solano and Michal Pióro. Lightpath reconfiguration in WDM networks. *IEEE/OSA J. Opt. Commun. Netw.*, 2(12):1010–1021, 2010.

[SS09] Chava Vijaya Saradhi and S. Subramaniam. Physical layer impairment aware routing (pliar) in WDM optical networks: issues and challenges. *IEEE Comm. Surveys and Tutorials*, 11(4):109–130, 2009.

[SS11] Alex Scott and Benny Sudakov. A bound for the cops and robbers problem. *SIAM J. Discrete Math.*, 25(3):1438–1442, 2011.

[SSV11] Robert Sámal, Rudolf Stolar, and Tomás Valla. Complexity of the cop and robber guarding game. In *22nd International Workshop on Combinatorial Algorithms (IWOCA)*, volume 7056 of *Lecture Notes in Computer Science*, pages 361–373. Springer, 2011.

[ST93] Paul D. Seymour and Robin Thomas. Graph searching and a min-max theorem for tree-width. *J. Comb. Theory, Ser. B*, 58(1):22–33, 1993.

[ST94] Paul D. Seymour and Robin Thomas. Call routing and the ratcatcher. *Combinatorica*, 14(2):217–241, 1994.

[ST99] Yannis C. Stamatiou and Dimitrios M. Thilikos. Monotonicity and inert fugitive search games. *Electronic Notes in Discrete Mathematics*, 3:184, 1999.

[ST07] Karol Suchan and Ioan Todinca. Pathwidth of circular-arc graphs. In *33rd Int. Workshop on Graph-Theoretic Concepts in Computer Sc. (WG)*, volume 4769 of *Lecture Notes in Computer Science*, pages 258–269. Springer, 2007.

[ST10] Ignasi Sau and Dimitrios M. Thilikos. Subexponential parameterized algorithms for degree-constrained subgraph problems on planar graphs. *J. Discrete Algorithms*, 8(3):330–338, 2010.

[SY92] Ichiro Suzuki and Masafumi Yamashita. Searching for a mobile intruder in a polygonal region. *SIAM J. Comput.*, 21(5):863–888, 1992.

[TE92] Leandros Tassiulas and Anthony Ephremides. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. *IEEE Transactions on Automatic Control*, 37(12):1936–1948, December 1992.

[The08] Dirk O. Theis. The cops and robber game on series-parallel graphs. Research report, 2008. URL: `arXiv:0712.2908`.

[Tho04] Mikkel Thorup. Compact oracles for reachability and approximate distances in planar digraphs. *J. ACM*, 51(6):993–1024, 2004.

[Tos88] R. Tosić. On cops and robber game. *Stud. Sci. Mat. Hungar*, 23:225–229, 1988.

[Tsa97] Panagiotis Tsaparas. Stability in adversarial queueing theory. Technical report, Master Thesis, Department of Computer Science, University of Toronto, 1997.

[TSB00] Dimitrios M. Thilikos, Maria J. Serna, and Hans L. Bodlaender. Constructive linear time algorithms for small cutwidth and carving-width. In *11th International Conference on Algorithms and Computation (ISAAC)*, volume 1969 of *Lecture Notes in Computer Science*, pages 192–203. Springer, 2000.

[TSB05a] Dimitrios M. Thilikos, Maria J. Serna, and Hans L. Bodlaender. Cutwidth i: A linear time fixed parameter algorithm. *J. Algorithms*, 56(1):1–24, 2005.

[TSB05b] Dimitrios M. Thilikos, Maria J. Serna, and Hans L. Bodlaender. Cutwidth ii: Algorithms for partial w-trees of bounded degree. *J. Algorithms*, 56(1):25–49, 2005.

[TZ01] Mikkel Thorup and Uri Zwick. Compact routing schemes. In *Annual ACM Symposium on Parallelism in Algorithms and Architectures (SPAA)*, pages 1–10, 2001.

[TZ05] Mikkel Thorup and Uri Zwick. Approximate distance oracles. *J. ACM*, 52(1):1–24, 2005.

[Ueh99] Ryuhei Uehara. Tractable and intractable problems on generalized chordal graphs. Technical Report COMP98-83, IEICE, 1999.

[Vaz01] Vijay V. Vazirani. *Approximation algorithms*. Springer, 2001.

[VYF06] Vidhyashankar Venkataraman, Kaouru Yoshida, and Paul Francis. Chunkyspread: Heterogeneous unstructured tree-based peer-to-peer multicast. In *14th IEEE International Conference on Network Protocols*, pages 2–11, 2006.

[wik] http://en.wikipedia.org/wiki/link_prefetching. URL: `http://en.wikipedia.org/wiki/Link_prefetching`.

[WLZC12] Zhen Wang, Felix Xiaozhu Lin, Lin Zhong, and Mansoor Chishtie. How far can client-only solutions go for mobile browser speed? In *21st Int. Conf. on World Wide Web*, pages 31–40, 2012.

[WS98] Duncan J. Watts and Steven H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:409–10, 1998.

[WS05] Xinzhou Wu and Ramakrishnan Srikant. Regulated maximal matching: A distributed scheduling algorithm for multi-hop wireless networks with node-exclusive spectrum sharing. In *44th IEEE Conference on Decision and Control, and the European Control Conference (CDC-ECC)*, pages 5342–5347, December 2005.

[Yan07] Boting Yang. Strong-mixed searching and pathwidth. *J. Comb. Optim.*, 13(1):47–59, 2007.

[YC07] Boting Yang and Yi Cao. Monotonicity of strong searching on digraphs. *J. Comb. Optim.*, 14(4):411–425, 2007.

[YC08a] Boting Yang and Yi Cao. Digraph searching, directed vertex separation and directed pathwidth. *Discrete Applied Mathematics*, 156(10):1822–1837, 2008.

[YC08b] Boting Yang and Yi Cao. Monotonicity in digraph search problems. *Theoretical Computer Science*, 407(1-3):532–544, 2008.

[YDA09] Boting Yang, Danny Dyer, and Brian Alspach. Sweeping graphs with large clique number. *Discrete Mathematics*, 309(18):5770–5780, 2009.

[ZTG05] Xianjin Zhu, Bin Tang, and Himanshu Gupta. Delay efficient data gathering in sensor network. In *1st International Conference on Mobile Ad-hoc and Sensor Networks (MSN)*, volume 3794 of *Lecture Notes in Computer Science*, pages 380–389. Springer-Verlag, 2005.

Complexité algorithmique: entre structure et connaissance.

Comment les jeux de poursuite peuvent apporter des solutions.

Résumé : Ce document présente les travaux que j'ai réalisés depuis ma thèse de doctorat. Outre la présentation de mes contributions, j'ai essayé de présenter des survols des domaines dans lesquels mes travaux s'inscrivent et d'indiquer les principales questions qui s'y posent.

Mes travaux visent à répondre aux nouveaux challenges algorithmiques que posent la croissance des réseaux de télécommunications actuels ainsi que l'augmentation des données et du trafic qui y circulent. Un moyen de faire face à la taille de ces problèmes est de s'aider de la structure particulière des réseaux. Pour cela, je m'attache à définir de nouvelles caractérisations des propriétés structurelles des graphes pour les calculer et les utiliser efficacement à des fins algorithmiques. Autant que possible, je propose des algorithmes distribués qui ne reposent que sur une connaissance locale/partielle des réseaux.

En particulier, j'étudie les jeux de poursuite - traitant de la capture d'une entité mobile par une équipe d'autres agents - qui offrent un point de vue intéressant sur de nombreuses propriétés de graphes et, notamment, des décompositions de graphes. L'approche de ces jeux d'un point de vue agents mobiles permet aussi l'étude de modèles de calcul distribué.

Le chapitre 1 est dédié à l'étude de plusieurs variantes des jeux de gendarmes et voleur. Le chapitre 2 traite des décompositions de graphes et de leur relation avec les problèmes d'encerclement dans les graphes. Le chapitre 3 se concentre sur les problèmes d'encerclement dans des contextes à la fois centralisé et distribué. Finalement, le chapitre 4 traite de problèmes de routage dans différents contextes, ainsi que de modèles de calcul distribué.

Mots-clés : Algorithmes, Théorie des Graphes, Jeux de Poursuite-Évasion, Gendarmes et voleur, Encerclement dans les graphes, Décompositions de graphes, Propriétés structurelles de graphes, Agents Mobiles, Réseaux de télécommunication, Routage, Calcul distribué

Algorithmic complexity: Between Structure and Knowledge

How Pursuit-evasion Games help.

Abstract: This manuscript describes the work I did since I obtained my Ph.D. in 2007. In addition to the presentation of my contributions, I tried to give overviews of the scientific areas my work takes part of and some of the main issues that arise here.

My work deals with new algorithmic challenges posed by the growth of nowadays networks and by the increased data and traffic arising in it. One way to cope with the size of these problems is to use the particular network structures. For this, I strive to develop new characterizations of graph structural properties in order to calculate and use them efficiently for an algorithmic perspective. Whenever possible, I propose distributed algorithms that are based only on a local/partial knowledge of networks.

In particular, I study the Pursuit-Evasion games - dealing with the capture of a mobile entity by a team of other agents - that offer an interesting perspective on many properties of graphs and, in particular, decompositions of graphs. This approach from mobile agents point of view also allows the study of various applications in telecommunication networks and of some distributed computing models.

Chapter 1 is dedicated to the study of several variants of turn-by-turn Pursuit-Evasion Games, mostly to the Cops and Robber games. Chapter 2 focuses on graph decompositions and their relationship with graph searching. Chapter 3 treats another aspect of Pursuit-Evasion games with a study of variants of Graph Searching games, both in centralized and distributed settings. Finally, Chapter 4 deals with routing problems in various environments and with distributed models of computation.

Keywords: Algorithm, Graph Theory, Pursuit-Evasion Games, Cops and Robber, Graph Searching, Graph decompositions, Graph Structural Properties, Mobile Agents Computing, Telecommunication Networks, Routing, Distributed Computing