Temporal distances and cops and robber games on temporal networks

Frédéric Simard

Thesis submitted to the University of Ottawa
in partial Fulfillment of the requirements for the
Doctor of Philosophy in Computer Science

School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa

Abstract

Temporal networks have gained in popularity in the last decade for their ability to model how connections vary over time. We are interested in understanding their structure and analyzing real-world data emerging from temporal networks. We approach those objectives by looking at the Cops and Robber Game from graph theory that is studied for its connection with the structure of graphs as well as by devising algorithms to compute metrics on temporal networks.

Cops and Robber Games have been first extended to the context of temporal networks by Erlebach and Spooner with algorithmic methods. The authors assumed a specific type of schedule that is *periodic*. We further the study of said games in the context of periodic temporal networks with algorithmic and analytical tools. Thus, we present characterizations of all periodic temporal networks on which a single cop can win, also called *copwin*. We suggest a structural characterization of copwin periodic graphs. Then, we venture on to explore periodic temporal networks that are not copwin. We also research lower and upper bounds on the number of cops required to capture the robber, i.e. the cop number. This then initiates a classification of periodic temporal networks into classes defined by their properties and their cop numbers.

Moreover, we present algorithms to study some connectivity properties of *link streams*, a model of temporal networks. These algorithms compute different types of distances on link streams, defined as the minimal number of hops between two temporal nodes, the minimal duration of a temporal path and a combination of both. Those distance functions are relevant for analyzing real-world data and, in particular, to compute an extension of the *betweenness centrality* on link streams.

Acknowledgements

I would like to extend my most sincere thanks to everyone who helped me, directly or indirectly, throughout this doctoral journey. Whether through academic guidance, encouragement, or social support, many people have contributed to this work. Words cannot perfectly describe how grateful I am to everyone involved.

First, I want to thank my advisors Paola, Jean-Lou, and Nicola, for their wonderful guidance. I am deeply thankful for the time they spent with me and the valuable knowledge they passed on to me with patience and kindness. Feeling that they believed in me for so long encouraged me to persevere through all these years.

I am also thankful to the members of the jury for how generous they were with their time as well as their important insights on this work. Their participation has greatly enhanced the quality of my research.

On a more personal level, I want to thank my parents whose unwavering support made it possible for me to dedicate so many years of my life to theoretical research. Similarly, the warm social support provided by my brother and my partner was invaluable in keeping me motivated through difficult times. They, as well as my friends, have made this long journey much brighter.

From the bottom of my heart, thank you everyone.

Contents

List of Figures

Chapter 1

Introduction

There is a saying attributed Heraclitus that is sometimes paraphrased as the following:

> Nothing endures but change.

Thus, trees grow, ice melts, people move countries, species evolve, etc. The real-world is never fully stable. Yet, mathematically, we tend to describe it as if it were fixed. This is traditionally what we do with graphs: we assume they are forever unchanging. However, a new framework has emerged in the last few decades to adapt to our changing reality that we will simply refer to as *Temporal Networks*.

1.1 Temporal Networks

The study of networks has emerged in the last few decades as a valuable tool to analyze the world around us. For example, actions performed online can influence one's "social credit" in China, drones must perform complex tasks with minimal communication and the Internet of Things creates cyber vulnerabilities in masses of interconnected objects. As such, Graph Theory and Network Science have provided important tools to analyze the issues those phenomena produce. One critical aspect that has long been overlooked is the varying nature of those networks. From a computer going offline to a drone flying out of communication range, networks are not fixed in time. Since the turn of the millenium and especially in the last decade, many researchers have been at work trying to fix this gap and develop knowledge about *temporal networks*.

A recent request on "Google Scholar" with the query "temporal network", limited to articles published since 2022 returned about 46 500 results. That is to say, research on temporal networks is booming. Researchers have recently used temporal networks to investigate sleep [128], the propagation of opinion on online social networks [206], generate better synthetic models of human movements [158], increase the prediction accuracy of air temperature [209], and even understand depressive symptoms of college students during the COVID-19 pandemic [152]. Those are but a small sample of the literature on applications of temporal networks to show that temporal networks have become ubiquitous in modern research. More examples of use cases are presented in Chapter 2

In order to gain a broader understanding of temporal networks, their uses and challenges, we want to look at two different problems on those structures. One is closer to applications while the other is more theoretical. This way we can learn about temporal networks from the two sides of the theoretical/practical "divide".

1.1.1 Models, Discrete vs Continuous

In order to simplify the investigations on temporal networks, one generally imposes assumptions on time. One common assumption is to consider time as *discrete*. This often comes with the assumption that the number of nodes does not grow infinitely. In this case, a *temporal graph*[1] is often described as an infinite sequence $\mathcal{G} = (G_0, G_1, \dots)$ of subgraphs $G_i = (V, E_i)$ of a common graph $G = (V, \cup_i E_i)$, a model that was originally described (independently) in [84] and [111]. Here, G_t is called the *snapshot* of G at time t while G is the *footprint*.

One way to work with *continuous* time is to assume the edges of a graph G have durations. Thus, each edge $uv \in E(G)$ appears continuously in an interval $[a, b] \subset \mathbb{R}$ (or set of intervals) that depends on uv. The *Link Stream* framework of Latapy et al. [144] is specifically designed for the simple case when all edges have real-time durations and where nodes are fixed.

Finally, although there are many models used for representing "temporal networks" (see [55, 115, 207]), there are two main general-purpose models: Stream Graphs [144] and Time-Varying Graphs [55].

[1] We will come back to the distinction between temporal networks and temporal graphs in Chapter 2.

We come back to the different models and assumptions in Subsection 2.1.2.

1.1.2 Connectivity and Periodicity

In temporal networks, it is also customary to apply assumptions on the variability of the changes. Some assumptions relate to the connectivity of G and each G_t. The strongest condition, 1-*interval connectivity* (e.g., [119, 141, 168]), requires that each G_t be connected. On the other hand, many weak conditions exist (see, e.g., [55]), for example requiring only that the temporal network be *connected over time* ([53, 104]). Casteigts et al. [55] initiated a classification of temporal networks through their connectivity properties.

In discrete-time, one can also wish to control the frequency of appearance of the edges in a temporal graph $\mathcal{G}$. In mobile ad hoc networks (MANETs), multiple mobility models exist to describe the movements of the wireless nodes, see for example the survey of [16]. Without explicitly describing the dynamics of those nodes, one could impose assumptions on their patterns of contacts, quantifying when the nodes interact. The most relevant such assumption for our study is *periodicity*: there exists a positive integer p such that $G_i = G_{i+p}$ for all positive integers i (e.g., [86, 121, 126]), so that $\mathcal{G}$ can be written as a finite sequence $\mathcal{G} = (G_0, \ldots, G_{p-1})^*$. We refer to such temporal graphs as *periodic graphs*. Periodic graphs can model, for example: how connections between transit systems vary or how the schedule of self-employed workers assigned to ressources, e.g. medical doctors assigned to hospitals or clinics, change on a weekly basis.

1.1.3 Distances

A first concern that appears when studying real-wold data coming from temporal networks is that there is an inconsistent notion of distance. How do we tell how far apart are two "temporal nodes"? By the number of hops required to reach the latter from the former (i.e. the graph distance)? By

[1]The use of one model over another often depends on the community it emerged from. Link streams and stream graphs were designed by researchers studying *complex networks*: huge datasets of interconnected datapoints without any assumed structure. As such, their problems are concerned with analyzing data. Time-varying graphs originate from research in distributed computing. As such, they are well-suited to analyze failures or variations in a network that influences independently acting entities. In our work, we focus on link streams and periodic graphs.

how long it takes to walk from one to the other (i.e. the duration of a walk)? Or a combination of both?

The notions of paths and distances are fundamental to the study of temporal networks. Kempe et al. [133] mention the use of time-respecting paths to study temporal networks. Because of the above-mentioned inconsistencies in the notions of distances, multiple different types of "time-respecting paths" (henceforth referred to as *journeys*) have been proposed over the years. In particular, shortest journeys use the minimal number of edges, in fastest journeys the difference in the times of the last and first used edges is minimal, while foremost journeys arrive the earliest at their destination (see for example [44]). Some researchers found an interest in combining different criterias. In particular, Latapy et al. [144] proposed to use *shortest fastest* journeys in their link stream model as a type of journeys that gather together the temporal as well as the structural information of a link stream. A shortest fastest journey is one that is shortest among the fastest journeys between two endpoints. This type of journey is used to define, for example, a *betweenness* centrality (see [164, 93]). A social network can thus be analyzed through different perspectives: using the *distance* to measure how the connectivity of a group varies over time, the *latency* to measure how quickly an information can spread into a group of people and the length of a shortest fastest journey to measure how efficiently this information is relayed. Shortest fastest journeys describe a natural notion of communication *efficiency*: when a viral rumour (such as a piece of disinformation) spreads over a network it can spread *quickly* and those actors on *short* fastest journeys from the source to any receiver can be considered as efficient spreaders.

1.1.4 Static and Mobile agents in (Temporal) Graphs

There are multiple problems about *mobile entities* in which the agents operate on graphs and temporal graphs under different conditions, such as GRAPH EXPLORATION, DISPERSION and GATHERING (e.g., [5, 40, 68, 70, 81, 104, 105] and [67] for a recent survey).

One common problem is the *pursuit-evasion* game, in which one team of agents tries to capture another team. A closely related variant is the *search problem* in which a team of agent tries to search for an object. Another closely related problem is the *decontamination problem* in which a team of agents seeks to decontaminate a graph. Thus, from a base mobile agent problem, by changing the objective slightly one can ask the agents to find

an object in a cave, to remove a leaky substance from it or to capture an escaping fugitive whose position may or not be known to the agents. Thus, search, exploration, decontamination and pursuit-evasion games are tightly connected. Fomin and Thilikos [90] present a history of graph searching up to around 2008, while Bonato [34] surveys pursuit-evasion games up to around 2022.

One of the simplest models of pursuit-evasion games on graphs is the Cops and Robber game [167, 38] in which a team of cops try to capture a visible fugitive, the robber, in discrete-time. Those games are frequently studied for the structures they help extract from graphs: families of graphs that are *copwin*, 2-*copwin*, etc. and the *cop number* is one among many graph parameters that defines a graph. Connections abound between cops and robber games and search problems and width parameters: treewidth, pathwidth, etc. Some researchers attribute the recent surge in interest for cops and robber games to their connections to a wide variety of applications such as artificial intelligence and distributed computing, see for example the recent paper of Gahlawat et al. [95].

Erlebach and Spooner [80] bridged the communities of cops and robber games and temporal networks by studying a game of Cops and Robber on periodic graphs. We come back to this game in Subsection 1.2.1.

1.2 Problems and Contributions

1.2.1 Cops and Robber Games in Temporal Networks

The game of Cops and Robber is a pursuit-evasion game played in turns, originally on a finite undirected graph G, between $k \geq 1$ cops and a single robber. There is perfect information. Initially, first the cops, then the robber, choose their positions on G. Then, in every turn each cop first moves to a neighbouring vertex or stays still, then the robber moves to a neighbouring vertex or stays still. The game ends and the k cops win if they ever step on the node occupied by the robber. The robber wins by forever evading capture.

This game was first described by Nowakowski and Winkler [167], and independently by Quilliot [172, 173] for $k = 1$. Later, Aigner and Fromme [6] extended the game to general values of k. The smallest integer $k \geq 1$ for which k cops can always capture the robber on G is called the cop number of

G, denoted as $c(G)$. We say a graph G is *copwin* if $c(G) = 1$. Determining whether $c(G) \leq k$ for an input of (G, k) is EXPTIME-COMPLETE in general [136]. Nevertheless, computing $c(G)$, finding good upper and lower bounds on this number and finding classes of graphs for which $c(G)$ is bounded (by a constant) are some of the main research objectives in this field.

Bonato and Nowakowski [38] summarize many results on the Cops and Robber game in their seminal book. See also the more recent work of Bonato [34]. We give more details about this game in Chapter 3.

Extending the game of Cops and Robber to periodic graphs is straightforward because it is generally played on a finite structure and a periodic graph $\mathcal{G} = (G_0, \ldots, G_{p-1})^*$[2] can be thought of as a finite sequence of finite graphs. The rules are easily extended. Initially, the cops first choose a set of nodes in V to occupy, then the robber chooses a vertex of V. Then, starting in G_0, first the cops, then the robber move to a node that is adjacent to their current positions. After both players (cops and robber) have moved in G_t, they start their next turn occupying their nodes in G_{t+1} where they will play their next move. The game ends and the cops win if and only if one cop can move on the node occupied by the robber in some snapshot. The robber wins otherwise.

Erlebach and Spooner [80] first introduced the game of cops and robber on periodic graphs by sending the input to an algorithm that solves *Reachability Games* [30]. This is unsatisfying first because it does not give a characterization of copwin periodic graphs in terms of the original input, thus it does not help understanding why a periodic graph might be copwin or not. Moreover, the temporal complexity of their algorithm is derived entirely from a call to another generic algorithm, so it might not be optimal.

Furthermore, so far not much else has been written about cops and robber games on periodic graphs. In particular, there are no general analytical results, in the style that are presented in Bonato and Nowakowski [38]. Thus, aside from executing an algorithm to determine if a specific periodic graph is copwin or not, one could not say anything related to the cop number of a periodic graph. This prevents one from computing or approximating the cop number of a specific periodic graph from some of its properties without

[2]Some authors, such as Fluschnik et al. [88], have questionned the periodic assumption of temporal graphs for cops and robber games and suggested that the game might be better defined on a temporal graph that is a finite sequence. This would impose a constraint that the robber should be captured in a certain amount of time, given by the length of the sequence, which is not generally assumed in cops and robber games.

having to do extra work. For example, we know from Aigner and Fromme [6] that $c(G) \leq 3$ if G is planar, which can be sufficient in some applications. There are yet no published results of this type in the literature, which is a problem that we seek to remedy in this work.

One motivation for studying cops and robber games on periodic graphs is that they, along with search problems more generally, are known to provide structural characterizations of graphs. This capacity to structurize graphs can be valuable to researchers on temporal networks given the surge of interest in those objects and that their classification is still in its infancy. Indeed, Casteigts' updated list of classes [52] still contains only 21 classes of temporal networks. In contrast, the cops and robber game provides an infinite number of temporal networks classes, one per cop number, and each with specific structure. Therefore, we wish to extend this classification with the help of the cops and robber game and, possibly, other properties of periodic graphs.

1.2.2 Link Streams

Link Streams offer an elegant framework to study how real-world networks change over time. However, at the time of publication, there was still an inconsistency in the way researchers computed "distances" on temporal networks. For example, Wu et al. [208] devised separate algorithms to compute lengths of shortest journeys and durations of fastest journeys, but there was no way of combining them. Furthermore, the latest algorithm that computed metrics from all sources to all targets dated back to Bui-Xuan et al. [44] in 2003. There was an obvious need to update this work on link streams. Moreover, Latapy et al. [144] specifically designed their *betweenness centrality* (see [164] for more explanations on network centralities), but this required a notion of distances that was a combination of shortest and fastest journeys. Thus, new algorithms were necessary to compute those values. Algorithms to compute this betweenness centrality are presented by Simard et al. [192].

1.2.3 Contributions

1.2.3.1 Characterizations of copwin periodic graphs

In Chapter 4, we characterize periodic graphs that are copwin and use this characterization to design an efficient algorithm to compute whether an input periodic graph is copwin or not. Our characterization involves a new struc-

ture called *augmented arena*. Our techniques apply to more than the classic game we have described. This characterization, presented in Theorem 4.3, is a process that involves adding edges to the "augmented arena" of an input periodic graph $\mathcal{G}$. We show that this process terminates with a specific output if and only if $\mathcal{G}$ is copwin. We then build from this characterization to describe precisely how this process works and when we can declare it has terminated, culminating in Algorithm 1. Notably, this algorithm is faster than the method presented by Erlebach and Spooner [80].

This is significant since the only way so far to determine if a periodic graph was copwin or not was to use Erlebach and Spooner's transformation into a reachability game, which is not expressive. This transformation works from an algorithmic perspective, but it does not explain why a particular periodic graph is copwin or not. We help remedy this with our results in Chapter 4.

1.2.3.2 Cop numbers of periodic graphs

In Chapter 5, our main contributions are many examples of periodic graphs with particular cop numbers that altogether highlight the elusive structure of periodic graphs, especially compared to static graphs. For example, we show in Proposition 5.1 that there exists a periodic graph $\mathcal{G}$ with outerplanar footprint G that has cop number 3. This is despite the known fact that outerplanar graphs have cop number at most 2 [60]. We similarly exhibit in Proposition 5.2 a periodic graph $\mathcal{G}$ with cube footprint G that also has cop number 3, even though G has cop number 2 by [153]. In Table 5.1, we present a table of periodic graphs and associated triples (a, b, c) such that the cop number of G, the highest cop number of all snapshots and the cop number of the periodic graph are, respectively, a, b and c. This table is filled with almost all possible sequences of integers between 1 and 3 (inclusive).

This chapter is a major contribution to the field for how it can help researchers build intuition on what properties of periodic graphs to look for in connection with the cops and robber game, and to determine what pitfalls to avoid when studying this problem.

1.2.3.3 Bounds on the cop number of periodic graph families

Our main contributions in Chapter 6 are general results on the cop numbers of periodic graphs. Those mainly come in the form of upper and lower bounds.

The spirit of this chapter is embodied by our main result, Theorem 6.4, that relates the treewidth of a graph G to the maximum cop number of all periodic graphs with a footprint G. Such results are so far non-existant in the literature and they provide a lot of information on how the footprint affects the dynamics of a periodic graph. The upper bound presented in Theorem 6.4 can be compared with some upper bounds on cop numbers of graphs or directed graphs that already in the literature such as those presented by [50, 148, 129]. Most results in this chapter are upper bounds.

Such results can have a major impact in the nascent study of the structure of temporal graphs as well as cops and robber games on periodic graphs, since they provide explicit connections between the footprint of a temporal graph and the temporal graph itself.

1.2.3.4 Evaluating metrics in link streams

Our main contributions in Chapter 7 are two groups of algorithms that compute metrics of shortest (fastest) journeys in a link stream. Two algorithms in the first group work from a single source, while the two in the latter work from multiple sources. The two sets of algorithms are split on whether they assume journeys have strictly positive or null *delays*: the time it takes to traverse an edge. All methods return the lengths of shortest journeys, the lengths of shortest fastest journeys as well as pairs of starting and arrival times of (fastest) journeys. Some of that information is summarized as *reachability triples* (a, b, c) (see Definition 7.1) such that for every fixed source node s and node v, there exists a time b such that a is a largest starting time from s to (b, v) and c is the *distance* from (a, s) to (b, v). There are three major novel aspects in this work. First, we compute multiple metrics at once, whereas many other authors devise separate algorithms for the same tasks. Second, we compute lengths of shortest fastest journeys, which is a novel metric in the literature. Finally, we present algorithms that work from multiple sources. This had not been considered in the recent years before publishing in the literature and one had to go back to the work of Bui-Xuan et al. [44] to find similar algorithms that work with multiple sources. In short, our major contributions are:

1. To provide four algorithms each of which compute multiple known metrics;

2. To design algorithms that compute the lengths of shortest fastest journeys;

3. To design algorithms that compute metrics from multiple sources.

A major impact of this study is the ability to efficiently compute values required to compute the betweenness centrality of a temporal node in a link stream. It also makes it faster to compute shortest journeys between large numbers of nodes in link streams. The novel ability to compute reachability triples for all destinations (and sources) is relevant due to the importance of shortest journeys in the analysis of temporal networks. We think our algorithms are also simple yet powerful enough that they could be extended to other metrics such as arrival times of foremost journeys and lengths of shortest foremost journeys. This seems to hold in particular if the temporal dimension is the first argument optimized over, such as in shortest fastest journeys or shortest foremost journeys.

1.2.4 List of Publications

The work in Chapter 4 was presented at the *30th International Colloquium on Structural Information and Communication Complexity* (SIROCCO) in 2023, [65].

Some results in Chapter 5 and Chapter 6 were presented at the *54th Southeastern International Conference on Combinatorics, Graph Theory & Computing* in 2023 [2], before being submitted to their proceedings[3]. Some results were also presented at the *11th Workshop on GRAph Searching, Theory and Applications* (GRASTA), in Bertinoro, Italy, in 2023 [1].

The work in Chapter 7 was presented first at the *International Conference on Advances in Social Networks Analysis and Mining* (ASONAM) conference in Vancouver, Canada in 2019 [189], then published as a longer version in the journal *Social Networks Analysis and Mining* (SNAM) [188].

1.2.5 Overview

We start by reviewing the literature in Chapter 2 and giving general definitions in Chapter 3.

[3]The papers are evaluated after the end of the conference.

In Chapter 4, we give our characterization of copwin periodic graphs in terms of *augmented arena*.

In Chapter 5, we initiate the construction of a list of examples of periodic graphs with specific cop numbers.

We then move on to Chapter 6 that presents more general results, mostly in the form of upper and lower bounds on the cop numbers of periodic graphs.

We present our work on link streams in Chapter 7.

We conclude in Chapter 8.

Chapter 2

Related work

2.1 Temporal Networks

The history of temporal networks, or temporal graphs[1], is recent. It has been brewing in the last two decades in communities of, for example, Network Science [21, 164] and Distributed Computing [183] as researchers recognized that empirical networks evolved over time and that tools to analyze them had to be adapted. Casteigts et al. [55] (see also Casteigts [52]) provide a survey on those developments, as do Holme and Saramäki [114] and Latapy et al. [144]. The community on Complex Networks studies *metrics* [165], *centralities* such as the *betweenness* and *closeness* [155, 10] and *community detection algorithms* [91] in order to analyze real-world (temporal) networks. On the other hand, the community of distributed algorithms use the temporal dimension to model communication networks and failures that can occur on them.

Temporal networks have been the focus of an increasing number of studies in recent years, as one can see by the number of workshops and conferences dedicated to the topic, such as *SAND 2023* ("2nd Symposium on Algorithmic Foundations of Dynamic Networks"), the "Temporal Graph Learning Workshop" at *NeurIPS 2022* as well as the workshop "Algorithmic Aspects of Temporal Graphs VI*" at *ICALP 2023*.

[1]Some authors like Holme and Saramäki [114] use the term *Temporal Networks* as a general description of a graph that varies over time. The term *Temporal Graphs* sometimes refers to a specific object in which time is discrete. We will define the term *temporal graph* in Chapter 3 and use use *temporal network* as the generic expression.

A different area of investigation concerns dynamic grahs in the context of algorithms, graphs and data structures. For instance, people have studied problems where we want to maintain a data structure which keeps track of the connectivity of a graph as it changes over time. For example, edges can be deleted and/or inserted over time and we do not know about the sequence of delete/insert operations at the beginning. See for example the survey of Eppstein et al. [78], as well as [127], [74], [150]. Let us stress that although the expression *dynamic graph* is used in that field, this refers to an area of investigation that is unrelated to our work.

2.1.1 Applications and Motivations

Temporal networks have been widely used in computer networking under the name of Delay-Tolerant, Opportunistic and (Mobile) Ad-Hoc Networks. In general, delay-tolerant networks (DTN, also known as disruptive-tolerant networks) are considered as highly dynamic networks that might never be connected at any point in time. Rodrigues and Soares [179] summarize the topic as follows:

> DTN is a network research topic focused on the design, construction, performance evaluation, and application of architectures, services, and protocols that intend to enable data communication among heterogeneous networks in extreme environments [...].

Delay-tolerant networks can suffer from poor connectivity, low reliability and long transmission delays. Thus, routing information on those networks is more challenging than on static networks. A delay-tolerant network can arise as a network of devices carried by pedestrians that can forward messages. Sending a message from one device to another requires a protocol that is robust to sudden changes in connectivity. Rodrigues and Soares [179] mention that delay-tolerant networks were originally designed to communicate with spacecraft. We refer to the book of Rodrigues [178] for more information on the subject.

Another field of study in which temporal networks have become important is the analysis of real-world (complex) networks. Information on how connections are ordered over time is valuable in the study of dynamic processes on networks such as the propagation of diseases, viruses and information. For example, if a person is infected with a virus at some time t, then stays hidden for the duration of its contagious phase before resuming contact with

the world at a later time $t' > t$, then they cannot infect anyone. This information is impossible to infer from a static network of contacts. Social networks arise from the dynamics of social interactions in groups of people, which according to Newman [164] was studied as early as the 1930s. Other networks that arise from real data are technological networks such as the above-mentinned delay-tolerant networks, the internet, the network of natural gas pipelines, networks that describe the chemical reactions that occur in cells, networks of interactions between proteins or even network of connections between neurons in the brain. An interesting piece of knowledge one can discover on real-world temporal networks is that networks of contacts (e.g. of phone calls) have been found to be often bursty [20, 115]. Steven Strogatz [196] summarized the subject in a Nature publication many years ago. We refer to Barabási [22] and Newman [164] and the references therein for more information and examples on complex networks. The key takeaway here is that most of those networks change over time, whether because the segment of a pipeline fails at a particular time or a protein is simply not interacting with any other protein. Thus, in the same way that one can follow the evolution of a random variable, such as the price of an asset, over time in the form of a time series, one can keep track of the evolution of a network as a *temporal network*. A classic reference on the subject of temporal networks is the text of Holme and Saramäki [114].

Recent trends in the study of temporal networks include Deep Learning. Many researchers have recognized the successes of Graph Neural Networks in, for example, identifying molecules found in food that can help beat cancer [203]. Graph Neural Networks are Neural Networks adapted to take graphs as inputs, we refer to the book of Goodfellow et al. [103] for a general discussion on Deep Learning and the book of Ma and Tang [151] for a more specific exposition on deep learning on graphs. See also, for example, Rossi et al. [181] for a description of Temporal Graph Networks, a model of deep learning on temporal networks, as well as Kazemi et al. [130] that present a survey on using temporal networks in deep learning.

Finally, a core concern about temporal networks is our lack of knowledge about their structure, especially when compared to (static) graphs. For example, whereas the treewidth is well-known (static) graph parameter (see [71] for explanations), there is still no good notion of "treewidth" that applies to temporal networks according to Fluschnik et al. [88].

2.1.2 Models

One difficulty in working with temporal networks is the lack of uniformity in the teminology. Researchers employ a variety of models, from Temporal Networks to Time-Varying Graphs [55] and it can be hard to keep track of them. The term *Temporal Network*, as used by Holme and Saramäki [114] seems to refer to any network that changes in time. Casteigts [52] (who uses *Dynamic Networks* instead for generality) notes that the term *Dynamic Graphs* describing a sequence of graphs has been around since the 1980s (see for example Shiloach and Even [187]).

The model of *Temporal Network* as a graph $G = (V, E, \lambda)$ with a function $\lambda : E \to \mathbb{R}$ that indicates at what time each edge appears is originally due to Kempe et al. [133]. In this model, a temporal network is represented as a graph with labels on the edges and it is still being used, for example in the work of Casteigts et al. [58]. This representation highlights the footprint (see Chapter 3) but makes the temporal dimension harder to visualize. Temporal networks as a sequence of graphs was also deemed *Evolving Graphs* by Ferreira and Viennot [85]. The first author and their coauthors would later incorporate the duration of each graph in the sequence under the label of evolving graphs, as well as the time it takes to cross an edge [44]. Kostakos [140] is credited by Casteigts [52] for a particular usage of the term *Temporal Graph* to describe a static representation of a discrete-time temporal network. This provides a description of a discrete-time temporal network with a single static directed graph, which is similar to what we will call *arena* Subsection 3.1.3. The term temporal graph is sometimes used interchangeably with temporal networks.

The *Time-Varying Graph* (TVG) framework of Casteigts et al. [55] provides a unifying model to describe many forms of temporal networks. Whether edges have non-zero, integer or real duration and whether crossing an edge takes a certain time is encompassed by this framework. Still more recently, Latapy et al. [144] introduced the *Link Stream* model of temporal networks. A link stream can be thought of as a set-theoretical version of a specific type of TVG, in the sense that the dynamics of a TVG are described by functions $\rho : E \times \mathcal{T} \to \{0, 1\}$ and $\zeta : E \times \mathcal{T} \to \mathbb{T}$, where $\mathbb{T}$ is often taken to be either $\mathbb{N}$ or $\mathbb{R}$ and $\mathcal{T} \subseteq \mathbb{T}$ is the lifetime of the network, to describe the times of appearances of an edge and the time it takes to cross it, respectively. In comparison, a link stream $L = (T, V, A)$ is a triple where T is a set of time instants (often $T \subseteq \mathbb{R}$), V is a finite set of vertices and A is a set of *temporal*

edges[2] of the form $(t, uv) \in A$, where uv denotes the unordered pair of vertices u and v and $t \in T$. This allows one to write (I, uv) with $I \subset T$ for the set of temporal edges $\{(t, uv) \in A \mid t \in I\}$. The time it takes to cross an edge, i.e. the *delay* $\gamma \in \mathbb{R}$, is often fixed in this model. Latapy et al. extended those ideas to *Stream Graphs* in the same work to allow nodes themselves to appear and disappear. We will have more to say about link streams in Section 3.2.

One common way to analyze temporal graphs that are not periodic is through a time-window: a contiguous subsequence of snaphsots of fixed length, introduced by Viard et al. [204]. We refer to the recent survey of Klobas et al. [138] for more information about this way of dealing with temporal graphs.

Other assumptions one can encounter about a temporal graph $\mathcal{G} = (G_0, G_1, \dots)$ include:

1. T-Interval connected: for every time t, the graph $H = (V, \bigcap_{i=t}^{t+T-1} E(G_t))$ is connected;

2. Recurrent: for every edge $e \in E(G)$ and time t, there exists a time $t' > t$ such that $e \in E(G_{t'})$;

3. δ-Recurrent: for every edge $e \in E(G)$ and time t, there exists a time $t' \in [t, t + \delta - 1]$ such that $e \in E(G_{t'})$.

Those assumptions are described in detail in [55].

2.1.3 Distances in Link Streams

In Chapter 7 we expose algorithms we developped to compute metrics in link streams. This study is related to the study of Wu et al. [208] and our algorithms can be applied in the same contexts as their shortest and fastest journeys methods. The main contribution of the present work is to compute *sf*-metrics, as well as distances and latencies, in a single pass over a dataset. Separately, Wu et al.'s fastest and shortest journeys methods are insufficient to compute centralities such as the betweenness of a link stream, while an algorithm combining them to produce *sf*-metrics is not efficient

[2] We write A instead of E to avoid ambiguity with models in which E is a set of edges and there is a presence function that assigns time instants to them.

because it requires iterating multiple times over the dataset. Meanwhile, our methods iterate only once over the dataset to produce the three metrics and are suitable for studying different aspects of a link stream. We also output information on the starting and arrival times of shortest (fastest) journeys. This study was instigated as a first step in computing Latapy et al.'s betweenness centrality defined in Latapy et al. [144] and computed in Simard et al. [192].

Bentert et al. [26] devised a generic algorithm to compute optimal journeys in temporal graphs from one source node to all other destinations. Their algorithm is generic in the sense that it can compute different types of optimal journeys such as shortest, fastest and foremost. Their main algorithm computes those journeys separately. They do consider a method to combine some optimization criteria, such as fastest and shortest, however this is done through a linear combination of optimization objectives. This is in contrast to the present work, which is focused on a bilevel optimization approach (see Colson et al. [62]): the criterion that a journey be fastest can never be violated, at the possible expense of journeys not being shortest. We also present algorithms from multiple sources, which is not the case in their work. Brunelli et al. [42] took a similar approach to Himmel et al. and devised a generic algorithm to compute Pareto optimal journeys to generalize multiple criteria (shortest journeys, foremost journeys, etc.). Again, their method only works from a single source. Moreover, they present the temporal complexity of their method but do not derive the explicit complexities when their method is applied to specific problems. In particular, it is not clear what complexity their method would achieve on the task of computing lengths of shortest fastest journeys and how a concrete implementation would fare against our programs. In 2019, Li et al. [145] improved on the work of Wu et al. by using dynamic programming approaches to compute shortest journeys, fastest journeys and (restricted) earliest-arrival journeys. Although interesting, this work comes with the same restrictions that we observed in the work of Wu et al.

Furthermore, this work is also close to Tang et al. [199] since these authors define a *betweenness* centrality on temporal networks in terms of fastest shortest journeys. Whether to use fastest shortest or shortest fastest journeys (or any other type of journey that combines temporal and structural information) depends on what information one wants to emphasize and on the context of the study. Shortest and fastest journeys were also studied by Bui-Xuan et al. [44] and we were inspired by their all-pairs fastest journey

method to develop Algorithm 3 and Algorithm 5. The latter are relevant to compute some centralities because metrics between all pairs of (temporal) nodes may be required. To our knowledge, Xuan et al.'s method is the only of its kind to return latencies between all pairs of nodes. The same problem of computing shortest and fastest journeys has been solved, with different strategies, in a distributed way by Casteigts et al. [54]

Casteigts et al. [55] also offer a survey of temporal networks that includes many applications of shortest and fastest journeys. In particular, such journeys can be used to study the reachability of a temporal node from another. Although simple to formulate on graphs, in temporal graphs one can define multiple notions of reachability and some are still being investigated, such as reachability when there are constraints on the allowed waiting time at each node [43]. It appears from the survey of Casteigts et al. [55] that either the distance or the latency is often used as a temporal metric to evaluate how well a temporal node can communicate with another. In this regard, the *sf*-metric can be used as another temporal function since it combines the temporal as well as the structural information into a single map. Note that the notion of *foremost* journeys (or journeys) is also used by some authors (such as Casteigts et al. [54]) to study temporal reachability. A foremost journey only has minimal arrival time, while its starting time is unconstrained. This type of journey is also useful in many studies and we expect that our algorithms can be extended to those cases to output lengths of *shortest foremost* journeys. Finally, Casteigts et al. [56] and Thejaswi et al. [200] studied another type of temporal journeys called *restless*, such that one cannot wait more that a prescribed amount of time on each node. This is an interesting constraint that we have not considered. However, we expect that our methods can also be extended to fit these constraints.

Finally, observe that the link stream framework is also close to the Time-Varying Graphs framework of Casteigts et al. [55]. Thus, all results presented in this paper carry to this other framework as well. We did not intend to survey the different models of temporal graphs present in the literature, as this is out of the scope of this work. To the best of our knowledge, there are two major models that allow edges to have duration and allow time to flow continuously: the Link Stream of Latapy et al. and the Time-Varying Graph of Casteigts et al. Other models, such as temporal networks (see Holme and Saramäki [115]), evolving graphs (Ferreira and Viennot [85]) and temporal-event graphs (Mellor [160]) mostly focus either on discrete timesteps or on zero transmission delay. Our algorithms also work on those models. The

time-dependent network model mentioned by Brunelli et al. [42] is slightly more general than the link stream we use since it allows the transmission delay over the edges to follow a non-constant function.

2.1.4 Tractability and Width Measures

Orlin [169] showed that many temporal analogues of NP-COMPLETE problems on graphs are PSPACE-COMPLETE, so many graph problems become harder when extended to temporal graphs.

For example, Akrida et al. [8] studied extensions of the Vertex Cover problem, NP-COMPLETE on graphs, to temporal graphs where they show, among other results, that it is NP-COMPLETE to solve on temporal graphs whose footprints are stars. Baste et al. [24] define a notion of temporal matching on link streams and show that computing a maximum matching in this case is NP-HARD under some restrictions, even though finding a maximum matching can be solved in polynomial-time on graphs. Casteigts et al. [57] show that some types of journeys are W[1]-HARD (see for example Downey and Fellows [73] for an overview of parameterized complexity) to find under some restrictions, for some parameters. In the same vein, Kempe et al. [133] showed that computing the number of node-disjoint s-t paths in temporal graphs is NP-COMPLETE, yet it is known to be polynomial-time computable on graphs via Menger's Theorem (see, e.g., [71]). Earlier, Berman [29] noted that Menger's Theorem fails on temporal graphs. To help alleviate this issue, Kempe et al. [133] define a graph G as *Mengerian* if Menger's Theorem holds for every temporal graph with footprint G. Trying to find a good adaptation of Menger's Theorem is still the subject of recent investigations [118].

Similarly, Marino and Silva [157] showed that a temporal analogue of the problem of finding an Euler walk on a graph is NP-COMPLETE, yet it is polynomial on graphs. Akrida et al. [7] and Bumpus [45] showed that the temporal analogue of exploring a star is NP-COMPLETE.

One modern idea when studying a NP-HARD problem P on a graph G is to look for a *fixed-parameter tractable* (FPT) algorithm for P *parameterized* by a measure of width of G, such as the *treewidth* ([177], see also [71] for explanations and Section 6.2 for a brief introduction), of G. For example, Downey and Fellows [73] present a linear-time algorithm for the INDEPENDENT-SET problem[3] on graphs of bounded treewidth.

[3]The INDEPENDENT-SET problem takes as input a graph G and a positive integer k

With this in mind, Bumpus [45] defined a novel width measure, *interval-width*, of temporal graphs that he used to show the existence of FPT algorithms for the above problem of exploring the star when parameterized by the interval-width of the temporal graph. In our notation (see Chapter 3 for full definitions, or Chapter 1 for an overview), let $\mathcal{G} = (G_0, G_1, \ldots, G_T)$ be a temporal graph with footprint G that is either periodic or ends at time T and for every edge $e \in E(G)$, let $f(e) := \min_{0 \leq t \leq T : e \in E(G_t)}(t)$ and $l(e) := \max_{0 \leq t \leq T : e \in E(G_t)}(t)$ be the first and last appearance times of e (respectively). Then, the *interval membership sequence of* $\mathcal{G}$ is the sequence $(F_t)_{0 \leq t \leq T}$ of subsets $F_t := \{e \in E(G) \,|\, f(e) \leq t \leq l(e)\}$ and the *interval-membership-width* of $\mathcal{G}$ is the integer $\text{imw}(\mathcal{G}) := \max_{0 \leq t \leq T} |F_t|$. That is, each F_t contains all edges that have appeared sometime before t and have not yet disappeared forever, and $\text{imw}(\mathcal{G})$ is the size of the smallest such set. This functions thus quantifies how spread out in time are the edges of $\mathcal{G}$.

Unfortunately, $\text{imw}(\mathcal{G})$ is not suitable for our context because of its "coarseness". That is, if all edges appear in a snapshot G_t, then $\text{imw}(\mathcal{G}) = |E(G_t)| = |E(G)|$, which does not help us analyze our game of Cops and Robber. Based on this definition, many of our examples of periodic graphs $\mathcal{G}$, with footprint G, in Chapter 5 will have interval-membership-width $|E(G)|$. Nevertheless, the interval-membership-width is so far the first usable width measure of temporal graphs.

More abstract width measures were presented very recently in [47, 11, 48], however they use constructions from Category Theory. Category Theory, see for example [176, 180], is a branch of mathematics whose purpose is to abstract other fields of mathematics (such as topology, group theory, set theory, graph theory, and so on) by studying objects and their relations. There is thus, for example, a category of sets that contains sets and morphisms on them and a category of graphs that contains graphs and homomorphisms. Bumpus [45] generalized, via category theory, an algebraic definition of the *treewidth* based on a graph function due to Halin [109]. See also the works of [11, 48] that follow from [45, 47]. This allows Bumpus et al. [48] to recover graph width measures such as the usual *treewidth*, but also the *complemented treewidth* [75], the *graph $\mathcal{G}$-decomposition width* [51], the *layered treewidth* [76, 185] and the *$\mathcal{H}$-treewidth* [125]. More recently, Bumpus et al. [46] pushed further into category theory to model large swaths of *time-varying data*, which

and asks whether or not there is a subset of vertices of size k all of which are pairwise non-adjacent.

includes temporal graphs. This opens the door for a future notion of width measure of temporal graphs from category theory.

On a similar note, Fluschnik et al. [88] studied temporal extensions of the *treewidth*, noting that there is yet no perfect analogue of this parameter in the temporal setting. This explains why, in Theorem 6.4 we use the treewidth of the footprint G as an upper bound on $c(\mathcal{G})$ instead of any "temporal treewidth".

As we saw, the decision problem associated with computing the cop number of a static graph is EXPTIME-COMPLETE [136]. Based on the results we saw on temporal graphs, we can expect this already difficult problem to be no easier on temporal graphs.

2.1.5 Mobile Agents

Claude Shannon originated the study of moving entities on graphs with his maze-solving machine [186], and since then researchers have wondered how to add more agents to this setting, which requires assumptions on how the agents are controlled. Assuming the agents are fully independent leads to the field of distributed computing by mobile agents, and we refer to the survey of Flocchini et al. [87] for an overview of this field. The motivations for studying mobile agents come about organically, for example a group of independant robots might have to coordinate together to capture an intruder [23] or to form a geometric pattern [197].

The traditional problems to solve with mobile agents on graphs are EXPLORATION, GATHERING and DEPLOYMENT (see for example [149]). In EXPLORATION, the agents are tasked with fully exploring the graph. GATHERING asks for the agents to regroup, while in DEPLOYMENT the agents must spread on the graph under some constraints. More modern problems include the BLACK-HOLE SEARCH introduced by Dobrev et al. [72]. In this problem, agents scour a graph in search of a node that destroys all information, such as a virus that is stationary on the graph. NETWORK DECONTAMINATION has also been vastly studied and we will have more to say about it in Section 2.2.

Temporal networks were used in distributed computing (see [183] for an introduction) as early as in the work of Awerbuch and Even [15]. One common natural source of the dynamics of temporal networks comes from the possibility of the network to be *faulty*. Faults are often studied as properties of robots, as in a robot might crash (see, e.g., Défago et al. [66]). However, the network itself can also be faulty, for example because of the presence of

black holes. Moreover, in asynchronous settings, when an entity sends a message along the network to other entities there is no guarantee that message is ever received.

The temporal assumption of *periodicity* was already present in the works of Flocchini et al. [86] and Ilcinkas and Wade [122] where periodic graphs model the periodic movements of mobile carriers such as subways, satellite systems or patrolling guards.

The 1-interval connected (always connected) assumption is used in some studies, such as [69, 119] and some also assume a specific footprint [120]. There are also two main contexts in which to study problems of mobile agents in temporal networks, according to Luna [149]: the postmortem model and the live model. In the postmortem model, the agents completely know the temporal graph $\mathcal{G}$. In the live model, some informations about $\mathcal{G}$ are unknown to the agents, for example they might only know the footprint G. The live setting was used in the cops and robber games by Balev et al. [18], where it was called "online".

The community on mobile agents has a longer tradition of studying dynamic topologies than the community on cops and robber games. Indeed, temporal graphs were introduced to cops and robber games by Erlebach and Spooner [80], while Luna [149] mention multiple studies on mobile agents years before that [14, 3, 25, 32].

To see how the dynamics of a temporal graph affects the time complexity of a distributed problem on a graph, consider the following results. We focus on the problem of EXPLORATION in postmortem setting, for simplicity. Michail and Spirakis [161] showed that given a 1-interval connected temporal graph $\mathcal{G}$ with finite lifetime and a single agent positionned on a node v, deciding whether this agent can explore $\mathcal{G}$ in the smallest amount of time is NP-COMPLETE. Erlebach et al. [79] showed that for every integer $n \geq 1$, there exists a 1-interval connected temporal graph $\mathcal{G}$ with $2n$ nodes such that the least amount of time it takes for a single agent to explore $\mathcal{G}$ is $\Omega(n^2)$. Ilcinkas and Wade [120] showed that given a 1-interval connected temporal graph $\mathcal{G}$ whose footprint G is a cycle with n nodes, an agent starting anywhere on G can explore the cycle in at most $2n - 2$ timesteps. The same authors also give explicit bounds on the number of rounds required to do this when $\mathcal{G}$ is T-interval connected. Erlebach et al. [79] showed that there exists a temporal graph with footprint a planar graph with n nodes and maximum degree 4 that requires $\Omega(n \log n)$ timesteps to explore. Aaron et al. [4, 3] show that it is impossible to approximate the minimum number of timesteps required

for a single agent to explore a recurrent temporal graph in general. Observe that the main assumptions are very simple: assumptions on the schedule (e.g. recurrent), the temporal connectivity (such as 1-interval connected) and assumptions on the footprint.

The live exploration setting is also difficult to solve. Luna [149] present many result on this case. We only mention the following result of Ilcinkas and Wade [120]. In the live exploration setting, a δ-recurrent and T-interval temporal graph whose footprint is a cycle with n nodes can be explored by a single agent in at most $n-1+\left\lceil \frac{n-1}{\max(1,T-1)} \right\rceil (\delta-1)$ timesteps by *stubbornly walking* along the cycle. In this context, an agent is said to perform a stubborn walk if it simply chooses a direction and crosses every edge as they appear. We use the idea of stubborn walks in this thesis multiple times, for example in Proposition 5.1. A slightly more general definition is given in Subsection 3.4.3. Surprisingly, such a simple idea can be nearly optimal: Ilcinkas and Wade [120] show that in their context, $n-1+\left\lfloor \frac{n-3}{\max(1,T-1)} \right\rfloor (\delta-1)$ timesteps are necessary to explore such a temporal graph.

2.2 Games

2.2.1 Games and Pursuit Evasion

Researchers in Game Theory, see for example [198] for a general overview of this field, typically assume that all agents in the same team are centrally controlled. Broadly speaking, Game Theory is the mathematical study of settings in which agents, or players, must each reach a personal goal under prespecified rules, and may either compete or cooperate to do so. Researchers look for *strategies*, when they exist, to assign to each player for them to reach their objective. Said strategies may be deterministic or not. One generally looks for an "optimal strategy". This is traditionally the point of view taken by researchers in Cops and robber games. Some explicitly conceptualize cops and robber games as games in which two opposing players each control a set of tokens: one player controls a set of tokens representing the cops and another controls tokens representing the robber. Then, each player moves its tokens according the the rules of the game. Thus, cops and robber games are often compared with more general games such as *Reachability Games* [131, 30] and *Stochastic Games* [100, 191]. Bonato and Nowakowski [38] also note the connection between cops and robber games and *Combinatorial Games*

[28].

Game theory in general has wide-ranging applications. For example, automatic auctions are conducted to determine the placement of publicities on websites ([156, 182]), evolutionary game theory helps understand the competition and cooperation in animals and plants [110], it can help understand how to better allocate water resources [154] and how to better secure networks [146].

One of our main subjects in this thesis, the Game of Cops and Robber, can be simultaneously conceptualized as a problem from game theory or as a problem of mobile agents. The language of game theory has mostly been left aside by those studying cops and robber games on graphs, a fact noted by Kehagias and Konstantinidis [131] who showed that, fortunately, the discussion of *optimal deterministic strategies* is mathematically grounded.

We refer to the bibliography of Fomin and Thilikos [90] for the early history of pursuit-evasion games on graphs as well as the survey by Nisse [166]. We note from this work the early connection with differential games (a type of game in which the position of the players depend continuously on time [174]) that are used to model, for example, missile guidance and autonomous systems [83]. This connection with differential games was noted early by Isaacs [124] in his study of the Homicidal Chauffeur problem: a slow but agile runner must maneuver in order to prevent a fast, but less agile, driver of a vehicle from running him over. Another instance of a pursuit game whose study involves differential games is the Lion and the Man game in which a man must escape a lion in the plane (see again [174]).

Researchers on pursuit-evasion games typically focus on contexts based on specific assumptions, of which the main ones are:

1. Is the space discrete or continuous;
2. Is time discrete or continuous;
3. Is the evader visible to the pursuers;
4. Are the players moving randomly or deterministically.

We focus on discrete space (graphs), discrete time (turn-based) games in which the players act deterministically. We will mention some variants for completeness.

On graphs, one alternatively speaks of *graph searching* or *pursuit-evasion games* depending on whether the target is invisible or visible (respectively).

In many cases, the parameter of interest is called the *search number*: the minimum number of searchers required to either search a graph or capture an evader on a graph[4].

Graph searching originally dates back to Golovach [102, 101]. In this game, the searchers are allowed to slide along edges. Kirousis and Papadimitriou [137] introduced a variant in which the searchers are moved along nodes, without sliding on edges. Seymour and Thomas [184] extended this game to the case of a visible fugitive. This paper is celebrated as presenting an alternative description of the *treewidth* of a graph in terms of a cop number. Similar characterizations exist for different contexts, for example: the pathwidth of a graph equals a search number ([137, 135, 77]), the D-width of a directed graph equals a cop number [82] and the DAG-width of a directed graph equals a cop number [31]. We also mention the recent work of Toruńczyk [201] that generalizes the treewidth with a connection to cops and robber games. The survey of Ganian et al. [97] takes a critical view of the applicability of width measures for directed graphs.

To the best of our knowledge, no work has been done on graph searching (i.e. with an invisible fugitive) in temporal graphs. The closest problem to graph searching that has been brought to the temporal setting would be EXPLORATION, see [79, 49], which can be thought of as a highly unconstrained form of graph searching.

2.2.2 Cops and Robber Games

The game of Cops and Robber was first studied by Nowakowski and Winkler [167] and, independently, by Quilliot [172][173]. In the decades since its formulation, many variants have been suggested on the original rules of the game, starting with allowing more cops to play (see Aigner and Fromme [6]). The game was studied on specific graph families, such as planar graphs [6]. A good summary on the first 30 years of development on the subject is the book by Bonato and Nowakowski [38], while the survey of Fomin and Thilikos [90] gives a broader perspective about graph search problems.

[4]Formally, every variant of the problem has its own associated search number, such as the edge search number, the node search number, the connected search number, the cop number, the helicopter cop number, the lazy cop number etc. For simplicity, we refer to search number here as a general term that encompasses all variants of graph searching and to cop number as a general term for all variants of pursuit-evasion games. We will specify later the variants used.

The cop number $c(G)$ of a graph G is the smallest number of cops required to capture a robber on this graph. It was shown by Kinnersley [136] that its decision variant is EXPTIME-COMPLETE to compute. Some researchers provide exact algorithms to determine if a graph G has cop number less than or equal to some integer k, which is solvable in time polynomial in n in k. The fastest algorithm so far has temporal complexity $O\left(kn^{k+2}\right)$ [170].

Many researchers focus on determining the exact cop number of specific graph families, for example Clarke [60] showed that outerplanar graphs have cop number two. This came after Aigner and Fromme [6] showed that planar graphs have cop number 3. Similarly, Maamoun and Meyniel [153] show that $c(Q_k) = \left\lceil \frac{k+1}{2} \right\rceil$ where Q_k is the hypercube of dimension k. Others devise upper or lower bounds on the cop number, either for graph families or for general graphs. Hill [113] uses a block decomposition of a graph to give upper and lower bounds on $c(G)$.

One important concept is that of the *retract*. A *retraction* $h : G \to H$ is a homomorphism from G to one of its subgraph H such that $h(H) = H$. The graph H itself is called a retract of G. Retracts are used for example to show that graphs with cop number one (also called *copwin*) are exactly the *dismantlable* graphs [167]. In particular, Berarducci and Intrigila [27] showed that whenever H is a retract of G, $c(H) \leq c(G)$ so the cop number is upper bounded under taking retracts (this is not the case for general subgraphs). This last result was used by Baird et al. [17] and Turcotte and Yvon [202] to prove the size of the smallest graphs with cop number 3 and 4 (respectively). Note that some researchers define retracts on *oriented* graphs without mentioning homomorphisms, see Gahlawat et al. [95], although this seems uncommon. Gahlawat and Zehavi [96] also use a function like a retraction to analyze the parameterized complexity of cops and robber games, but without mentionning retractions.

Joret et al. [129] noticed an interesting connection between tree decompositions and Cops and Robber game and showed that $c(G) \leq \mathrm{tw}(G)/2 + 1$. They also show that every P_l-free graph has cop number at most $l - 2$ for every $l \geq 3$, and that every graph with no induced cycle of length at least l has cop number at most $l - 2$. Sivaraman [194] slightly improved the latter result, replacing $l-2$ with $l-3$ when $l \geq 4$. On the other hand, Chudnovsky et al. [59] focus on P_5-free graphs. The study of the cop number of graphs without a specific induced subgraph was initiated by Andreae [12].

A famous open problem is Meyniel's conjecture that the cop number of

a graph with n nodes is $O(\sqrt{n})$. This conjecture was first mentionned by Frankl [92] and Bonato and Nowakowski [38] dedicate a full chapter to it.

The capture time of the robber is the smallest number of turns required to catch the robber, when this is possible. It is a subject of study of its own, akin to the cop number and researchers produce the same types of result: exact results on graph families and lower/upper bounds. A notable result states that the capture time of any copwin graph with n nodes is at most n, due to Clarke and Nowakowski [61].

The cops and robber game on directed graphs was studied by Hahn and MacGillivray [107] and the authors presented an algorithm to solve it. However, they could not provide any structural characterization of directed copwin graphs. In 2018, Khatri et al. [134] also studied the cops and robber game on directed graphs. The authors provide bounds on oriented graphs as functions of different parameters such as the number of sources, sinks or even the size of some long cycles. Nevertheless, no structural characterization is found in their work. More results were presented by Das et al. [63] in a variant of the game on oriented graphs. The same setting was then further investigated by Gahlawat et al. [95].

Loh and Oh [148] exhibited a planar directed graph with cop number 4, which shows that Aigner and Fromme's upper bound on planar graphs does not hold in the directed case. They go on to prove that planar directed graphs have cop number at most in $O(\sqrt{n})$. So far, the gap between cop numbers 4 and $O(\sqrt{n})$ has not been bridged. The directed/oriented graphs setting was also studied recently by Hosseini and Mohar [116], Slivova [195], and Frieze et al. [94] and Bradshaw et al. [41].

Another variant of the original game is the *fully active game* in which no player is allowed to stay on their node at any turn. Gromovikov et al. [106] presents some results on the cop numbers of graphs in this case and how they compare with the cop numbers in the original game.

Some classes of graphs can be defined in terms of shortest paths and distances, such as bridged graphs and graphs with diameter two. An important result due to Anstee and Farber [13] states that bridged graphs, which include chordal graphs, are copwin. Wagner [205] showed that if G has n vertices and diameter two, then $c(G) \leq \sqrt{2n}$, so diameter-two graphs satisfy Meyniel's conjecture.

Two teams of researchers have independently devised and solved general formulations of cops and robber games. Bonato and MacGillivray [37] formulated a fully deterministic game that includes the usual variants of cops

and robber games. They present general methods to solve them, one of which generalizes the *elimination ordering* and another the classic *relational characterization* of Nowakowski and Winkler [167]. In comparison, Simard et al. [191] devised another general framework that can incorporate random events. This allows them to include the game of Cop and Drunk Robber[5] [132, 193, 139] that does not fit into Bonato and MacGillivray's description. The novelty of this variant is that the robber moves according to a random walk. Simard et al. provide a relational characterization in the form of a w_n recursion to solve those games. Both models describe plays in pursuit-evasion games as sequences $(s_0, a_0, s_1, a_1, \dots)$ of states s_i and actions a_i such that (s_i, a_i) determines (deterministically or with some probability) the state s_{i+1} for every index $i \geq 0$. In both models, a state s_i contains the positions X and Y of the cops and the robbers. Since Bonato and MacGillivray's model is fully deterministic, they deduce a relation $\preceq$ on the positions of the game such that, informally, $X \preceq Y$ if and only if when the robbers are on X and the cops are on Y, the cops eventually win. In a similar way, the recursion of Simard et al. [191] is a function of the form $w_n(C, R) \in [0, 1]$ that means, informally, that when the cops are on C and the robbers on R, the cops win in at most n steps *with probability* $w_n(C, R)$.

In the original game, one can also let the robber move faster. We say the robber has speed $s \geq 1$ if on its turn it can move to any node of the graph that is reachable by a path of length s, without stepping on a node occupied by a cop. If the speed of the robber is not bounded, we write $s = \infty$. The cop number of a graph in this game is written $c_s(G)$. Mehrabian [159] presents many results when $s = \infty$. Simply increasing the speed of the robber by one leads to fascinating results, such as Fomin et al.'s [89] that the $n \times n$ grid has cop number $\Omega(\sqrt{\log n})$ when the robber has speed two. This was later generalized to large, but finite, speeds and grid sizes by Balister et al. [19].

Another new interesting setting is the recent extension of the game of cops and robber to metric spaces ([162, 99, 123]), which recalls early pursuit-evasion games such as the Homicidal Chauffeur and the Lion and the Man, that we mentionned in Section 2.2.

Finally, Kehagias and Konstantinidis [131] observed that many variants of Cops and Robber games fit into the larger framework of *Reachability Games*

[5]Nearly all of the variants of cops and robber games are deterministic. It would be interesting to study the cop and drunk robber game on periodic temporal networks, however we focus on the deterministic version since not much is known on this structure.

[30]. Those games can also be solved recursively, with a method similar to Hahn and MacGillivray's algorithm [107] and the relational characterization of Bonato and MacGillivray. Reachability Games are simple two-player games with perfect information in which each player seeks to move the game into a finite set of configurations.

2.3 Cops and Robber Games on Temporal Networks

Erlebach and Spooner [80] first introduced the game of cops and robber on periodic temporal networks. They devised algorithms to determine if any periodic graph is copwin or not through a transformation into a reachability game [30]. This gives an algorithm with temporal complexity of $O(pn^3)$ (in our notation). They also mentioned a method to extend this to $k > 1$ cops with a number of operations at most $O\Big(kn^{k+2}p\Big)$, but did not write it explicitly. Using this same connection to reachability games, Balev et al. [18] also present algorithms to solve the game of cops and robber on periodic temporal networks. However, they also present a novel variant, called *online*, in which the players do not initially know the *schedule* of the temporal network: when each edge is set to appear. One notable result in their work is the following. Let G be a graph with $m = n - 1 + \lambda$ edges and $c_t(G)$ be the highest cop number of any temporal graph with footprint G, when the game is played in the online case, then $c_t(G) \leq \lambda + 1$. This is tight, for example, when G is a tree or a cycle. We will present an upper bound on a similar quantity, $c_{\circledcirc}(G)$, in Theorem 6.4 with $\mathrm{tw}(G) + 1$, where $\mathrm{tw}(G)$ is the *treewidth* of G. Although $\mathrm{tw}(G) + 1 = \lambda + 2$ on some graphs whose number of edges is minimal for their treewidths, such as trees and cycles, λ is unbounded even on classes of graphs with bounded treewidths such as outerplanar graphs. This makes Theorem 6.4 more adequate for our context.

The online game can be thought of as a case in which the amount of information available to the cops is restricted. Other authors had previously studied restrictions on information available to the cops in the game played on undirected graphs, such as Clarke [61]. Nevertheless, it is much more common to assume perfect information. Finally, the last published work on the subject comes from Morawietz et al. [163]. These authors study a question left open by Erlebach and Spooner on the complexity of solving the cops and

robber game on periodic temporal networks. Specifically, they employ the framework of Parameterized Complexity (see Downey and Fellows [73]) in order to devise hardness results for this game. Of particular interest are their NP-HARD and W[1]-HARD results on the question of whether a single cop has a winning strategy on a periodic graph. Unlike the rest of the literature (see for example [52, 55]), they treat the number of time instants (the period, see Chapter 3) as a function of the number of nodes in order to derive their complexity results. We follow the literature and consider the period and the number of nodes as two independent parameters, treating the temporal (via the period) and structural (via the number of nodes) dimensions as orthogonal. In that case, Erlebach and Spooner and Balev et al. already presented polynomial-time algorithms to answer this question as a function of both parameters.

More specific references are given throughout this document as they become relevant to the subject at hand.

Chapter 3

General definitions and basic results

3.1 Graphs and Time

In what follows, we write $\mathbb{Z}^+$ for the set of positive integers including zero and $\mathbb{Z}_k$ for the set of integers modulo k. Given any integer i, we let $[i]_k$ be the integer in $\mathbb{Z}_k$ such that $i \equiv [i]_k \pmod{k}$. We also write $\mathbb{R}$ for the set of real numbers. We use the notation $[a, b]$ for the closed interval $\{x \in \mathbb{R} \mid a \leq x \leq b\}$.

3.1.1 Static Graphs

We denote by $G = (V, E)$, or sometimes by $G = (V(G), E(G))$, the graph (directed or undirected) with set of vertices V and set of edges E. We write (u, v) for a *directed* edge from u to v and uv for an *undirected* one. A self-loop is an edge of the form (u, u) or uu. We say that G is reflexive if every node has a self-loop. Unless stated otherwise, we consider all graphs to be undirected and reflexive. We say a graph G' is a *subgraph* of G, written $G' \subseteq G$, if $V(G') \subseteq V(G)$ and $E(G') \subseteq E(G)$. For any subset $V' \subseteq V$, we write $G[V'] = (V', E')$ for the subgraph of G such that E' contains all edges of E that have both endpoints in V' and say $G[V']$ is the subgraph of G *induced* by V'. We also write $G \setminus V'$ for the subgraph $G[V \setminus V']$ of G and $G \setminus H$ for $G[V \setminus V(H)]$ when $H \subseteq G$. When G is undirected, we write $N_G(u) := \{v \in V \mid vu \in E\} \setminus \{u\}$ and $N_G[u] := N_G(u) \cup \{u\}$ for any node u. The *degree* of u in G, $\deg_G(u)$, is given by $|N_G(u)|$. We write $d_G(x, y)$ for the

distance from x to y in G given by the length of a shortest path of G from x to y. We sometimes leave out the subscript when G is easily inferred from the context. The *diameter* of G is given by $d(G) = \max_{x \in V} \max_{y \in V} d_G(x, y)$.

For reasons apparent later, we shall refer to a graph G so defined as a *static graph*.

The distance in G from a node u to another node v, written $d_G(u, v)$ is the length of a shortest path from u to v.

3.1.2 Temporal Graphs

A *time-varying graph* $\mathcal{G}$ is a graph whose set of edges changes in time[1]. A *temporal graph* is a time-varying graph where the set of time instants is $\mathbb{Z}^+$.

A temporal graph $\mathcal{G}$ is represented as an infinite sequence $\mathcal{G} = (G_0, G_1, \dots)$ of static graphs $G_i = (V, E_i)$ on the same set of vertices V. Unless stated otherwise, we generally assume each G_i is *reflexive* and *undirected*. We shall denote by $n = |V|$ the number of vertices of $\mathcal{G}$. The graph G_i is called the *snapshot* of $\mathcal{G}$ *at time* $i \in \mathbb{Z}^+$ and the aggregate (undirected) graph $G = (V, \bigcup_i E_i)$ is called the *footprint* of $\mathcal{G}$.

Given two nodes $x, y \in V$, a *journey*, from x to y starting at time t is any finite sequence $\pi(x, y) = ((z_0, z_1), (z_1, z_2), \dots, (z_{k-1}, z_k))$ where $z_0 = x, z_k = y$, and $z_i z_{i+1} \in E(G_{t+i})$ for $0 \leq i < k$.

A temporal graph $\mathcal{G}$ is *temporally connected* if for any $u, v \in V$ and any time $t \in \mathbb{Z}^+$ there is a journey from u to v that starts at time t. Observe that if $\mathcal{G}$ is temporally connected, then its footprint is connected even when all its snapshots are disconnected. A temporal graph $\mathcal{G}$ is said to be *always connected* (or 1-*interval connected*) if all its snapshots are connected.

A temporal graph $\mathcal{G}$ is *periodic* if there exists a positive integer p such that for all $i \in \mathbb{Z}^+$, $G_i = G_{i+p}$. If p is the smallest such integer, then p is called the *period* of $\mathcal{G}$. We shall represent a periodic graph $\mathcal{G}$ with period p as $\mathcal{G} = (G_0, \dots, G_{p-1})^*$. An example of a periodic graph $\mathcal{G}$ with period $p = 4$ is shown in Figure 3.1. Observe that $\mathcal{G}$ is temporally connected, however most of its snapshots are disconnected graphs. Note also that when $\mathcal{G}$ is periodic, then it is temporally connected if and only if its footprint G is connected. In this work we assume all periodic graphs are temporally connected and their footprints are reflexive unless specified otherwise.

Given a node $u \in V$ and a time $t \in \mathbb{Z}_p$, we write $N_t[u, \mathcal{G}] := N_{G_t}[u]$ for

[1]The terminology in this section is mainly from [55].

the neighbourhood of u at time t and $\deg_t(u) := |N_{G_t}(u)|$ for the degree of u at time t. Thus, $|N_t[u, \mathcal{G}]| = \deg_t(u) + 1$.

Let us point out the obvious but useful fact that static graphs are periodic graphs with period $p = 1$.

For simplicity, we sometimes write $d_t(u, v)$ instead of $d_{G_t}(u, v)$ for the distance from u to v in G_t. Note that this is the length of a shortest path from u to v in G_t, so it only depends on G_t.

3.1.3 Arena

All graphs in this work are undirected, except for the following class of directed graphs which we call *arenas*. This is similar to the *temporal graph* in [140] and to the *static expansion* in [88].

Definition 3.1 (Arena)**.** *Let $k \geq 1$ be any integer and W be a non-empty finite set. An* arena *of length k on W is any static directed graph $\mathcal{D} = (\mathbb{Z}_k \times W, E(\mathcal{D}))$ where $E(\mathcal{D}) \subseteq \{((i, w), ([i+1]_k, w')) \mid i \in \mathbb{Z}_k \text{ and } w, w' \in W\}$.*

A periodic graph $\mathcal{G} = (G_0, \ldots, G_{p-1})^*$ with period p and set of nodes V has a unique correspondence with the arena $\mathcal{D} = (\mathbb{Z}_p \times V, E(\mathcal{D}))$ where, for all $i \in \mathbb{Z}_p$, $((i, u), ([i + 1]_p, v)) \in E(\mathcal{D}) \iff uv \in E(G_i)$, called the *arena of* $\mathcal{G}$. Therefore, the arena $\mathcal{D}$ of $\mathcal{G}$ explicitly preserves the snapshot structure of $\mathcal{G}$. An example of a periodic graph $\mathcal{G}$ and its arena $\mathcal{D}$ is shown in Figure 3.1.

The vertices of an arena $\mathcal{D}$ will be called *temporal nodes*. Given a temporal node $(i, u) \in V(\mathcal{D})$ we shall denote by $N_i[u, \mathcal{D}]$ the set of its outneighbours (which includes $([i + 1]_p, u)$), and by $\Gamma_i(u, \mathcal{D}) = \{v \in V \mid ([i + 1]_p, v) \in N_i[u, \mathcal{D}]\}$ the corresponding set of nodes in G_i. Given an arena $\mathcal{D} = (\mathbb{Z}_p \times V, E)$ and a node $u \in V$, we write $\mathcal{D} \setminus \{u\}$ for the arena $\mathcal{D}' = (\mathbb{Z}_p \times V \setminus \{u\}, \{((t, x), (t + 1, y)) \in E \mid x \neq u \neq y\})$.

For every time t, the directed subgraph $S_t = (\{(t', u) \mid t \leq t' \leq t + 1, u \in V\}, \{((t, u), (t + 1, v)) \in E_{\mathcal{D}}\})$ of $\mathcal{D}$ is called a *slice*, and it corresponds to snapshot G_t of $\mathcal{G}$. We only use the symbol Γ when we work with an arena and N when we work with a periodic graph.

Let $\mathcal{G} = (G_0, \ldots, G_{p-1})^*$ and $\mathcal{H} = (H_0, \ldots, H_{p-1})^*$ be two periodic graphs with footprints G and H, respectively, and the same period. We say $\mathcal{H}$ is a *periodic subgraph* of $\mathcal{G}$, written $\mathcal{H} \subseteq \mathcal{G}$, if $H_i \subseteq G_i$ for every time i. Observe that, since $\mathcal{H}$ is a periodic graph, we must have $V(H_i) = V(H_{i+1})$ for every time i. Furthermore, the induced periodic subgraph $\mathcal{G}[V'] \subseteq \mathcal{G}$, induced by a subset of nodes $V' \subseteq V$, is the periodic subgraph $\mathcal{G}[V'] :=$

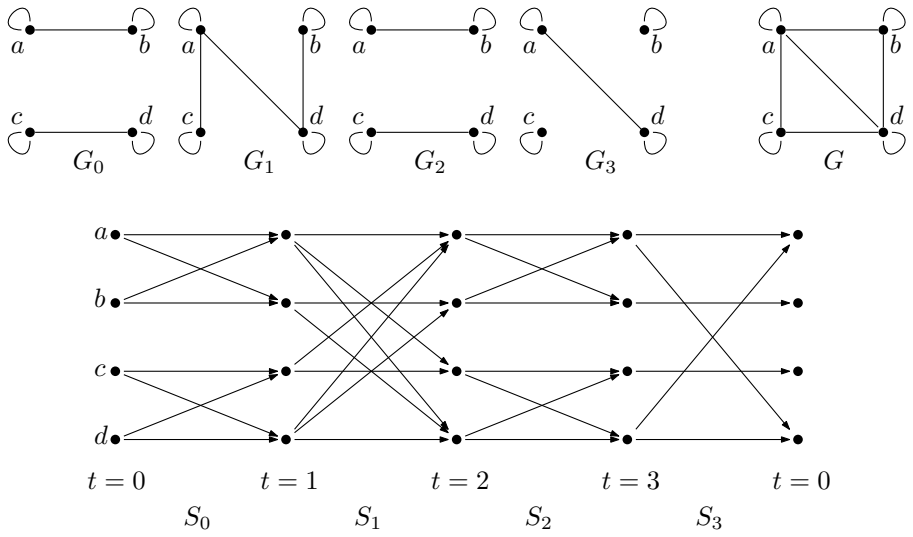


Figure 3.1: A periodic graph $\mathcal{G} = (G_0, G_1, G_2)^*$ with its footprint G and corresponding arena.

$(G_0[V'], \ldots, G_{p-1}[V'])^*$. If $H \subseteq G$, we write $\mathcal{G}[H] := (G_0[V(H)], \ldots, G_{p-1}[V(H)])^*$. We sometimes leave out the subset V' when it is clear from the context and simply say $\mathcal{H}$ is an induced periodic subgraph of $\mathcal{G}$ when there exists a subset $V' \subseteq V$ such that $\mathcal{H} = \mathcal{G}[V']$. Similarly, let $\mathcal{D} = (\mathbb{Z}_p \times V, E_{\mathcal{D}})$ be an arena. A subgraph $\mathcal{D}' \subseteq \mathcal{D}$ with the same length as $\mathcal{D}$ is called a *subarena* of $\mathcal{D}$.

In the rest of this work we consider mostly periodic graphs and their arenas. We always write $\mathcal{G}$ for a periodic graph, G for its footprint and $\mathcal{D}$ for the arena that corresponds to $\mathcal{G}$. Similarly, we reserve the letters n and p for the number of nodes of $\mathcal{G}$ and its period, respectively. The graph G_t is always meant as the snapshot at time t of a periodic graph $\mathcal{G}$.

3.2 Link Streams

A link stream[2] L is a tuple $L = (T, V, A)$ where $T \subseteq \mathbb{R}$ is a set of time instants, V is a finite set of nodes (vertices) and A is a set of temporal edges[3]. For our usage in Chapter 7, we assume that T is a non-empty, continuous and closed interval. However, the algorithms also work if T is a union of closed intervals or if L is equivalent to a periodic graph. As in a temporal graph, every time

[2]Most definitions about link streams come from Latapy et al. [144].

$t \in T$ induces an undirected graph (snapshot) $G_t = (V, A_t)$ such that A_t contains every edge of L that appears at time $t \in T$. Then, a temporal edge is a tuple (t, uv) such that uv is an undirected edge of G_t. We say an element of $T \times V$ is a temporal vertex, or temporal node. We do not assume G_t is reflexive.

Given an interval $I \subseteq T$, we write $(I, uv) \subseteq A$ as a shorthand for the set $I \times \{uv\} \subseteq A$ such that $(t, uv) \in A$ for every $t \in I$. We also call (I, uv) a temporal edge. Moreover, we say a temporal edge $(I, uv) \subseteq A$ is *maximal* if there exists no other temporal edge $(J, uv) \subseteq A$ such that $I \subset J$. We say a maximal edge $([a, b], uv)$ starts at a, ends on b and has *duration* $b - a$. Similarly, let (a, uv) be a temporal edge and $[a, b]$ be the interval such that there exists no other temporal edge $([a, c], uv) \subseteq A$ with $[a, b] \subset [a, c]$. Then, we write $\mathrm{dur}(a, uv) = b - a$ for the duration of the temporal edge (a, uv). Let $\mathcal{I}$ be the set of intervals $[a, b]$ such that $([a, b], uv)$ is a maximal edge of A for some $u, v \in V$. The set of endpoints of intervals of $\mathcal{I}$, written Ω, is called the set of *event times* of T. Elements of $\Omega \times V$ are called *event nodes*. We write $A_\Omega := \{(t, uv) \in A \mid t \in \Omega\}$.

A maximal edge, as well as Ω and $\Omega \times V$ are illustrated on the link stream of Figure 3.2. On this link stream, $([1, 2], cb) \subset A$ is a maximal edge, whereas $([1, 1.5], cb) \subset A$ is not. Thus, $\Omega = \{0, 1, 2, 3\}$.

Remark 3.2. *We represent a link stream similarly to an arena. The most obvious differences are that we only draw one* undirected *edge between two temporal nodes and that we do not explicitly draw all temporal nodes, since there is an uncountable number of them.*

Other definitions, specific to the context of Chapter 7 are defined in this chapter.

3.3 Cops and Robber Game in Graphs

Let us recall the original game of Nowakowski and Winkler, and Quilliot. A Cops and Robber game is a two-player, perfect information and turn-based game in which a team of cops try to capture a robber on an *undirected and reflexive* graph. The cops always play first. A turn is defined as one move of the cops, then one move of the robber. Initially, the cops position themselves

[3] We use the symbol A (as in arcs), to distinguish the set of temporal edges A of a link stream from the set of edges E of the footprint of a periodic graph.

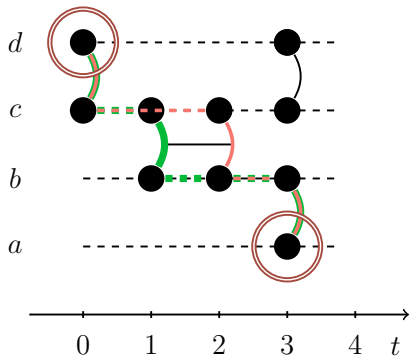


Figure 3.2: A simple link stream with maximal edge $([1,2],cb)$. The lowest horizontal line represent the timeline $T = [0,4]$ and dotted lines parallel to the timeline represent the temporal nodes. Curved lines represent temporal edges, such as the the temporal edge $(1,cb)$ drawn in thick green. Finally, the full horizontal line that starts in the middle of the temporal edge $(1,cb)$ represents the maximal edge $([1,2],cb)$. Two journeys are drawn in thick green and thin red between the same encircled temporal nodes.

on a set of nodes, then the robber positions itself on a node. Afterwards, each player (we use the term *player* to mean either the cops or the robber) stands on a node and has to move along an edge to an adjacent vertex. Because the graph is reflexive, both players are allowed to stay on their positions. When multiple cops are playing, they can move simultaneously and more than one cop can stand on the same vertex. The robber is captured if a cop ever stands on the node where the robber is standing. In this case, the cops have won while the robber wins if it can forever escape. That is, for the robber to win, the game must never end.

A graph is *copwin* if a single cop can capture the robber, otherwise it is *robberwin*. More generally, to every graph G there corresponds a *cop number* $c(G)$ that is the smallest number of cops required to catch the robber on G. A copwin graph has cop number 1.

Let us focus on the case of a single cop. In order to win, the cop must *constrain* the robber on a node. Therefore, when the undirected graph G is copwin it must contain a node u that is a *corner* of another node v, that is:

$$N_G[u] \subseteq N_G[v].$$

Then, if the robber is on u while the cop is on v and the robber is next to play, no matter where it goes it gets captured.

It is known that a graph is copwin if and only if it is *dismantlable*. A graph G is dismantlable if it contains a corner u_1 and for each $1 < i < n$, either there exists a corner u_i in $G \setminus \{u_1, \ldots, u_{i-1}\}$ or the latter contains only one node. When G is dismantlable, the sequence $u_1, \ldots, u_{n-1}$ is called either a *dismantling order* or an *elimination ordering* of G.

Equivalently, if u is a corner of v in G, then there is a graph homomorphism[4] h, called a *retraction*, that maps u to v and is the identity everywhere else. Then, G is dismantlable if recursively retracting corners results in a graph with a single node. More generally, a *retraction* h of a graph G is a homomorphism from G to one of its subgraphs H that is the identity on H. In this case, H is called a *retract* of G.

3.4 Cops and Robber Game in Periodic Graphs

3.4.1 Basics

The extension of the game of Cops and Robber from static to temporal graphs is quite natural. Initially, first the cops, then the robber, choose a starting position on the vertices of G_0. Then, at each time $t \in \mathbb{Z}^+$, first the cops, then the robber, move to vertices adjacent to their current positions in $G_{[t]_p}$. Thus, in round t, the players are in $G_{[t]_p}$ and, after making their moves, they find themselves in $G_{[t+1]_p}$ in the next round. The game ends and the cops win if and only if at least one cop moves to the vertex currently occupied by the robber. The robber wins by forever preventing the cops from winning.

A play on the arena $\mathcal{D}$ of $\mathcal{G}$ follows the play on $\mathcal{G}$ in a direct obvious way: at each time $t \in \mathbb{Z}^+$, first the cop, then the robber, chooses a new node in the out-neighbourhood of its current position and moves there. The cop wins and the game ends if it manages to move to a temporal node $([t+1]_p, u)$ while the robber is on $([t]_p, u)$. The robber wins by forever escaping capture from the cop, in which case the game never ends.

[4]We refer to Hahn and Tardif [108] for a survey on graph homomorphisms.

3.4.2 Configurations and Strategies

Let $k \geq 1$ cops play on $\mathcal{G}$. A configuration is a pair of possible positions for k cops and the robber when the game is played on $\mathcal{G}$, written as $C((t, c_1, \ldots, c_k), (t', r))$ where $t' \in \{t-1, t\}$. We let $t' \in \{t-1, t\}$ because on every turn, the cops start in the same snapshot as the robber and end in the next snapshot. Thus, the positions of the cops and the robber cannot be further than one unit of time apart. A strategy for the cops is a function σ_c that maps each configuration to a new position for the cops and robber strategies are similarly defined. We say a cops strategy σ_c is *feasible* if whenever $\sigma_c((t, c_1, \ldots, c_k), (t, r)) = (t+1, c'_1, \ldots, c'_k)$, then $c'_i \in N_t[c_i, \mathcal{G}]$ for every $1 \leq i \leq k$. The same holds for robber strategies. A cops strategy σ_c is said to be *k-copwin*, or simply *winning*, from a configuration $C((t, c_1, \ldots, c_k), (t, r))$ if, starting from this configuration and provided they follow σ_c, the cops can win the game regardless of the strategy used by the robber. In this case, we also say $C((t, c_1, \ldots, c_k), (t, r))$ is k-copwin. Moreover, σ_c is said to be k-copwin on $\mathcal{G}$ if there exists k nodes $u_1, \ldots, u_k$ such that for any node v, σ_c is winning from $C((0, u_1, \ldots, u_k), (0, v))$.

We say $\mathcal{G}$ and $\mathcal{D}$ are k-copwin if the cops have a k-copwin strategy on $\mathcal{G}$. When $k = 1$, we write *copwin* instead of 1-copwin. The smallest integer k such that $\mathcal{G}$ is k-copwin is the cop number of $\mathcal{G}$, written $c(\mathcal{G})$.

3.4.3 Stubborn walks

We borrow from the field of mobile agents the simple notion of *stubborn walk*. A player is said to perform a stubborn walk over a periodic graph $\mathcal{G}$ from a temporal node (t, u) to another node v simply if it greedily crosses every edge on a path of G from u to v as they appear in $\mathcal{G}$. If the next edge on the path being followed is not available, then the player stubbornly waits until it appears before crossing it (hence the name stubborn). If there are multiple paths from u to v in G, we will specify which one the player chooses. This results in a journey over $\mathcal{G}$.

3.4.4 Temporal Corners

A temporal node (t, u) in an arena $\mathcal{D}$ is a *temporal corner* of a *temporal cover* $(t+1, v)$ if $u \neq v$ and

$$\Gamma_t(u, \mathcal{D}) \subseteq \Gamma_{t+1}(v, \mathcal{D}).$$

In $\mathcal{D}$, every time it moves, a single cop ends its turn in the snapshot ahead of the robber. Thus, this definition of temporal corner encapsulates the usual meaning of corner that *after the cop has moved, no matter where the robber plays, the robber gets captured the next time the cop moves.* We add the restriction $u \neq v$ since, if the robber stands at (t, u), when the cop moves to $(t+1, u)$, the game ends with the cop winning. This situation is different from the situation when the cop moves to a temporal cover of the robber's position: then the robber is still not captured, but cannot escape from the cop.

More generally, we say (t, x) is a *k-temporal corner* of $(t+1, y_1), \ldots, (t+1, y_k)$, if $x \notin \{y_1, \ldots, y_k\}$ and

$$\Gamma_t(x, \mathcal{D}) \subseteq \bigcup_{i=1}^{k} \Gamma_{t+1}(y_i, \mathcal{D}).$$

In this definition, the y_is are allowed the be equal.

Given a graph $G = (V, E)$, we say a vertex u is *universal* in G if $N_G[u] = V$. Similarly, given a periodic graph $\mathcal{G}$, we say (t, u) is a universal temporal node if $N_t[u, \mathcal{G}] = V$.

The following two results relate k-temporal corners to k-copwin periodic graphs.

Lemma 3.3. *Every copwin arena contains a temporal corner.*

Proof. Let $\mathcal{D}$ be a copwin arena. Consider a configuration right before the cops win the game. Observe that if the cop wins in a single move in G_0, then G_0 has a universal vertex u and every temporal node $(p-1, x)$ is a temporal corner of $(0, u)$. Otherwise, consider a configuration $C((t+1, v), (t, u))$ from which no matter where the robber moves to the robber gets captured by the cop. This configuration exists because $\mathcal{D}$ is copwin. Since it is the robber's turn to play, for every $w \in \Gamma_t(u, \mathcal{D})$, there exists a $z \in \Gamma_{t+1}(v, \mathcal{D})$ such that $z = w$. In other words, $\Gamma_t(u, \mathcal{D}) \subseteq \Gamma_{t+1}(v, \mathcal{D})$ and (t, u) is a temporal corner of $(t+1, v)$. □

Proposition 3.4. *Every k-copwin arena contains a k-temporal corner.*

Proof. Let $\mathcal{D}$ be a k-copwin arena. Consider a configuration right before the cops win the game. If the cops win in a single move in G_0, then G_0 has a dominating set $\{v_1, \ldots, v_k\}$ of size k and every temporal node $(p-1, x)$ is a k-temporal corner of $(0, v_1), \ldots, (0, v_k)$. Suppose now the robber is on

(t, x) while the k cops are on $(t+1, y_1), \ldots, (t+1, y_k)$ and the robber is about to get captured, but is alive. Since it is the robber's turn to play, for every $w \in \Gamma_t(x, \mathcal{D})$, there exists a node y_i and a neighbour $z \in \Gamma_{t+1}(y_i, \mathcal{D})$ such that $z = w$. In other words, $\Gamma_t(x, \mathcal{D}) \subseteq \bigcup_{i=1}^{k} \Gamma_{t+1}(y_i, \mathcal{D})$. Therefore, $\mathcal{D}$ contains a k-temporal corner. □

Proposition 3.4 implies that if $\mathcal{G}$ does not contain any k-temporal corner then it cannot be k-copwin. This contrapositive will often come in handy in the next sections. However, the converse is not true in general.

3.4.5 Preliminary results

Lemma 3.5. *Let $\mathcal{G}$ be a periodic graph with set of nodes V and $V' \subseteq V$ be such that $\mathcal{G}[V']$ is temporally connected. Suppose that for every time t, $u \in V'$ and $v \in V$, $\Gamma_t(u, \mathcal{G}[V']) \not\subseteq \Gamma_{t+1}(v, \mathcal{G})$. Then, $\mathcal{G}$ is not copwin.*

Proof. Let $V' \subseteq V$ be as in the statement. By assumption, $\mathcal{G}[V']$ does not contain any temporal corner. Thus, $G_0[V']$ cannot contain a universal node since otherwise every $(p-1, x)$ would be a temporal corner of that universal node. The robber can thus safely start the game on $G_0[V']$ and survive at least one turn. It is possible for the robber to always move on $\mathcal{G}[V']$ since it is temporally connected. Furthermore, the only way for the cop to catch the robber is for the robber to first step on a temporal corner, by Lemma 3.3. However, none of the elements of $\mathcal{G}[V']$ are temporal corners in $\mathcal{G}$. Therefore, the robber survives in $\mathcal{G}$ by always playing in $\mathcal{G}[V']$. □

Corollary 3.6. *Let $k \geq 1$ be any integer. Let $\mathcal{G}$ be a periodic graph with set of nodes V and $V' \subseteq V$ be such that $\mathcal{G}[V']$ is temporally connected. Suppose that for every $(t, u) \in \{0, \ldots, p-1\} \times V'$ and $(t+1, v_1), \ldots, (t+1, v_k) \in \{0, \ldots, p-1\} \times V$, $\Gamma_t(u, \mathcal{G}[V']) \not\subseteq \bigcup_{i=1}^{k} \Gamma_{t+1}(v_i, \mathcal{G})$. Then, $\mathcal{G}$ is not k-copwin.*

Proof. This follows by the same reasoning as in Lemma 3.5, along with Proposition 3.4. □

When a static graph G is copwin, there exists a retract $h : G \to H$ that maps a corner u to its cover v and is the identity everwhere else. Using this retract, the cop can play as if the robber moves along the edge $h(x)h(u) = xv$ in H whenever it moves along the edge xu in G. We show in Lemma 3.7 that this does not hold in general on periodic graphs. We come back to

the importance of retracts in Cops and Robber games in Section 6.1 and Theorem 6.2.

In Lemma 3.7, we first exhibit a copwin arena with a temporal corner (t, y) of some $(t+1, z)$ and a temporal node $(t-1, x)$ such that $y \in \Gamma_{t-1}(x, \mathcal{D})$ but $z \notin \Gamma_{t-1}(x, \mathcal{D})$. Then, if the robber is on $(t-1, x)$ while the cop is on an inneighbour of $(t+1, z)$, even though the robber can move to the temporal corner (t, y) the cop cannot act *as if* the robber is really on (t, z), contrary to the situation on static graphs. Second, we exhibit a copwin arena with two temporal corners (t_1, y_1) and (t_2, y_2) of (t_1+1, z_1) and (t_2+1, z_2), respectively, and a node u such that from $(0, u)$ the cop needs to move to y_2 to reach (t_1+1, z_1) and y_1 to reach (t_2+1, z_2). Then, if y_1 was removed from the arena, the (t_2+1, z_2) is unreachable for the cop *unless* the cop goes directly on y_2 and vice-versa. Thus, contrary to static graphs, *removing* a node from an arena can destroy copwin strategies.

Lemma 3.7. *There exists a copwin arena $\mathcal{D}$ and three temporal nodes $(t-1, x), (t, y), (t+1, z)$ such that (t, y) is a temporal corner of $(t+1, z)$, $y \in \Gamma_{t-1}(x, \mathcal{D})$ but $z \notin \Gamma_{t-1}(x, \mathcal{D})$.*

Furthermore, there exists a copwin arena with two temporal corners (t_1, y_1) and (t_2, y_2) of (t_1+1, z_1) and (t_2+1, z_2) and a node u such that from $(0, u)$ the cop needs to move to y_2 to reach (t_1+1, z_1) without going through y_1 and y_1 to reach (t_2+1, z_2) without going through y_2.

Proof. Consider the arena $\mathcal{D}$ in Figure 3.3. This arena is copwin. Let the cop start on $(0, y_2)$. The robber cannot start on u, z_2, y_2 nor z_1 for fear of being caught (possibly after some waiting). Thus, the robber starts on $(0, y_1)$. The cop moves to $(3, z_1)$ when possible and corners the robber on $(2, y_1)$.

Thus, let us show the second claim. The arena $\mathcal{D}$ contains the two temporal corners $(2, y_1)$ of $(3, z_1)$ and $(6, y_2)$ of $(7, z_2)$. From $(0, u)$, the cop can reach $(3, z_1)$ by going through y_2 and reach $(7, z_2)$ through y_1. In $\mathcal{D} \setminus \{y_1\}$, the only journey from $(0, u)$ to $(7, z_2)$ goes through y_2, while in $\mathcal{D} \setminus \{y_2\}$, the only journey from $(0, u)$ to $(3, z_1)$ goes through y_1.

For the first claim, we can build a second arena $\mathcal{D}'$ from $\mathcal{D}$ by adding the edges $((1, y_2), (2, y_1))$ and $((1, y_1), (2, y_2))$ and removing $((2, y_2), (3, z_1))$ and $((2, z_1), (3, y_2))$. This is still copwin, with the cop again starting the game on $(0, y_2)$. However, in $\mathcal{D}'$ we have that $(2, y_1)$ is a temporal corner of $(3, z_1)$, $y_1 \in \Gamma_1(y_2, \mathcal{D})$ and $z_1 \notin \Gamma_1(y_2, \mathcal{D})$. Thus, $\mathcal{D}'$ statisfies the first statement. □

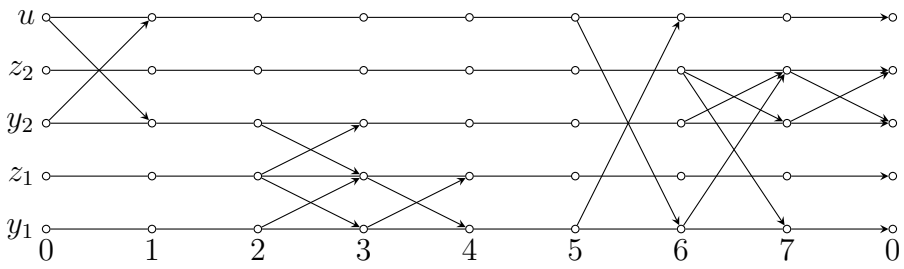


Figure 3.3: If node y_1 is removed, the cop cannot use it to reach the temporal cover $(7, z_2)$ of $(6, y_2)$.

Lemma 3.8. *If any snapshot of a periodic graph contains a universal temporal node, then it is copwin.*

Proof. It suffices for the cop to reach this temporal node in order to catch the robber on her next turn. □

Let $\gamma(G)$ be the *domination number* of a graph G: the size of a smallest dominating set of the graph. The following result gives a simple upper bound on the cop number of a periodic graph.

Lemma 3.9. *Let $\mathcal{G} = (G_0, \ldots, G_{p-1})^*$ be a periodic graph. Then,*

$$c(\mathcal{G}) \leq \min_{0 \leq i \leq p-1} \gamma(G_i)$$

and this bound is tight on some periodic graphs.

Proof. Let G_t be a snapshot of $\mathcal{G}$ that contains a dominating set U of size $k = \min_{0 \leq i \leq p-1} \gamma(G_i)$. Let k cops position themselves on distinct nodes of U in G_0. By definition, for every node $v \in V$, there is an edge $vu \in E(G_t)$ for some node $u \in U$. Thus, no matter where the robber is in G_t it gets captured. Therefore, $c(\mathcal{G}) \leq k$.

Let $\mathcal{G} = (G_0)^*$ have a single node. Clearly, $c(\mathcal{G}) = 1 = \gamma(G_0)$, so this bound is tight. □

The domination number of G does not in general upper bound the cop number of $\mathcal{G}$. To help with the proof of Proposition 3.10, first recall the classic theorem of Berarducci and Intrigila [27]: if G is connected and H is a retract of G, then $c(H) \leq c(G)$.

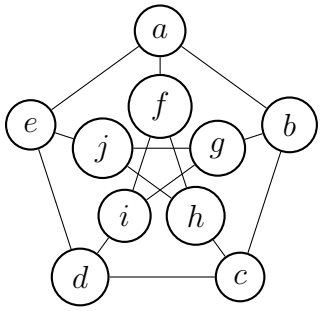


Figure 3.4: A labeled Petersen graph

Proposition 3.10. *There exists a periodic graph $\mathcal{G}$ with footprint G such that*

$$c(\mathcal{G}) > \gamma(G).$$

Proof. Let P_e be the Petersen graph and x, y be two nodes that are not in $V(P_e)$. Suppose P_e is represented as in Figure 3.4, with the cycles $C_o = (a, b, c, d, e)$ and $C_i = (f, h, j, g, i)$. Connect x to every node of C_o and y to every node of C_i. Let G be the resulting graph. Observe that $\gamma(G) = 2$, with $\{x, y\}$ as the minimal dominating set.

Let $\mathcal{G}$ be a periodic graph such that every snapshot contains a full copy of P_e. Furthermore, let E_0 be the sequence of edges $(xa, yf, yi, xd, xe, yj, yh, xc, xb, yg, yi, yf)$. For every integer $i \geq 1$, place the i^{th} edge of E_0 in snapshot G_t such that $t = 11i$.

Let two cops play on $\mathcal{G}$. Observe that for every time t, there is a retraction $h_t : G_t \to P_e$, so $3 = c(P_e) \leq c(G_t)$ by [27]. Suppose that $h_t(x) = h_t(y) = v$ where $v \in V(P_e)$ is the last node to be connected to $\{x, y\}$ no later than time t, since either x or y has degree zero in G_t. The sequence of nodes of P_e connected to $\{x, y\}$ in $\mathcal{G}$ forms a walk in P_e. Thus, for every time t, if $uv \in E(G_t)$ and $vw \in E(G_{t+1})$, then $h(u)h(v) \in E(G_t)$ and $h(v)h(w) \in E(G_{t+1})$. Hence, let the robber play against the cops' images under h_t in every snapshot G_t. This is winning for the robber.

Therefore, $c(\mathcal{G}) > \gamma(G)$. □

Nevertheless, if we replace the criteria of covering the nodes of G by that of covering the edges of G, then we can use a cover number of G as an upper bound on $c(\mathcal{G})$. Thus, let $\tau(G)$ be the *vertex cover number* of a graph G. Lemma 3.12 shows that, when it comes to the footprint of $\mathcal{G}$, $c(\mathcal{G}) \leq \tau(G)$. Before presenting this result, we need the following definition.

Definition 3.11. *Given any graph G, let $c_{\circlearrowleft}(G)$ be the maximal cop number $c(\mathcal{G})$ of any periodic graph $\mathcal{G}$ with footprint G.*

The value of $c_{\circlearrowleft}(G)$ is always well defined since we trivially have $c_{\circlearrowleft}(G) \leq n$. We will sometimes refer to $c_{\circlearrowleft}(G)$ as the *periodic cop number of* G. Observe that we also have $c_{\circlearrowleft}(G) \geq c(G)$ for every graph G.

Similarly, given a periodic graph $\mathcal{G} = (G_0, \ldots, G_{p-1})^*$, we define $c(G_{\min}) := \min_{G_i} c(G_i)$ as well as $c(G_{\max}) := \max_{G_i} c(G_i)$.

Lemma 3.12. *For any connected graph G,*

$$c_{\circlearrowleft}(G) \leq \tau(G).$$

Proof. Let $k = \tau(G)$ cops start the game on the k vertices $u_1, \ldots, u_k$ of a minimal vertex cover of G. The robber starts on a node r. Since G is connected and $\{u_1, \ldots, u_k\}$ is a vertex cover, the first edge rx that appears in $\mathcal{G}$ has an endpoint $x \in \{u_1, \ldots, u_k\}$. Then, a cop makes the catch. □

Lemma 3.13. *Let G be any graph.*

1. *If G is a tree, then $c_{\circlearrowleft}(G) = 1$ and the capture time is at most $p\left\lceil \frac{n}{2} \right\rceil$;*
2. *If G is a cycle, then $c_{\circlearrowleft}(G) \leq 2$ and the capture time is at most $p\left\lceil \frac{n}{4} \right\rceil$.*

Proof. Let us consider each case separately.

1. It suffices for the cop to start on any node of the tree and perform a stubborn walk to catch the robber. The size of the region controlled by the cop never decreases since there are no cycles. Moreover, the cop can start on a node of G at distance at most $\left\lceil \frac{n}{2} \right\rceil$ from the robber. Since $\mathcal{G}$ is periodic, it can move across at least one edge per period, hence the capture time.

2. Two cops can perform a stubborn walk to catch the robber. The capture time follows from the same reasoning as in the previous case.

□

Therefore, we have $c(G) = c_{\circlearrowleft}(G)$ when G is a tree or a cycle. Observe that G might be a cycle while $\mathcal{G}$ contains a single journey per period in which case the periodic graph is copwin.

Chapter 4

Characterizations of copwin periodic graphs

This chapter was presented at the *30th International Colloquium on Structural Information and Communication Complexity* (SIROCCO) in 2023, [65].

For the sake of generality, in this chapter we assume that every snapshot G_t of a *periodic graph* $\mathcal{G}$ is *directed*. We do not assume G_t to be reflexive, but will make an assumption on the temporal connectivity of $\mathcal{G}$.

In any variant of the game of cops and robber, we say the players are *restless* if they must move at every turn. Other researchers have labelled the resulting game the *fully active* game (e.g. [106]).

We study here the game of cops and robber in which the players are restless and the snapshots are directed. Let us point out that the standard version, both in the original or restless variant, as well as the non-restless directed version can actually be redefined as a restless game played on (appropriately chosen) directed graphs: a pair of directed edges between a pair of nodes corresponds to an unidirected link between them, and the presence of a self-loop at a node allows the players currently there not to move to a different node in the current round. In other words, the *restless directed* version of the game includes all the different versions mentioned earlier. In this *unified* version, for the *C&R* game to be defined, and thus *playable*, the only requirement is that every node in the graph must have an outgoing edge. In our investigation, we will use this simple unified version.

For the unified game, we provide a complete characterization of copwin periodic graphs, establishing several basic properties on the nature of a copwin game in such graphs. We do so by using a compact representation of

periodic temporal graphs as static directed graphs, we call arenas, introducing the novel notion of augmented arenas, and using these structures to extend to the temporal domain classical concepts such as *covers* and *corners*.

These characterization results are *general*, in the sense that they do not rely on any assumption on properties such as connectivity, symmetry, reflexivity held (or not held) by the individual snapshot graphs in the sequence.

Based on these results, we design an algorithm for determining if a periodic temporal graph is copwin, prove its correctness and analyze its time complexity.

The total cost of the algorithm is $O(pn^2 + nm)$, where $m = \sum_{i \in \mathbb{Z}_p} |E_i|$ is the number of edges in the first p snapshots. Thus, it improves on the existing $O(pn^3)$ bound established by [80]; in particular, in periodic graphs with sparse snapshots the proposed algorithm terminates in $O(pn^2)$ time. Let us stress that, in the static case, the complexity becomes $O(nm)$, improving the best existing $O(n^3)$ bound Petr et al. [170]; in particular our bound becomes $O(n^2)$ for sparse graphs.

All our results are established for the unified version of the game. Therefore, all the caracterization properties and algorithmic results hold for the standard and for the directed games studied in the literature, both when the players are restless and when they are not. They hold also for all those settings, not considered in the literature, where there is a mix of nodes: those where the players must leave and those where the players can wait; furthermore such a mix might be time-varying (i.e., different in every round).

4.1 Augmented Arenas and Characterization

In this chapter, we focus on the case $k = 1$ and on configurations where it is the cop's turn to play. In that chapter, we write $C(t, c, r)$ instead of $C((t, c), (t, r))$.

The crucial element in the characterization of copwin periodic graphs is the notion of *augmented arena*.

Definition 4.1 (Augmented Arena)**.** *Let $\mathcal{D}$ be the arena of $\mathcal{G}$. An* augmented arena *$\mathcal{A}$ of $\mathcal{D}$ is an arena such that $\mathcal{D} \subseteq \mathcal{A}$ and, for each edge $((t, x), (t + 1, y)) \in E(\mathcal{A})$, the configuration $C(t, x, y)$ is winning for the cop in $\mathcal{D}$.*

We shall refer to the edges of the augmented arena $\mathcal{A}$ of $\mathcal{D}$ as *shadow edges*. Observe that, by definition, all edges of $\mathcal{D}$ are shadow edges of $\mathcal{A}$.

Let $\mathbb{A}(\mathcal{D})$ denote the set of augmented arenas of $\mathcal{D}$. Observe that, by definition, $\mathcal{D} \in \mathbb{A}(\mathcal{D})$. Further observe the following:

Property 4.2 (Closure Property)**.** *The partial order* $(\mathbb{A}(\mathcal{D}), \subset)$ *induced by edge-set inclusion on* $\mathbb{A}(\mathcal{D})$ *is a complete lattice. Hence* $(\mathbb{A}(\mathcal{D}), \subset)$ *has a maximum which we denote by* $\mathcal{A}^*$*.*

Proof. It follows from the fact that, by definition of augmented arena, the set $\mathbb{A}(\mathcal{D})$ is closed under union of the edge-sets. □

Recall from Subsection 3.1.3 that S_t is the *slice* of an arena $\mathcal{D}$ that corresponds to the snapshot at time t. A temporal node $(t, u) \in V(S_t)$ is said to be a *star* if $\Gamma_t(u, \mathcal{D}) = V$. It is said to be *anchored* if there exists a journey from some node $(0, v) \in V(S_0)$ to (t, u).

We have now the elements for the characterization of copwin periodic graphs.

Theorem 4.3 (Characterization)**.** *An arena* $\mathcal{D}$ *is copwin if and only if* $\mathcal{A}^*$ *contains an anchored star.*

Proof. (**if**) Let $\mathcal{A}^*$ contain an anchored star (t, u), $t \in \mathbb{Z}_p$. By definition of star, $\Gamma_t(u, \mathcal{A}^*) = V$; thus, by definition of augmented arena, for every $v \in V$ the configuration $C(t, u, v)$ is copwin, i.e. there is a copwin strategy σ_c from $C(t, u, v)$.

Since (t, u) is anchored, there exists a journey $\pi((0, x), (t, u))$, starting at time 0 and ending at time t, to (t, u) from some temporal node $(0, x)$. Consider now the cop strategy σ'_c of: (1) initially positioning itself on the temporal node $(0, x)$, (2) then moving according to the journey $\pi((0, x), (t, u))$ and, once on (t, u), (3) following the copwin strategy σ_c from $C(t, u, w)$, where w is the position of the robber at the beginning of round t. This strategy σ'_c is winning for all $C(0, x, v)$, $v \in V$; hence $\mathcal{D}$ is copwin.

(**only if**) Let $\mathcal{D}$ be copwin. We then show that there must exist an augmented arena $\mathcal{A}$ of $\mathcal{D}$ that contains an anchored star. Since $\mathcal{D}$ is copwin, by definition, there must exist some starting position $(0, c)$ for the cop such that, for all positions $(0, r)$ initially chosen by the robber, the cop eventually captures the robber. In other words, all the configurations $C(0, c, v)$ with $v \in V$ are copwin; thus the arena $\mathcal{A}$ obtained by adding to $E(\mathcal{D})$ the set of edges $\{((0, c), (1, v)) | v \in V\}$ is an augmented arena of $\mathcal{D}$ and $(0, c)$ is an anchored star. By Property 4.2, $E(\mathcal{A}) \subseteq E(\mathcal{A}^*)$ and the theorem follows. □

The characterization of copwin periodic graphs provided by Theorem 4.3 indicates that, to determine whether or not an arena $\mathcal{D}$ is copwin, it suffices to check whether $\mathcal{A}^*$ contains a star. To be able to transform this fact into an effective solution procedure, some additional concepts need to be introduced and properties established.

4.2 Shadow Corners and Augmentation

Other crucial elements in the analysis of copwin periodic graphs are the concepts of corner and cover, introduced in Subsection 3.4.4 for arenas, now in the context of augmented arenas.

Definition 4.4 (Shadow Corner and Shadow Cover)**.** *Let $\mathcal{A}$ be an augmented arena of $\mathcal{D}$. A temporal node (t, u) is a* shadow corner *of a temporal node $(t+1, v)$, with $v \neq u$, if*

$$\Gamma_t(u, \mathcal{D}) \subseteq \Gamma_{t+1}(v, \mathcal{A}).$$

The temporal node $(t+1, v)$ is called the shadow cover *of (t, u).*

By definition, any temporal corner is a shadow corner, and its temporal covers are shadow covers. An example is shown in Figure 4.1; the red links indicate the neighbours of node (t, u) in $\mathcal{D}$, while in green are indicated the edges to the neighbours of $(t+1, v)$ that exists in $\mathcal{A}$ but not in $\mathcal{D}$.

The role that shadow corners play with regards to the set $\mathbb{A}(\mathcal{D})$ of augmented arena of $\mathcal{D}$ is expressed by the following.

Theorem 4.5 (Augmentation Property)**.** *Let $\mathcal{A} \in \mathbb{A}(\mathcal{D})$, $(t, x), (t, y) \in V(\mathcal{D})$ and $z \in \Gamma_t(x, \mathcal{D})$. If (t, y) is a shadow corner of $(t+1, z)$, then the arena $\mathcal{A}' = \mathcal{A} \cup \{((t, x), (t+1, y))\}$ is an augmented arena of $\mathcal{D}$.*

Proof. Let $\mathcal{A}$ be an augmented arena of $\mathcal{D}$ and let $(t, x), (t, y), (t+1, z) \in V(\mathcal{D})$ where $z \in \Gamma_t(x, \mathcal{D})$ and (t, y) is a shadow corner of $(t+1, z)$. The theorem follows if $((t, x), (t+1, y))$ is already an edge of $\mathcal{A}$. Consider the case where $((t, x), (t+1, y)) \notin E(\mathcal{A})$. Since (t, y) is a shadow corner of $(t+1, z)$, then for every $w \in \Gamma_t(y, \mathcal{D})$ we have that $((t+1, z), (t+2, w)) \in E(\mathcal{A})$, that is $C(t+1, z, w)$ is winning for the cop. Since $z \in \Gamma_t(x, \mathcal{D})$, if the cop moves from (t, x) to $(t+1, z)$ when the robber is on (t, y), then regardless of the robber's move, the resulting configuration would be winning for the cop. In other words, $C(t, x, y)$ is a winning configuration for the cop. It follows that $\mathcal{A}' = \mathcal{A} \cup \{((t, x), (t+1, y))\}$ is an augmented arena of $\mathcal{D}$. □

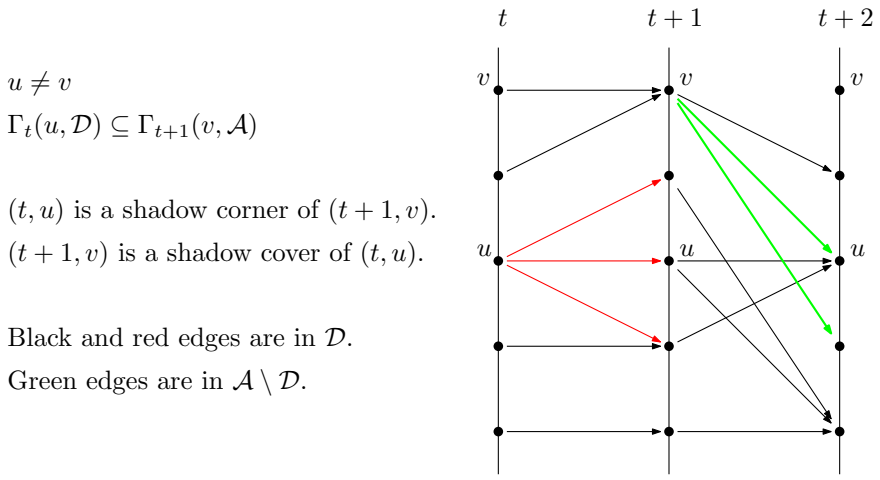


Figure 4.1: Node (t, u) is a shadow corner of $(t+1, v)$.

In other words, given an augmented arena, by identifying a (still unconsidered) shadow corner and its covers, new shadow edges may be determined and added to form a denser augmented arena.

4.3 Determining $\mathcal{A}^*$

The properties expressed by Theorem 4.5, in conjunction with that of Theorem 4.3, provide an algorithmic strategy to construct $\mathcal{A}^*$: start from an augmented arena; determine new shadow edges; add them to the set of shadow edges, creating a denser augmented arena; repeat this process until the current augmented arena $\mathcal{A}$ either contains an anchored star or is $\mathcal{A}^*$.

To be able to employ the above strategy, a condition is needed to determine if the current augmented arena of $\mathcal{D}$ is indeed $\mathcal{A}^*$. This is provided by the following.

Theorem 4.6 (Maximality Property)**.** *Let $\mathcal{A} \in \mathbb{A}(\mathcal{D})$. Then $\mathcal{A} = \mathcal{A}^*$ if and only if, for every edge $((t, x), (t+1, y)) \notin E(\mathcal{A})$, there exists no $z \in \Gamma_t(x, \mathcal{D})$ such that $\Gamma_t(y, \mathcal{D}) \subseteq \Gamma_{t+1}(z, \mathcal{A})$.*

Proof. **(only if)** By contradiction, let $\mathcal{A} = \mathcal{A}^*$ but there exists an edge $((t, x), (t+1, y)) \notin E(\mathcal{A})$ and a temporal node $z \in \Gamma_t(x, \mathcal{D})$ such that $\Gamma_t(y, \mathcal{D}) \subseteq \Gamma_{t+1}(z, \mathcal{A})$. This means that (t, y) is a shadow corner of $(t+1, z)$.

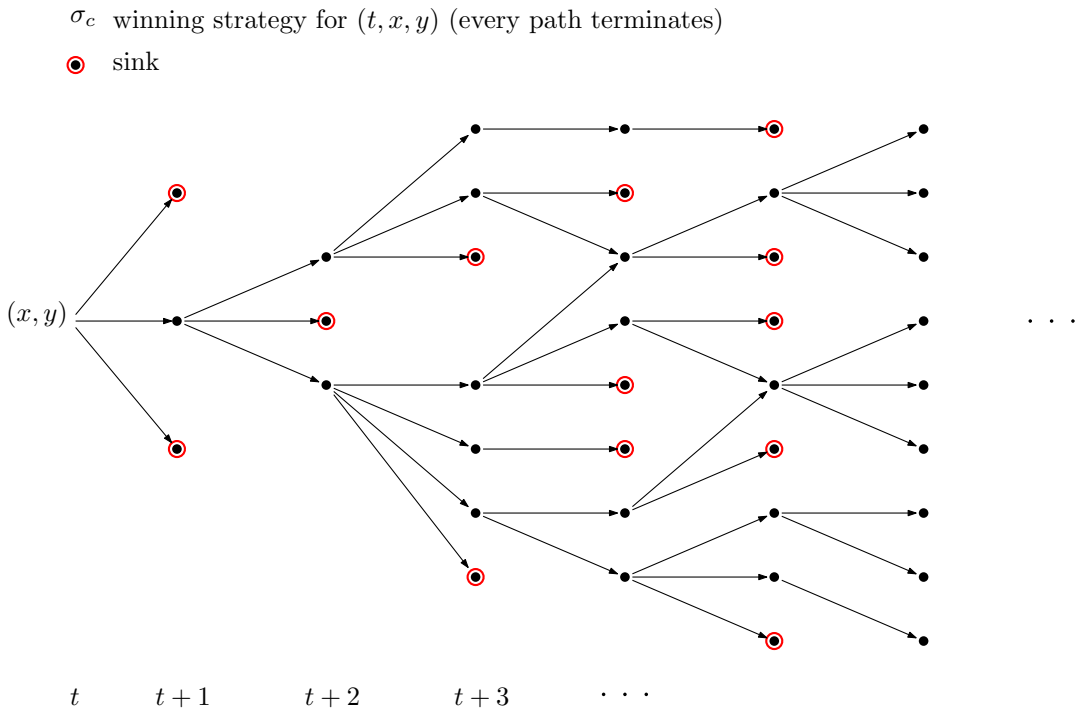


Figure 4.2: The directed acyclic graph $\mathcal{C}$ of configurations induced by σ_c starting from $C(t, x, y)$.

By Theorem 4.5, $\mathcal{A}' = \mathcal{A} \cup \{((t, x), (t + 1, y))\}$ is an augmented arena of $\mathcal{D}$; however, $E(\mathcal{A}')$ contains one more edge than $E(\mathcal{A})$, contradicting the assumption that $\mathcal{A}$ is maximum.

(if) Let $\mathcal{A} \neq \mathcal{A}^*$; that is, there exists $((t, x), (t + 1, y)) \in E(\mathcal{A}^*) \setminus E(\mathcal{A})$. By definition, the configuration $C(t, x, y)$ is copwin; let σ_c be a cop winning strategy for the configuration $C(t, x, y)$; i.e., starting from $C(t, x, y)$, the cop wins the game regardless of the strategy σ_r of the robber.

Let $\mathcal{CG} = (V(\mathcal{CG}), E(\mathcal{CG}))$ be the infinite directed graph that describes all the possible configurations $C(t, u, v)$ and their temporal connection in $\mathcal{D}$:

$$V(\mathcal{CG}) = \{(t, u, v) \,|\, t \in \mathbb{Z}^+, ([t]_p, u), ([t]_p, v) \in V(\mathcal{D})\}$$
$$E(\mathcal{CG}) = \{((t, u, v), (t + 1, u', v')) \,|\, t \in \mathbb{Z}^+, u \neq v, u' \in \Gamma_{[t]_p}(u, \mathcal{D}), v' \in \Gamma_{[t]_p}(v, \mathcal{D})\}.$$

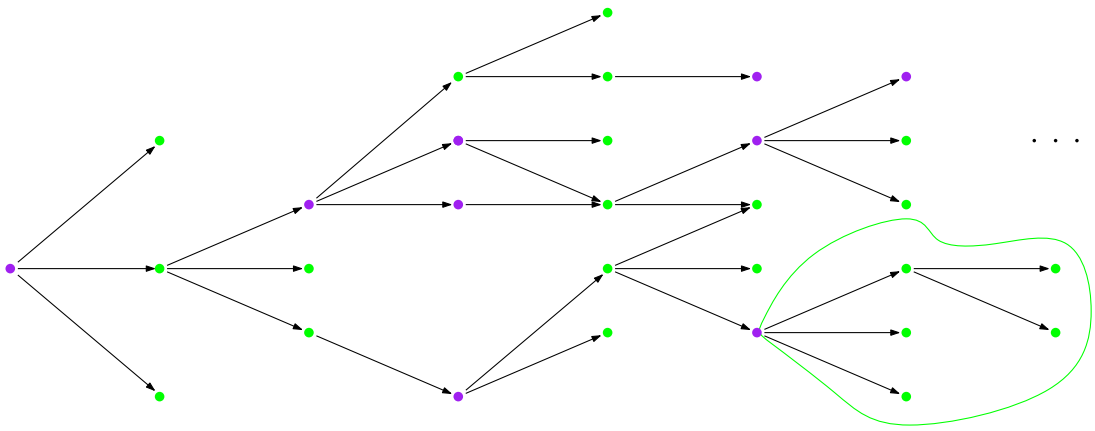
Observe that $\mathcal{CG}$ is acyclic. The *source* nodes (i.e., the nodes with no in-edges) are those with $t = 0$, while the *sink* nodes (i.e., the nodes with no

out-edges) are those with $u = v$.

Moreover, let $\mathcal{C} = (V(\mathcal{C}), E(\mathcal{C})) \subseteq \mathcal{CG}$ be the directed acyclic graph of configurations induced by σ_c starting from $C(t, x, y)$, and defined as follows: (1) $C(t, x, y) \in V(\mathcal{C})$; (2) if $C(t', u, v) \in V(\mathcal{C})$ with $t' \geq t$ and $u \neq v$, then, for all $w \in \Gamma_{t'}(v, \mathcal{D})$, $C(t'+1, \sigma_c(t'+1, u, v), w) \in V(\mathcal{C})$ and $(C(t', u, v), C(t'+1, \sigma_c(t'+1, u, v), w))) \in E(\mathcal{C})$.

Observe that in $\mathcal{C}$ there is only one source (or root) node, $C(t, x, y)$, and every $C(t', w, w) \in V(\mathcal{C})$ is a sink (or terminal) node. Since σ_c is a winning strategy for the root, every node in $\mathcal{C}$ is a copwin configuration, and every path from the root terminates in a sink node. See Figure 4.2.

Partition $V(\mathcal{C})$ into two sets, U and W where $U = \{C(i, u, v) | ((i, u), (i+1, v)) \in E(\mathcal{A})\}$ and $W = V(\mathcal{C}) \setminus U$. Observe that every sink of $V(\mathcal{C})$ belongs to U; on the other hand, since $((t, x), (t+1, y)) \notin E(\mathcal{A})$ by assumption, the root belongs to W (see Figure 4.3).



W: Shadow edges are missing. U: Shadow edges are not missing.

Figure 4.3: The sets U (green) and W (purple).

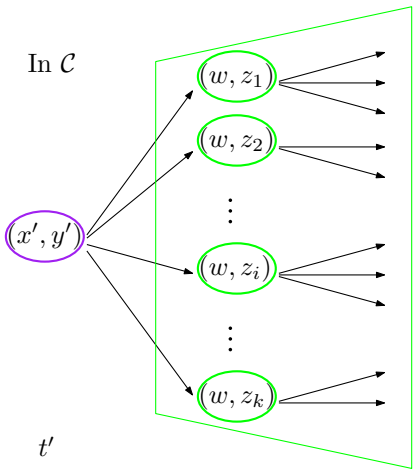
Given a node $\kappa = C(i, u, v) \in V(\mathcal{C})$, let $\mathcal{C}[\kappa]$ denote the subgraph of $\mathcal{C}$ rooted in κ.

Claim. *There exists $\kappa \in V(\mathcal{C})$ such that all nodes of $\mathcal{C}[\kappa]$ except the root belong to U.*

Proof of Claim. Let P_0 be the set of sinks of $\mathcal{C}$. Starting from $k = 0$, consider the set P_{k+1} of all in-neighbours of any node of P_k; if P_{k+1} does not contains an element of W, then increase k and repeat the process. Since

$(t, x, y) \in W$, this process terminates for some $j \geq 1$, and the Claim holds for every $\kappa \in P_j$. $\square$

Let (t', x', y') be a node of $V(\mathcal{C})$ satisfying the above Claim(see Figure 4.4). Thus $((t', x'), (t'+1, y')) \notin E(\mathcal{A})$ but, since (t', x', y') is copwin, $((t', x'), (t'+1, y')) \in \mathcal{A}^*$. By the Claim, all other nodes of $\mathcal{C}[(t', x', y')]$ belong to U, in particular the set of nodes $\{(t'+1, w, z) | w = \sigma_c(t', x', y'), z \in \Gamma_{t'}(y', \mathcal{D})\}$. This means that, for every $z \in \Gamma_{t'}(y', \mathcal{D})$, $(t'+1, w, z) \in E(\mathcal{A})$. In other words, $\Gamma_{t'}(y', \mathcal{D}) \subseteq \Gamma_{t'+1}(w, \mathcal{A})$; that is, (t', y') is a shadow corner of $(t'+1, w)$ (see Figure 4.5).



No shadow edges missing

Figure 4.4: (t', x', y') satisfies the Claim.

Summarizing: by assumption $\mathcal{A} \neq \mathcal{A}^*$; as shown, $((t', x'), (t'+1, y')) \in E(\mathcal{A}^*) \setminus E(\mathcal{A})$, and $w \in \Gamma_{t'}(x', \mathcal{D})$ is a shadow cover of (t', y'); that is, $\Gamma_{t'}(y', \mathcal{D}) \subseteq \Gamma_{t'+1}(w, \mathcal{A})$, concluding the proof of the **if** part of the Theorem. $\square$

4.4 Algorithmic Determination

In this section we show that the results established in the previous sections provide all the tools necessary to design an algorithm to determine whether or

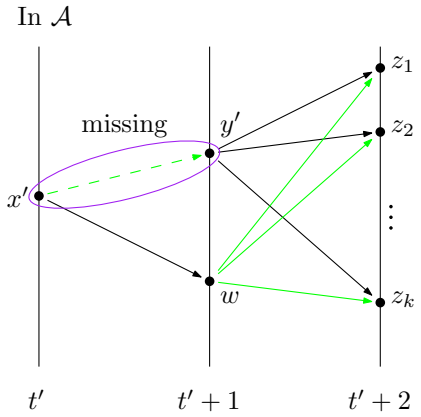


Figure 4.5: (t', y') is a shadow corner of $(t'+1, w)$.

not a periodic graph $\mathcal{G}$ is copwin. Furthermore, if $\mathcal{G}$ is copwin, the algorithm can actually provide a winning cop strategy σ_c.

4.4.1 General strategy of the algorithm

Given a periodic graph $\mathcal{G}$, or equivalently its arena $\mathcal{D}$, to determine whether or not it is copwin, by Theorem 4.3, it is sufficient to determine whether or not its maximal augmented arena $\mathcal{A}^*$ contains an anchored star. Hence, informally, a basic solution approach is to start from $\mathcal{A} = \mathcal{D}$, repeatedly determine a "new" shadow edge (i.e., in $E(\mathcal{A}^*) \setminus E(\mathcal{A})$), using Theorem 4.5, and consider the new augmented arena obtained by adding such an edge. This process is repeated until either the current augmented arena $\mathcal{A}$ contains an anchored star, or no other "missing" shadow edge exists. In the former case, by Theorem 4.3, $\mathcal{D}$ is copwin; in the latter case, by Theorem 4.6, the current augmented arena is $\mathcal{A}^*$ and, if it does not contain an anchored star, $\mathcal{D}$ is robberwin.

A general strategy based on this approach operates in a sequence of iterations, each composed of two operations: the examination of a shadow edge, and the examination of new shadow corners (if any) determined in the first operation. More precisely, in each iteration:

1. A "new" (i.e., not yet examined) shadow edge $e = ((t, x), (t+1, y))$ is examined to determine if its presence transforms some nodes into new shadow corners of (t, x).

GENERAL STRATEGY

1. While there is a still unexamined shadow edge
   $e = ((t, x), (t + 1, y))$ in $\mathcal{A}$ do:
2. If there are still unexamined shadow corners covered by (t, x) then:
3. For each such shadow corner $(t - 1, z)$ do:
4. If there are new shadow edges due to $(t - 1, z)$ then:
5. Add them to $\mathcal{A}$ to be examined.
6. Remove $(t - 1, z)$ from consideration as a shadow corners of (t, x) (i.e., mark it as examined).
7. Remove e from consideration (i.e., mark it as examined).
8. If there is an anchored star in $\mathcal{A}$, then $\mathcal{D}$ is copwin else it is robberwin.

Figure 4.6: Outline of general strategy where the iterative process terminates when $\mathcal{A} = \mathcal{A}^*$.

2. Each of these new shadow corners is examined, determining if its presence generates new shadow edges.

By the end of the iteration, the shadow edge e and the new shadow corners of (t, x) examined in this iteration are removed from consideration. This iterative process continues until there are no new shadow edges to be examined (i.e. $\mathcal{A} = \mathcal{A}^*$) or there is an anchored star in $\mathcal{A}$.

An outline of the strategy, where the iterative process is made to terminate when $\mathcal{A} = \mathcal{A}^*$, is shown in Figure 4.6.

4.4.2 Description of the algorithm

Let us present the proposed algorithm, CopRobberPeriodic, which follows directly the general strategy described above to determine whether or not an arena $\mathcal{D} = ((\mathbb{Z}_p \times V), E(\mathcal{D}))$ is copwin, where $V = \{v_1, \ldots, v_n\}$.

We denote by $\mathcal{A}$ the current augmented arena of $\mathcal{D}$, by A its adjacency matrix, and by A_t the adjacency matrix of slice S_t of $\mathcal{A}$. Auxiliary structures used by the algorithm include the queue SEDGES, of the known shadow edges that have not been examined yet; a $n \times n$ Boolean matrix SE_t for each t, initialized to A_t, used to indicate shadow edges already known; a $n \times n$ Boolean matrix SC_t for each t, initialized to zero and used to indicate the

detected shadow corners; more precisely, $B_t[x, y] = 1$ indicates that (t, x) has been determined to be a shadow corner of $(t + 1, y)$ [1].

The algorithm is composed of two phases:

Initialization In which all the necessary structures are set up and preliminary computations are performed;

Iteration A repetitive process where the two basic operations of the general strategy (described in Subsection 4.4.1) are performed in each iteration: examination of a "new" shadow edge (to determine "new" shadow corners generated by that edge) and examination of the "new" shadow corners (to determine "new" shadow edges generated by that corner).

The structure used to determine new shadow corners is the set $\{\mathrm{DIF}(t, x, y) : t \in \mathbb{Z}_p, x, y \in V\}$ of n^2p Boolean arrays of dimension n. For all $x, v \in V$ and $t \in \mathbb{Z}_p$, the value of the cell $\mathrm{DIF}(t, x, y)[i]$ indicates whether

$$v_i \in \Gamma_t(x, \mathcal{D}) \setminus \Gamma_{t+1}(y, \mathcal{A})$$

(in which case $\mathrm{DIF}(t, x, y)[i] = 1$), or

$$v_i \in \Gamma_t(x, \mathcal{D}) \cap \Gamma_{t+1}(y, \mathcal{A})$$

(in which case $\mathrm{DIF}(t, x, y)[i] = 0$). Note that, if $v_i \notin \Gamma_t(x, \mathcal{D})$, the value of $\mathrm{DIF}(t, x, y)[i]$ is left undefined. Indeed, the algorithm only initializes and uses the $|\Gamma_t(x, \mathcal{D})|$ cells corresponding to the elements of $\Gamma_t(x, \mathcal{D})$. We will call those cells the *core* of $\mathrm{DIF}(t, x, y)$.

The algorithm also maintains a variable $s\,(\mathrm{DIF}(t, x, y))$ indicating the current number of core cells with value "1" in array $\mathrm{DIF}(t, x, y)$, which is initialized to $|\Gamma_t(x, \mathcal{D})|$. Observe that, by definition of $\mathrm{DIF}(t, x, y)$, $s\,(\mathrm{DIF}(t, x, y)) = 0$ if and only if (t, x) is a shadow corner of $(t + 1, y)$.

In each iteration of the *Iteration* phase, a new shadow edge is taken from SEDGES, added to the augmented arena $\mathcal{A}$, and examined. The examination of a shadow edge $((t, x), (t+1, y))$ involves (i) the update of $\mathrm{DIF}(t-1, z, x)[y]$ for any in-neighbour $(t - 1, z)$, in $\mathcal{D}$, of (t, y) and, for any such in-neighbour, (ii) the test to see if the presence of the edge $((t, x), (t + 1, y))$ in the augmented arena has created new shadow corners among such in-neighbours[2]. If

[1] The algorithm was implemented in the Python programming language and it was used determine if some of the arenas in Chapter 5 were copwin or not.

new shadow corners exist, they may in turn have created new shadow edges originating from the in-neighbours, in $\mathcal{D}$, of (t, x). In fact, any in-neighbour $(t-1, w)$ of (t, x) such that $((t-1, w), (t, z))$ is not already in the augmented arena is a new shadow edge: a move of the cop from $(t-1, w)$ to (t, x) is fatal for the robber wherever it goes. In such a case, the algorithm then adds $((t-1, w), (t, z))$ to SEDGES.

The pseudo code of the algorithm is shown in Algorithm 1. Not shown are several very low level (rather trivial) implementation details. These include, for example, the fact that the core cells of $\mathrm{DIF}(t, x, y)$ are connected through a doubly linked list, and that, for efficiency reasons, we also maintain two additional doubly linked lists: one going through the core cells of the array containing "1", the other linking the core cells containing "0".

In Algorithm 1 as well as in the following section, we write $\Gamma^{-}_{t+1}(v, \mathcal{D}) := \{z \in V \mid ((t, z), (t+1, v)) \in E(\mathcal{D})\}$ for any temporal node $(t+1, v)$ in $\mathcal{D}$.

[2]Such would be any $(t-1, z)$ for which the update has resulted in an array $\mathrm{DIF}(t-1, z, x)$ that contains only zero entries.

Algorithm 1: CopRobberPeriodic

Input: Arena $\mathcal{D} = (\mathbb{Z}_p \times V, E(\mathcal{D}))$, with $V = \{v_1, \ldots, v_n\}$

Initialization
$\mathcal{A} := \mathcal{D}$
$SE := A$
SEDGES $= \emptyset$
$SC :=$*Zero* /* a table of p zero matrices, each of size $n \times n$ */
foreach $t \in \mathbb{Z}_p$, $u, v \in V$ **do**
    $s\,(\mathrm{DIF}(t, u, v)) := |\Gamma_t(u, \mathcal{D})|$
    foreach $w \in \Gamma_t\,(u, \mathcal{D})$ **do**
        if $A_{t+1}[v, w] = 1$ **then**
            $\mathrm{DIF}(t, u, v)[w] := 0$
            $s\,(\mathrm{DIF}(t, u, v)) := s\,(\mathrm{DIF}(t, u, v)) - 1$
            if $s\,(\mathrm{DIF}(t, u, v)) = 0$ **and** $SC_t[u, v] = 0$ **then**
                $SC_t[u, v] := 1$
                foreach $z \in \Gamma^-_{t+1}\,(v, \mathcal{D})$ **do**
                    if $SE_t[z, u] = 0$ **then**
                        $SE_t[z, u] := 1$
                        SEDGES $\leftarrow ((t, z), (t + 1, u))$
        else
            $\mathrm{DIF}(t, u, v)[w] := 1$
Iteration
while SEDGES $\neq \emptyset$ **do**
    $((t, x), (t + 1, y)) \leftarrow$ SEDGES
    $A_t(x, y) := 1$
    foreach $z \in \Gamma^-_t\,(y, \mathcal{D})$ **do**
        if $\mathrm{DIF}(t - 1, z, x)[y] = 1$ **then**
            $\mathrm{DIF}(t - 1, z, x)[y] := 0$
            $s\,(\mathrm{DIF}(t - 1, z, x)) := s\,(\mathrm{DIF}(t - 1, z, x)) - 1$
        if $s\,(\mathrm{DIF}(t - 1, z, x)) = 0$ **and** $SC_{t-1}[z, x] = 0$ **then**
            $SC_{t-1}[z, x] := 1$
            foreach $w \in \Gamma^-_t\,(x, \mathcal{D})$ **do**
                if $SE_{t-1}[w, z] = 0$ **then**
                    $SE_{t-1}[w, z] := 1$
                    SEDGES $\leftarrow ((t - 1, w), (t, z))$
if $\mathcal{A}$ contains an anchored star **then** $\mathcal{D}$ is copwin
else $\mathcal{D}$ is robberwin.

4.4.3 Correctness of the algorithm

Let us prove the correctness of Algorithm 1. Let $\mathcal{D} = (\mathbb{Z}_p \times V, E(\mathcal{D}))$ be the arena of a periodic graph with $n = |V|$ and $m = |E(\mathcal{D})|$ and period p.

Lemma 4.7. *Algorithm 1 terminates after at most $|E(\mathcal{A}^*)| - |E(\mathcal{D})|$ iterations.*

Proof. In the *Initialization* phase, all of the $m = |E(\mathcal{D})|$ edges of $\mathcal{D}$ are examined, and their entry in the shadow edge matrix SE is set to 1 (Line 3). Any new shadow edge discovered in this phase is inserted in SEDGES (Line 17).

Observe that, when a shadow edge e is inserted in SEDGES, its entry in the shadow edge matrix SE is set to 1 (this is done in Line 16 for the edges of $\mathcal{D}$, and in Line 32 for the others); this means that, once extracted and examined, e will fail the test of Line 15 (or the test of Line 31) in any subsequent iteration and, therefore, it will never be inserted in SEDGES again. Since only one shadow edge is extracted from SEDGES and examined in each iteration, the number of iterations is at most the total number $|E(\mathcal{A}^*)| - |E(\mathcal{D})|$ of shadow edges not originally in $\mathcal{D}$. □

Given an augmented arena $\mathcal{A}$ and a shadow edge $e = ((t, x), (t + 1, y)) \in E(\mathcal{A}^*) \setminus E(\mathcal{A})$, we shall say that e is an *implicit* shadow edge of $\mathcal{A}$ if there exists $z \in \Gamma_t(x, \mathcal{D})$ such that (t, y) is a shadow corner of $(t + 1, z)$ in $\mathcal{A}$.

Lemma 4.8. *At the end of the* Initialization *phase: (i) for all and only the temporal corners (t, x) of $(t+1, y)$ in $\mathcal{D}$, $SC_t[x, y] = 1$ and $s(\mathrm{DIF}(t, x, y)) = 0$; (ii) all implicit shadow edges of $\mathcal{D}$ are in* SEDGES*; furthermore, the entry in SE of all edges of $\mathcal{D}$ and implicit shadow edges of $\mathcal{D}$, is* 1*.*

Proof. (**i**) Observe that, in the *Initialization* phase, by construction, $\forall t \in \mathbb{Z}_p$, $\forall (t, u), (t+1, v) \in V(\mathcal{D})$, and $\forall w \in \Gamma_t(u, \mathcal{D})$, $\mathrm{DIF}(t, u, v)$ is initialized so that $\mathrm{DIF}(t, u, v)[w] = 1$ if and only if $w \in \Gamma_t(u, \mathcal{D}) \setminus \Gamma_{t+1}(v, \mathcal{D})$. Every time it is determined that $\mathrm{DIF}(t, u, v)[w] = 0$, the counter $s(\mathrm{DIF}(t, u, v))$, initialized to $|\Gamma_t(u, \mathcal{D})|$, is decreased by one; thus, by definition, $s(\mathrm{DIF}(t, u, v)) = 0$ if and only if (t, u) is a temporal corner of $(t+1, v)$; in such a case $SC_t[u, v] = 1$. Recall that, by definition, all temporal corners are also shadow corners.

(**ii**) First observe that all edges of $\mathcal{D}$ are by definition shadow edges, and that their corresponding entry in the shadow edges matrix SE is set to 1 (Line 3); hence, for them, the lemma holds.

Let us now consider the implicit shadow edges of $\mathcal{D}$. Recall that, by Theorem 4.5, given a shadow corner (t, u) of $(t+1, v)$, any edge originating from an in-neighbour (t, z) of $(t+1, v)$ and terminating in $(t+1, u)$ is a shadow edge (Lines 14-17); hence, it is immediate to identify the implicit shadow edges corresponding to a given shadow corner. An implicit shadow edge (i.e., one whose entry in SE is 0), once identified, is added to the queue SEDGES, and the corresponding entry in the shadow edges matrix is set to 1. Since, by part (i) of this Lemma, all the shadow corners present in $\mathcal{D}$ are identified, it follows that all the implicit shadow edges are queued in SEDGES and their entry in SE is set to 1. □

Let us consider the *Initialization* phase as iteration 0 of the *Iteration* phase. Hence, the entire algorithm can be viewed as a sequence of iterations. Denote by $\mathcal{A}_j$ the augmented arena at the beginning of the j-th iteration, with $\mathcal{A}_0 = \mathcal{D}$. We now show that, at the beginning of iteration j, all shadow corners of $\mathcal{A}_{j-1}$ have been examined and all implicit shadow edges of $\mathcal{A}_{j-1}$ are in SEDGES.

Lemma 4.9. *At the beginning of iteration $j > 0$:*

(a) $s\,(\mathrm{DIF}(t, x, y)) = 0$ *if and only if* (t, x) *is a shadow corner of* $(t+1, y)$ *in* $\mathcal{A}_{j-1}$*; furthermore, in such a case,* $SC_t[x, y] = 1$.

(b) SEDGES *contains all the implicit shadow edges of* $\mathcal{A}_{j-1}$*; furthermore, in* SE*, the entry of the edges of* $\mathcal{A}_{j-1}$ *and of the implicit shadow edges of* $\mathcal{A}_{j-1}$ *is* 1.

Proof. By induction on j. Observe that, when $j = 1$, both statements of the lemma follow directly from Lemma 4.8. Let they hold for $j \geq 1$; we now prove that they hold for $j + 1$.

Let $e_j = ((t, x), (t+1, y))$ be the shadow edge extracted from SEDGES and examined in iteration j. This edge is added to $\mathcal{A}_{j-1}$ (Line 23), which is thus transformed into $\mathcal{A}_j$. This addition, which modifies only the out-neighbourhood of (t, x), might create new shadow corners of (t, x) among the in-neighbours of (t, y). The algorithm therefore checks the set $\Gamma_t^-(y, \mathcal{D})$ to verify if this has happened (Lines 24-33). This is done by considering, for each element $(t-1, z)$ of that set, the entry $\mathrm{DIF}(t-1, z, x)[y]$.

If $\mathrm{DIF}(t-1, z, x)[y] = 1$, then the shadow edge e_j was one of those missing edges; hence, in that case (Lines 25-27) the value of $\mathrm{DIF}(t-1, z, x)[y]$ is set to

zero and $s\,(\mathrm{DIF}(t-1,z,x))$ is decreased by one. If now $s\,(\mathrm{DIF}(t-1,z,x))$ becomes 0, then $(t-1,z)$ is a shadow corner of (t,x) in $\mathcal{A}_j$ and $SC_{t-1}[z,x]$ is set to 1 (Line 29). This means that, at the end of this iteration, all shadow corners of $\mathcal{A}_j \setminus \mathcal{A}_{j-1}$ have their entry in SC set to 1 and the corresponding entry in $s\,(\cdot)$ set to 0. Thus, by the induction hypothesis on the shadow corners of $\mathcal{A}_{j-1}$, statement *(a)* of the lemma holds for iteration j.

Any new shadow corner $(t-1,z)$ of (t,x), created by the addition of e_j, might in turn have created new implicit shadow edges in $\mathcal{A}_j$. By Theorem 4.5, any edge originating from an in-neighbour $(t-1,w)$ of (t,x) and terminating in the shadow corner (t,z) is a shadow edge (Lines 30-33). Let P denote the set of these shadow edges; among them, the only implicit ones for $\mathcal{A}_j$ are, by inductive hypothesis, the ones whose entry in SE was 0 in $\mathcal{A}_{j-1}$. Any such implicit shadow edge is thus identified, added to the queue SEDGES, and the corresponding entry in SE is set to 1. Since, by part (a) of this Lemma, all the shadow corners present in $\mathcal{A}_j$ are identified, it follows that all the new implicit shadow edges of $\mathcal{A}_j$ are queued in SEDGES and their entry in SE is set to 1.Thus, by inductive hypothesis on the shadow edges of $\mathcal{A}_{j-1}$, statement (b) of the lemma holds for iteration j. □

Theorem 4.10. *Algorithm 1 correctly determines whether or not an arena $\mathcal{D}$ is copwin.*

Proof. By Lemma 4.7, the algorith terminates after a finite number $q \geq 1$ of iterations, when SEDGES becomes empty and no other shadow edges are added to it during the iteration. By Lemma 4.9, the fact that SEDGES $= \emptyset$ at the end of the iteration means that in $\mathcal{A}_q$ there are no implicit shadow corners identified in previous iterations. Furthermore, during this iteration, regardless of the new shadow corners found and examined, no implicit shadow corners were found. In other words, the set $E(\mathcal{A}^*) \setminus E(\mathcal{A}_q) = \emptyset$, that is $\mathcal{A}_q = \mathcal{A}^*$. Hence, by Theorem 4.6, the test in the last operation of the algorithm (Lines 34-35) determines correctly whether or not $\mathcal{D}$ is copwin. □

4.4.4 Complexity of the algorithm

Let us analyze the cost of Algorithm 1. Given $\mathcal{D} = (\mathbb{Z}_p \times V, E(\mathcal{D}))$, let m_i denote the number of edges of slice S_i of $\mathcal{D}$, $i \in \mathbb{Z}_p$, and $m = |E(\mathcal{D})| = \sum_{i=0}^{p-1} m_i$ the total number of edges of $\mathcal{D}$. As usual, $n = |V|$.

Theorem 4.11. *Algorithm 1 determines in time $O(n^2p + nm)$ whether or not $\mathcal{D}$ is copwin.*

Proof. We first derive the cost of the *Initialization* phase. Observe that the initialization of $\mathcal{A}$, SE, SC (Lines 2-4) can be performed with $O(n^2p)$ operations. Line 7 will be executed n^2p times. The cost of the initialization of DIF and of $s\,$(DIF) (Lines 6-13,18-19), which includes the update of some entries of SC, plus the cost of the initialization of SEDGES (Lines 14-17), which includes the update of some entries of SE, require at most

$$O\Big(n^2p\Big) + \sum_{i\in\mathbb{Z}_p,u,v\in V} O(|\Gamma_i\,(u,\mathcal{D})|) + \sum_{i\in\mathbb{Z}_p,u,v\in V} O\Big(\Big|\Gamma_i^-\,(v,\mathcal{D})\Big|\Big)$$
$$= O\Big(n^2p\Big) + \sum_{i=0}^{p-1} O(n(m_i+m_{i-1}))$$

operations, which sums up to $O(n^2p+nm)$ operations for the *Initialization* phase.

Let us consider now the *Iteration* phase. The while loop will be repeated until in the current augmented arena $\mathcal{A}$ there are no more shadow edges to be examined (i.e. $\mathcal{A}=\mathcal{A}^*$). By Lemma 4.7, the total number of iterations is $|E(\mathcal{A}^*)|-|E(\mathcal{D})| \le n^2p-m$. Further observe that every operation performed during an iteration requires constant time.

In each iteration, two processes are being carried out.

The first process (Lines 24-27) is the determination of all new shadow corners (if any) of (t,x) created by (the addition of) the shadow edge $((t,x),(t+1,y))$ being examined. The total cost of this process in this iteration is at most two operations for each in-neighbour of (t,y), i.e., at most $2c_1|\Gamma_t^-\,(y,\mathcal{D})\,|$, where $c_1\in O(1)$ is the constant cost of performing a single operation in this process.

This process is repeated in all iterations, each time with a different shadow edge being examined. Thus, the cost of $2c_1|\Gamma_t^-\,(y,\mathcal{D})\,|$ will be incurred for all $((t,x),(t+1,y)) \in E(\mathcal{A}^*)$, so at most n times. Summarizing, for each $y\in V, t\in\mathbb{Z}_p$ this process costs $2c_1n|\Gamma_t^-\,(y,\mathcal{D})\,|$. Hence the total cost of this process over all iterations is

$$\sum_{y\in V,t\in\mathbb{Z}_p} 2\ c_1\ n\ |\Gamma_t^-\,(y,\mathcal{D})\,| = 4\ c_1\ n\ \sum_{t=0}^{p-1} m_t = O(nm).$$

The second process, to be performed only if new shadow corners of (t,x) have been found in the first process, is the determination (Lines 28-33) of all the new shadow edges (if any) created by the found new shadow corners,

and their addition to SEDGES. The cost of this process for a new shadow corner in this iteration is $c_2|\Gamma_t^-(x,\mathcal{D})|$, where $c_2 \in O(1)$ is the constant cost of performing a single operation in this process. Observe that, if a new shadow corner of (t,x) is found in this iteration, it will not be considered in any subsequent iteration (Lines 28-29). Hence, the cost $c_2|\Gamma_t^-(x,\mathcal{D})|$ will be incurred at most once for each shadow corner of (t,x); that is, at most n times. Summarizing, for each $x \in V, t \in \mathbb{Z}_p$ this process costs at most $2c_2n|\Gamma_t^-(x,\mathcal{D})|$. Hence the total cost of this process over all iterations is

$$\sum_{x\in V, t\in\mathbb{Z}_p} 2\ c_2\ n\ |\Gamma_t^-(x,\mathcal{D})| = 4\ c_2\ n\ \sum_{t=0}^{p-1} m_t = O(nm).$$

Consider now the last step of the algorithm, of determining if the constructed $\mathcal{A}$ contains an anchored star. To determine all the stars (if any) in $\mathcal{A}^*$ can be done by checking the degree of each temporal node in $\mathcal{A}^*$, i.e., in $O(np)$ time. To determine if at least one of them is anchored can be done by a DFS traversal of $\mathcal{A}^*$ starting from each root node $(0,x)$, for a total of at most $O(n^2+nm)$ operations.

It follows that the total cost of the algorithm is $O(pn^2+nm)$ as claimed. □

The bound established by Theorem 4.11 improves on the existing $O(pn^3)$ bound [80]; in particular, in periodic graphs with sparse snapshots the proposed algorithm terminates in $O(pn^2)$ time. Furthermore, since in static graphs $p=1$, the bound of Theorem 4.11 becomes $O(nm)$, improving the existing $O(n^3)$ bound [170]. Our bound becomes $O(n^2)$ for sparse graphs.

4.5 Extensions and Improvements

4.5.1 Determining a Copwin Strategy

Algorithm 1, as described, determines whether or not an arena $\mathcal{D}$ (and, thus, the corresponding temporal graph $\mathcal{G}$) is copwin. Simple additions to the algorithm would allow it to easily determine a copwin strategy σ_c if $\mathcal{D}$ is copwin.

For any shadow edge $e=((t,x),(t+1,y)$, let $\rho(t,x,y)$ be defined as follows:

1. If $e = ((t, x), (t + 1, y) \in E(\mathcal{D})$, then $\rho(t, x, y) = y$.

2. If $e = ((t, x), (t+1, y) \in E(\mathcal{A}^*) \setminus E(\mathcal{D})$, when e is inserted in SEDGES, either during the Initialization or the Iteration phase, then $\rho(t, x, y) = z$ where $(t + 1, z)$ is the shadow cover of (t, y) determined in the corresponding phase of the algorithm (Line 12 if Initialization, Line 28 if Iteration).

Recall that, if $\mathcal{D}$ is copwin, $\mathcal{A}^*$ must contain an anchored star, say (t, x), Since (t, x) is a star, if the cop is located on (t, x) and the robber is located on (t, y), by moving according to ρ (starting with $\rho(t, x, y)$) the cop will eventually capture the robber. Since (t, x) is anchored, it is reachable from some node in G_0, say $(0, v)$; that is, there is a journey $\pi((0, v), (t, x))$ from $(0, v)$ to (T, x), where $[T]_p = t$.

Consider now the following strategy σ_c for the cop: (1) choose as initial location $(0, v)$; (2) follow $\pi((0, v), (t, x))$; (3) follow ρ. Using this strategy, the cop will eventually capture the robber.

4.5.2 Improvements

The time costs of Algorithm 1 can be reduced by simple modifications and/or by exploiting properties of the temporal graphs.

First of all observe that the algorithm can be made to stop as soon as a temporal node becomes an anchored star (e.g., testing if (t, x) is an anchored star in Line 22) possibly reducing the overall cost with the early termination.

Observe next that some of the costs of the algorithm can be reduced and some of its processes simplified if the *periodic* graph $\mathcal{G}$ has special properties. Consider for example the properties of *reflexivity* and (temporal) *connectivity* with respect to the last step of the algorithm, determining if $\mathcal{A}^*$ contains an anchored star.

We say a temporal node of $\mathcal{G}$ is a source if it is a source in $\mathcal{D}$.

Lemma 4.12.

1. *If $\mathcal{G}$ is reflexive, then every temporal node of $\mathcal{D}$ is anchored.*

2. *If $\mathcal{G}$ is temporally connected without sources, then every temporal node of $\mathcal{D}$ is anchored.*

3. *If $\mathcal{G}$ is reflexive and temporally connected, then $\mathcal{D}$ is strongly connected.*

4. *Let $\mathcal{G}$ be reflexive and temporally connected. If an augmented arena $\mathcal{A}$ of $\mathcal{G}$ contains a star, then every temporal node of $\mathcal{A}^*$ is an anchored star.*

Proof. Let us prove each statement separately.

1. If $\mathcal{G}$ is reflexive, for every node u and time t, (t, u) is reachable from $(0, u)$, so every temporal node of $\mathcal{D}$ is anchored.

2. Let (t, u) be any temporal node of $\mathcal{D}$. If $t = 0$, then (t, u) is already anchored, so suppose $t > 0$. Then, since (t, u) is not a source, there exists an edge $((t-1, u_1), (t, u))$ in $\mathcal{D}$. If $t - 1 = 0$, then (t, u) is anchored, so suppose $t - 1 > 0$. Since $(t - 1, u_1)$ is not a source, there exists another edge $((t-2, u_2), (t-1, u_1))$ in $\mathcal{D}$. Recalling that $\mathcal{G}$ is periodic, we can apply this argument successively to construct a journey from (t, u) to some temporal node $(t - k, u_k)$ with $t - k = 0$. This shows that (t, u) is anchored.

3. Suppose $\mathcal{G}$ is reflexive and temporally connected. Since $\mathcal{G}$ is reflexive, for every time t and node u, (t, u) is reachable from $(0, u)$. Because $\mathcal{G}$ is temporally connected, for every two nodes u, v and time t, u is reachable from v by a journey that starts at time t. Thus, let (t, u) be any temporal node. For every temporal node $(t', v) \neq (t, u)$, let t_v be the first time such that (t_v, u) is reachable from (t', v). Since $\mathcal{D}$ is periodic and reflexive, (t, u) is reachable from (t_v, u). Thus, (t, u) is reachable from (t', v) by a journey that starts at (t', v), goes to (t_v, u) before ending on (t, u). Thus, $\mathcal{D}$ is strongly connected.

4. Suppose that $\mathcal{G}$ is reflexive and temporally connected and that some augmented arena $\mathcal{A}$ of $\mathcal{D}$ has a star (t, u). By Item 3, $\mathcal{D}$ is strongly connected, so for every temporal node $(t', v) \neq (t, u)$, there is a journey from (t', v) to (t, u) in $\mathcal{D}$. For any such (t', v), the strategy that moves the cop from $(0, v)$ to (t', v), then to (t, u) is copwin since (t, u) is a star in $\mathcal{A}$. Therefore, (t', v) is also a star in some augmented arena $\mathcal{A}' \supseteq \mathcal{A}$ and every temporal node of $\mathcal{A}^*$ is an anchored star.

□

The standard game in a static graph assumes the graph to be reflexive and connected. Hence, if its extension to a periodic graph likewise assumes

the graph to be reflexive and temporally connected, by Lemma 4.12, the last step of the algorithm would consist of just testing if the degree of an arbitrary temporal node in $\mathcal{A}^*$ is n. That is, instead of $O(nm)$ operations, a single one suffices.

4.5.3 Game Variations

All our results are established for the unified version of the game. Therefore. all the caracterization properties and algorithmic results hold for the standard and for the directed games studied in the literature, both when the players are restless and when they are not. They hold also for all those settings (and, thus, variants of the game) not considered in the literature, where there is a mix of nodes: those where the players must leave and those where the players can wait. Furthermore such a mix might be time-varying, so different in every round.

Chapter 5

Examples of cop numbers of periodic graphs

Reasonable bounds on the cop number do not hold

Some results in this chapter were presented at the *54th Southeastern International Conference on Combinatorics, Graph Theory & Computing*. A preprint of this text can be found here: [64].

As we can see from the literature so far on cops and robber games on periodic graphs, much of the work on this problem is focused on algorithmic results. With the aim of understanding the relationship between the static and temporal settings in terms of the cop number, we start by focusing on the differences between the cop number of a periodic graph and the cop numbers of its constituent static graphs. Our results show that the temporal dimension introduces huge differences with the static setting, and we discover some properties of graphs that help control those variations.

We investigate the maximum cop number $c_{\circledcirc}(G)$ (recall Definition 3.11) of any periodic graph with footprint G. From this quantity we seek to understand how the footprint constrains the cop number of a periodic graph. One takeaway from our investigation of $c_{\circledcirc}(G)$ is that copwin strategies on periodic graphs, when only G is known, need to be resilient to change. This setting is akin to planning under uncertainty, when failures can occur on the graph G, and this is easier when G has good separation properties.

First, we show that the result of Clarke [60] that outerplanar graphs have cop number at most two does not extend to periodic graphs with outerplanar footprints, by exhibiting a counterexample in Proposition 5.1. Then, we contrast Baird et al's result [17] on the minimum order of a 3-copwin graph by exhibiting a smaller (in the order of its footprint) periodic graph with the same cop number in Proposition 5.2. We show that no value of $c(G), c(G_0), \ldots, c(G_{p-1})$ can, in general, be used as either lower or upper bound on $c(\mathcal{G})$ by presenting two counterexamples (Theorem 5.6 and Theorem 5.7). We complete this presentation by filling Table 5.1 that summarizes many examples we present and their different cop numbers. This table serves to highlight the counterintuitive nature of periodic graphs and the difficulty of deriving $c(\mathcal{G})$ from $c(G), c(G_0), \ldots, c(G_{p-1})$. Those examples are presented in order to help researchers build intuition and avoid pitfalls when moving from the context of graphs to the context of periodic graphs. One such pitfall is that results on $c(G)$ carry over to $c(\mathcal{G})$ and we show this is not true even for a simple extension of Berarducci and Intrigila's result [27] in Proposition 6.1.

We delay presenting more general results, such as lower and upper bounds to the next chapter. Our goal here is, informally, to "break things": to show pitfalls that one must avoid when working with periodic graphs. The next chapter is more constructive. From the mistakes we learn to avoid here, we create general results there.

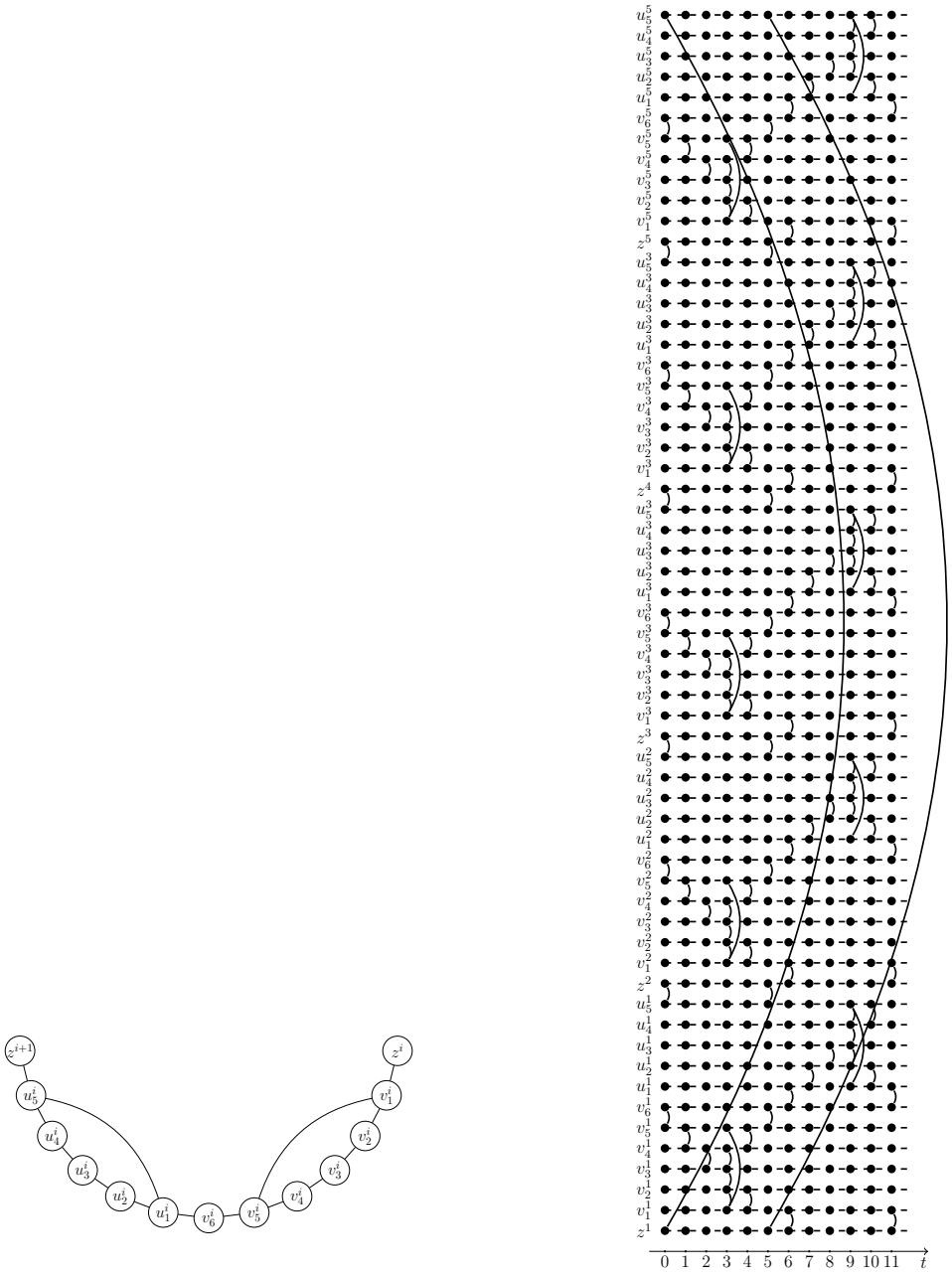
5.1 Motivational examples

A periodic graph $\mathcal{G}$ is defined as a sequence of (possibly disconnected) subgraphs of a graph G on the same set of nodes V. Because those subgraphs define the structure of $\mathcal{G}$, it seems natural to wonder if $c_{\circledcirc}(G) \leq f(c(G))$ for some function f. We start with the simplest function and inquire if $c_{\circledcirc}(G) \leq c(G)$. If so, then we would have a "simple" upper bound on $c_{\circledcirc}(G)$. Computing $c(G)$ is EXPTIME-COMPLETE in general (see [136]). Thus, not only would this bound still be hard to compute if it held, but if it were not true, then $c_{\circledcirc}(G)$ might be much more difficult to compute in general.

Unfortunately, we answer this question in the negative.

Our first counterexample, presented in Proposition 5.1, will be useful later on. Recall from [60] that outerplanar graphs have cop number 2.

In Proposition 5.1 we draw some periodic graphs as *link streams* instead of arenas for simplicity. We introduced link streams in Section 3.2. The



(a) H_i (b) $\mathcal{G}$

Figure 5.1: Subraph $H_i \subset G$ and periodic graph $\mathcal{G}$ drawn as a link stream, used the proof of Proposition 5.1

representations are highly similar, the biggest difference being that each pair of directed edges in an arena is replaced with a single undirected edge in a link stream.

Proposition 5.1 (3-copwin outerplanar periodic graph)**.** *There exists an outerplanar graph G such that $3 = c_{\circledcirc}(G) > c(G) = 2$.*

Proof. Let us show that the robber, denoted by R, can win against two cops, denoted by C_1 and C_2 on some periodic graph $\mathcal{G}$ with an outerplanar footprint G. Given a player $X \in \{R, C_1, C_2\}$, we write $X = a$ to mean that player X is on node a.

Consider the graph H_i shown in Figure 5.1(a), where $i \in \mathbb{N}$. Let G be the outerplanar graph obtained from $H_1 \cup H_2 \cup \cdots \cup H_5$ by identifying z^6 with z^1.

The periodic graph $\mathcal{G}$ is shown in Figure 5.1(b), where it is drawn as a link stream (recall Section 3.2). We say the robber *escapes* from H_i if it can move to $G \setminus H_i$.

We show the following.

1. For every $1 \leq i \leq 5$, there is a configuration $\mathcal{C}_i$ in $\mathcal{G}[H_i]$ from which the robber can either prevent its capture or escape to $\mathcal{G}[G \setminus H_i]$.

2. For every $1 \leq i \leq 5$, if the robber moves from $\mathcal{G}[H_i]$ to $\mathcal{G}[H_j]$ for some $i \neq j$, it can move the game into configuration $\mathcal{C}_j$ in $\mathcal{G}[H_j]$.

3. No matter where the cops start the game, the robber can move the game into configuration $\mathcal{C}_i$ of $\mathcal{G}[H_i]$ for some $1 \leq i \leq 5$.

Let $1 \leq i \leq 5$.

Part 1 Let us show first that if $(C_1, C_2, R) = (v_1^i, u_5^i, v_3^i)$ in G_0, then either the robber escapes from H_i through z^i or z^{i+1} (or z^1 when $i = 5$), or forever moves back and forth between v_3^i and u_3^i.

Since the cops play first, in G_3 cop C_1 either moves to v_2^i, v_5^i or stays still. As we can see from Figure 5.1(b), if C_1 moves to v_5^i, the robber escapes to $G \setminus H_i$ through z^i along the journey $((v_3^i, v_2^i), (v_2^i, v_1^i), (v_1^i, z^i))$ from time 3 to time 6. Otherwise, let the robber move to u_3^i, threatening the cop C_2, before moving back to v_3^i. In the meantime, cop C_1 either moves along the edge $v_1^i v_2^i$ or stays still on z^i. In the second case, the robber moves back to v_3^i and repeats the same configuration as the original one. In the first case, the cop

needs another period to move to v_3^i. Thus, when the robber arrives back on v_5^i in G_0, it notices this cop on v_2^i and stays on v_5^i. Cop C_1 cannot move until G_3. In G_3, this cop either moves to v_3^i, to v_1^i or stays on v_2^i. If it moves to v_3^i, the robber moves to v_1^i along the edge $v_5^i v_1^i$ and escapes to $G \setminus H_i$. Otherwise, the cop moves to v_1^i or stays on v_2^i. Then, the robber moves back to u_3^i to threaten cop C_2. In this case, the robber moves back and forth *between* v_3^i and u_3^i.

By inspecting Figure 5.1(b), we can see that the same situation happens with cop C_2: either the robber can escape to $G \setminus H_i$ through z^{i+1}, move to u_3^i or cannot move further than u_1^i to avoid capture. In the two latter cases it can move back to threaten C_1.

This shows the first part of the proof.

Part 2 The link streams in Figure 5.2 and Figure 5.3 show the only situations in which a cop, C_1 or C_2, moves out of its occupied node v_1^i or u_5^i. The robber escapes from H_i to some other H_j in both cases and moves the game either into the configuration $\mathcal{C}_j$ or one in which there is a single cop in H_j. This is winning for the robber.

This shows the second part of the proof.

Part 3 Finally, notice that initially the cops might be positioned in some H_i, H_j while the robber is in another H_k, with $i \neq j \neq k$. Then, the robber can apply its strategy that we described above in H_k as if the cops were on v_1^k and u_5^k because *eventually* they must move to those positions if they hope to capture the robber.

This shows the last part of the proof.

Therefore, the robber strategy we described above is always winning for the robber no matter where the cops start the game. It follows that $\mathcal{G}$ cannot have cop number 2 or less.

Finally, we show 3 cops are sufficient to capture the robber. If the two cops above do not move from their configuration, the robber cannot escape from H_i. Thus, a third cop can stubborn walk toward the robber. Eventually, both endpoints of a chord would be occupied by a cop and one of them could safely move toward the robber without the robber being able to escape. The robber thus eventually gets captured. □

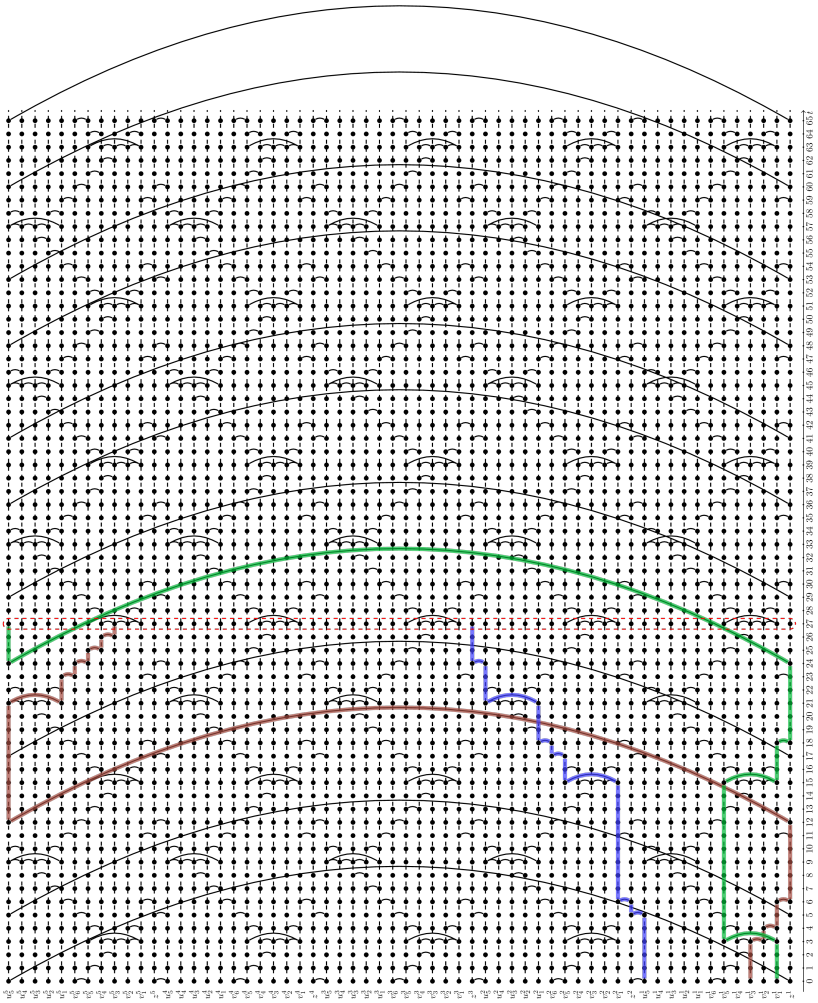


Figure 5.2: The first 66 snapshots of $\mathcal{G}$, drawn as a link stream. The robber moves along a journey, in red, against cop C_1, in green, who moved from v_1^1 to v_5^1 in G_3, and cop C_2 in . Timestep 27 is highlighted, when a robberwin configuration is created. The figure is rotated to fit the page.

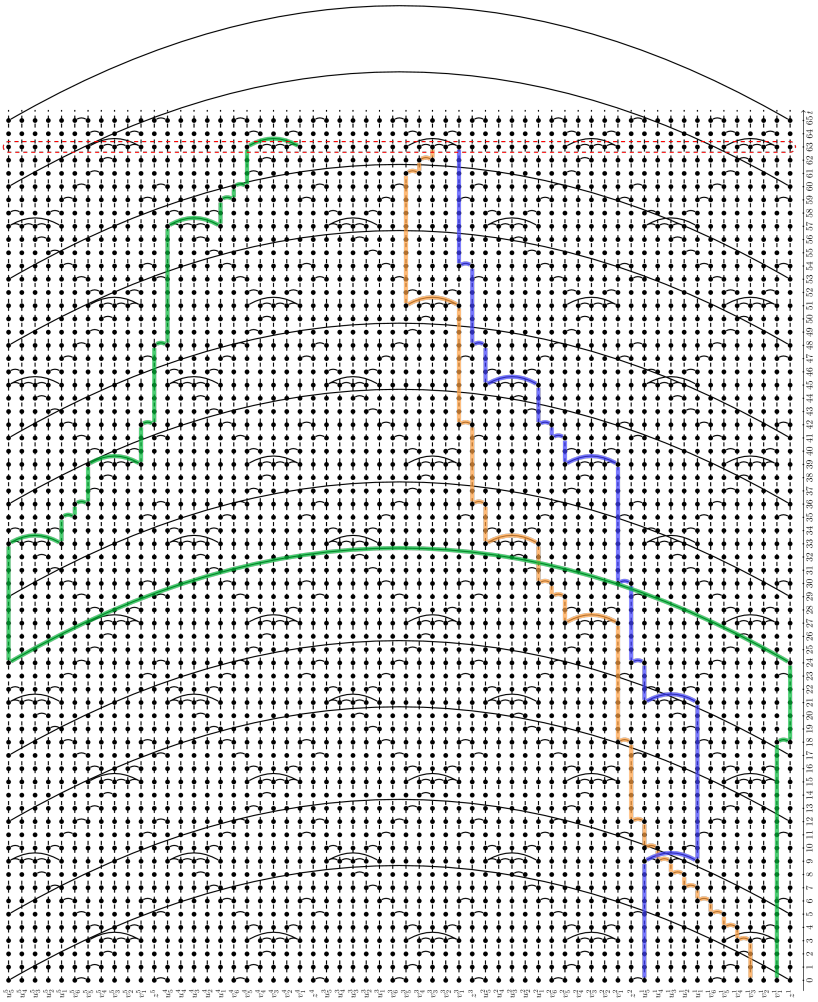


Figure 5.3: The first 66 snapshots of $\mathcal{G}$, drawn as a link stream. The robber moves along a journey, in orange, against cop C_1, in green, and cop C_2, in blue, who moved from u_5^1 to u_1^1 in G_9. Timestep 63 is highlighted, when a robberwin configuration is created. The figure is rotated to fit the page.

The following construction is based on the hypercube Q_3. Maamoun and Meyniel [153] showed that[1] $c(Q_k) = \left\lceil \frac{k+1}{2} \right\rceil$ for every $k \geq 0$. Hence, $Q_3 = 2$. We use the usual construction for Q_3 and write each node as a bit sequence of length 3.

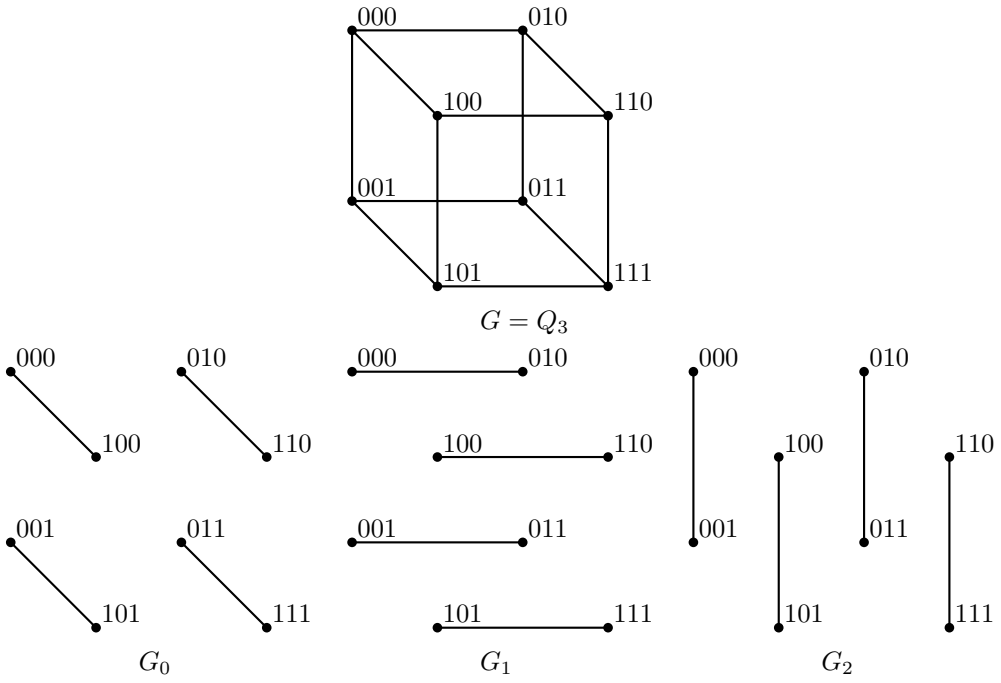


Figure 5.4: Periodic graph used in Proposition 5.2

Proposition 5.2 (3-copwin cube periodic graph)**.** *There exists a periodic graph $\mathcal{G}$ with footprint $G = Q_3$ such that $c(\mathcal{G}) = 3$.*

Proof. Let us describe a periodic graph $\mathcal{G} = (G_0, G_1, G_2)^*$ with footprint $G = Q_3$ such that $c(\mathcal{G}) = 3$. In G_0, only edges that change the first bit appear. In G_1, only those that change the second bit appear and so on for G_2. This is shown in Figure 5.4. We claim two cops cannot catch the robber.

Let us call a 4-cycle of Q_3 a *face*. Any subgraph of any snapshot of $\mathcal{G}$ that induces a face in Q_3 is also called a face.

We wish to preserve the following invariant for the robber:

[1] Maamoun and Meyniel [153] actually showed the more general result that the cartesian product of k trees has cop number $\left\lceil \frac{k+1}{2} \right\rceil$. The k-dimensional hypercube Q_k can be written as the cartesian product of k paths of length two, hence the result.

(I) Two cops are never in the same face as the robber at the beginning of their turn.

Let us show that if invariant I holds in G_t for all $t \geq 0$, then no matter where the cops move, the robber will not be on a 2-temporal corner of the cops' positions. Then, by Proposition 3.4, $c(\mathcal{G}) > 2$.

Suppose otherwise, that (t, u) is a 2-temporal corner of $(t+1, v), (t+1, w)$. Observe that for any time t and nodes $r, c \in V$ with $r \neq c$, $|N_t[r, \mathcal{G}] \cap N_{t+1}[c, \mathcal{G}]| \leq 1$. Moreover, let $N_t[u, \mathcal{G}] = \{x, u\}$. Then, $\deg_t(u) = \deg_{t+1}(v) = \deg_{t+1}(w) = 1$ and $E(G_t) \cap E(G_{t+1}) = \emptyset$. Thus, either $N_{t+1}[v, \mathcal{G}] = \{v, x\}$ and $N_{t+1}[w, \mathcal{G}] = \{w, u\}$, or $v = x$ and $N_{t+1}[w, \mathcal{G}] = \{w, u\}$, without loss of generality. In both cases, $(uxvw)$ is (part of) a face of G. Indeed, only the edges that change the same bit appear at time $t + 1$, so either $wv \in E(G)$ (when $v \neq x$) or $wy \in E(G)$ where $N_{t+1}[v, \mathcal{G}] = \{v, y\}$ (when $v = x$). Then, we either have $xu, xv, vw, wu \in E(G)$ or $xu, vy, yw, wu \in E(G)$ which are both faces of Q_3.

Therefore, no matter where the robber moves to, the cops will start their turn in G_{t+1} in the same face as the robber. This contradicts our invariant I.

Let us show now that the robber can play so that I is always true.

Since Q_3 has 6 faces, the robber can avoid choosing an initial position in a face that contains both cops in G_0.

Let us show that if I is true before the cops have played, then it will remain true after the robber has moved. Without loss of generality because of the symmetries in the snapshots of $\mathcal{G}$, suppose the robber is on $(0, 000)$ and the cops are on $c_1 \neq 000 \neq c_2$ in G_1. We moreover assume that $\{000, c_1, c_2\}$ is contained in a face of Q_3 since otherwise the robber could easily move so that I is true.

The node 000 is contained in 3 faces of Q_3, but no cop is on 000, so there are 9 cases.

1. The cops are on (001)(011). In G_0, the robber moves to 100 and avoids ending in the same face as the cops.

2. The cops are on (001)(101). The only way for the cops to occupy this edge in G_1 was for them to start on this edge in G_0 and stay on their position. Therefore, in G_0 the robber is in the same face $(000, 100, 101, 001)$ as the cops before the cops have played. This violates our assumption that I was true before the cops played.

3. The cops are on (010)(011). In G_0, the robber moves to 100 and avoids ending in the same face as the cops.

4. The cops are on (010)(110). The only way for the cops to occupy this edge in G_1 was for them to start on this edge in G_0 and stay on their position. Therefore, in G_0 the robber is the same face $(000, 100, 110, 010)$ as the cops. This violates our assumption that I was true before the cops played.

5. The cops are on (100)(101). Before the cops moved in G_0, all players were in the same face $(000, 100, 101, 001)$. This violates our assumption that I was true before the cops played.

6. The cops are on (100)(110). Before the cops moved in G_0, all players were in the same face $(000, 100, 110, 010)$. This violates our assumption that I was true before the cops played.

7. The cops are on $\{010, 100\}$. Before the cops moved in G_0, all players were in the same face $(000, 010, 110, 100)$. This violates our assumption that I was true before the cops played.

8. The cops are on $\{010, 001\}$. The robber moves to 100 and avoids ending in the same face as the cops.

9. The cops are on $\{100, 001\}$. Before the cops moved in G_0, all players were in the same face $(000, 100, 101, 001)$. This violates our assumption that I was true before the cops played.

Thus, from $(0, 000)$ the robber can move so that I will be true in G_1. By symmetry, this holds for every robber position under optimal play. Therefore, the invariant I always holds and $c(\mathcal{G}) > 2$.

We argue that $c(\mathcal{G}) \leq 3$. Let three cops start on 000, 010 and 111 in G_0. The robber starts either on 001 or 101 in order to avoid getting captured in the first turn. The cops stay still in G_0. The robber must end its turn on 001, otherwise the cop on 111 would make the catch in G_1. In G_1, no cops move so that the robber end its turn either on 001 or 011. Since there are cops on 000 and 010, the robber gets captured in G_2. □

Notice that in the previous result, $c(\mathcal{G}) = 3 > c(G) = 2 = \gamma(G)$, where $\gamma(G)$ is the domination number of G. In the static case, the domination

number is a trivial upper bound on $c(G)$ and $c_s(G)$, where $c_s(G)$ is the cop number when the robber has speed $s \geq 1$. So this also shows that there exists periodic graphs $\mathcal{G}$ such that for every $s \geq 1$, $c(\mathcal{G}) > c_s(G)$.

Proposition 5.2 shows the existence of a periodic graph with eight vertices and cop number 3. Yet, the smallest 3-copwin graph has 10 vertices [17].

5.2 About the maximum and minimum cop numbers of the snapshots

In the previous section, we showed examples where $c(\mathcal{G}) > c(G)$. Here, we take the opposite direction and show examples where $c(\mathcal{G}) < c(G)$, to highlight that both cases are possible. The main question here is: How small can $c(\mathcal{G})$ be compared to $c(G)$?

The following result, along with its corollary, answers this.

Lemma 5.3. *For any $k \geq 3$ and $1 \leq k' < k$ such that there exists a k-copwin graph with a spanning k'-copwin subgraph, there exists an at most k'-copwin periodic graph whose footprint is k-copwin.*

Proof. Let G be any graph with $c(G) = k$ and H a spanning k'-copwin subgraph of G. It suffices to let H appear long enough in a periodic graph $\mathcal{G}$ so that $c(\mathcal{G}) \leq k'$. Then, we can cover the remaining edges of G in the remaining snapshots with spanning trees, so that $c(G) = k$ and $c(\mathcal{G}) \leq k'$. □

Corollary 5.4. *There exists a copwin periodic graph whose footprint is 3-copwin.*

Proof. Given any graph H, let $r(H) := \min_{x \in V(H)} \max_{y \in V(H)} d_H(x, y)$ be the radius of H. Let G be the Petersen graph, so $c(G) = 3$. Let $\mathcal{G} = (G_0, \ldots, G_{p-1})^*$ be such that the first snapshots $G_0, \ldots, G_l$ contain a minimum spanning tree T of G, with $l \geq r(T)$. The remaining snapshots contain different spanning trees to cover the edges of G. The cop starts on a node x such that $\max_{y \in V} d_T(x, y) = r(T)$, so it can walk along T and reach every node of $\mathcal{G}$ in the first l snapshots. The robber cannot escape, so $c(\mathcal{G}) = 1$. □

Remark 5.5. *We mostly focus on always connected periodic graphs here. Occasionally, a construction will only be temporally connected. On one hand, periodic graphs that are always connected are easier to construct systematically than their temporally connected counterpart since they often come from*

decomposing the footprint into a sequence of connected subgraphs. On the another hand, our constructions of temporally connected periodic graphs tend to be sparser. As such, always connected periodic graphs can also be easier to analyse when the subgraphs are highly symmetric. We mostly use temporally connected periodic graphs in order to construct nontrival schedules and help one player or the other.

Given the two parameters $c(G_{\min})$ and $c(G_{\max})$, introduced in Subsection 3.4.5, one might expect the following pair of inequalities to hold:

$$\min(c(G), c(G_{\min})) \overset{?}{\leq} c(\mathcal{G}) \overset{?}{\leq} \max(c(G), c(G_{\max})).$$

Indeed, this forms the widest range of cop numbers that only uses parameters from $c(G_0), c(G_1), \ldots, c(G_{p-1})$ along with $c(G)$. Nevertheless, we show both are false. We introduce the following notation for convenience and say that $\mathcal{G}$ is (a, b, c)-copwin if $c(G) = a$, $c(G_{\max}) = b$ and $c(\mathcal{G}) = c$.

Theorem 5.6 ((1,1,2)-copwin)**.** *The inequality* $c(\mathcal{G}) \leq \max(c(G), c(G_{\max}))$ *is false.*

Proof. Consider the periodic graph $\mathcal{G}$ whose snapshots are shown in Figure 5.5. The footprint G is shown in Figure 5.6. Each snapshot is a path on 9 vertices, so it is copwin. The footprint G has a universal vertex (node 8), thus it is also copwin. The list of neighbourhoods of $\mathcal{G}$ is shown in Table A.1 on Page 161 and a careful inspection shows that $\mathcal{G}$ contains no temporal corner. Therefore, it cannot be copwin by Lemma 3.3.

We now show that $\mathcal{G}$ is 2-copwin by describing a winning strategy for two cops. Let two cops start on nodes 2 and 4 in G_0. The robber must start on either $3, 5$ or 7 to avoid capture in G_0.

1. If the robber starts on 3, the cop on 4 moves to 1 and the cop on 2 moves to 0. The robber cannot move to 1 nor to 5, since $15 \in E(G_1)$, so it stays on 3. In G_1, 03 appears and one cop makes the catch.

2. If the robber starts on 5, the cop on 4 moves to 1 and the cop on 2 moves to 6. The robber cannot move to 6 nor to 3 because $63 \in E(G_1)$. Therefore, the robber stays on 5 and gets captured by the cop on 1 that moves along the edge 15 in G_1.

3. If the robber starts on 7, the cops move to 0 and 8 in G_0. The robber cannot move to 0, so it stays on 7 and gets captured by the cop on 8 in G_1.

□

Theorem 5.7 ((2,2,1)-copwin)**.** *The inequality* $\min(c(G), c(G_{\min})) \leq c(\mathcal{G})$ *is false.*

Proof. Let G be a bow tie graph formed with two 4-cycles joined on a vertex v. Let $G_0 = G_1 = G_2$ be the subgraph of G induced by removing one edge of a cycle and $G_3 = G_4 = G_5$ be the subgraph of G induced by removing one edge of the other cycle. Thus, every snapshot consists in a 4-cycle joined to a path of length at most 3 on vertex v. Moreover, v is a cutvertex of every snapshot. Clearly, G, $G_0, \ldots, G_5$ are all 2-copwin. Let $\mathcal{G} = (G_0, \ldots, G_5)^*$. Place one cop on v in G_0. The robber initially starts on the cycle of G_0. The cop waits until G_3 for the path to appear underneath the robber. The robber cannot have crossed onto the other cycle since the cop is on the cutvertex v. The robber is now on a path of length at most 3, so the cop walks to capture the robber. □

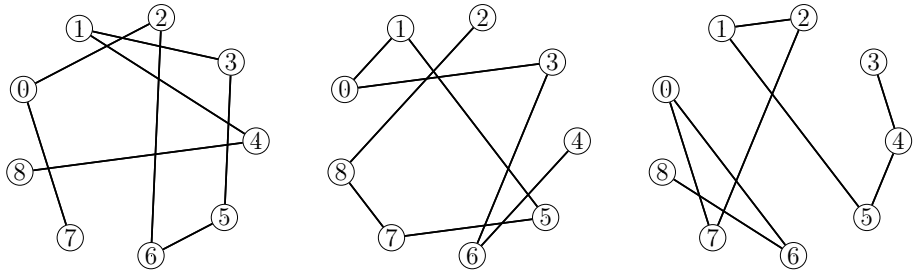
Theorem 5.7 is a dual result of Theorem 5.6 and together they serve to highlight how loose the connection is between the cop numbers of a periodic graph, its snapshots and footprint.

5.3 Completing the table of copwin periodic graphs

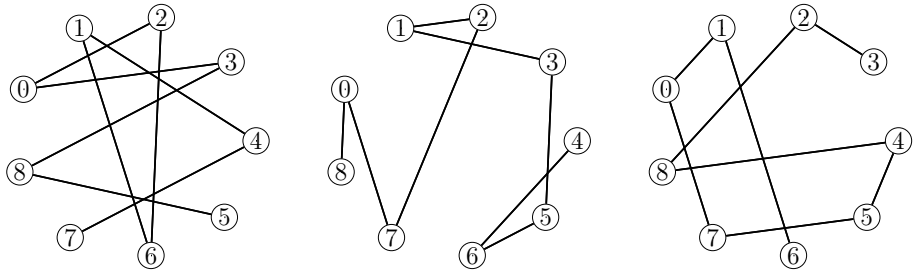
In the previous sections, we showed that simple lower and upper bounds on $c(\mathcal{G})$ that depend on $c(G), c(G_0), \ldots, c(G_{p-1})$ do not hold in general. In this section, we want to further emphasize the disconnection between those values by presenting Table 5.1 that shows that nearly all combinations of values (between 1 and 3) of $c(\mathcal{G})$, $c(G)$ and $c(G_{\max})$ are possible.

Out of the 27 possible combinations of those parameters, 3 remain to be determined.

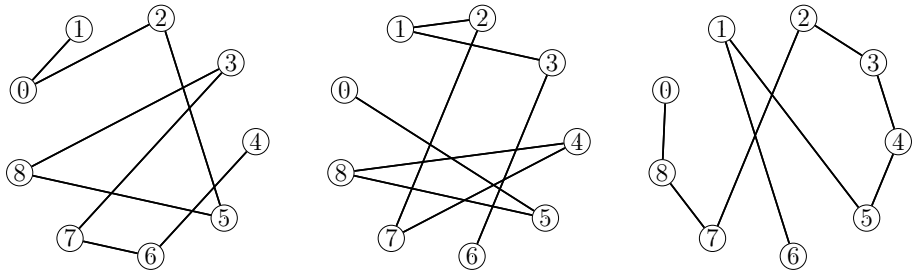
First, let us show that from any particular result, we can always increase the number of nodes without changing the period.



(a) G_0 (b) G_1 (c) G_2



(d) G_3 (e) G_4 (f) G_5



(g) G_6 (h) G_7 (i) G_8

Figure 5.5: The $(1, 1, 2)$-copwin periodic graph presented in Theorem 5.6.

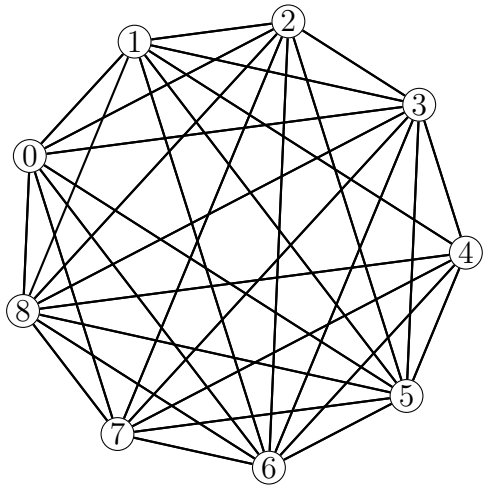


Figure 5.6: The footprint of the periodic graph from Theorem 5.6 and Theorem 5.25.

Lemma 5.8. *For any (a, b, c)-copwin periodic graph $\mathcal{G}$, with n nodes and period $p \geq 2$, and any integer $N \geq n$, there exists a periodic graph $\mathcal{G}'$ with N nodes and period p that is (a, b, c)-copwin.*

Proof. Let $\mathcal{G} = (G_0, \ldots, G_{p-1})^*$ with footprint G be as in the statement and $P = (u_1, \ldots, u_{N-n+1})$ be a path with $N - n + 1$ nodes, where $N \geq n$ is some integer. Let $u \in V(G)$. For every $1 \leq i \leq p-1$, let H_i be obtained by identifying u with u_1 and let $\mathcal{G}' = (H_0, \ldots, H_{p-1})^*$ with footprint H. By construction, $c(\mathcal{G}'[G]) = c(\mathcal{G})$.

The robber on $\mathcal{G}'$ can play on $\mathcal{G}'[G]$ and win against less than $c(\mathcal{G}'[G]) = c(\mathcal{G})$ cops. Therefore, $c(\mathcal{G}') \geq c(\mathcal{G}'[G]) = c(\mathcal{G})$.

Let us show that $c(\mathcal{G}') \leq c(\mathcal{G})$. By construction, the homomorphism $h : H \to G$ that maps $V(P)$ to u and is the identity on G is a retraction of H and every snapshots. Thus, $c(\mathcal{G})$ cops can play on $\mathcal{G}'$ so that if at any time the robber moves to P the cops act as if the robber moved to u. Since u is a cutvertex between P and G, eventually the robber is either captured on u, since $c(\mathcal{G}) = c(\mathcal{G}'[G])$, or it is somewhere on P while a cop is on u. This latter cop eventually makes the catch. Therefore, $c(\mathcal{G}') \leq c(\mathcal{G})$ and the equality holds. Similarly, $c(H) = c(G)$ and $c(H_{\max}) = c(G_{\max})$. □

$c(G)$	$c(G_{\max})$	$c(\mathcal{G})$	Reference	$c(G)$	$c(G_{\max})$	$c(\mathcal{G})$	Reference
1	1	1	Lemma 5.20	2	2	3	Lemma 5.19
1	1	2	Theorem 5.6	2	3	1	Proposition 5.11
1	1	3	*Undetermined*	2	3	2	Corollary 5.12
1	2	1	Lemma 5.21	2	3	3	Proposition 3.10
1	2	2	Lemma 5.16	3	1	1	Corollary 5.4
1	2	3	Proposition 5.18	3	1	2	Theorem 5.25
1	3	1	Lemma 5.22	3	1	3	*Undetermined*
1	3	2	Lemma 5.10	3	2	1	Lemma 5.13
1	3	3	Proposition 5.28	3	2	2	Corollary 5.15
2	1	1	Lemma 5.23	3	2	3	Proposition 5.26
2	1	2	Lemma 5.24	3	3	1	Lemma 5.27
2	1	3	*Undetermined*	3	3	2	Lemma 5.14
2	2	1	Theorem 5.7	3	3	3	Lemma 5.20
2	2	2	Lemma 5.20				

Table 5.1: Summary of results on periodic graphs. These are existence results of periodic graphs $\mathcal{G}$ with cop number $c(\mathcal{G})$. The cop number of the footprint is noted $c(G)$ and the maximum cop number of the snapshots is $c(G_{\max})$.

5.3.1 Constructions based on the Petersen graph

Some footprints are particularly useful. For instance, we derived a lot of constructions from the Petersen graph.

The following result about the diameter $d(G)$ of a graph G is useful for some constructions. This is a simple and well-known result, so we leave out the proof.

Lemma 5.9. *Every graph G has a spanning tree T such that*

$$d(T) \leq 2d(G).$$

Lemma 5.10 ((1,3,2)-copwin)**.** *There exists a periodic graph $\mathcal{G}$ with $c(G) < c(\mathcal{G}) < c(G_{\max})$.*

Proof. Let P_e be the Petersen graph, labelled as in Figure 5.7 and x be a node that is not in $V(P_e)$. Let C_o be the cyle (a, b, c, d, e) and C_i the cycle (f, h, j, g, i). Let G be the graph that contains P_e and x as a universal vertex. Let $H \subset G$ be the subgraph with $E(H) = E(C_o) \cup \{af, bg, ch, di, ej, ax\}$. Let

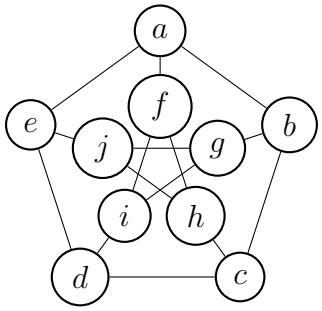


Figure 5.7: A labelled Petersen graph

$\mathcal{G}$ be the periodic graph such that for every $t \equiv 0 \pmod 5$, G_t contains P_e along with a single edge from x to P_e until all such edges have been exhausted, while for every $t \not\equiv 0 \pmod 5$, $G_t = H$. Observe that every snapshot is connected.

Since x is a universal vertex in G, $c(G) = 1$. Moreover, $c(G_{\max}) = c(G_0) = 3$.

Let us show that $c(\mathcal{G}) = 2$. In every snapshot x can be retracted either to C_o or C_i, so a single cop gains nothing in starting on x. Moreover, $c(C_o) = 2$ and $C_o \subset G_t$ for every time t, so the robber can escape from a single cop by forever moving on C_o. Thus, $c(\mathcal{G}) > 1$. But, $c(H) = 2$ and H appears for 4 consecutive snapshots. Let two cops start on a and d in G_0 and stay on their positions on their turn. The robber must be somewhere in $\{g, h, j\}$ on the cops' turn in G_1 if it wants to avoid direct capture in this snapshot. Any cop closest to the robber will take two turns to move to the robber's position while the latter cannot escape. Thus, the robber gets captured in G_2 at the latest and $c(\mathcal{G}) = 2$.

□

Proposition 5.11 ((2,3,1)-copwin)**.** *There exists a periodic graph $\mathcal{G}$ with $c(\mathcal{G}) < c(G) < c(G_{\max})$.*

Proof. Let P_e be the Petersen graph and x, y be two nodes that are not in $V(P_e)$. Identify the outer and the inner cycle of P_e as C_o and C_i as in the proof of Lemma 5.10. Then, connect x to every node of C_o and y to every node of C_i. Let G be the resulting graph. Let us show that G does not contain any corner. Let u be any node of G. Clearly, neither x nor y can be corners, so suppose $u \in V(P_e)$. Since P_e has no corner and is induced

in G, u cannot be a corner of another vertex in P_e. But, u also cannot be a corner of x nor y, since u has neighbours in both C_i and C_o, while x is only adjacent to vertices of C_o and y to vertices of C_i. Therefore, G is not dismantlable, so $c(G) > 1$ by [167]. Moreover, observe that $\gamma(G) = 2$, with $\{x, y\}$ as the minimal dominating set. Since $c(G) \leq \gamma(G) = 2$ and $c(G) > 1$, we get $c(G) = 2$.

Let $\mathcal{G}$ be the following periodic graph. For every time $t \equiv 0 \pmod{11}$, $E(P_e) \subset E(G_t)$ and G_t contains one edge from x to C_0 and one edge from y to C_i until all edges from $\{x, y\}$ to P_e appear in $\mathcal{G}$. Thus, every such G_t is connected.

Let also T be the spanning tree of P_e shown in Figure 5.8, where $d(T) = 3 = \max_{x \in V(P_e)} d_T(c, x)$. Let $T' = T \cup \{ax, cy\}$, so that $d_{T'}(c, x) = 3$ and $d_{T'}(j, y) = 4$. Let $G_t = T'$ for every $t \not\equiv 0 \pmod{11}$.

Then, G_t is a spanning tree of G that appears for at least $10 > d(G_t) = 7$ consecutive snapshots, so a single cop can win on $\mathcal{G}$. That is, $c(\mathcal{G}) = 1$. Nevertheless, for every $t \equiv 0 \pmod{11}$, P_e is a retract of G_t given by mapping x to its unique neighbour and similarly for y. Then, by the classic result of Berarducci and Intrigila [27] that $c(H) \leq c(H')$ whenever H is a retract of H', $3 = c(P_e) \leq c(G_t)$. Finally, the smallest 4-copwin graph has 19 nodes (see [202]) while G has 12 nodes, so $c(G_t) \leq 3$ and $c(G_t) = c(G_{\max}) = 3$. Therefore, $\mathcal{G}$ is $(2, 3, 1)$-copwin.

□

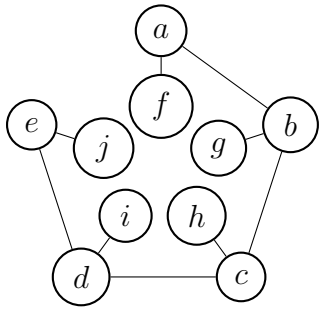
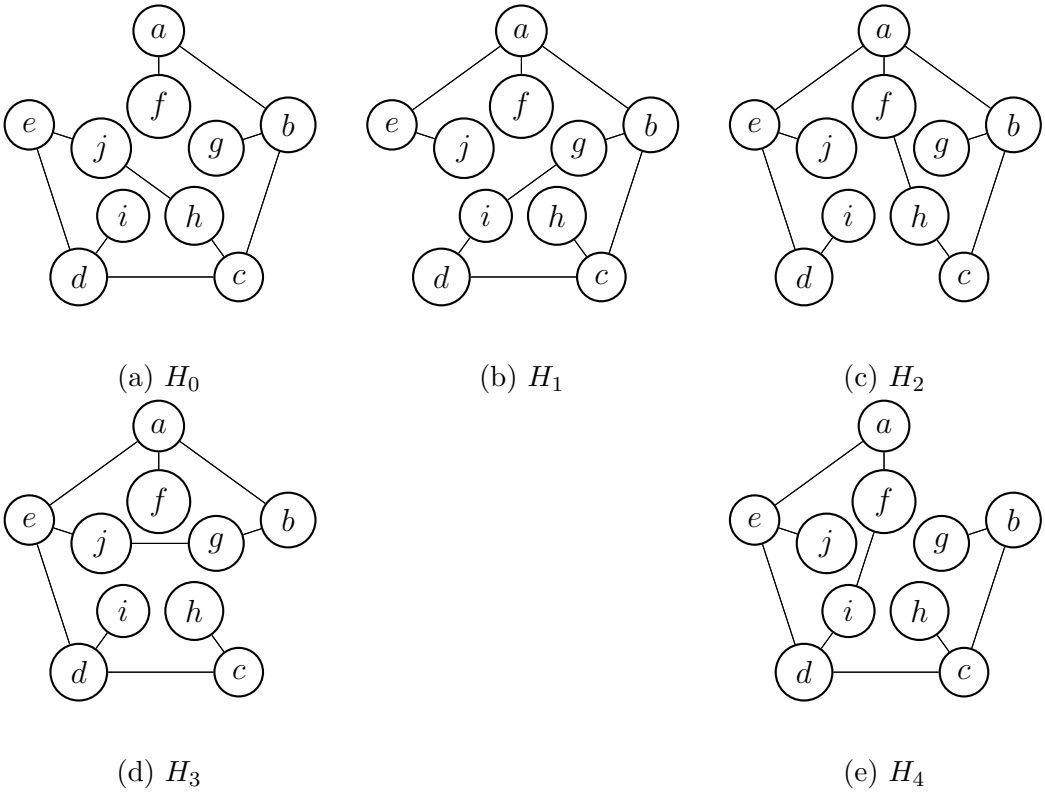


Figure 5.8: Spanning tree T used in the proof of Proposition 5.11

Corollary 5.12 ((2,3,2)-copwin)**.** *There exists a periodic graph with* $2 = c(G) = c(\mathcal{G}) < c(G_{\max}) = 3$.



(a) H_0 (b) H_1 (c) H_2

(d) H_3 (e) H_4

Figure 5.9: Sequence of graphs used in Lemma 5.13

Proof. This similar to Proposition 5.11. However, we replace the spanning tree of G by G itself. □

Lemma 5.13 ((3, 2, 1)-copwin)**.** *There exists a periodic graph $\mathcal{G}$ with $c(\mathcal{G}) < c(G_{\max}) < c(G)$.*

Proof. Let G be the Petersen graph labelled as in Figure 5.7 and $H_0, \ldots, H_4$ be the graphs shown in Figure 5.9. Then, let $\mathcal{G} = (G_0, \ldots, G_{p-1})^*$ be the

periodic graph where

$$G_t = \begin{cases} H_0, & \text{if } 0 \leq t \leq 3 \\ H_1, & \text{if } 4 \leq t \leq 7 \\ H_2, & \text{if } 8 \leq t \leq 11 \\ H_3, & \text{if } 12 \leq t \leq 15 \\ H_4, & \text{if } 16 \leq t \leq 19. \end{cases}$$

The footprint is the Petersen graph, so $c(G) = 3$. Every snapshot is always connected and contains a cycle of length five, so $c(G_{\max}) \geq 2$. Moreover, two cops can capture the robber on every snapshot, so $c(G_{\max}) \leq 2$ and $c(G_{\max}) = 2$.

Finally, consider the first 8 snapshots $G_0, \ldots, G_7$. Let a single cop start on c. The robber cannot start in $N_{H_0}[c] = \{b, c, h, d\}$. Neither can it start in $\{a, f, g\}$, since it would be stuck on a tree rooted at c. Finally, if it starts on i, it gets stuck on i when the cop moves to d. Thus, the robber must start on e or j. Let the cop wait until G_4, when H_1 appears. The robber cannot move out of the set of nodes $\{e, j, h, c, d, i\}$. In order for the robber to move to i before G_4, it must move to d, in which case the cop makes the catch on the next snapshot. Similarly, $h \in N_4[c, \mathcal{G}]$. Thus, the robber cannot safely end its turn on $\{d, i, h\}$ in G_3. Therefore, the robber must end its turn in G_3 somewhere in $\{e, j\}$. In this case, let the cop move to b in G_4. The robber is on a tree rooted at b until snapshot G_7 and the cop makes the catch.

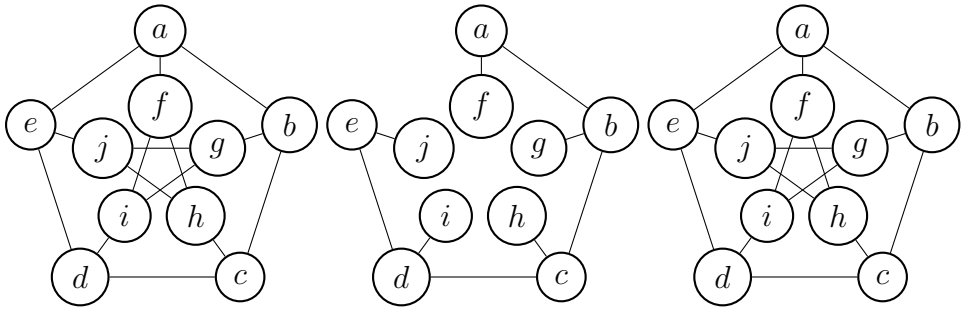
This shows that $c(\mathcal{G}) = 1$. □

Lemma 5.14 ($(3, 3, 2)$-copwin)**.** *There exists an always connected periodic graph $\mathcal{G}$ such that $c(G_{\min}) < c(\mathcal{G}) < c(G_{\max})$.*

Proof. Consider the periodic graph $\mathcal{G}$ drawn in Figure 5.10. The footprint is the Petersen graph that is 3-copwin. The first and third snapshot are the Petersen graph, while the second one is a spanning tree of the Petersen graph. Thus, the second snapshot is copwin and $c(G_{\min}) = 1 < c(G_{\max}) = 3$.

We call the cycle $C_i = (f, h, j, g, i)$ of G the *inner* cycle and the cycle $C_o = (a, b, c, d, e)$ the *outer cycle*.

Let us show that $c(\mathcal{G}) \leq 2$. If two cops are on nodes b and d in G_1, then the robber gets caught at the latest in G_2. Indeed, $N_1[b, \mathcal{G}] \cup N_1[d, \mathcal{G}] = \{a, b, c, d, e, g, i\}$, so the robber gets captured in G_1 if it stands on either of those nodes. Otherwise, the robber is on a node $x \in \{f, h, j\}$ in G_1. One



(a) Snapshots G_0 (b) Snapshot G_1 (c) Snapshots G_2

Figure 5.10: The periodic graph of Lemma 5.14

cop can move to the node y of G_1 such $xy \in E(G_1)$ since $y \in \{a, c, e\} \subset N_1[b, \mathcal{G}] \cup N_1[d, \mathcal{G}]$. Then, $(1, x)$ is a temporal corner of $(2, y)$ and the robber gets captured in G_2. Therefore, $c(\mathcal{G}) \leq 2$.

Let us show that $c(\mathcal{G}) > 1$. A single cop can only capture the robber in $\mathcal{G}$ on a leaf of the spanning tree T of G_1. But, all the leaves of T belong to C_i and their neighbours to C_o. Thus, the cop must visit the neighbour x of a leaf y of T in G_1 while the robber is on y in order to make the catch. But, G has girth five and $G = G_0$. Thus, there is a 5-cycle $C = (u_1, u_2, u_3, u_4, u_5)$ in G_0 such that $u_5 = y$ and $u_4 = x$. In G_0, the cop must be on either u_3 or u_2 in order to end its turn on u_4 in G_1. In any case, in G_0 the robber can either move to u_1 or to some other node of $V(G) \setminus V(C)$ and avoid getting captured.

Hence, $c(\mathcal{G}) = 2$ and this concludes the proof. □

Given a periodic graph $\mathcal{G}$ with footprint G, we say that a subgraph H of G is *stable* in $\mathcal{G}$ if it appears in all snapshots of $\mathcal{G}$.

Corollary 5.15 ($(3, 2, 2)$-copwin)**.** *There exists an (a, b, c)-copwin periodic graph with $a = 3$ and $c < a$.*

Proof. Let G again be the Petersen graph, so $c(G) = 3$, and let G be drawn as in Figure 5.11. We know that G is the smallest 3-copwin graph, in number of nodes, and is the unique 3-copwin graph with 10 nodes (see Baird et al. [17]). Let x be the centre node in Figure 5.11. From Baird et al.'s result, it follows that for every edge xu, $c(G \setminus \{xu\}) = 2$. Let $N_G(x) = \{a, b, c\}$ and $\mathcal{G} = (G_0, \ldots, G_{p-1})^*$ be such that the first snapshots $G_0, \ldots, G_{p-3}$ are

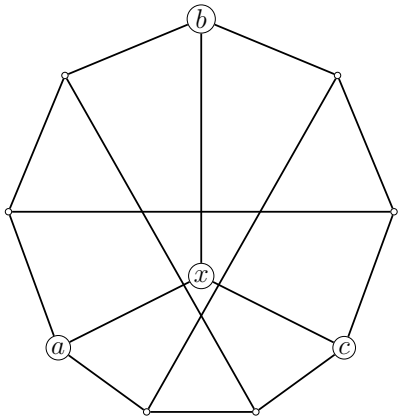


Figure 5.11: Another representation of the Petersen graph

equal to $G \setminus \{xa\}$. Then, we let $G_{p-2} = G \setminus \{xb\}$ and $G_{p-3} = G \setminus \{xc\}$. For our purposes, we choose p such that $p-3$ is greater or equal to the capture time of the robber on $G \setminus \{xa\}$ by two cops. Then, $c(\mathcal{G}) \geq 2$ since there is a stable 2-copwin subgraph in $\mathcal{G}$, however $c(\mathcal{G}) \leq 2$ since two cops can capture the robber on the first $p-3$ snapshots. Therefore, $c(\mathcal{G}) = 2$. By the same argument, $c(G_{\max}) = 2$. □

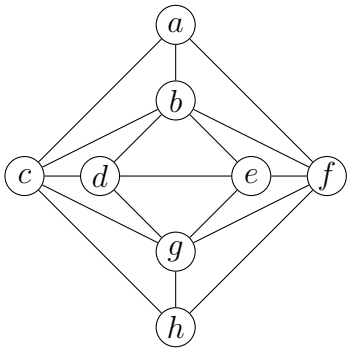
5.3.2 Arguments from the non-existence of temporal corners

Lemma 5.16 ((1,2,2)-copwin)**.** *There exists an always connected periodic graph with copwin footprint and* 2*-copwin snapshots that is* 2*-copwin.*

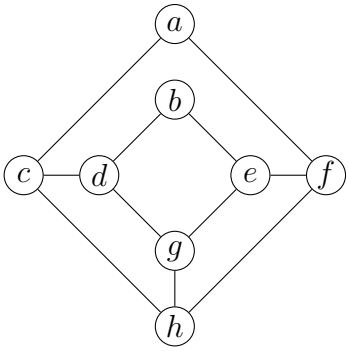
Proof. The periodic graph $\mathcal{G}$ in Figure 5.12 is formed with 2-copwin snapshots and copwin footprint. We can see every snapshot is 2-copwin since each of them contains an induced 4-cycle.

At most two cops are necessary to capture the robber on $\mathcal{G}$. Indeed, two cops can start on d and f in G_0, which is a dominating set of G_0. Then, $c(\mathcal{G}) \leq \gamma(G_0) = 2$ by Lemma 3.9.

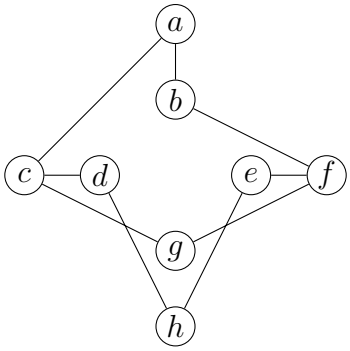
Inspection shows that $\mathcal{G}$ has no temporal corner, so it cannot be copwin by Lemma 3.3. Therefore, 2 cops are necessary and sufficient to capture the robber. □



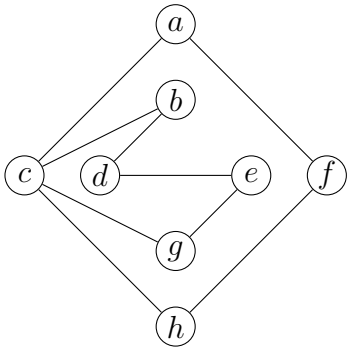
(a) Footprint G



(b) Snapshot G_0



(c) Snapshot G_1



(d) Snapshot G_2

Figure 5.12: The footprint G is copwin, the periodic graph $\mathcal{G} = (G_0, G_1, G_2)^*$ is not.

By the contrapositive of Proposition 3.4, if $\mathcal{G}$ does not have a k-temporal corner, then it cannot be k-copwin. This leads to a generic method to produce $(k+1)$-copwin periodic graphs that we further expend on in Section 5.4. We present some examples that result from this method here.

Lemma 5.17. *Let (t,u) be some temporal node in a periodic graph $\mathcal{G}$. Suppose that the sets $N_{t+1}[v_i,\mathcal{G}]$ and $N_{t+1}[v_j,\mathcal{G}]$ are pairwise disjoint for any two distinct $v_i,v_j \in N_t[u,\mathcal{G}]$. Then, (t,u) is not a $\deg_t(u)$-temporal corner in $\mathcal{G}$.*

Proof. Suppose otherwise, that (t,u) is a $\deg_t(u)$-temporal corner of $(t+1,w_1),\dots,(t+1,w_{\deg_t(u)})$. By definition, $|N_t[u,\mathcal{G}]| = \deg_t(u)+1$, since $\deg_t(u) = |N_{G_t}(u)|$. Therefore, at least two neighbours v_i, v_j of u in G_t are adjacent to the same node w_l in G_{t+1}, for some $1 \leq l \leq \deg_t(u)$. In turn, $w_l \in N_{t+1}[v_i,\mathcal{G}] \cap N_{t+1}[v_j,\mathcal{G}]$ so $N_{t+1}[v_i,\mathcal{G}] \cap N_{t+1}[v_j,\mathcal{G}] \neq \emptyset$, which is a contradiction. □

Proposition 5.18 ($(1,2,3)$-copwin)**.** *For every integers $n \geq 11$ and $p \geq 5$ odd, there exists an always connected periodic graph $\mathcal{G}$ with period p such that $c(G) < c(G_{\max}) < c(\mathcal{G})$.*

Proof. Let $\mathcal{G}$ be the periodic graph shown in Figure 5.13, where every snapshot is an 11-cycle. By construction, $c(G_{\max}) = 2$.

Let us show that $\mathcal{G}$ has no 2-temporal corner. By symmetry, it suffices to show that $(t,0)$ is not a 2-temporal corner of any $(t+1,x),(t+1,y)$ for any time t. Table 5.2 shows the set of neighbours $N_t[0,\mathcal{G}]$ of $(t,0)$ as well as the set of neighbours at time $t+1$ of neighbours of $(t,0)$, written $N_{t+1}[N_t[0,\mathcal{G}]]$. One can verify that the sets in $N_{t+1}[N_t[0,\mathcal{G}]]$ are pairwise disjoints for every time t, so $(t,0)$ is not a 2-temporal corner by Lemma 5.17.

Thus, $c(\mathcal{G}) \geq 3$ by Proposition 3.4. Moreover, the footprint is the complete graph K_{11}, see Figure 5.14, so $c(G) = 1$.

Let us show that $c(\mathcal{G}) \leq 3$. Let us place one cop on node 0, one on node 3 and another on 8. Let the cops wait until G_3. The robber must be on either 5 or 6 in order not to be captured in G_3. Observe that $N_3[5,\mathcal{G}] \cup N_3[6,\mathcal{G}] = \{4,5,6,7\} \subseteq N_4[0,\mathcal{G}] \cup N_4[2,\mathcal{G}] \cup N_4[9,\mathcal{G}] = \{0,2,4,5,6,7,9\}$, so when the cops in G_3 move to 0, 2 and 9, the robber gets stuck on a 3-temporal corner of the cops' positions. Then, the cops win in G_4.

We can extend the period of $\mathcal{G}$ without creating 2-temporal corners by adding pairs (G_3,G_4) at the end of $\mathcal{G}$. Note that this only generates odd

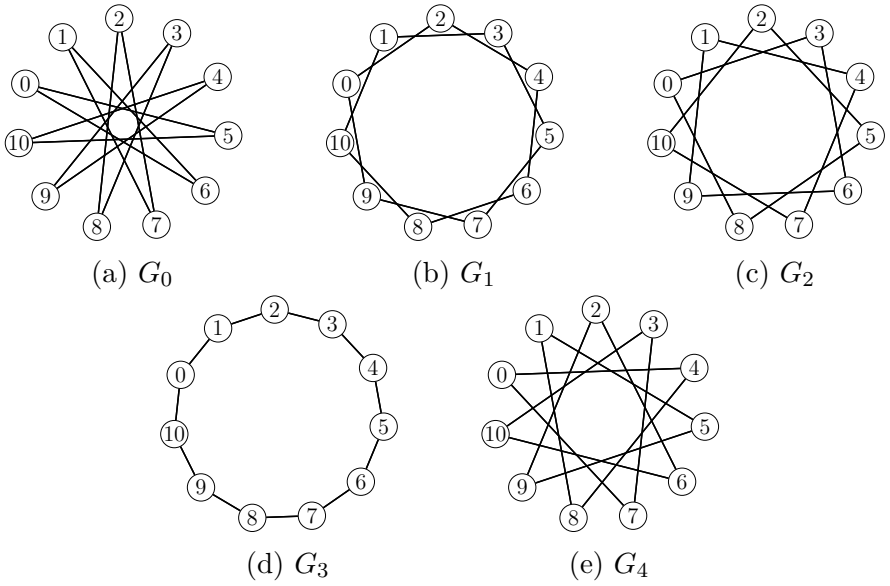


Figure 5.13: The periodic graph used in Proposition 5.18. Each snapshot is a cycle of length 11, the footprint is copwin and the periodic graph is 3-copwin.

periods. Similarly, we can always apply Lemma 5.8 to increase the number of nodes without changing the properties of $\mathcal{G}$. □

The construction in Proposition 5.18 is generic. Given an integer n, we simply fix a sequence of integers $(i_0, \ldots, i_{p-1})$ such that for every $0 \leq t \leq p-1$, $1 \leq i_t \leq \left\lfloor \frac{n}{2} \right\rfloor$ and $N_t[u, \mathcal{G}] = \{u, u + i_t \pmod n, u - i_t \pmod n\}$ for every node $u \in V$. When i_t is prime, $G_t \equiv {C_n}^2$. One can then play with the values of n and $(i_0, \ldots, i_{p-1})$ to generate more examples of periodic graphs.

Lemma 5.19 ($(2,2,3)$-copwin)**.** *There exists a periodic graph with* $2 = c(G) = c(G_{\max}) < c(\mathcal{G}) = 3$.

Proof. Consider the periodic graph $\mathcal{G}$ shown in Figure 5.15. Let us show that $\mathcal{G}$ has no 2-temporal corner. By symmetry, it suffices to show that for every time t, $(t, 0)$ is not a 2-temporal corner of any $(t+1, x), (t+1, y)$.

[2]Indeed, let n be prime and consider a cycle $C \subseteq C_n$ that starts and ends in 0. Then, $C = (0, i_t, 2i_t, \ldots, li_t)$. Since n is prime, for each $ki_t \in V(C)$ with $k \neq 1$, $k \nmid n$. Thus, $l + 1 = n$ and $C = C_n$. When n is not prime, G_t might contain a collection of disjoint cycles of length less than n.

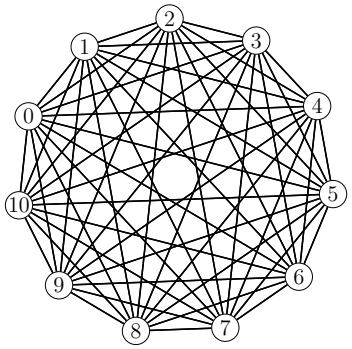


Figure 5.14: The footprint G of the periodic graph shown in Figure 5.13 is a complete graph K_{11}.

t	$N_t[0, \mathcal{G}]$	$N_{t+1}[N_t[0, \mathcal{G}]]$
0	$\{0, 5, 6\}$	$\{\{0, 2, 9\}, \{3, 5, 7\}, \{4, 6, 8\}\}$
1	$\{0, 2, 9\}$	$\{\{0, 3, 8\}, \{2, 5, 10\}, \{1, 6, 9\}\}$
2	$\{0, 3, 8\}$	$\{\{0, 1, 10\}, \{2, 3, 4\}, \{7, 8, 9\}\}$
3	$\{0, 1, 10\}$	$\{\{0, 4, 7\}, \{1, 5, 8\}, \{3, 6, 10\}\}$
4	$\{0, 4, 7\}$	$\{\{0, 5, 6\}, \{4, 9, 10\}, \{1, 2, 7\}\}$

Table 5.2: Neighbours of $(t, 0)$ for every t, as well as neighbours at time $t+1$ of neighbours of $(t, 0)$

Table 5.3 shows, for every time t, the set of neigbhours $N_t[0, \mathcal{G}]$ of $(t, 0)$ as well as the set of neighbours at time $t+1$ of neighbours of $(t, 0)$, denoted by $N_{t+1}[N_t[0, \mathcal{G}]]$. One can verify that the sets in $N_{t+1}[N_t[0, \mathcal{G}]]$ are pairwise disjoints for every t, so $(t, 0)$ is not a 2-temporal corner of any pair of temporal nodes for any time t, by Lemma 5.17.

Therefore, $c(\mathcal{G}) \geq 3$ by Proposition 3.4. Let us show that $c(\mathcal{G}) \leq 3$. Let us put three cops on $0, 3, 6$. The cops pass their turn in G_0, so the robber must be on either 7 or 10 in G_1.

1. Suppose the robber is on 7. Let the cops move to $9, 5, 4$. The robber must stay on 7, which is adjacent to 4 in G_2. The cop on 4 makes the catch in G_2.

2. Suppose the robber is on 10. Let the cops move to $2, 1, 8$. The robber must stay on 10, which is adjacent to 2 in G_2, so the cop on 2 makes

the catch in this snapshot.

Thus, $c(\mathcal{G}) = 3$.

The footprint G is shown in Figure 5.16. One can verify that G has no corner, so $c(G) > 1$. This graph is symmetric, so it suffices to show that 0 is not a corner of any other node. We can see that $N_G[0] = \{0, 2, 3, 5, 6, 8, 9\}$. But, $23, 56, 89 \notin E(G)$ so none of $2, 3, 5, 6, 8, 9$ can be covers of 0 in G.

Nevertheless, $N_G[0] \cup N_G[4] = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10\} = V$, so G has domination number $\gamma(G) = 2$ and $c(G) = 2$. Finally, every snapshot is a cycle of length 11, so $c(G_{\max}) = 2$. □

t	$N_t[0, \mathcal{G}]$	$N_{t+1}[N_t[0, \mathcal{G}]]$
0	$\{0, 5, 6\}$	$\{\{0, 2, 9\}, \{3, 5, 7\}, \{4, 6, 8\}\}$
1	$\{0, 2, 9\}$	$\{\{0, 3, 8\}, \{2, 5, 10\}, \{1, 6, 9\}\}$
2	$\{0, 3, 8\}$	$\{\{0, 2, 9\}, \{1, 3, 5\}, \{6, 8, 10\}\}$
3	$\{0, 2, 9\}$	$\{\{0, 5, 6\}, \{2, 7, 8\}, \{3, 4, 9\}\}$

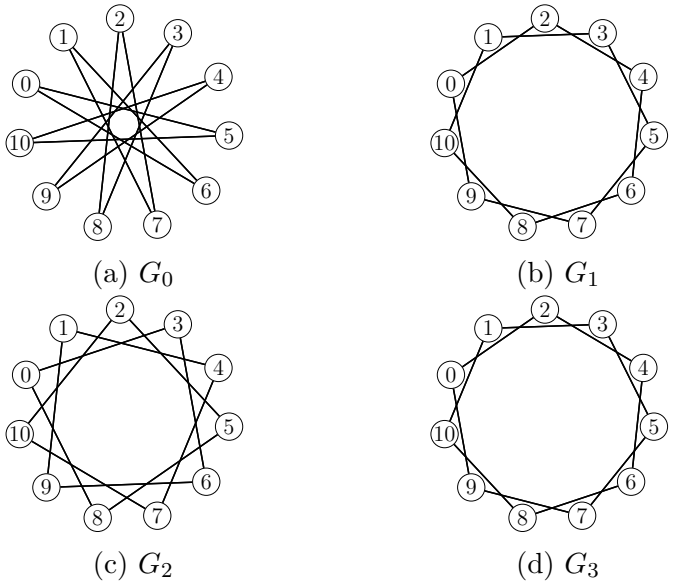
Table 5.3: Neighbours of $(t, 0)$ for every t, as well as neighbours at time $t+1$ of neighbours of $(t, 0)$

5.3.3 Completing the table

In some constructions, we use the following definition. Let G be a k-copwin graph. We say a node u of G (or $\mathcal{G}$) is *critical* if k cops have a k-copwin strategy such that if the robber ever moves to u, it gets caught on the *next* turn. Observe that if G is *vertex-transitive* and contains a critical node, then every node of G is critical because, by definition of vertex-transitive, for every node v there exists an automorphism ϕ_v of G such that $\phi_v(u) = v$.

The examples presented in this section are varied and do not use ideas presented in the previous sections. Some, like Lemma 5.20, simply serve to fill Table 5.1. Others, like Theorem 5.25, Proposition 5.26 and Lemma 5.27 are interesting in their own rights for how they present more ways to produce periodic graphs with nontrivial combinations of cop numbers, for example by using retracts.

Lemma 5.20 ((a, a, a)-copwin periodic graphs)**.** *For every integer $a \geq 1$, there exists an always connected (a, a, a)-copwin periodic graph.*



(a) G_0 (b) G_1

(c) G_2 (d) G_3

Figure 5.15: The periodic graph used in Lemma 5.19. Each snapshot is a cycle of length 11 and the periodic graph is 3-copwin.

Proof. Let G be a connected a-copwin graph. The periodic graph $\mathcal{G} = (G)^*$ is such an example. □

Lemma 5.21 ($(1,2,1)$-copwin)**.** *There exists an always connected periodic graph that is* $(1,2,1)$*-copwin.*

Proof. The periodic graph $\mathcal{G} = (K_6, C_6)^*$ is copwin since the cop wins in a single move. The footprint is K_6, which is copwin, while $G_{\max} = C_6$ that is 2-copwin. □

Lemma 5.22 ($(1,3,1)$-copwin)**.** *There exists an* (a,b,c)*-copwin periodic graph with* $\max(a,c)+2 = b$ *for some positive integers* a, b, c*.*

Proof. Let P_e be the Petersen graph. The periodic graph $\mathcal{G} = (K_{10}, P_e)^*$ satisfies the statement. □

Lemma 5.23 ((2,1,1)-copwin)**.** *There exists an always connected periodic graph with* 2*-copwin footprint and copwin snapshots that is copwin.*

Proof. Let G be a four-cycle (a,b,c,d), $G_0 = G_1 = G_2$ be the path (a,b,c,d) and G_3 the path (d,a,b,c). The footprint of $\mathcal{G} = (G_0, G_1, G_2, G_3)^*$ is G, so it

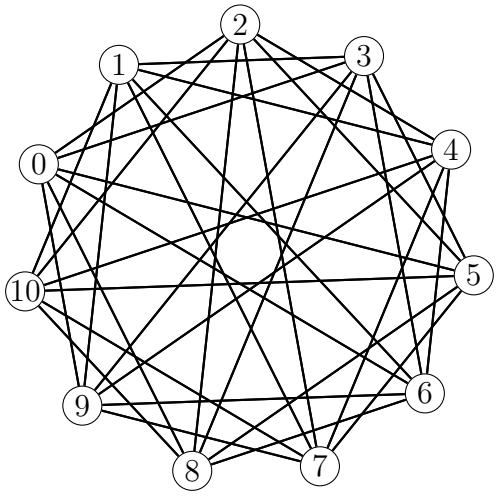


Figure 5.16: The footprint of the periodic graph used in Lemma 5.19 is 2-copwin

has cop number two, while every snapshot has cop number one. Furthermore, $\mathcal{G}$ is copwin. The cop starts on b in G_0, forcing the robber to start on d. The cop moves to c, while the robber stays on d. In G_1, the cop makes the catch. □

Lemma 5.24 ((2,1,2)-copwin)**.** *There exists an always connected periodic graph, with* 2*-copwin footprint and all copwin snapshots, that is* 2*-copwin.*

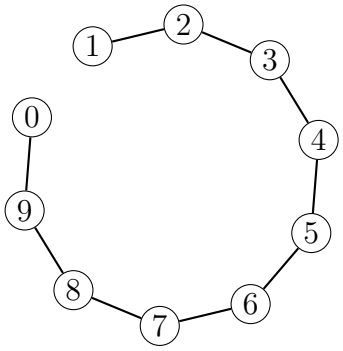
Proof. Let $G = C_{10}$, and $\mathcal{G}$ be the periodic graph shown in Figure 5.17. By construction, G is 2-copwin and both snapshots are copwin. However, we show that $\mathcal{G}$ cannot be copwin.

Suppose a player stubbornly walks along the cycle of G in $\mathcal{G}$. We say that player is *blocked* if it either tries to move along the edge 01 in G_0 or 56 in G_1. One can show that under any of the following conditions that player is never blocked:

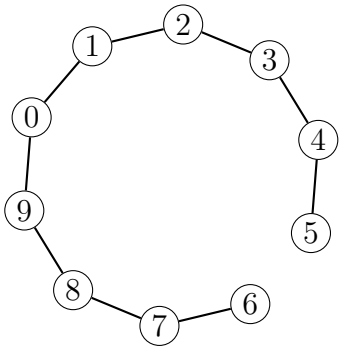
1. If it starts in G_0 on an odd-numbered node and walks along nodes with increasing labels;

2. If it starts in G_0 on an even-numbered node and walks along nodes with decreasing labels;

3. If it starts in G_1 on an even-numbered node and walks along nodes with increasing labels;



(a) G_0



(b) G_1

Figure 5.17: The periodic graph used in Lemma 5.24. Each snapshot is a cycle of length 10 and the periodic graph is 2-copwin.

4. If it starts in G_1 on an odd-numbered node and walks along nodes with decreasing labels;

This follows from the facts that $\mathcal{G}$ has period 2, G is an even cycle and that $x \in V(G)$ has the same parity as both $x + 2 \pmod{10}$ and $x - 2 \pmod{10}$. Furthermore, the player can be blocked at most once.

Therefore, here is a winning robber strategy against a single cop. Let the cop start on some node x. The robber starts on some node y with the same parity as x and such that $d_G(x, y) \geq 3$. The robber always moves in the same direction as the cop. The robber will be blocked at most once, so its distance to the cop cannot go lower than two. Thus, the cop cannot catch the robber and $\mathcal{G}$ cannot be copwin.

□

Theorem 5.25 ($(3, 1, 2)$-copwin)**.** *There exists a periodic graph $\mathcal{G}$ with $1 = c(G_{\max}) < c(\mathcal{G}) < c(G) = 3$.*

Proof. Let H be the graph in Figure 5.6 on Page 80. Let G be given by identifying any node of the Petersen graph P_e with node 2 of H. Let $\mathcal{G}$ be as follows. The schedule is chosen so that $\mathcal{G}[H] = \mathcal{H}$ where $\mathcal{H}$ is the periodic graph of Theorem 5.6 and Figure 5.5. Since this has period 9 and P_e has diameter two, we let a spanning tree of P_e with diameter at most four (by Lemma 5.9) appear for the first four snapshots, then cover the remaining edges with different spanning trees in the remaining fives snaphots.

Since $\mathcal{G}[H]$ is 2-copwin and 2 is a cutvertex of G, $c(\mathcal{G}) \geq 2$. Indeed, the robber can survive against a single cop by forever playing on $\mathcal{G}[H]$. Moreover, every snapshot is a tree, so $c(G_{\max}) = 1$. Finally, every node of P_e is critical, so a robber on $G[P_e]$ cannot escape from 3 cops by moving to $G[H]$ and $c(G) = c(P_e) = 3$.

Let us show that $c(\mathcal{G}) \leq 2$. Let one cop start on 2 and another on node 4 of H. The cop on 2 can capture the robber on $\mathcal{G}[P_e]$ because the first four snapshots of $\mathcal{G}[P_e]$ contain a spanning tree of diameter at most four. The robber cannot move to $\mathcal{G}[H]$ because the first cop moves along cutvertices of G. Thus, the robber must start on $\mathcal{G}[H]$. But, in Theorem 5.6, we also showed that if a cop starts on 2 and another starts on 4, the cops can capture the robber. Furthermore, in the winning strategy we described, the robber never gets to visit node 2. Thus, if the robber tries to move to $\mathcal{G}[P_e]$, through node 2, it gets captured by the cop on this node. Therefore, $c(\mathcal{G}) \leq 2$ which implies that $c(\mathcal{G}) = 2$. □

The previous construction also leads to a $(3, 2, 3)$-copwin periodic graph.

Proposition 5.26 ($(3, 2, 3)$-copwin)**.** *There exists a periodic graph $\mathcal{G}$ with $3 = c(G) = c(\mathcal{G}) > c(G_{\max}) = 2$.*

Proof. Let H be the graph in Figure 5.16. Let the Petersen graph P_e be labelled as in Figure 5.18 and let G be obtained by identifying node a of P_e and node 0 of H. Let also $\mathcal{H}$ be the periodic graph in Lemma 5.19 with footprint H and $\mathcal{K}$ be the periodic graph of Corollary 5.15 with footprint P_e. Then, let $\mathcal{G}$ be such that $\mathcal{G}[H] = \mathcal{H}$ and $\mathcal{G}[P_e] = \mathcal{K}$. The period of $\mathcal{K}$ is left undefined in Corollary 5.15 while the period of $\mathcal{H}$ can be any odd integer greater than 4. Thus, we can match the periods of both periodic graphs by adding snapshots to either or both of them.

By construction, $c(G) = 3$. Indeed, let three cops be on $0, 4, 6$. If the robber is on H, it gets captured by the cops on 0 and 4 because $N_H[0] \cup N_H[4] = V(H)$. Otherwise, the robber must be on P_e. Then, let the cops on 4 and 6 walk to nodes i and b of P_e while the other cop waits on $a = 0$. The robber cannot escape from P_e because a is a cutvertex. When this is done, the robber is only safe from immediate capture if it is on j or h. Let the cops on a and b move to e and c while the other cop waits on i. Then, $N_{P_e}[j] \cup N_{P_e}[h] = \{c, e, f, g, h, j\} \subseteq N_{P_e}[e] \cup N_{P_e}[c] \cup N_{P_e}[i] = \{a, e, d, j\} \cup \{b, c, d, h\} \cup \{d, f, g, i\}$. Therefore, no matter where the robber is, it gets captured on the next move

of the cops. Observe that in this case, the robber can never move to a and b once the cops there started moving.

Let us show that $c(G_{\max}) = 2$. Clearly, $c(G_{\max}) > 1$, so let us show that two cops can win. Let G_t be any snapshot. By construction, $G_t = P_e \setminus \{cx\}$ for some $x \in \{b, d, h\}$. Without loss of generality, $dc \notin E(G_t)$. Recall that a joins P_e and H. Here is a winning strategy for two cops on G_t. The cops start on b and h, so the robber must start on either e, d or i. Without loss of generality, let the robber start on e as the other cases are similar. The cops move to b and j, so the robber must move to d. The cops then move to g and j, so the robber stays on d since $dc \notin E(G_t)$. Finally, the cops move to e and i and the robber gets captured on the next turn. Notice that at no point the robber had access to a. By symmetry, the same reasoning holds when $bc \notin E(G_t)$ and $hc \notin E(G_t)$, which both eventually happen in $\mathcal{G}$. The same reasoning holds even if the robber decides to start on a, when for example $bc \notin E(G_t)$. In this case, the robber can escape to $G_t[H]$. But, $G_t[H]$ is an 11-cycle and the two cops will move to a without the robber being able to move back to $G_t[P_e]$. Thus, the cops make the catch on H in this case.

Finally, $c(\mathcal{G}) \geq \max(c(\mathcal{G}[H]), c(\mathcal{G}[P_e])) = 3$ since the robber can forever play in $\mathcal{G}[H]$ and $c(\mathcal{G}[H]) = c(\mathcal{H}) = 3$. Hence, let us show that $c(\mathcal{G}) \leq 3$.

Let 3 cops start on $0, 3, 6$. If the robber is in $\mathcal{G}[P_e]$, the two cops on 3 and 6 move to make the catch in $\mathcal{G}[P_e]$. Otherwise, the robber is in $\mathcal{G}[H]$. The cops on $0, 3, 6$ can apply their winning strategy described in Lemma 5.19. From the proof of Lemma 5.19, the robber cannot move to 0 without being captured, so it cannot escape from $\mathcal{G}[H]$. Thus, the three cops make the catch on $\mathcal{G}[H]$ and $c(\mathcal{G}) \leq 3$. Therefore, $c(\mathcal{G}) = 3$. □

The periodic graph in Proposition 5.26 shows an example where we can divide the footprint into two "blocks". We will see later, in Chapter 6, what we mean specifically by this.

Lemma 5.27 ($(3, 3, 1)$-copwin)**.** *There exists a periodic graph $\mathcal{G}$ with $1 = c(\mathcal{G}) < c(G_{\max}) = c(G) = 3$.*

Proof. Let H and H' be two copies of the Petersen graph and G be obtained by identifying any node $u \in V(H)$ with any node $v \in V(H')$. Since the Petersen graph has diameter two, H and H' contain spanning trees T and T' with diameter at most four each, by Lemma 5.9. We let $\mathcal{G}$ be such that the first four snaphots contain $H \cup T'$, while the last four snapshots contain $T \cup H'$. Since the Petersen graph is a retract of G and every snapshot,

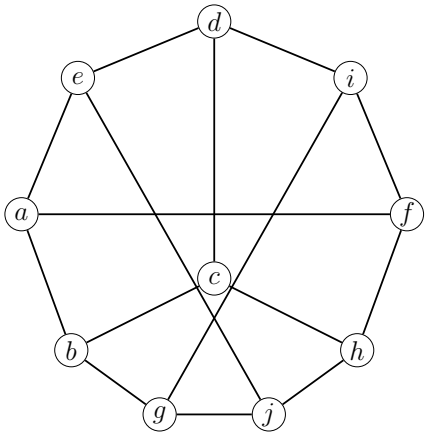


Figure 5.18: A labelled Petersen graph used in Proposition 5.26

$3 \leq c(G)$ and $3 \leq c(G_{\max})$ by [27]'s Theorem. Moreover, let 3 cops play on $H \cup T'$ such that if the robber moves to T', the cops act as if the robber moves to u instead: this is 3-copwin. The same holds for $T \cup H'$, so $c(G_{\max}) \leq 3$. On another hand, every vertex of the Petersen graph is critical, so 3 cops can apply their 3-copwin strategy on either H or H' (whichever copy contains the robber) and prevent the robber from escaping to the other copy. Then, $c(G) = c(G_{\max}) = 3$. Nevertheless, on $\mathcal{G}$ the robber is eventually stuck on a tree that appears for long enough for a single cop to make the catch. Therefore, $c(\mathcal{G}) = 1$. □

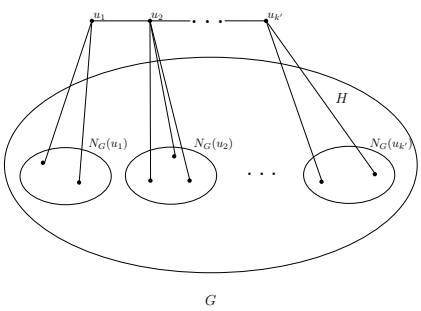
5.4 General construction

It follows from Lemma 5.8 that every example showed in Table 5.1 can be extended to any number of vertices.

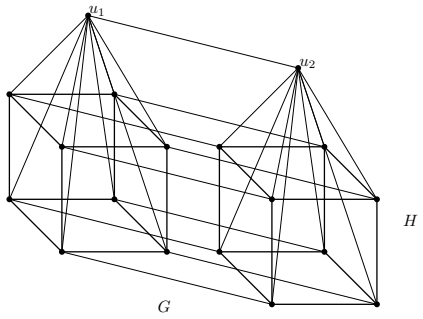
Theorem 5.7 shows one way to construct a copwin periodic graph whose footprint is a 2-copwin graph. We can generalize this idea and construct an at most k'-copwin periodic graph whose footprint is k-copwin for any integers $1 \leq k' \leq k$ with similar ideas, when the period is not bounded.

Let us focus on the opposite question: can we build a k-copwin periodic graph whose footprint is k'-copwin for every integers $1 \leq k' \leq k$?

We saw in Theorem 5.6 that it is possible to generate a $(1, 1, 2)$-copwin periodic graph by considering a sequence of paths such that no snapshot contains a temporal corner. This begs the question of whether this idea can



(a) Sketch of the construction used in the proof of Proposition 5.28, where H is a k-copwin graph and $\{u_1, \ldots, u_{k'}\}$ is a dominating set in G



(b) Example of a footprint G used in the proof of Proposition 5.28, where H is a hypercube Q_4, with cop number $k = 3$, and $P_2 = (u_1, u_2)$

Figure 5.19: General sketch of the construction used in Proposition 5.28 along with an example on the hypercube Q_4 with $k' = 2$

be extended further, to construct, for any $k \geq 1$, a $(k, k, k{+}1)$-copwin periodic graph from a sequence of k-copwin snapshots that contains no k-temporal corners.

In Proposition 3.10 we used the Petersen graph that is 3-copwin and does not contain any 2-corner. We use this in the next result to construct examples with large gaps between the cop number of the footprint and the cop number of the periodic graph. We begin the proof of Proposition 5.28 by initializing a k-copwin graph, given an integer k. This is always possible. One could take for example the family of hypercubes, by a result of Maamoun and Meyniel [153], however Proposition 5.28 is not restricted to this family of graphs.

Proposition 5.28. *For every integers $1 \leq k' < k$, there exists a periodic graph $\mathcal{G}$ such that $k' = c(G) < c(G_{\max}) = c(\mathcal{G}) = k$.*

Proof. Let H be any k-copwin graph and $P_{k'} = (u_1, \ldots, u_{k'})$ be a path with k' vertices. Let G be obtained by letting $\{u_1, \ldots, u_{k'}\}$ be a dominating set of H such that the neighbourhoods of each u_is are pairwise disjoint. Figure 5.19 shows a general sketch of G as well as an example.

Let $Q = (v_1, \ldots, v_m)$ be a walk of H that visits all vertices and starts and ends on the same node. We assume that $m = xk'$ for some $x \in \mathbb{N}$, which we can achieve by using self-loops multiple times. Let $E_0 = (u_1v_1, u_2v_2, \ldots, u_{k'}v_{k'}, u_1v_{k'+1}, u_2v_{k'+2}, \ldots, u_{k'}v_{xk'})$ be a sequence of edges.

Let $\mathcal{G}$ be the following periodic graph. The graph H along with the path $P_{k'}$ are stable in $\mathcal{G}$. For every time t, G_t contains the t^{th} edge of E_0, until all edges have been used.

By construction, $c(G) \leq \gamma(G) = k'$. Also, H is k-copwin and $k > k'$, so less than k' cops cannot capture the robber on $G[H]$. Thus, $c(G) \geq k'$ and $c(G) = k'$.

Let us show that $c(G_{\max}) = k$. In every snapshot G_t there is a unique node u_i such that u_i is connected to a node x of H in G_t. Thus, the map $h_t : G_t \to H$ that sends $\{u_1, \ldots, u_{k'}\}$ to x is a homomorphism, hence a retraction. That is, H is a retract of every snapshot G_t. It follows by [27] that $k = c(H) \leq c(G_t)$ for every time t. By the same argument, $c(G_t) \leq k$: the k cops in G_t play on H so that if the robber moves to P, the cops act as if the robber moved to the unique node u_i connected to H. Then, either the robber is captured on H, or one cop is on u_i when the robber is alive on P. Since P is a path, this cop makes the catch. Therefore, $c(G_t) = k$.

Finally, let us show that $c(\mathcal{G}) = k$. Since Q is a walk of H, the retractions in $\{h_t : G_t \to H \,|\, 0 \leq t \leq p-1\}$ map journeys on G to journeys on H. Therefore, the images of the cops under those retractions perform valid moves in the cops and robber game on $\mathcal{G}[H]$. But, H is k-copwin and stable in $\mathcal{G}$, so $c(\mathcal{G}) = c(H) = k$.

□

Aside from the family of hypercubes, a classic family of graphs with unbounded cop number is the family $\mathbb{G}_q$ of incidence graphs of the projective plane $\mathrm{PG}(2, q)$ of order q, for any prime power q. For any $G \in \mathbb{G}_q$, $c(G) = q+1$ by Pralat [171] (see also [38]). Bonato and Burgess [35] present more families of graphs with unbounded cop numbers. Thus, by choosing $k' = 1$ and H in a family of graphs with unbounded cop number that respects Meyniel's conjecture in Proposition 5.28, we can produce an infinite class of periodic graphs $\mathbb{G}_{\circledcirc}$ such that for every $\mathcal{G} \in \mathbb{G}_{\circledcirc}$, $c(G) = 1$ and $c(\mathcal{G}) \in O(\sqrt{n})$.

Chapter 6

Bounds on the cop number of periodic graph families

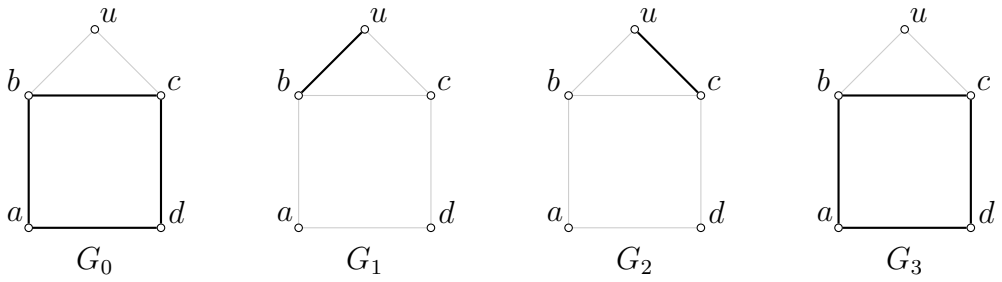
Some results in this chapter were presented at the *54th Southeastern International Conference on Combinatorics, Graph Theory & Computing*. A preprint of this text can be found here: [64].

This chapter is in direct continuation with Chapter 5. In the previous chapter, we stressed that the different cop numbers of the constituents of a periodic graph $\mathcal{G} = (G_0, \ldots, G_{p-1})^*$ are unrelated in general. In this chapter, we take a more constructive approach to find upper and lower bound on $c(\mathcal{G})$ that often use properties of G.

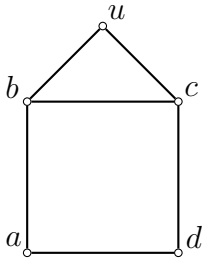
6.1 Retracts

An important tool in the study of Cops and Robbers games is the concept of retract, that we explained in Section 3.3. Retracts are used to show that copwin graphs are dismantlable (see for example [38]), a tighter upper bound on $c(G)$ with block decompositions [113], even to help show that the Petersen graph is the smallest 3-copwin graph [17]. One important result on retracts is a theorem of Berarducci and Intrigila [27] stating that $c(H) \leq c(G)$ when H is a retract of G. In other words, upper bounds on $c(G)$ are carried over when taking retracts. Unfortunately, this is not true in general in periodic graphs.

Proposition 6.1. *There exists a periodic graph $\mathcal{G}$ with footprint G and a retract H of G such that $c(\mathcal{G}[H]) > c(\mathcal{G})$.*



(a) $\mathcal{G} = (G_0, G_1, G_2, G_3)^*$



(b) G

Figure 6.1: $c(\mathcal{G}[\{a, b, c, d\}]) > c(\mathcal{G})$

Proof. Let G be the graph drawn in Figure 6.1(b). Then, $H = C_4$ is a retract of G obtained by mapping u to either b or c. Let us construct a periodic graph $\mathcal{G}$ on G such that $c(\mathcal{G}) = 1$ while $c(\mathcal{G}[H]) = 2$.

Consider the periodic graph $\mathcal{G}$ shown in Figure 6.1(a). Let the cop start on a in G_0. The robber gets captured in one move if it starts on a, b or d, so let it start on either u or c. Let the cop move along the journey $((a, b), (b, u), (u, c), (c, d))$, crossing the first edge in G_0 and the last one in G_3. The cop captures the robber no matter where the latter goes. Thus, $\mathcal{G}$ is copwin. However, $\mathcal{G}[H]$ contains two identical 4-cycles, so $\mathcal{G}[H]$ is 2-copwin. Therefore, $c(\mathcal{G}) = 1 < c(\mathcal{G}[H]) = 2$. □

With one extra assumption, we can recover Berarducci and Intrigila's original result.

Theorem 6.2. *Let $\mathcal{G}$ be a periodic graph and $h : V \to V'$ be a retraction of every snapshot, where $V' \subseteq V$. Then*

$$c(\mathcal{G}[V']) \leq c(\mathcal{G}).$$

Proof. For every time t, we write $H_t = h(G_t)$. For any edge $xy \in E(G_t)$, $h(x)h(y) \in E(H_t)$ because h is a retraction on the snapshots.

Let $k = c(\mathcal{G})$ and σ_c be any cops strategy on $\mathcal{G}$ for k cops. Let C be a set of nodes occupied by the cops at any time t and C_i the position of the i^{th} cop. If $\sigma_c((t, C), (t, r)) = (t + 1, C')$ in $\mathcal{G}$, then $C_iC'_i \in E(G_t)$ for every cop $1 \leq i \leq k$. By the above argument, $h(C_i)h(C'_i) \in E(H_t)$. Thus, every cops strategy σ_c on $\mathcal{G}$ has a corresponding strategy σ_c^h on $\mathcal{G}[H]$ such that if $\sigma_c((t, C), (t, r)) = (t + 1, C')$, then σ_c^h moves the i^{th} cop from $h(C_i)$ to $h(C'_i)$ for every $1 \leq i \leq k$.

Thus, let the cops play a k-copwin strategy σ_c on $\mathcal{G}$ while the robber is restricted to play on $\mathcal{G}[H]$. Since $\mathcal{G}$ is k-copwin, the cops eventually move to a position $(t+1, C_1), \ldots, (t+1, C_k)$ while the robber is on (t, r) such that no matter where the robber moves to it gets captured on the next turn. That is, for every $x \in N_t[r, \mathcal{G}[H]]$, there exists $xC_i \in E(G_{t+1})$. Then, by the above argument, $h(x)h(C_i) \in E(H_t)$. Thus, $C((t + 1, h(C_1), \ldots, h(C_k)), (t, r))$ is a winning configuration for the cops in $\mathcal{G}[H]$. As we argued, this configuration can be reached when the cops follow the strategy that corresponds to σ_c in $\mathcal{G}[H]$. Therefore, the robber gets captured by the k cops in $\mathcal{G}[H]$.

Thus, $c(\mathcal{G}[H]) \leq c(\mathcal{G})$. □

We did not assume that h is a retraction of G in Theorem 6.2 because we can show this is implied from h being a retraction of all the snapshots.

Consider again the arena in Figure 6.1. Let us explain why Theorem 6.2 does not apply in this case. Let $h : \{a, b, c, d, u\} \to \{a, b, c, d\}$ be any retraction. The node u can only be mapped to either b or c. Suppose that $h(u) = b$. Then, $uc \in E(G_2)$ implies that $h(u)h(c) = bc \in E(H_2)$, which is not the case. Similarly, if $h(u) = c$, then $ub \in E(G_1)$ implies that $h(u)h(b) = cb \in E(H_1)$, which is also not the case. Thus, h cannot be a retraction of every snapshot.

We explicit an important consequence of this theorem that we will use later. Its proof is straightforward.

Corollary 6.3. *Let $\mathcal{G}$ be a periodic graph and $h : V \to V'$ a retraction of all the snapshots, where $V' \subseteq V$. Then, the cops on $\mathcal{G}[V']$ can capture the image by h of the robber on $\mathcal{G}[V']$.*

6.2 Tree decompositions

Joret et al. [129] proved that $c(G) \leq \mathrm{tw}(G)/2+1$ for every connected graph[1] G. We prove the following.

Theorem 6.4. *For every connected graph G, $c_{\circlearrowleft}(G) \leq \mathrm{tw}(G)+1$.*

Proof. Let G have treewidth k and $(\mathcal{T}, \mathcal{B})$ be a minimal tree decomposition of G that is *smooth*[2], where $\mathcal{B} = \{X_0, \ldots, X_l\}$. Recall that for any $XY \in E(\mathcal{T})$, $X \cap Y$ is a cutset of G. Also, we write $\mathcal{T}_{X,Y}$ for the subtree of $\mathcal{T} \setminus \{X\}$ that contains Y.

Let $\mathcal{G} = (G_0, \ldots, G_{p-1})^*$ be any periodic graph with footprint G. For any bag X of $\mathcal{T}$, $k+1$ cops can guard X so that if the robber moves into X at any time it gets captured immediately. Thus, let $k+1$ cops start on the $k+1$ nodes of any bag X_0 in G_0. Let the robber start on some node $r_0 \in V$.

Let X_1 be the unique neighbour of X_0 in $\mathcal{T}$ such that r_0 is in a bag of $\mathcal{T}_{X_0,X_1}$. Because $\mathcal{T}$ is smooth, $|X_0 \cap X_1| = k$ and there exists a unique node $x_1 \in X_1 \setminus X_0$ and a unique node $x_0 \in X_0 \setminus X_1$. Recall that $\mathcal{G}$ is temporally connected, so let the cop on x_0 walk to x_1, crossing edges whenever possible. Meanwhile, all other cops stay still. In order to escape from $\mathcal{T}_{X_0,X_1}$, the robber has to move trough a node of $X_0 \cap X_1$ because this is a cutset of G. But, the nodes in this set will all be occupied while the travelling cop moves to x_1. Therefore, once this cop arrives on x_1, the robber is still in $\mathcal{T}_{X_0,X_1}$. Furthermore, at that time all nodes of X_1 are occupied.

When this happens, the robber's territory is reduced. It follows by successively applying this reasoning that $c(\mathcal{G}) \leq k+1$. □

We call the cop strategy outlined in Theorem 6.4 a *smooth strategy* since it moves the cops on a smooth tree decomposition.

Observe that Theorem 6.4 provides a proof that every periodic graph with outerplanar footprint has cop number at most three since outerplanar graphs have treewidth at most two. Also, compare this with the bound shown in

[1] Specifically, Joret et al. [129] defined $c(G)$ as the maximum cop number of the connected components of G, so that $c(G) \leq \mathrm{tw}(G)/2+1$ without requiring that G is connected. In the present work, $c(G)$ is the sum of the cop numbers of all connected components of G, otherwise there are not enough cops to capture the robber.

[2] A tree decomposition $(\mathcal{T}, \mathcal{B})$ of G with width k is *smooth* if every bag $X \in \mathcal{B}$ has size $k+1$ and for any two adjacent bags $X_i, X_j \in \mathcal{B}$, $|X_i \cap X_j| = k$. Any tree decomposition of G can be transformed into a smooth tree decomposition with the same width. Smooth tree decompositions are similar to *normalized* tree decompositions ([112]).

[129] that was refined in [36]. Joret et al.'s strategy of letting the cops guard shortest paths, which gives the $\frac{1}{2}$ factor to their bound, is not applicable here because of the temporal nature of $\mathcal{G}$. Since the graph varies, the only safe way for a cop to guard a node seems to be to stand on it.

In Lemma 2.5 of [33], Bodlaender shows that if G has treewidth k and $(\mathcal{T}, \mathcal{B})$ is a smooth tree decomposition of G, then $\mathcal{T}$ has $n - k$ nodes. In Theorem 6.4, a team of cops performs a stubborn walk over a smooth tree decomposition of the footprint and must go through at most $\left\lceil \frac{n-k}{2} \right\rceil$ nodes of the decomposition. We deduce Corollary 6.6 that uses the *temporal diameter*, that we define here and reuse in Section 6.5.

Definition 6.5 (Foremost distance and temporal diameter)**.** *Let $\mathcal{G}$ be a periodic graph with footprint G. A journey from (t, u) to (t', v) in $\mathcal{G}$ is* foremost *if $\mathcal{G}$ contains no other journey from (t, u) to (t'', v) such that $t \leq t'' < t'$. We also say this journey is foremost from (t, u) to v. We define the foremost distance $\delta_t(u, v)$ from (t, u) to v as ∞ if u and v are in distinct connected components of G, 0 if $u = v$, and $t' - t + 1$ if $u \neq v$ and there is a journey from (t, u) to (t', v) that is foremost.*

The temporal diameter *d_f of $\mathcal{G}$ is the largest foremost distance from any temporal node to any node.*

Observe that it is possible that $\delta_t(u, v) > p$ for some t, u, v in a periodic graph $\mathcal{G}$, for example if only one edge of every foremost journey from (t, u) to v can be crossed in p consecutive snapshots.

When $0 < \delta_t(u, v) < \infty$, $\delta_t(u, v)$ also corresponds to the length of a foremost journey from u to v that starts at time t in $\mathcal{D}$.

Corollary 6.6. *Let $\mathcal{G}$ be a periodic graph with footprint G. If $\mathcal{G}$ has temporal diameter d_f, then a team of $\mathrm{tw}(G)+1$ cops can capture the robber in at most $d_f \left\lceil \frac{n-\mathrm{tw}(G)}{2} \right\rceil$ turns.*

Proof. From Theorem 6.4, $c(\mathcal{G}) \leq \mathrm{tw}(G) + 1$. Consider the smooth strategy outlined Theorem 6.4. Since $\mathcal{G}$ has temporal diameter d_f, in every sequence of d_f snapshots of $\mathcal{G}$ the cop on x_0 can move to x_1. Furthermore, we can choose X_0 as the centre of the tree, so the cop team would have to move along at most $\left\lceil \frac{n-\mathrm{tw}(G)}{2} \right\rceil$ bags. Thus, the capture time of the robber is at most $d_f \left\lceil \frac{n-\mathrm{tw}(G)}{2} \right\rceil$. □

In general, we do not have that $c(\mathcal{G}) \leq \max_{0 \leq i \leq p-1} \mathrm{tw}(G_i)$, as one can attest from our $(1, 1, 2)$-copwin periodic graph presented in Theorem 5.6.

The next result shows that, in general, it is unlikely that less than $\mathrm{tw}(G)+1$ cops can capture the robber on every periodic graph $\mathcal{G}$ with footprint G, even if we assume $\mathcal{G}$ to be always connected. Given a tree decomposition $(\mathcal{T}, \mathcal{B})$ of a graph G, we say the cops *guard* a bag $X \in \mathcal{B}$ if they occupy some nodes of X in such a way that if the robber tries to move on X it gets captured by the cops on the next turn.

Lemma 6.7. *There exists an always connected periodic graph $\mathcal{G}$ with footprint G and an optimal tree decomposition $(\mathcal{T}, \mathcal{B})$ of G such that $\mathrm{tw}(G)$ cops cannot always guard every bag of $\mathcal{B}$.*

Proof. Let $\mathcal{G} = (G_0, G_1)^*$ be the periodic graph shown in Figure 6.2 with footprint $G = P_e$ the Petersen graph. Figure 6.2 also shows an optimal tree decomposition $(\mathcal{T}, \mathcal{B})$ of P_e. Note that $\mathrm{tw}(P_e) = 4$. We can also show that $c(\mathcal{G}) \leq 3$ by placing 3 cops on a, h, i in G_1, so that in this snapshot the robber must be on either b or c to avoid direct capture. The cops make the catch in G_0.

Suppose the robber is on i in G_0, while 4 cops are on a, c, h and j in G_1. The cops are in the bag $X := \{a, c, j, i, h\}$ of $\mathcal{B}$, while the robber can be in either bag $\{a, c, d, i, j\}$ or $\{a, c, g, i, j\}$ of $\mathcal{B}$. Without loss of generality, suppose the robber is in $Y := \{a, c, d, i, j\}$. Then, the robber can move to g in G_0. No matter where the cops move, the robber cannot be captured in G_1. The bags that contain g are in another branch of $\mathcal{T} \setminus \{X\}$ that the one that contains Y. Therefore, the cops were not able to guard the bag X. □

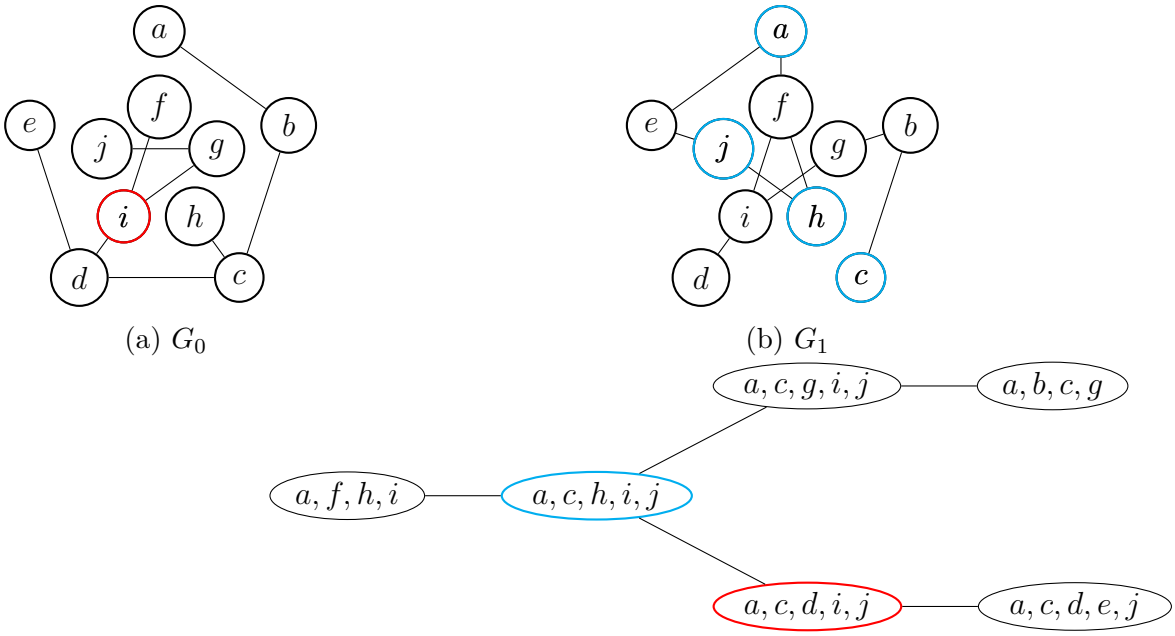
Consider again Figure 6.2. In the proof of Theorem 6.4, there would be a cop on node i, so the robber could not move across the edge ig. From Lemma 6.7, it follows that a cop strategy that would use less than $\mathrm{tw}(G)+1$ cops could not be stubborn.

We say a tree decomposition $(\mathcal{T}, \mathcal{B})$ of a graph G is *temporally regular* in a periodic graph $\mathcal{G}$ if for every bag $X \in \mathcal{B}$ and time t, $E(X) \cap E(G_t) \neq \emptyset \implies E(X) \subseteq E(G_t)$.

Lemma 6.8. *If $\mathcal{G}$ admits an optimal tree decomposition $(\mathcal{T}, \mathcal{B})$ that is temporally regular, then $c(\mathcal{G}) \leq \left\lceil \frac{\mathrm{tw}(G)}{2} \right\rceil + 1$.*

Furthermore, if G is also chordal, then $c(\mathcal{G}) = 1$.

Proof. Let $(\mathcal{T}, \mathcal{B})$ be a temporally regular optimal tree decomposition of G. Since the tree decomposition is temporally regular, for any bag $X \in \mathcal{B}$, either



(a) G_0

(b) G_1

(c) A tree decomposition of the Petersen graph

Figure 6.2: Petersen tree-decomposition, from [117]

the cops can apply the strategy described by Joret et al. [129] in X or no player can move in X. Thus, while the cops move in X to reach a neighbour Y of X in $\mathcal{T}$, the robber cannot move to another branch of $\mathcal{T} \setminus \{X\}$ than the one it is currently in. Thus, $\mathrm{tw}(G)/2$ cops can guard a bag while one more cop moves toward the robber.

Furthermore, recall that G is chordal if and only if it admits a tree decomposition where each bag induces a clique (see for example [71]). When every bag of $\mathcal{B}$ induces a clique, a single cop can guard any bag by itself. Furthermore, this same cop can progress toward the robber while guarding its bag, since the bags are temporally regular. Thus, in this case one cop is sufficient to capture the robber. □

6.3 Block decompositions

Block decompositions can be used to show that the cop number of outerplanar graphs is at most two. They also seem to fit with the way we design winning strategies for $c_{\circledcirc}(G)$ so far, since they highlight the cutvertices of G, so we present them here. A block B of a graph G is a connected maximal induced subgraph of G without cutvertex. A block decomposition of G is a set $\{B_1, \ldots, B_m\}$ of blocks of G that contains all the nodes of the graph. The following result is originally due to Hill [113] for static graphs. We adapted it for periodic graphs. The original result uses retracts to guarantee a stronger bound, which we cannot do here. We write $\Delta(G)$ for the maximum degree of a graph G. The proof follows from similar arguments as those used by Hill.

Proposition 6.9. *Suppose $\{B_1, \ldots, B_m\}$ is a block decomposition of G that forms a tree T. Then,*

$$\max_{1 \leq i \leq m} c_{\circledcirc}(B_i) \leq c_{\circledcirc}(G) \leq \max_{1 \leq i \leq m} c_{\circledcirc}(B_i) + \Delta(T).$$

Proof. The lower bound is clear: if the robber forever plays in the block B_i of highest cop number $c_{\circledcirc}(B_i)$, then less than $c_{\circledcirc}(B_i)$ cops cannot capture it. Thus, let us show the upper bound.

Let the blocks be linearly ordered so that B_1 is a centre[3] of T and each path from B_1 to a leaf B_j is a sequence of blocks whose indices are strictly increasing. We write $c_1, \ldots, c_{m-1}$ for the cutvertices of G. By construction, if $B_i \in N_T(B_1)$, then $V(B_1 \cap B_i) = \{c_j\}$ for some c_j. Thus, there are at most $\Delta(T)$ cutvertices in B_1. Let $\Delta(T)$ cops start on those cutvertices. Then,

the robber is in a unique branch P of $T \setminus B_1$, from which it cannot escape because of the $\Delta(T)$ cops covering the cutvertices. Let B_i be the neighbour of B_1 on the branch P and c_j the cutvertex of $B_1 \cap B_i$. With the cop on c_j staying still, let the $\Delta(T) - 1$ cops *stubbornly* move from their cutvertices to the cutvertices of B_i. While this is happening, the robber cannot move from P to another branch of $T \setminus B_1$. Therefore, once the cops have finished moving, the robber is either in B_i or in the unique branch of $P \setminus (B_1 \cup B_i)$. We can recursively apply the previous operation until the robber is surrounded by cops in a bag B_l. Then, the robber cannot escape from B_l and another team of $c_{\textcircled{\scriptsize R}}(B_l) \leq \max_{1 \leq i \leq m} c_{\textcircled{\scriptsize R}}(B_i)$ cops can make the catch on B_l. □

In the following corollaries, we add assumptions to lower the added term $\Delta(T)$ to 1. We use again the notion of critical vertex that we defined in Subsection 5.3.2.

Corollary 6.10. *Suppose $\{B_1, \ldots, B_m\}$ is a block decomposition of G that forms a tree T such that for every two adjacent blocks B_i, B_j in T, the unique node $u \in V(B_i \cap B_j)$ is a critical vertex in G. Then,*

$$\max_{1 \leq i \leq m} c_{\textcircled{\scriptsize R}}(B_i) \leq c_{\textcircled{\scriptsize R}}(G) \leq \max_{1 \leq i \leq m} c_{\textcircled{\scriptsize R}}(B_i) + 1.$$

Proof. This follows from Proposition 6.9 and the definition of critical vertex. Then, when the cop on a cutvertex c_i at the intersection of two adjacent blocks B_p and B_q moves toward the cutvertex c_j at the intersection of B_q and $B_r \in N_T(B_q)$, the robber cannot move back to c_i. Therefore, the robber eventually gets trapped on a block B_l that is a leaf of T and gets captured by $c_{\textcircled{\scriptsize R}}(B_l)$ cops. □

Corollary 6.11. *Suppose $\{B_1, \ldots, B_m\}$ is a block decomposition of G that forms a tree T. Furthermore, suppose there is a set of retractions $\{h_i : G \to B_i \mid 1 \leq i \leq m\}$ such that for every time t and block B_i, $h_i : G_t \to B_i$ is a retraction. Then,*

$$\max_{1 \leq i \leq m} c_{\textcircled{\scriptsize R}}(B_i) \leq c_{\textcircled{\scriptsize R}}(G) \leq \max_{1 \leq i \leq m} c_{\textcircled{\scriptsize R}}(B_i) + 1.$$

Proof. This follows from Proposition 6.9 and Corollary 6.3. We apply the same reasoning as Hill's [113] to the cops on each B_i to capture the image of the robber on the cutvertices between each block. □

[3]The centre(s) of a tree T are the one or two nodes whose distances to every other nodes are minimal.

6.4 Comments on planar graphs

We suspect that for planar graphs we cannot do better than the upper bound of $O(\sqrt{n})$. The gap between cop numbers of periodic graphs with planar footprint we showed as examples and this bound is similar to the gap in the cop number of planar oriented graphs. That is, Loh and Oh [148] exposed a planar oriented graph that is 4-copwin and gave an upper bound of $O(\sqrt{n})$. Similarly, Gao and Yang [98] showed a planar graph G with cop number at least 4 in the lazy cops and robber game, but no upper bound. We show that Theorem 1.1. of Loh and Oh [148] also works on periodic graphs in Proposition 6.12. We expose the full argument here despite the fact that we already know that $c_{\circledcirc}(G) \in O(\sqrt{n})$ because $c_{\circledcirc}(G) \leq \mathrm{tw}(G) + 1 \in O(\sqrt{n})$ by Theorem 6.4 and the planarity of G. We think this argument is interesting by itself and it might lead to other interesting results. A similar idea was also used by Bose et al. [39].

Proposition 6.12. *For any planar or toroidal graph G, $c_{\circledcirc}(G) \in O(\sqrt{n})$.*

Proof. Suppose G is planar and let $\mathcal{G}$ be any periodic graph with footprint G. From Lipton and Tarjan [147], we know G has a separator S_1 of size $c_1\sqrt{n}$ between some sets of vertices A_1 and B_1 such that $|A_1|, |B_1| \leq 2n/3$. Let $c_1\sqrt{n}$ cops stubborn walk to S_1 in $\mathcal{G}$. This prevents the robber from ever moving from A_1 to B_1 in $\mathcal{G}$, or vice-versa. Without loss of generality, suppose the robber is in A_1. We apply this argument recursively, since planar graphs are closed under taking subgraphs. That is, $c_2\sqrt{2n/3}$ cops stubborn walk in $\mathcal{G}$ to a separator S_2 of $G[A_1]$ between sets of vertices A_2 and B_2 of size at most $4n/9$ each and so on. Let c be the maximum of all constants. Then, this requires at most

$$c\sum_{i=0}^{\infty}\sqrt{n\left(\frac{2}{3}\right)^i} = c\sqrt{n}\sum_{i=0}^{\infty}\sqrt{\left(\frac{2}{3}\right)^i} \in O\big(\sqrt{n}\big)$$

many cops, so $c(\mathcal{G}) \in O(\sqrt{n})$.

Suppose now that G is toroidal. Then, by Theorem 3.1. of Aleksandrov and Djidjev [9], G contains a separator S of size at most $\sqrt{12n}$ between any two sets A and B of size of at most $2n/3$. A subgraph of a toroidal graph is either toroidal or planar and the size of A and B are at most $2n/3$ in both cases. Thus, the same reasoning applies. □

The strategy highlighted in Theorem 6.4 is not optimal in general on planar graphs since $\mathrm{tw}(G) \in O(\sqrt{n})$ when G is planar, yet $c(G) \leq 3$ in that case. This holds because strategies on planar graphs rely on the result that isometric paths are "guardable" [6], in the sense that a single cop can move on this path so that if the robber hops on it, the robber gets captured on the next cop move. We can show that isometric paths are not guardable in general in periodic graphs, therefore preventing a wide array of known cop strategies in the static case to be adapted to the periodic case.

Lemma 6.13. *An isometric path in the footprint is* not *guardable in a periodic graph in general.*

Proof. Consider the grid $G = P_{10} \square P_3$. Let $\mathcal{G}$ be as follows. The column edges, those of P_3, always appear. Similarly, rows 0 and 2 always appear. Finally, the edges of row 1 appear one per snapshot, in order, starting with the first edge. The path induced by this row is an isometric path of G. However, one cop cannot prevent the robber from moving on it. □

One might object that the natural extension of isometric paths from static graphs to periodic graphs should consider the temporal dimension as well. Thus, let us consider *foremost journeys* (see Definition 6.5). We say a foremost journey J from (t, u) to (t', v) is *guardable* if a single cop can move on J such that if the robber moves to $V(J)$ between t and t' (included), the cop captures the robber.

Proposition 6.14. *A foremost journey is not guardable in periodic graphs in general.*

Proof. Let G be a 3×3 grid, with nodes labelled in the form (i, j), $0 \leq i, j \leq 2$. Let $\mathcal{G}$ be the periodic graph with footprint G such that:

$$
\begin{aligned}
E(G_0) &= \{((0,1),(1,1)),((0,2),(1,2))\} \\
E(G_1) &= \{((1,2),(2,2))\} \\
E(G_2) &= \{((2,2),(2,1))\} \\
E(G_3) &= \{((2,1),(2,2))\} \\
E(G_4) &= \{((1,1),(2,1))\} \\
E(G_5) &= E(G) \setminus \bigcup_{i=0}^{4} E(G_i).
\end{aligned}
$$

Let J be the journey that crosses the edges $((0,1),(1,1))$ in G_0 and $((1,1),(2,1))$ in G_4.

If the cop starts on $(0,1)$ while the robber starts on $(0,2)$, the robber can move to $(2,1)$ from $(2,2)$ and back to $(2,2)$ before the cop has been able to move to $(2,1)$. Thus, J is not guardable. Yet, J is the only foremost journey from $(0,1)$ to $(2,1)$. □

Therefore, we suspect that Proposition 6.12 is the best we can say about planar graphs since we showed in Lemma 6.13 and Proposition 6.14 that isometric paths and foremost journeys are not guardable in general in periodic graphs.

6.5 Periodic graphs of temporal diameter two

So far, we have showed upper and lower bounds on the periodic cop number of arbitrary graphs G. In this section and the next ones, we want to design specific families of graphs for which we can give stronger results on their periodic cop numbers. We start with periodic graphs of temporal diameter two.

In a periodic graph with temporal diameter d_f, every *node* is reachable in at most d_f steps from every other temporal node. A known class of graphs with bounded diameter that is studied in Cops and Robber games is the class of graphs with diameter two. Wagner [205] showed that the cop number of any graph with n nodes and diameter two is at most $\sqrt{2n}$, so they respect Meyniel's conjecture. In this section, we show that Wagner's bound also holds on periodic graphs.

Given a periodic graph $\mathcal{G}$, we write $\Delta(\mathcal{G})$, or simply Δ, its maximal degree. Recall that degrees in $\mathcal{G}$ do not account for self-loops.

Lemma 6.15. *If $\mathcal{G}$ has temporal diameter 2, then $c(\mathcal{G}) \leq \Delta + 1$.*

Proof. Let $\mathcal{G}$ have temporal diameter 2. Let $\Delta + 1$ cops start the game on the nodes $w_1, \ldots, w_{\Delta+1}$ in G_0 and the robber on x. Since $d_f = 2$, for every $v_j \in N_0[x, \mathcal{G}]$, v_j is reachable in G_2 from every node u by a journey that starts in G_0. Thus, there exists $z_1, \ldots, z_{\Delta+1}$ such that $N_0[x, \mathcal{G}] \subseteq \bigcup_{i=1}^{\Delta+1} N_1[z_i, \mathcal{G}]$, so let the cops move to $z_1, \ldots, z_{\Delta+1}$. By definition, $(0, x)$ is a $(\Delta + 1)$-temporal corner of $((1, z_1), \ldots, (1, z_{\Delta+1}))$ and the cops win on their next move.

It follows that $\Delta + 1$ cops are sufficient to catch the robber. □

Corollary 6.16 follows from Lemma 6.15 by constraining the robber's movements. We say the robber is forced to move on a periodic subgraph $\mathcal{H}$, with vertex set $V_{\mathcal{H}}$, of $\mathcal{G}$ if the rules of the game force the robber to always choose its next position in $V_{\mathcal{H}}$.

Corollary 6.16. *Suppose $\mathcal{G}$ has temporal diameter two and let $V' \subset V$. If the robber is forced to move on the periodic subgraph $\mathcal{G}[V']$ while the cops move on $\mathcal{G}$, then $\Delta(\mathcal{G}[V']) + 1$ cops can catch the robber.*

Proof. Let $\mathcal{G} = (G_0 \dots G_{p-1})^*$ have temporal diameter two and let the robber start on x in $G_0[V']$. By assumption, x has $l \leq \Delta(\mathcal{G}[V'])$ neighbours, so let $\{x, v_1, \dots, v_l\} = N_0[x, \mathcal{G}[V']]$. Let $k = \Delta(\mathcal{G}[V']) + 1$ cops be on $w_1, \dots, w_k$ in G_0. The cops move on $\mathcal{G}$. Therefore, since $\mathcal{G}[V'] \subseteq \mathcal{G}$ and $\mathcal{G}$ has temporal diameter two, the cops can follow their strategy from Lemma 6.15 and move to a set of nodes $\{z_1, \dots, z_k\}$ such that for each $v_j \in \{x, v_1, \dots, v_l\}$, $v_j \in N_1[z_i, \mathcal{G}]$ for at least one $z_i \in \{z_1, \dots, z_k\}$. The cops capture the robber at time 2. □

Let $\mathcal{G}$ be a periodic graph with n nodes. Clearly, we always have $c(\mathcal{G}) \leq n$. Given a periodic subgraph $\mathcal{H} \subseteq \mathcal{G}$, let $c_{\mathcal{G}}(\mathcal{H})$ be the cop number of $\mathcal{H}$ when the robber is forced to move on $\mathcal{H}$ while the cops move on $\mathcal{G}$. We write $|\mathcal{G}| = n$ and define $c_{\mathcal{G}}(m) := \max_{V' \subseteq V : |V'| = m} c_{\mathcal{G}}(\mathcal{G}[V'])$ for every $1 \leq m \leq n$. By definition, $c_{\mathcal{G}}(m)$ is the maximal cop number of any induced periodic subgraph $\mathcal{H}$ of $\mathcal{G}$ with m nodes when the robber only moves on $\mathcal{H}$ and the cops on $\mathcal{G}$. We always have $c_{\mathcal{G}}(m) \leq c_{\mathcal{G}}(n) = c(\mathcal{G})$.

We write the proof of Wagner in the context of periodic graphs and show that it still works. Recall that we assume that the footprints of our periodic graphs are connected and reflexive.

Theorem 6.17. *For every periodic graph with n nodes and temporal diameter 2, $c(\mathcal{G}) \leq \sqrt{2n} + 1$.*

Proof. Let $\mathcal{G}$ be a periodic graph with temporal diameter two. Let us prove by induction on $1 \leq m \leq n$ that the bound holds for every induced periodic subgraph of $\mathcal{G}$ of m nodes when $\mathcal{G}$ has temporal diameter two. Since $\mathcal{G}$ has temporal diameter two, it must be temporally connected. The results holds for $1 \leq m \leq 3$ because every periodic graph with $1 \leq m \leq 3$ nodes and temporal diameter 2 is either copwin or 2-copwin. This is clear for $m = 1$ and $m = 2$. For $m = 3$, one can verify this by observing that the footprint

of $\mathcal{G}$ is either a path P_3 or a cycle C_3. Suppose the result holds for every periodic subgraph of $\mathcal{G}$ with at most $k \geq 3$ nodes and let $\mathcal{H}$ be an induced periodic subgraph of $\mathcal{G}$ that is temporally connected with $m = k + 1$ nodes.

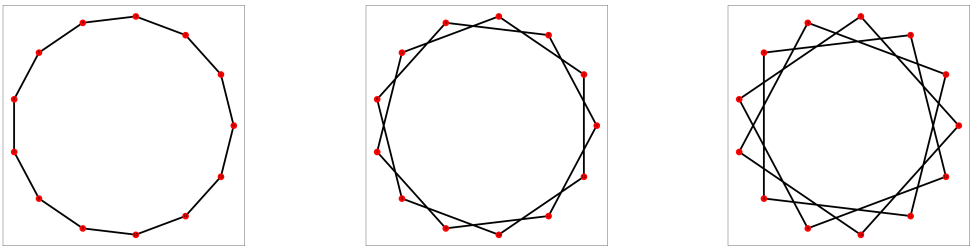
If $\Delta(\mathcal{H}) \leq \left\lfloor \sqrt{2m} \right\rfloor$, then $c_{\mathcal{G}}(\mathcal{H}) \leq \Delta(\mathcal{H}) + 1 \leq \left\lfloor \sqrt{2m} \right\rfloor + 1$ by Corollary 6.16. The result holds in this case. Thus, let some temporal node (t, v) of $\mathcal{H}$ have degree $d > \left\lfloor \sqrt{2m} \right\rfloor$. Put one cop on $(0, v)$ that will move on $(0, v), (1, v), \ldots, (p-1, v), (0, v)$. This cop prevents the robber from moving on v at any time. Let us now remove v together with its neighbourhood from the footprint of $\mathcal{H}$. This constructs an induced periodic subgraph $\mathcal{H}' \subset \mathcal{H} \subseteq \mathcal{G}$ with $m' < m$ nodes and, by the induction hypothesis, $c_{\mathcal{G}}(\mathcal{H}') \leq \left\lfloor \sqrt{2m'} \right\rfloor + 1$. Observe that removing v from $\mathcal{H}$ removes at least $\left\lfloor \sqrt{2m} \right\rfloor + 2$ nodes, since v itself counts for one and it has degree at least $\left\lfloor \sqrt{2m} \right\rfloor + 1$ in $\mathcal{H}$. Thus, $m' \leq m - \left(\left\lfloor \sqrt{2m} \right\rfloor + 2 \right)$. Moreover, since one cop guards v and $\mathcal{H}' \subseteq \mathcal{H}$, we must have $c_{\mathcal{G}}(\mathcal{H}) \leq 1 + c_{\mathcal{G}}(\mathcal{H}')$. It follows that:

$$
\begin{aligned}
c_{\mathcal{G}}(\mathcal{H}) &\leq 1 + c_{\mathcal{G}}(\mathcal{H}') \\
&\leq 1 + \left(\left\lfloor \sqrt{2m'} \right\rfloor + 1 \right) \quad \text{By the induction hypothesis} \\
&\leq 2 + \left\lfloor \sqrt{2 \left(m - \left(\left\lfloor \sqrt{2m} \right\rfloor + 2 \right) \right)} \right\rfloor \quad \text{By Wagner's argument} \\
&\leq 1 + \left\lfloor \sqrt{2m} \right\rfloor \quad \text{By Corollary 6.16, with } \mathcal{H}' \text{ an induced periodic subgraph of } \mathcal{H}
\end{aligned}
$$

The proof follows from induction when $m = n$. □

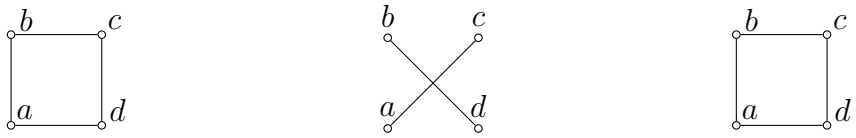
6.6 A family of periodic graphs with clique footprint

Let G be the complete graph K_n on n nodes. Suppose the nodes of G are labelled from 0 to $n-1$ and let $C = (0, \ldots, n-1)$ be a cycle. Imagine the nodes of C are drawn along a circle in increasing order, clockwise. Then, the *clockwise distance* from i to j is the length of the subpath $(i, i+1, \ldots, j-1, j)$ of C. Consider the *chordal labelling* of G such that for every node u, the *undirected* edge uv is labelled with the clockwise distance from u to v. Observe that every edge uv has two labels corresponding to the clockwise distance from u to v and from v to u. The subgraph $G_k \subset G$ on the same set of nodes of G is the reflexive, *undirected* graph containing all the edges at least



(a) G_0 (b) G_1 (c) G_2

Figure 6.3: The first three snapshots of a $\mathcal{K}_{13}$ with 13 nodes and period 12. Every snapshot is undirected.



(a) G_0 (b) G_1 (c) G_2

Figure 6.4: The periodic graph $\mathcal{K}_4$ is copwin.

one of whose labels is $1 \leq k+1 \leq n-1$. We define the periodic graph $\mathcal{K}_n = (G_0, \ldots, G_{n-2})^*$ with n nodes and period $n-1$. Observe that there is a unique periodic graph $\mathcal{K}_n$ per positive integer n. Figure 6.3 displays the first snapshots of the periodic graph $\mathcal{K}_{13}$ with 13 nodes.

Lemma 6.18. *Let i be any node of $\mathcal{K}_n$. For every $0 \leq t \leq n-2$,*

$$N_t[i, \mathcal{K}_n] = \{i, i+(t+1) \mod n, i-(t+1) \mod n\}.$$

Furthermore, when $n \geq 3$ is odd

$$N_{\frac{n-3}{2}}[i, \mathcal{K}_n] = N_{\frac{n-1}{2}}[i, \mathcal{K}_n].$$

Proof. For any time $0 \leq t \leq n-2$ and node i, i is connected in snapshot G_t in a symmetric way to nodes j and k at distance $(t+1) \mod n$ and $-(t+1) \mod n$ by construction.

Consider the second statement. The times $\frac{n-3}{2}$ and $\frac{n-1}{2}$ correspond to

the middle of the period. By the previous argument, for any node i,

$$\begin{aligned}
N_{\frac{n-3}{2}}[i, \mathcal{K}_n] &= \left\{ i, i + \left(\frac{n-3}{2} + 1 \right) \mod n, i - \left(\frac{n-3}{2} + 1 \right) \mod n \right\} \\
&= \left\{ i, i + \frac{n-1}{2} \mod n, i + \frac{n+1}{2} \mod n \right\} \\
N_{\frac{n-1}{2}}[i, \mathcal{K}_n] &= \left\{ i, i + \left(\frac{n-1}{2} + 1 \right) \mod n, i - \left(\frac{n-1}{2} + 1 \right) \mod n \right\} \\
&= \left\{ i, i + \frac{n+1}{2} \mod n, i + \frac{n-1}{2} \mod n \right\}
\end{aligned}$$

and the result follows. □

Theorem 6.19. *For every $n \geq 1$, $c(\mathcal{K}_n) \leq 2$.*

Proof. Let $n \geq 1$ be any integer and $\mathcal{K}_n = (G_0, \ldots, G_{n-2})^*$ have nodes $V = \{0, \ldots, n-1\}$. We show that two cops can capture the robber on $\mathcal{K}_n$. This is clear when $n \leq 4$, so suppose that $n \geq 5$. We write c_i^t, with $i \in \{0, 1\}$, for the position of cop i at time t, with operations on the indices taken modulo 2, and r^t for the position of the robber.

For every $0 \leq i \leq n-1$ and $0 \leq t \leq n-2$, if the cops are on $(t+1, i-1)$ and $(t+1, i+1)$ while the robber is on (t, i), then the cops are on a winning configuration. If the robber moves from (t, i), it gets caught by either of the two cops. Indeed, by Lemma 6.18,

$$\begin{aligned}
N_t[i, \mathcal{G}] \setminus \{i\} &= \{i - (t+1) \pmod n, i + (t+1) \pmod n\} \\
&\subseteq N_{t+1}[i-1, \mathcal{G}] \cup N_{t+1}[i+1, \mathcal{G}] \\
&= \left\{ \begin{array}{l} i+1, i-1, \\ i+1+(t+2) \pmod n, i+1-(t+2) \pmod n, \\ i-1+(t+2) \pmod n, i-1-(t+2) \pmod n \end{array} \right\} \\
&= \left\{ \begin{array}{l} i+1, i-1, \\ i+t+3 \pmod n, i-t-1 \pmod n, \\ i+t+1 \pmod n, i-t-3 \pmod n \end{array} \right\}.
\end{aligned}$$

Thus, the robber must stay on i at time t and the cops can catch it when $t = 0$ or $t = n-2$. Let us write $C = G_0$ and $d_C(u, v)$ as the distance, measured in C, between nodes u and v. We show the cops can play so as to be in the configuration above.

Let the cops stay still on their first move in G_0, so they can react to the robber's actions. For any time t, if the robber does not move after the cops

in G_t, then in G_{t+1} the cops can keep their distances to the robber, because it is measured in C, and decrease it at least once per period. It follows that the robber must move at every turn. Without loss of generality, let the robber move to $r^{t+1} \equiv r^t + (t+1) \pmod{n}$ in G_t.

Cop i has three choices in G_{t+1}, either stay on c_i^{t+1}, move to $c_i^{t+1} + (t+2) \pmod{n}$ or move to $c_i^{t+1} - (t+2) \pmod{n}$.

Since we can relabel nodes, we can always write $c_i^t = 0$. This gives an orientation to G_{t+1} and we can write the robber's position relative to cop i as either $-r^t \equiv n - r^t \pmod{n}$ or r^t. Conceptually, the robber is either to the "left" of the cop or to the "right".

Let the cops play so that one is on c_0^{t+1} with the robber on r_0^t, while the other cop is on c_1^{t+1} and the robber on $-r_1^t$. Thus, from the cops' perspectives, $1 \leq r_0^t < \left\lfloor \frac{n}{2} \right\rfloor$ and $n - 1 \geq r_1^t > \left\lfloor \frac{n}{2} \right\rfloor$. We show both cops can get closer to the robber.

1. Let us start with cop 0. We have $d_C(c_0^{t+1}, r_0^t) = r_0^t$. Let the cop move to $t + 2 \pmod{n}$, so that

$$\begin{aligned} d_C(c_0^{t+2}, r_0^{t+1}) &= \left| r_0^{t+1} - c_0^{t+2} \right| \\ &= \left| r_0^{t+1} - (0 + t + 2) \right| \\ &= \left| r_0^t + (t+1) - (t+2) \right| \\ &= \left| r_0^t - 1 \right| \\ &= d_C(c_0^{t+1}, r_0^t) - 1. \end{aligned}$$

   That is, cop 0 can get closer to the robber in C.

2. Consider now cop 1. The robber's position relative to this cop is $-r_1^{t+1} \equiv n - r_1^{t+1} \pmod{n}$. Let the cop move to $-t - 2 \pmod{n} \equiv n - t - 2 \pmod{n}$. We have

$$\begin{aligned} d_C(c_1^{t+2}, r_1^{t+1}) &= \left| n - t - 2 - (n - r_1^{t+1}) \right| \\ &= \left| n - t - 2 - (n - (r_1^t + t + 1)) \right| \\ &= \left| r_1^t - 1 \right| \\ &= d_C(c_1^{t+1}, r^t) - 1. \end{aligned}$$

   That is, cop 1 moves strictly closer to the robber.

It follows that two cops can move closer to the robber in C to produce the configuration presented above. Therefore, the cops are eventually on $(t+1, i+1), (t+1, i-1)$ while the robber is on (t, i), which is a winning position for the cops. Therefore, two cops can capture the robber on $\mathcal{K}_n$. □

Corollary 6.20. *For every $n \geq 5$ prime, $c(\mathcal{K}_n) = 2$.*

Proof. First, let us show that if $\mathcal{G} = (G_0, \ldots, G_{p-1})^*$ is a sequence of edge-disjoint hamiltonian cycles with $n \geq 4$ nodes, then $\mathcal{G}$ is not copwin.

Let $\mathcal{G}$ be such a periodic graph, with $n \geq 4$ nodes and period $p \geq 1$. The result already holds when $p = 1$, so suppose $p > 1$. For every time t and node i, i has degree exactly two (because the degree of i does not account for self-loops) in G_t, by construction. Thus, for any two nodes i and j and time t, if (t, i) is a temporal corner of $(t+1, j)$, then j is adjacent in G_{t+1} to the two neighbours u and v of i in G_t, and $ij \in E(G_{t+1})$. Because G_t and G_{t+1} are edge-disjoint, $j \neq u$ and $j \neq v$. That is, $N_{t+1}[j, \mathcal{G}] = \{i, j, u, v\}$ and j has degree three in G_{t+1}, which is impossible. Therefore, $\mathcal{G}$ contains no temporal corner and is not copwin by Lemma 3.3.

Let now $\mathcal{K}_n = (G_0, \ldots, G_{n-2})^*$ with $n \geq 5$ prime. One can verify that $(G_0, \ldots, G_{\frac{n-3}{2}})$ and $(G_{\frac{n-1}{2}}, \ldots, G_{n-2})$ are two sequences of edge-disjoint hamiltonian cycles, while $G_{\frac{n-3}{2}} = G_{\frac{n-1}{2}}$ and $G_{n-2} = G_0$ by Lemma 6.18.

Therefore, the periodic subgraph $\mathcal{G}_0 = \left(G_0, \ldots, G_{\frac{n-3}{2}}\right)^*$ cannot be copwin by the above reasoning, and similarly for $\mathcal{G}_{\frac{n-1}{2}} = \left(G_{\frac{n-1}{2}}, \ldots, G_{n-2}\right)^*$ by the same result. It follows that if the robber starts at distance at least two from a single cop in G_0, it can play so as to end its turn at distance at least two from the cop in $G_{\frac{n-3}{2}}$. Since $G_{\frac{n-1}{2}} = G_{\frac{n-3}{2}}$, when the robber starts at distance two from the cop it can end its turn at distance two from it. If the robber starts at distance two from the cop in $G_{\frac{n-1}{2}}$, then it can escape in $\mathcal{G}_{\frac{n-1}{2}}$ by the above argument. Finally, $G_{n-2} = G_0$ and by the same argument as above the robber can survive from G_{n-2} to G_0. It follows that $\mathcal{K}_n$ is not copwin when $n \geq 5$ is prime.

Therefore, by Theorem 6.19, $c(\mathcal{K}_n) = 2$. □

One interesting aspect of $\mathcal{K}_n$ is its systemic construction and how we can deduce its cop number from the shape of its neighbourhoods. When $n \geq 5$ is prime the cop number of $\mathcal{K}_n$ equals its minimum degree. Thus, we wonder if we can generalize this construction to graphs with larger minimum degrees such as cubic graphs.

6.7 A family of periodic graphs with maximal outerplanar footprint

Every maximal outerplanar graph is 2-connected[4] and can be represented with an *outer cycle* that bounds the outer face. The remaining edges are called the *chords*.

Recall that a periodic graph can have a *stable* subgraph that is a subgraph of the footprint and of every snapshot.

A maximal outerplanar graph is also chordal and we know that chordal graphs are copwin [13]. We show a special case when a periodic graph with maximal outerplanar footprint is also copwin. In general, if $\mathcal{G}$ has a maximal outerplanar footprint it might not be copwin since we can construct each snapshot so that they only contain large induced cycles.

We say a periodic graph $\mathcal{G}$ is 1-*bounded* if every snapshot is missing at most one edge and every edge is missing in at most one snapshot per period.

Given a cycle C and three distinct nodes x, y, u, we write C_{xy} for a shortest path of C from x to y and C^u_{xy} for the subpath of C from x to y that contains u.

Lemma 6.21. *Let $\mathcal{G}$ be a periodic graph with maximal outerplanar footprint G with $n \geq 5$ vertices and period $p > 2$. Let $\mathcal{G}$ be 1-bounded and the outer cycle C of G be stable. Suppose u is a node of degree two of G with neighbours v and w. We make two claims.*

1. *Suppose G has a node x that is incident to $k \geq 2$ chords $(xv_1, \ldots, xv_k)$, with v_1 closest to u on C. Let the cop be on x at time t and suppose the robber moves on $(v_k, \ldots, v_i)$. If the robber ever moves from v_i to either v_i or v_{i+1}, it gets captured by the cop. Furthermore, if $u \in N_G[x]$, the cop catches the robber on u.*

2. *Let $x \neq u$ be the other common neighbour of v and w. A cop on x can prevent the robber on u from moving out of $\mathcal{G}[\{u, v, w, x\}]$ and capture the robber.*

Proof. Let us prove the first claim. The only way the robber can safely move on v_k while the cop is on x is by ending its turn on v_k at time t such

[4]Indeed, suppose otherwise and let u be a cutvertex of a maximal outerplanar graph G. Let $v, w \in N_G(u)$ be in different connected components of $G \setminus \{u\}$. Then, $G \cup \{vw\}$ is maximal outerplanar, which is a contradiction.

that $xv_k \notin E(G_{t+1})$. Let the robber do so. Because $\mathcal{G}$ is 1-bounded, we have $xv_k \in E(G_{t+2})$, so the robber must move to v_{k-1} in G_{t+1}. In turn, we must have $xv_{k-1} \notin E(G_{t+2})$. More generally, there exists a sequence of times $(t_1, t_2, \ldots, t_k)$ such that $xv_i \notin E(G_{t_i})$ for every $1 \leq i \leq k$. Let us show that the only way for the robber to survive on the shortest path $C_{v_1v_k}$ is if $t_{j-1} \equiv t_j + 1 \mod p$ for all $2 \leq j \leq k$. We know $t_{k-1} \equiv t_k + 1 \mod p$ by the argument above. Thus, suppose otherwise and take the greatest index $j < k$ such that $t_{j-1} \not\equiv t_j + 1 \mod p$. Then, the robber on v_j can neither move to v_{j-1}, stay on v_j, nor move to v_{j+1} since all three chords xv_{j+1}, xv_j, xv_{j-1} appear in $G_{t_{j+1}}$. The robber gets captured, which is a contradiction. Thus, $t_{j-1} \equiv t_j + 1 \mod p$ for all $2 \leq j \leq p$. Therefore, the robber is forced to move toward v_1 and the cop can force it to stay in $C^u_{xv_k}$. It follows that if $u \in N_G[x]$, the cop captures the robber on u.

Let us show the second part of the statement. Suppose $vw, xw \in E(G \backslash C)$, $xv \in E(C)$ and $vw \notin E(G_t)$. Let the robber start on u and the cop on x.

Let $t_x > t$ be the next time xw disappears, which is unique because $\mathcal{G}$ is 1-bounded. As long as the cop is on x, the robber does not move to v, since xv is stable. Similarly, while the cop is on x, the robber can safely move on w only at time $t_x - 1$. At time $t_x - 1$, let the cop move to w. In G_{t_x}, the chord wv appears, because $xw \notin E(G_{t_x})$ and $\mathcal{G}$ is 1-bounded. Therefore, we have $N_{t_x-1}[u, \mathcal{G}] = \{u, v, w\} \subseteq N_{t_x}[w, \mathcal{G}] = \{u, w, v\}$, so the robber is on a temporal corner of the cop's position and gets captured on the next turn. □

Lemma 6.22. *Let $\mathcal{G}$ be a periodic graph with maximal outerplanar footprint with $n \geq 5$ vertices and period $p = 2$. If $\mathcal{G}$ is 1-bounded and the outer cycle of G is stable, then it is copwin.*

Proof. Let G have outer cycle C. Because $\mathcal{G}$ has period 2 and is 1-bounded, G has at most two chords that are not stable. If all chords of $\mathcal{G}$ are stable, then $\mathcal{G}$ is copwin for it suffices for the cop to follow a copwin strategy from G. Among the copwin strategies on G, at least one is monotone: such that the cop can constrain the robber on a subset of nodes that always decreases in size. Also, for every subset $V' \subseteq V$, $G[V']$ is maximal outerplanar. Thus, if $G[V']$ is connected and every edge of $G[V']$ is stable in $\mathcal{G}$, $\mathcal{G}[V']$ is copwin.

Suppose xy is the only chord of G that is not stable and let the cop start on the node of x or y with highest degree in G. Without loss of generality, let this be x. The robber starts on r. The vertices on the path C^r_{xy} induce a connected subgraph $H \subset G$ that is stable except for xy. Thus, once the cop

has left x, it moves on a stable connected subgraph of G and it follows that $\mathcal{G}$ is copwin.

Suppose now that two chords xy and uv are not stable in $\mathcal{G}$ and let us show that $\mathcal{G}$ is still copwin. Suppose $xy \in E(G_0)$, so $uv \in E(G_1)$, and that x is closer to u than v on C. There are two cases to consider.

1. Suppose $u = x$. Let the cop start on y. If the robber is in $N_0[y, \mathcal{G}]$, it gets captured in G_0. In general, the robber is either on the subpath C_{xy}^{v} or the other path $C_{xy}^{-v} := C \setminus C_{xy}^{v}$. If the robber starts on C_{xy}^{-v}, the cop moves to x in G_0, to a neighbour $z \in N_{C_{xy}^{-v}}(x)$ with $z \neq y$, then plays on a connected induced subgraph of G that is stable in $\mathcal{G}$. This is copwin. Otherwise, the robber is in C_{xy}^{v}. Because G is maximal outerplanar, x has $k \geq 1$ neighbours $(v_1 = y, \ldots, v_k = v)$ in C_{xy}^{v}. Let the cop walk along $(v_1, \ldots, v_k)$. When $k > 2$, for every $2 \leq i \leq k-1$, the cop can wait on v_i and prevent the robber from moving to x since xv_i is stable. When $k = 2$, the cop moves directly to v. Thus, the cop can move on $v_k = v$ in G_0 and prevent the robber from moving to x because $xv \in E(G_1)$. When the cop is on v, the robber is somewhere in $C_{xy}^{v} \setminus \{v, x\}$. The connected subgraph H of G induced by the nodes of $C_{xy}^{v} \setminus \{v, x\}$ is stable. Therefore, the cop follows a copwin strategy on H and captures the robber.

2. Suppose $u \neq x$. The footprint G must have at least three chords because it is maximal outerplanar and $\{u, v\} \cap \{x, y\} = \emptyset$, so at least one is stable. Let ab be a chord of G such that uv and xy are in different connected components C_u and C_x of $G \setminus \{a, b\}$. Then, ab is stable in $\mathcal{G}$. Let the cop start on a. The robber must be in either C_u or C_x. But, both contain only one chord, so the cop can apply its winning strategy above on either $\mathcal{G}[C_u]$ or $\mathcal{G}[C_x]$ and make the catch.

In all cases, $\mathcal{G}$ is copwin. □

Proposition 6.23. *Let $\mathcal{G}$ be a 1-bounded periodic graph with maximal outerplanar footprint. If the outer cycle of G is stable, then it is copwin.*

Proof. Let G have n nodes and outer cycle C. One can verify that when $n \leq 4$, G has at most one chord and $\mathcal{G}$ is copwin. Therefore, suppose $n \geq 5$. The case when $\mathcal{G}$ has period $p = 2$ was handled in Lemma 6.22, so let us assume $p > 2$.

Observe that $\mathcal{G}$ is always connected since C is stable. Since G is connected and maximal outerplanar, it contains at least two nodes of degree two. If $xy \in E(G)$ is a chord, then either $\deg_{G \setminus E(C)}(x) \geq 2$ or $\deg_{G \setminus E(C)}(y) \geq 2$. Moreover, for every node x such that $\deg_{G \setminus E(C)}(x) \geq 2$, either x is universal or there exists a node y such that $\deg_{G \setminus E(C)}(y) \geq 2$, again because G is maximal outerplanar.

Let u be a vertex of degree two of G and let the cop start on u. Let also $R_c \subseteq V$ be the *region protected by the cop*. Initially, $R_c = N_G[u]$. We show by induction that while $|R_c| < n - 1$ the cop can move on a node $x \in V \setminus R_c$ such that:

- If the robber moves in R_c then it eventually gets captured;
- The node x is adjacent to two nodes v and w such that $v \in R_c$ and $w \in V \setminus R_c$;
- The cop can augment R_c with the nodes of $V(C^u_{xv}) \setminus R_c$.

Figure 6.5 sketches some of the cases that follow to help visualize the proof.

Let the robber start on some node y in G_0. Let $N_G[u] = \{u, v, w\}$, so $vw \in E(G \setminus C)$ and suppose $\deg_{G \setminus E(C)}(w) \geq 2$. Let the cop move to w, which is possible because C is stable. If w is universal in $G[V \setminus R_c]$, then $\mathcal{G}$ is copwin by Item 1 in Lemma 6.21. Otherwise, let $ww' \in E(G \setminus C)$ be such that $w' \in C^u_{wy}$, $\deg_{G \setminus E(C)}(w') \geq 2$ and w' is closest to y on C. This chord exists because G is maximal outerplanar. The set of nodes $R_c = \{u, v, w\}$ is protected by the cop, if the robber moves in R_c it gets captured. Indeed, let $v_r \neq u$ be the common neighbour of v and w on C and t_v the next time $vw \notin E(G_{t_v})$. If the robber is on v_r and wants to move in R_c while the cop is on w, its only option is to move on v at time $t_v - 1$. The cop can move to v_r along the chord $wv_r \in E(G_{t_v})$ because $\mathcal{G}$ is 1-bounded. The robber must move to u in G_{t_v}. By Item 2 in Lemma 6.21, this situation is winning for the cop. By Item 1 of the same result, if instead the robber is on w' at time $t_{w'}$ such that $ww' \notin E(G_{t_{w'}})$, it must move toward u on $C_{w'u}$ and it eventually gets captured. Therefore, the robber ends its turn, at any time, either on w' or on the subpath $C \setminus C^u_{ww'}$. Finally, w' is incident to a chord $w'z$ with $z \notin R_c$ because $\deg_{G \setminus E(C)}(w') \geq 2$. The cop can move to w' and augment R_c with $V(C^u_{w'w}) \setminus R_c$.

Suppose the result holds for the first $k \geq 1$ nodes visited by the cop and let the cop move from x' to $x \in V \setminus R_c$ in G_{t-1} such that $\deg_{G \setminus E(C)}(x) \geq 2$.

Suppose the robber is on y in G_t. If $y \in N_t[x, \mathcal{G}]$, the cop makes the catch. Similarly, if x is universal in $G[V \setminus R_c]$, then $\mathcal{G}$ is copwin. By construction, if $|R_c| < n-1$, then there exists a chord xz such that $z \in V \setminus R_c$. Because G is maximal outerplanar, either $N_G[y] \subset N_G[z]$ or z is incident to another chord za with $a \in V \setminus R_c$ and $\deg_{G \setminus E(C)}(a) \geq 2$. If $N_G[y] \subset N_G[z]$, then $N_G[y] = \{y, z, y'\}$ for some y'. We let the cop move to the common neighbour $z' \neq y$ of z and y' and play as in Item 2 of Lemma 6.21.

Therefore, let the cop choose such a chord xz with $z \in C^u_{xy}$ such that z is closest to y on C. By the induction hypothesis, $R_c = V(C^u_{x'x}) \cup N_G[u]$. Suppose $xz \notin E(G_t)$ and the robber ends its turn on z in G_{t-1}. Let the cop stay on x in G_t and the robber move to the neighbour z_c of z on C^u_{xz}. If the robber does not keep moving toward u, it gets captured by the cop by Item 1 in Lemma 6.21. Therefore, the robber must keep moving closer to u, eventually reaching x'. But, by the induction hypothesis, $x' \in R_c$ and the robber eventually gets captured if it steps in $\mathcal{G}[R_c]$. Thus, the robber does not move to z in G_{t-1} in order to avoid capture. The robber ends its turn in G_t in $C \setminus (C^u_{xz} \cup N_t[x, \mathcal{G}])$. The cop crosses the chord xz in G_{t+1}, since $\mathcal{G}$ is 1-bounded. The cop protects the new region $R_c = V(C^u_{xz})$. A similar reasoning holds if $xz \in E(G_t)$.

By induction, we eventually have $|R_c| = n-1$ and the robber is stuck on a node a of degree two of G, with neighbours b and c. The cop can move to the common neighbour $d \neq a$ of b and c. Both $ab \in E(C)$ and $ac \in E(C)$, so the players are in similar positions as in Item 2 of Lemma 6.21. The cop can then capture the robber. It follows that $\mathcal{G}$ is copwin when $p > 2$. Hence, $\mathcal{G}$ is copwin for every period. □

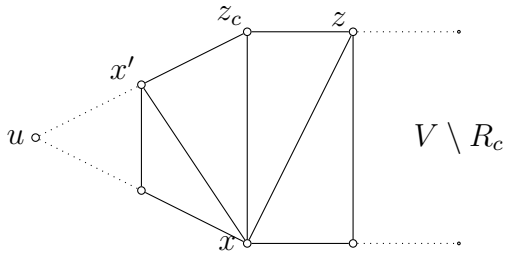


Figure 6.5: Sketch of the maximal outerplanar graph used in the proof of Proposition 6.23.

We expect a stronger version of this argument would also hold when the

outer cycle in the footprint of $\mathcal{G}$ is not stable, but $\mathcal{G}$ is still 1-bounded.

Chapter 7

Evaluating metrics in link streams

This work was presented first at the *International Conference on Advances in Social Networks Analysis and Mining* conference in Vancouver, Canada in 2019 [189], then published as a longer version in the journal *Social Networks Analysis and Mining* [188]. The latter version is transcribed here.

In latter chapters we focus on the game of cops and robber. In this chapter, we study a more applied problem of computing metrics efficiently in a link stream.

Our algorithms are evaluated on datasets taken from the KONECT library of networks [143]. A description of the datasets used can be found in Appendix B. The datasets are temporal networks found online that were obtained from real-world data and are often used in experiments. They are varied as they comprise, for example, citation networks, social networks and technological networks. We did not look for heterogeneity or homogeneity.

A state of the art was presented in Subsection 2.1.3. Basic definitions about link streams were presented in Section 3.2. More specific definitions are presented in Section 7.1. Then, we present our main methods in Section 7.2 and experiments in Section 7.3.

7.1 Background

Recall from Section 3.2 that a link stream is a triple $L = (T, V, A)$.

In a link stream L, a journey P from $(\alpha, u) \in T \times V$ to $(\omega, v) \in T \times V$ is a

sequence $(t_0, u_0v_0), (t_1, u_1v_1), \ldots, (t_k, u_kv_k)$ of temporal edges such that $u_0 = u$, $v_k = v$, $t_0 \geq \alpha$, $t_k \leq \omega$ and for all $i, t_i \leq t_{i+1}$ and $v_i = u_{i+1}$[1]. We say that such a journey starts at t_0, arrives at t_k, has length $k+1$ and duration $t_k - t_0$. We write $(\alpha, u) \rightsquigarrow (\omega, v)$ to mean that there exists a journey from (α, u) to (ω, v) and say (ω, v) is reachable from (α, u). We also call t_0 a starting time and t_k an arrival time from (α, u) to (ω, v). Each journey between two fixed temporal nodes (α, u) and (ω, v) defines a pair of starting time and associated arrival time. On the link stream of Figure 7.1, two journeys are illustrated: $P_1 = ((0, dc), (1, cb), (3, ba))$ and $P_2 = ((0, dc), (2, cb), (3, ba))$. Both have the same starting and arrival times from $(0, d)$ to $(3, a)$, namely times 0 and 3. We can also say t_s is a starting time from a temporal node $(\alpha, u) \in T \times V$ to a node $v \in V$, in which case there exists some time $t \in T$ such that t_s is the starting time of a journey from (α, u) to (t, v). The same goes for the arrival times.

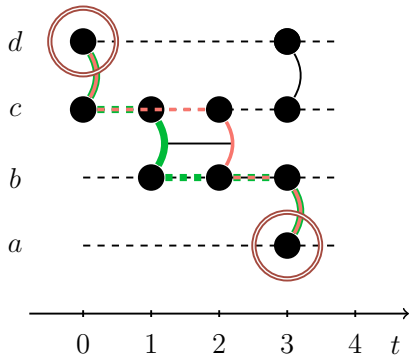


Figure 7.1: A simple link stream with two journeys P_1 and P_2, respectively, drawn in thick green and thin red between the same encircled temporal nodes.

We say a journey P from a temporal node (α, u) to another temporal node (ω, v) is *shortest* if it has minimal length among all journeys from (α, u) to (ω, v) and call its length the *distance* from (α, u) to (ω, v), written $d((\alpha, u), (\omega, v))$. Similarly, P is *fastest* if it has minimal duration, we call this duration the *latency* from (α, u) to (ω, v) and write it $l((\alpha, u), (\omega, v))$.

[1]For our purposes here, we assume that each G_t is not reflexive. Unlike in our definition of periodic graph (see Chapter 3), an agent can wait on a node without having to cross a reflexive edge, so journeys such as $((0, ab), (2, bc))$ are valid in a link stream.

Note that if $(\alpha, u) \rightsquigarrow (\omega, v)$, there exists at least one pair of starting time and arrival time (t_s, t_a) such that $l((\alpha, u), (\omega, v)) = t_a - t_s$. Then, we say t_s is a *latest* starting time and t_a an *earliest* arrival time from (α, u) to (ω, v). For example, on Figure 7.1, the times 0 and 3 are latest starting times for journeys from $(0, d)$ to (t, c) for some times t, whereas only 0 is a latest starting time between $(0, d)$ and $(0, c)$. Similarly, 0 and 3 are earliest arrival times for journeys from $(0, d)$ to (t, c) for some times t. Finally, P is called shortest fastest if it has minimal length among the set of fastest journeys from (α, u) to (ω, v). We call its length the *sf-metric* from (α, u) to (ω, v) and write it $m_{sf}((\alpha, u), (\omega, v))$. In general, this is not a distance as it does not respect the triangle inequality and is only a premetric, a simple counterexample is shown on Figure 7.2. On the same figure are drawn a shortest journey, two fastest journeys and a unique shortest fastest journey.

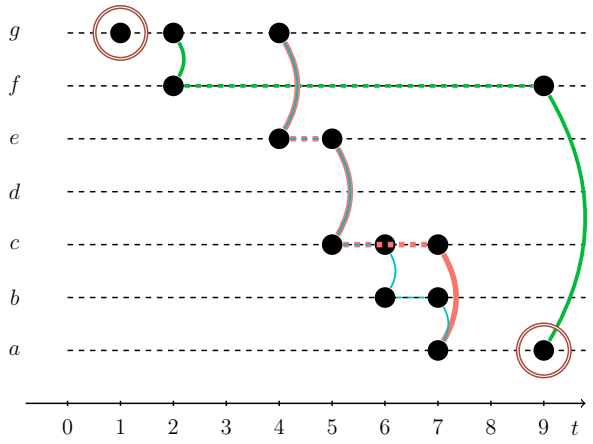


Figure 7.2: The shortest journey from $(1, g)$ to $(9, a)$ (both encircled ◎), of length 2, is drawn in green ▬. From $(1, g)$ to $(9, a)$, the two fastest journeys, with duration 3, are drawn in red ▬ and in blue ▬. The sole shortest fastest journey, with length 3, is the red one. Observe that, $m_{sf}((1, g), (9, a)) = 3 > m_{sf}((1, g), (9, f)) + m_{sf}((9, f), (9, a)) = 2$.

7.2 Multiple-targets shortest fastest journeys algorithms

The full implementations of the algorithms presented here, in C++, can be found online [190].

We present here two main methods, Algorithm 2 and Algorithm 3 that compute the distances, latencies and *sf*-metrics from one source event node to all other event nodes. Algorithm 3 builds on the first method to compute those values for all pairs of event nodes. Subsection 7.2.4 also presents Algorithm 4 that was derived from Algorithm 2. This new method works with positive *delays*, a case that is often considered in the literature. Algorithm 5 also works with positive delays and is derived from Algorithm 3. We focus on the first two algorithms.

We present some small results that lead the way to those algorithms. The strategy for all methods is the same: we compute the distances from any temporal node (s_v, u) to any other temporal node (t_v, v) such that s_v is the *largest* (or maximal) starting time from any (t_u, u), with $t_u \leq s_v$, to (t_v, v). If it happens that $t_v - s_v = l\left((s_v, u), (t_v, v)\right)$, then this distance is the *sf*-metric from the former to the latter temporal node. Otherwise, since we iterate chronologically over Ω, this latency must have been computed at a time earlier than t_v and is saved in memory.

All algorithms described in this section are presented in Appendix A in order not to disrupt the reading.

7.2.1 Two simple lemmas

The algorithms we present compute what we call *reachability triples* that contain information about the lengths of shortest journeys from one temporal node to another as well as the starting and arrival times of those journeys.

Definition 7.1 (Reachability triples)**.** *Let (t_s, s) be an event node. If there exists a shortest journey of length l from (t_s, s) to the event node (t_y, y) that starts on a largest starting time $t \in \Omega$, then we say (t, t_y, l) is a reachability triple from (t_s, s) to y.*

In the following, for any node v, we write R_v for the dictionary of reachability triples from a fixed source event node to v. In order to reduce to cost of operations in R_v, we assume this dictionary is implemented in such

a way that R_v holds keys s_v and that $\mathrm{R}_v[s_v]$ holds pairs (a_v, d_v) that form reachability triples (s_v, a_v, d_v). This makes it easier to search for a starting time s_v.

All algorithms compute distances *from largest starting times only*. One could define reachability triples without the constraint that starting times are largest, however the algorithms would not be as efficient because the dictionaries would grow larger with $|\Omega|$. Note that if T is a singleton, then each R_v will contain the graph distances from a fixed source to v. The temporal nature of a link stream forces us to take starting and arrival times into account when looking for shortest journeys.

Lemma 7.2 below, due to Wu et al. [208], states that shortest journeys are prefix-shortest. We say a journey $P_{(t_s,s)(t_u,u)}$ from a temporal node (t_s, s) to another temporal node (t_u, u) is a prefix of another journey $P_{(t_s,s)(t_v,v)}$ from the same source to temporal node (t_v, v) if $P_{(t_s,s)(t_u,u)}$ is a subsequence of $P_{(t_s,s)(t_v,v)}$.

Lemma 7.2. *Let $P_{(t_s,s)(t_v,v)}$ be a shortest journey from a temporal node (t_s, s) to another temporal node (t_v, v). Then, every prefix $P_{(t_s,s)(t_u,u)}$ of $P_{(t_s,s)(t_v,v)}$ is a shortest journey from (t_s, s) to (t_u, u).*

Let (t_s, s) and (t, v) be two temporal nodes. We define the *outer distance* from (t_s, s) to (t, v) as

$$d((t_s, s), (t^-, v)) := \begin{cases} d((t_s, s), (t, v)), & \text{if } t_s = t, \\ \min\limits_{t_s \leq t_0 < t} d((t_s, s), (t_0, v)), & \text{otherwise,} \end{cases}$$

For example, on the link stream of Figure 7.2, $d((0, g), (9^-, a)) = 3$, while $d((0, g), (9, a)) = 2$. In other words, $d((t_s, s), (t^-, v))$ is either equal to $d((t_s, s), (t, v))$, when $t_s = t$, or it is the smallest length of all shortest paths from (t_s, s) to (t_0, v) such that $t_0 < t$.

Lemma 7.3 suggests it suffices to compute the graph distances in the snapshots to deduce the distances between two temporal nodes. Given a journey $P = (t_1, u_1u_2), \ldots, (t_n, u_nu_{n+1})$, a subsequence $P' = (t_i, u_iu_{i+1}), \ldots, (t_{i+k}, u_{i+k}u_{i+k+1})$ of P for some $1 \leq i \leq n$ and $k \geq 0$ is called a subjourney of P from (t_i, u_i) to (t_{i+k}, u_{i+k+1}).

Lemma 7.3. *Let (t_s, s) be a source temporal node and (t_y, y) be a temporal node reachable from the source by a non-empty journey. For every time*

$t_s \leq t \leq t_y$, there exists a connected component C of G_t such that

$$d((t_s, s), (t_y, y)) = \min_{u,v \in C} d((t_s, s), (t^-, u)) + d((t, u), (t, v)) + d((t, v), (t_y, y)). \tag{7.1}$$

Proof. Let $P = (t_1, u_1u_2), \ldots, (t_n, u_nu_{n+1})$ be a non-empty shortest journey from (t_s, s) to (t_y, y). By definition, $t_y \geq t_n \geq \cdots \geq t_1 \geq t_s$ and $u_1 = s, u_{n+1} = y$. For every time t_j, let $Q_j = (t_j, u_ju_{j+1}), \ldots, (t_j, u_ku_{k+1})$ be the longest subjourney of P such that $u_iu_{i+1} \in E(G_{t_j})$ for every $j \leq i \leq k$. Moreover, let Q_j end on node v_j. By Lemma 7.2, Q_1 is the prefix of P from (t_s, s) to (t_1, v_1), is shortest and has length $d((t_s, s), (t_1, v_1))$. Similarly, for every $j > 1$, by Lemma 7.2 the prefix of P from (t_s, s) to (t_{j-1}, v_{j-1}) is shortest and its length is $d((t_s, s), (t_{j-1}, v_{j-1}))$. Moreover, $d((t_s, s), (t_{j-1}, v_{j-1})) = d((t_s, s), (t_j^-, v_{j-1}))$ by definition of outer distance.

We can apply the same exhange argument for prefixes from Lemma 7.2 to the subjourney of P from (t_j, v_j) to (t_y, y), replacing prefix with suffix. Indeed, if this subjourney was not shortest, we could replace it with a shorter one and reduce the total length of P, contradicting P's property of being shortest. Thus, the subjourney of P from (t_j, v_j) to (t_y, y) has length $d((t_j, v_j), (t_y, y))$. Finally, since P is shortest and the two subjourneys formed by $P \setminus Q_j$ are shortest, Q_j must also be a shortest journey from (t_j, v_{j-1}) to (t_j, v_j). Otherwise, one could replace Q_j with a shorter journey from (t_j, v_{j-1}) to (t_j, v_j) and obtain a journey from (t_s, s) to (t_y, y) that is shorter than P, which would be a contradiction. Then, Q_j has length $d((t_j, v_{j-1}), (t_j, v_j))$. Let C_j be the connected component of G_{t_j} that contains the nodes of Q_j. Then, we have

$$\begin{aligned} d((t_s, s), (t_y, y)) &= d((t_s, s), (t_j^-, v_{j-1})) + d((t_j, v_{j-1}), (t_j, v_j)) \\ &\quad + d((t_j, v_j), (t_y, y)) \\ &= \min_{u,v \in C_j} d((t_s, s), (t_j^-, u)) + d((t_j, u), (t_j, v)) \\ &\quad + d((t_j, v), (t_y, y)). \end{aligned}$$

□

Let us revisit the link stream of Figure 7.2 and compute $d((0, g), (9, a))$ with Lemma 7.3. Since this result applies to every time $0 \leq t \leq 9$, let us choose $t = 9$. Then, there exists a connected component C of G_t such that

$$d((0, g), (9, a)) = \min_{u,v \in C} d((0, g), (9^-, u)) + d((9, u), (9, v)) + d((9, v), (9, a)).$$

We can choose $C = \{f, a\}$ and deduce that

$$\begin{aligned} d((0,g),(9,a)) &= \min_{u,v \in \{f,a\}} d((0,g),(9^-,u)) + d((9,u),(9,v)) + d((9,v),(9,a)) \\ &= d((0,g),(9^-,f)) + d((9,f),(9,a)) + d((9,a),(9,a)) \\ &= 1 + 1 + 0 \\ &= 2, \end{aligned}$$

which is the correct length of the shortest path from $(0, g)$ to $(9, a)$.

7.2.2 A single-source method

In this section, we present Algorithm 2 that computes the distances (from largest starting times), latencies and *sf*-metrics from a source event node (t_s, s) to all other reachable event nodes. This algorithm mixes iterations on the induced graphs G_t for each time $t \in \Omega$ with an all-pairs distances method on their connected components. If t_s^* is the largest starting time from the source (t_s, s) to some temporal node (t, v), then either $t - t_s^* = l((t_s, s), (t, v))$ or not. If so, then $d((t_s^*, s), (t, v))$ is the *sf*-metric $m_{sf}((t_s, s), (t, v))$. This length is computed with Lemma 7.3 by using the outer distances saved in memory as well as the all-pairs distance method on G_t. Thus, when we iterate over all pairs (s_v, d_v) of starting time and outer distance from the source to (t, v), we can deduce the duration and the length of the shortest fastest journeys from the source to (t, v). This method uses a set D that is assumed sorted in lexicographic order. Sorting D helps lower the temporal complexity, but is not fundamental to understand the algorithm.

Remark 7.4. *In all algorithms, we assume the dictionaries use self-balanced binary trees in order to obtain logarithmic worst-case complexities with simple structures. In our implementations, we used hash tables and heaps (see Remark 7.7) to lower the running times.*

Furthermore, all operations on dictionaries such as min *and* max *are well-defined whenever the relevant keys are present in the dictionaries. We assume the absence of a key is properly handled.*

Before proving that Algorithm 2 is correct, let us go through a small example in order to build intuition. Other algorithms are highly similar.

Example 7.5. *Consider again the link stream of Figure 7.2. Suppose the source is again* $(1, g)$*,* $t = 7$ *and* $C = \{a, b, c\}$*. Thus, Algorithm 2 will look*

for shortest (fastest) journeys that can reach temporal nodes $(7, a), (7, b)$ *and* $(7, c)$*. The unique largest starting time from the source to* C *at time* 7 *is* $s_v = 4$*. This time is given by the greatest key in* R_u *such that* $u \in C$*. Then, we iterate over the outer distances from* $(4, g)$ *to* $(7, v)$ *for each* $v \in C$*. Note how the time of the source has changed from* 1 *to* 4*. The outer distances are given as the distances from* $(4, g)$ *to* $(6, v)$ *for each* $v \in C$*. Thus, we find outer distances* 2 *from* $(4, g)$ *to* $(7, c)$ *and* 3 *from* $(4, g)$ *to* $(7, b)$*. Node* a *is discovered at time* 7 *and its outer distance does not exist before that. Finally, combining the outer distances with the distances inside the graph induced by* C *at time* 7*, we find that the distance from* $(4, g)$ *to* $(7, c)$ *is* 2*,* 3 *from* $(4, g)$ *to* $(7, b)$ *and also* 3 *from* $(4, g)$ *to* $(7, a)$*. This last distance is given by the combination between the outer distance from* $(4, g)$ *to* $(7, c)$ *and the distance in* C *from* $(7, c)$ *to* $(7, a)$*. Since node* a *is discovered first at time* 7*, that is its first arrival time from* $(1, g)$ *is* 7*, then the latency from* $(1, g)$ *to* $(7, a)$ *is* $l\left((1, g), (7, a)\right) = 7 - 4 = 3$ *and the distance from* $(1, g)$ *to* $(7, a)$ *is the sf-metric from the former to the latter.*

In Algorithm 2, we use a set D that contains triples of the form $(-s_u, d_u, u)$. The term s_u is a starting time from the source to u at the time of the current iteration. The negative sign in $-s_u$ is used to sort D, so that tuples with higher starting times are iterated on first.

Proposition 7.6. *Algorithm 2 correctly computes the latencies and sf-metrics from a source temporal node to all reachable event nodes as well as the set of dictionaries* $\{\mathrm{R}_v \,|\, v \in V\}$*. It requires at most* $O\Big(|V|^2|\Omega|\log|\Omega| + |V|^3|\Omega|\Big)$ *operations in the worst case.*

Proof of correctness. Let $(t_v, v) \in \Omega \times V$ be some reachable destination from a fixed source (t_s, s). Let us show by induction on $\Delta := |\{t_0 \in \Omega \,|\, t_v \geq t_0 \geq t_s\}|$ that $d[(t_v, v)] = m_{sf}\left((t_s, s), (t_v, v)\right)$, $f[(t_v, v)] = l\left((t_s, s), (t_v, v)\right)$ and R_w is correct up to time t_v for every $w \in V$.

- When $\Delta = 1$, we iterate only on event time t_v, since $t_v \in \Omega$ by assumption. Then, G_{t_v} is a graph and the method all_pairs_distances(H) correctly returns the distances in every connected subgraphs H of G_{t_v} between every node. The result follows.

- Suppose the result holds for every $1 \leq \Delta \leq k$ and let $\Delta = k + 1$. Let $(t_1, \ldots, t_{\Delta-1})$ be the sequence of times previously iterated over on line 3

and t_Δ be the current event time. By the induction hypothesis, by time $t_{\Delta-1}$, all values of R_w, for all $w \in V$, are correctly updated. Let C_v be the connected component of G_{t_Δ} containing v. If $s \in C_v$, there is a fastest journey from (t_s, s) to (t_v, v) that starts at time t_Δ. Then, the result follows as in the case with $\Delta = 1$. Thus, suppose $s \notin C_v$. Since each R_w is correctly updated up to time $t_{\Delta-1}$ for each reachable $w \in V$, D contains triples $(-s_w, d_w, w)$ for each $w \in C_v$ that have been visited prior to $t_{\Delta-1}$ from the source from a starting time s_w. In particular, D contains the largest starting time s_w from the source to (t_Δ, w). Then, either $t_\Delta + s_w = l\left((t_s, s), (t_\Delta, w)\right)$ or this latency is given by some $f[(t_0, w)]$ such that $t_0 < t_\Delta$. In the latter case, the desired distance is already computed. Hence, let us iterate on $(-s_w, d_w, w)$.

By Lemma 7.3, for every node $u \in V$ and time $-s_w \leq t_i \leq t_\Delta$, there exists a connected component C_i of G_{t_i} such that $d((-s_w, s), (t_\Delta, u)) = \min_{x,y \in C_i} d((-s_w, s), (t_i^-, x)) + d((t_i, x), (t_i, y)) + d((t_i, y), (t_\Delta, u))$. The sequence of distances

$$d((-s_w, s), (-s_w, u)), \ldots, d((-s_w, s), (t_{\Delta-1}, u))$$

is non-increasing because each element is minimal. Thus, let u be any node of C_v. We can choose $t_i = t_\Delta$ and $C_i = C_v$, so that:

$$\begin{aligned} d((-s_w, s), (t_\Delta, u)) = &\min_{x,y \in C_v} d((-s_w, s), (t_\Delta^-, x)) \\ &+ d((t_\Delta, x), (t_\Delta, y)) \\ &+ d((t_\Delta, y), (t_\Delta, u)) \\ &= \min_{x \in C_v} d((-s_w, s), (t_{\Delta-1}, x)) \\ &+ d((t_\Delta, x), (t_\Delta, u)). \end{aligned}$$

By the induction hypothesis, the outer distance $d_x = d((-s_w, s), (t_{\Delta-1}, x))$ can be recovered from $(-s_w, t_{\Delta-1}, d_x) \in \mathrm{R}_x$ for each $x \in C_v$. Then, using d_x and the dictionary d' returned by the all-pairs distances algorithm on line 6, the expression above reduces to

$$d((-s_w, s), (t_\Delta, u)) = \min_{x \in C_v} d_x + d'[(x, u)].$$

In the last equation, the intermediary node $x \in C_v$ over which the minimum is taken is irrelevant. Observe that if (t_i, z), with $z \in C_v$ and

$t_i \in (t_1, \ldots, t_{\Delta-1})$, is reachable from $(-s_z, s)$, then $(t_i, d_0) \in R_z[-s_z]$ for some d_0 and s_z. Thus, for any node $z \in C_v$ such that $(t_\Delta - 1, d_z) \in \mathrm{R}_z[-s_z]$ with $d_z = d((-s_z, s), (t_{\Delta-1}, z))$ and $t_s \leq s_z \leq t_{\Delta-1}$, let $U_z = \{y \in C_v \mid (t_{\Delta-1}, d_z) \in \mathrm{R}_y[-s_z]\}$, which is computed on line 15. In particular, the set U_w contains the nodes of C_v that are reachable from $(-s_w, s)$ at time $t_{\Delta-1}$ with a shortest journey of length d_w. By the induction hypothesis, $d_w = d((-s_w, s), (t_{\Delta-1}, w))$ is minimal. Therefore,

$$\begin{aligned} d((-s_w, s), (t_\Delta, u)) &= \min_{x \in C_v} d_x + d'[(x, u)] \\ &= \min_{w \in V} \min_{y \in U_w} d_w + d'[(y, u)] \\ &= \min_{w \in V} d_w + \min_{y \in U_w} d'[(y, u)]. \end{aligned}$$

Thus, the distance $d((-s_w, s), (t_\Delta, u))$ is correctly computed when we iterate over all elements of D that have starting time s_w. It follows that $\mathrm{R}_u[-s_w]$ is correctly updated with $(t_\Delta, d((-s_w, s), (t_\Delta, u)))$ for every $u \in C_v$. In particular, $\mathrm{R}_v[-s_w]$ is also updated correctly. Finally, observe that once $f[(t_\Delta, v)]$ is updated with its final value, then by definition the update of $d[(t_\Delta, v)]$ on line 24 yields the *sf*-metric from (t_s, s) to (t_Δ, v).

□

Proof of complexity. Let us write $n := |V|, m_t := |A_t|$ and $\omega := |\Omega|$ for any time t. On each time $t \in \{t_0 \in \Omega \mid t_0 \geq t_s\}$, we first look up the connected components of G_t, which requires at most $O(n + m_t)$ operations. On each component C of G_t, we run an all-pairs distances method, which makes at most $O(n^2 + |C|m_t)$ operations. Observe that the $O(n^2)$ factor is for the initialization of a $V \times V$ array, which can be done once for the whole link stream. In order to see if a node has been reached at time t, it suffices at all previous times i to write distance d_{uv} in $d[u, v]$ as (i, d_{uv}).

For each node $v \in V$ and starting time s_u, the list in $\mathrm{R}_v[-s_u]$ contains at most ω elements since there can be at most as many pairs in $\mathrm{R}_v[-s_u]$ as there are arrival times on v. The same goes for the number of keys in R_v.

We only need one distance in $\mathrm{R}_v[-s_u]$, thus one distance per node, and D can be constructed with at most $O(|C|)$ operations for all $v \in C$ and connected component C of G_t, at any time t. Inserting and removing an element from $\mathrm{R}_v[-s_u]$ takes at most $O(\log \omega)$ operations. The costliest operation is

finding the value of $d_{\min}$, which involves searching in the set U of size $|C|$. This value is recomputed for every source $u \in C$, destination $w \in C$ and the set $U \subseteq C$ changes with every iteration. The **for** loop on line 14 will make at most $O\Big(|C|^2(|C| + \log|\omega|)\Big)$.

The **for** loop over G_t will make at most

$$\begin{aligned} &O\Big(n^2\Big) + \sum_{C \subseteq V} O\Big(|C|m_t + |C|^2(|C| + \log\omega)\Big) \\ &\qquad \leq O\Big(nm_t + n^2 \log\omega + n^3\Big). \end{aligned}$$

Recall that $\sum_{t\in\Omega} m_t = |A_\Omega|$. Summing over all times in Ω, we deduce a complexity of $O(nm_\Omega + n^2\omega\log\omega + n^3\omega)$. The final complexity follows from the fact that $A_\Omega \subseteq \Omega \times V \times V$. □

The temporal complexity of computing the distances between every node in every snapshot G_t, at every time $t \in \Omega$ is $O\Big(|V|^2 + |V||A_\Omega|\Big) \subset O\Big(|V|^3|\Omega|\Big)$. Thus, it is not asymptotically more efficient to compute all graph distances on all snapshots once before starting the **for** loop of line 3 to return the same output as Algorithm 2.

Computing the minimum distance between all pairs of nodes in a connected component C, while accounting for the outer distances leads to the factor of $|V|^3$. We do not expect this can be much reduced. Indeed, in the worst case, every snapshot G_t is connected and all nodes of V are reachable from the source from different starting times. Thus, $|V|$ link stream distances would have to be updated. Following Equation (7.1), in order to update the distance from the source to any node w, we need to look at the distances from the source to any reachable node of G_t as well as the distance from those nodes to w. Because G_t is undirected, in the worst case this amounts to running a single-source shortest journey method on G_t from w, which would require at least $\Omega\left(|A_t|\right)$ operations. Doing this $|V|$ times on each induced graphs leads to a lower bound of $\Omega\left(|V||A_\Omega|\right)$ on the temporal complexity of computing all *sf*-metrics in a link stream from a single source.

Remark 7.7. *The dictionary* R_v *might hold multiple triples* (s, a, d) *with the same starting time* s*, that is* $\mathrm{R}_v[s]$ *can grow linearly with* $|\Omega|$*. This is the simplest implementation of this dictionary that contains exactly all information. Since we only query for the largest starting time and the smallest distance (given that starting time) in Algorithm 2, a more convenient way to implement* R_v *is with max and min heaps. Thus,* R_v *can be a max-heap while*

$\mathrm{R}_v[s_v]$, *for any key* s_v, *would be a min-heap of elements* $(d_v, [a_v^1, a_v^2])$, *sorted on the distance* d_v. *Here,* a_v^1 *is the first known arrival time on* v *from* (s_v, s) *with distance* d_v, *while* a_v^2 *is the last one. This defines the interval* $[a_v^1, a_v^2]$ *during which shortest journeys of length* d_v *are known which summarizes with one pair* $|\Omega|$ *amount of information. This would remove the logarithmic factor from the complexity of Algorithm 2, which in our experiments leads to significant improvement. However, since the data structure is different, accessors to* R_v *would have to be modified. Thus, if one specifically needs to know exactly when a journey of length* d_v *arrives on* v *from* (s_v, s), *then this structure would be inadequate.*

Corollary 7.8. *Algorithm 2 can be implemented with heaps to make at most* $O\big(|V|^3|\Omega|\big)$ *operations.*

Proof. This follows from Proposition 7.6 and Remark 7.7. □

We use the sets V, A_Ω and Ω as parameters to evaluate the temporal complexities of our algorithms. These appear as natural choices since Ω indicates how the temporal dimension affects the number of operations while A_Ω is a surrogate for A, which is in general uncountable.

7.2.3 A multiple-sources *sf*-metrics method

Let L be a link stream and a be the first event time of L. Algorithm 3 returns a set of dictionaries of *sf*-metrics D_{uv} for each pair of nodes $(u, v) \in V^2$ of dictionary $D_{uv}[s_{uv}] = (a_{uv}, d_{uv})$ such that $l\left((s_{uv}, u), (a_{uv}, v)\right) = a_{uv} - s_{uv}$ and $d((s_{uv}, u), (a_{uv}, v)) = d_{uv}$. That is, $d_{uv} = m_{sf}\left((s_{uv}, u), (a_{uv}, v)\right)$. During its execution, it updates a dictionary D^0 such that $D_{uv}[t] = (a_{uv}, d_{uv})$, $t \in \mathrm{R}_v$ and $(a_{uv}, d_{uv}) \in \mathrm{R}_v[t]$ where R_v is computed from the source temporal node (a, u). This dictionary helps in computing D and in constructing R_v from any source. It also returns a set of dictionaries F_{uv} of latencies.

Proposition 7.9. *Algorithm 3 returns the latencies, sf-metrics and dictionaries* R_v, *from every source, between all pairs of nodes in at most* $O\big(|V|^3|\Omega|\log|\Omega|\big)$ *operations.*

Proof of correctness. Let us show that $D_{uv}^0[t_v]$ holds correct reachability triples from (a, u) to every reachable temporal node (t_v, v) for any two nodes u, v and a fixed time t_v. We also show that SA_{uv} contains pairs (s, t) such that (s, t, d) is a reachability triples from (a, u) to v for some d. We show this by induction on $\Delta := |\{t \in \Omega \,|\, a \leq t \leq t_v\}|$.

- Suppose $\Delta = 1$ and let u, v be two nodes. Then either u and v are in the same connected component C of G_{t_v} or not. In both cases, the result follows.

- Suppose the result holds for every $\Delta \leq k$ and let $\Delta = k + 1$. Let $(t_1, \ldots, t_{\Delta-1})$ be the sequence of times previously iterated over and u, v be two nodes. Let C_v be the connected component containing v at time t_Δ. If $u \in C_v$, then we argue as in the first case and the result follows. Otherwise, since (t_Δ, v) is reachable from (a, u), then by the induction hypothesis there must exist a largest starting time s_v from u to $(t_{\Delta-1}, v)$ that can be found in SA_{uw}, for some $w \in C_v$ since each node $w \in C_v$ is connected to v. Also by the induction hypothesis, $D^0_{uw}[t_{\Delta-1}]$ holds reachability triples for every $w \in C_v$. There remains to compute the distance from (s_v, u) to (t_Δ, v) to obtain a reachability triple (s_v, t_Δ, d_v) from (a, u) to v. We argue as in the proof of Algorithm 2 that Algorithm 3 returns this distance d_v. The update $D_{uv}[t_\Delta][s^*] \leftarrow d^*$ and SA_{uv} follows the same reasoning.

□

Proof of complexity. Again, let $n := |V|, m_t := |E_t|$ and $\omega := |\Omega|$ for every time t. The costliest operations occur in the **for** loop that starts on line 11. For any node u and v, the dictionary SA_{uv} can be implemented such that for any pair of elements $(s, a) \in SA_{uv}$, $a \in SA_{uv}[s]$. Then, there are at most ω keys in SA_{uv} and ω keys in $SA_{uv}[s]$. It follows that at most $O(\log \omega)$ operations are required to insert and search in this dictionary. Finding the largest starting time s_v on line 13 requires in the worst case $O(|C_v| \log \omega)$ operations. Similarly, dictionary $D^0_{uv}[t]$, for any $t \in \Omega$, has a size at most ω^2 and the loop over C_v (on line 16) to find $d_{\min}$ requires at most $O(|C_v| \log \omega)$ operations. The **for** loop on line 11 thus makes at most:

$$\begin{aligned}\sum_{u \in C} \sum_{v \in V \setminus C} O(|C_v| \log \omega) &\leq \sum_{u \in C} \sum_{v \in V} O(|C_v| \log \omega) \\ &\leq \sum_{u \in C} O\Big(n^2 \log \omega\Big)\end{aligned}$$

operations. This loop is itself repeated for all connected components $C \subseteq V$,

which in turn yields:

$$\sum_{C \subseteq V} \sum_{u \in C} O\Big(n^2 \log \omega\Big) = \sum_{u \in V} O\Big(n^2 \log \omega\Big)$$

operations. Thus, this method should make at most $O(n^2 + nm_t)+O(n^3 \log \omega)$ operations in the worst case on each time t, with the first term accounting for the all-pairs distances method on G_t. This number of operations is repeated at most ω times. Observe again that $O(nm_\Omega) \subset O(n^3\omega)$ and the result follows. □

Observe that Algorithm 2 needs to be called $|V|$ times in order to output the lengths of all shortest fastest journeys from any source to any destination, since it discovers all starting times from each source. The multiple-sources algorithm is empirically faster when the desired output is the set of *sf*-metrics from all sources to all destinations (see Section 7.3). The temporal complexity of both methods are affected mostly by the computations on the snapshots. In Subsection 7.2.4, we will see how to drastically speed up those algorithms when using γ-journeys with $\gamma > 0$ since in that case we can remove the dependency on the all-pairs distances methods on the snapshots. The temporal complexity of Algorithm 3 could also be reduced sligthly with the use of heaps, however this would make accessing information more difficult.

7.2.4 Shortest journeys with delays

In the literature on temporal walks, it is commonly assumed that a journey has a (transmission) *delay* γ that is the time it takes to cross a temporal edge. When every temporal edge of a journey has the same delay γ, we call the journey a γ-*journey*. This is the case with Wu et al. whose algorithm we wish to compare Algorithm 2 against since their shortest journey procedure is the most efficient known in temporal networks. Transmission delays are natural when modeling the spread of information and the time required for spreading is commensurable with the time interval during which the network is observed.

A γ-journey in a link stream is a journey $(t_1, u_1u_2), \ldots, (t_n, u_nu_{n+1})$ such that $t_i \geq t_{i-1} + \gamma$ for all $1 < i \leq n$ and some $\gamma \in {\mathbb{R}_+}^2$. We call γ the *delay* and note that a journey corresponds to a 0-journey. A simple γ-journey is shown on Figure 7.3 from $(0, d)$ to $(2, a)$ with $\gamma = \frac{1}{2}$. Note that with this

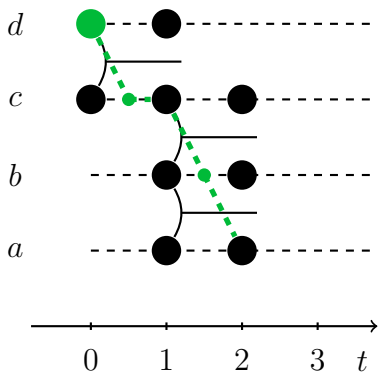


Figure 7.3: A γ-journey (in green ▬) with $\gamma = \frac{1}{2}$ from $(0, d)$ to $(2, a)$.

choice of γ, the subjourney from $(1, c)$ to $(2, a)$ is the unique journey from the former to the latter temporal node.

When $\gamma > 0$, it is not necessary to iterate over the connected components of the snapshots of a link stream, since all nodes of a component are not connected to each other via a γ-journey, and we can simplify Algorithm 2 and Algorithm 3 in order to reduce the number of operations they perform. Thus, we present Algorithm 4 and Algorithm 5 that are deduced from Algorithm 2 and Algorithm 3 and assume $\gamma > 0$. Their correctness and temporal complexities follow from the same arguments used in Proposition 7.6 and Proposition 7.9.

When journeys have delays, we must ensure a temporal edge appears long enough to be crossed given that delay.

Proposition 7.10. *When $\gamma > 0$, Algorithm 4 computes the latencies and sf-metrics from a source event node to all reachable event nodes as well as the set of dictionaries R_v, for all $v \in V$, in at most $O(|V| + |A_\Omega| \log|\Omega|)$ operations.*

Proof. Correctness follows from the same reasoning as in Proposition 7.6. For the complexity, we first initialize a dictionary of $|V|$ elements, then we iterate over all temporal edges of A_Ω and perform logarithmic searches on lists that grow with $|\Omega|$. □

[2]More generally, we could let $\gamma : A \to \mathbb{R}_+$ be a function of the temporal edge to traverse. This was not considered here, but our methods could be extended for this case.

Note that, by the same argument as for Algorithm 2, the complexity can be lowered to $O(|V| + |A_\Omega|)$ by using a combination of max and min heaps.

Finally, in Algorithm 4, the dictionaries d and f are implemented such that the keys are nodes and values are pairs (t, k) such that t is the time value k is computed at that node. For example, if $(t, f_v) \in f[v]$, then the latency from the source to (t, v) is f_v. This enables us to sort dictionaries by time.

We also extended Algorithm 3 to the case of $\gamma > 0$ and devised Algorithm 5. This method is also guaranteed by the proof of Algorithm 3.

Proposition 7.11. *When $\gamma > 0$, Algorithm 5 returns the latencies, sf-metrics and dictionaries R_v between all pairs of nodes in at most $O\Big(|V|^2 + |V||A_\Omega| \log|\Omega|\Big)$ operations.*

Proof. Correctness follows from Proposition 7.9. For complexity, initialization alone of the dictionaries requires $O\Big(|V|^2\Big)$ operations. Let (t, uv) be a temporal edge of A_Ω. Then, searching in the dictionaries for each values takes at most $O(\log|\Omega|)$ operations when they are implemented as balanced binary trees. This is repeated for all nodes $w \in V \setminus \{v\}$ and all temporal edges $(t, uv) \in A_\Omega$. Thus, this algorithm makes at most $O\Big(|V|^2 + |V||A_\Omega| \log|\Omega|\Big)$ operations. □

7.3 Experiments

We present some experiments to highlight the running times of our algorithms. In the first one, we compare Algorithm 4 with the single-source shortest journey method from Wu et al. Algorithm 4 (SSMD_γ) acts as a surrogate for Algorithm 2 (SSMD) since Wu et al. designed their method to work on journeys with strictly positive delays. Moreover, these authors evaluated their method from a small set of source nodes on large datasets and we follow the same procedure. In a second experiment, we compared the running times of our two methods for null delays (0-journeys) on synthetic link streams. We also compared Algorithm 3 (MSMD) and Algorithm 5 (MSMD_γ) on synthetic link streams. Finally, we compared Algorithm 4 and Algorithm 5 on the same task on some datasets.

Algorithm 3 was inspired by the fastest journeys method of Bui-Xuan et al. [44] that returns durations of fastest and foremost journeys. Comparing the two methods would be unfair against ours.

Remark 7.12 (Experimental setup)**.** *All experiments were run on a single machine with* 2.6 *GHz Intel Core i7 processor and* 16 *Gb of RAM. All methods were implemented in C++ with standard libraries, including Wu et al.'s method. We implemented standard approaches to compute connected components and all pairs distances in graphs. The full code can be found online [190].*

7.3.1 Runtime comparison with the literature

Let us compare how Algorithm 4 (SSMD_γ) fares against Wu et al.'s shortest journey algorithm. This is our comparison with the literature.

Wu et al. analyzed their method with the framework of temporal graphs. We translate their temporal complexity with link stream parameters, upper bounding M with $|A_\Omega|$ and $d_{\max}$ with $|\Omega|$. Thus, the shortest journey algorithm of Wu et al. makes at most $O(|V| + |A_\Omega| \log|\Omega|)$ operations in the worst case, which is the same temporal complexity as SSMD_γ.

We ran experiments on link streams of various sizes, as measured with $|V|$, $|\Omega|$ and $|A_\Omega|$. We used the same datasets as Wu et al. and added some, randomly chose 200 different source nodes from each and ran both methods one after the other. The full results (in seconds) can be found in Table 7.1 on Page 147. We extracted some parameters from the datasets used, they are shown in Table 7.2. The running times of Wu et al.'s method are comparable to those of SSMD_γ. Our method does more operations, since it must compute latencies as well and ensure the distances correspond to the *sf*-metrics, so this is encouraging and unexpected. All datasets are heterogenous, which explains the variability in running times and we have not yet pinpointed any hidden link stream parameter[3] that would precisely explain this variability, including the measure used for the visualizations of Figure 7.4.

Remark 7.13 (Notes on the datasets used)**.** *The datasets are only used as benchmarks. They all describe discrete temporal networks and can be found as part of the* KONECT *library of networks [142]. Only the values of the parameters $|V|, |\Omega|$ and $|A_\Omega|$ were extracted in Table 7.2 since only these were required for our experiments.*

A short description of the datasets used follows in Appendix B. Figure 7.4 shows the temporal evolution of some network measures on two of the datasets used: Arxiv-Hepph and Wikiconflict. On each, we sampled 0.1%

[3]Such as $|V|, |\Omega|, |A_\Omega|$ and the density of each snapshots.

of the dataset to build a sublink stream. On each sublink stream and on each snapshot G_t we evaluated the density *of G_t and normalized the values to $[0, 1]$. The density of an undirected graph $G = (V, E)$ is given by the ratio $\frac{|E|}{\binom{|V|}{2}}$. We chose this measure as an indicator of the temporal evolution of each dataset and it shows how varied the datasets are. Even though in both cases we observe strong fluctuations in the density, in the Wikiconflict dataset the density appears stratified. Meanwhile, in the Arxiv-Hepph dataset this density is mostly low and takes more distinct values in the interval $[0, 0.4]$, which hints that its connectivity varies more than in Wikiconflict. The Wikiconflict dataset therefore seems to have groups of temporal edges of similar sizes appear at regular times, while in Arxiv-Hepph there is a less obvious pattern in the appearance times of temporal edges.*

Remark 7.14 (Further implementation details)**.** *Previous results [189] showed* SSMD_γ *to be unstable on some datasets compared with Wu et al. This can be explained by the data structure used to implement each* R_v*. Note that* $\mathrm{R}_v[s_v]$*, for any node v and starting time s_v, can grow as $O(|\Omega|)$ which can be large. Thus, even logarithmic search in this structure can be costly. Notice also that we only ever need the largest key of* R_v *and the smallest distance of* $\mathrm{R}_v[s_v]$*. Thus, we reimplemented each dictionary* R_v *with (max/min)-heaps as explained in Remark 7.7. This provides constant time access to the largest/smallest element. This is a choice of implementation to increase the performance of this method and all results presented here that specifically test the performance of* SSMD_γ *used that implementation. The new method is considerably faster and is comparable with Wu et al.'s method. We did not reimplement the latter's method using heaps because at each step they remove* dominated *elements which naturally tends to make their structures smaller.*

Table 7.3 shows statistics on the running times and ratios between Algorithm 4 and Wu et al. Observe that even with large running times, the ratios $\left(\frac{\mathrm{SSMD}_\gamma}{\text{Wu et al.}}\right)$ are still decent.

7.3.2 Comparison between algorithms SSMD and MSMD

Algorithms SSMD and MSMD were run on a set of randomly generated link streams of size $|V|$ ranging from 100 to 170, with increments of 10. Although the link streams are small, the running times are significant since we compute the distances from every source to every destination. The link streams were

constructed by generating Erdös-Renyi graphs $G(n, p)$, with $n = |V|$ and $p = 0.7$. For each edge uv drawn from $G(n, p)$, we drew a time instant $t \in \{0, 1, \ldots, 7\}$ uniformly at random and added the temporal edge (t, uv) to A. In this case, temporal edges have no duration and the time instants are integers: this helps ensure the size of Ω is fixed and small, so the running times scale only with $|V|$ and $|A_\Omega|$.

Figure 7.5(a) shows the running times of each algorithms on a link stream with a fixed number of nodes. We observe that, as the number of nodes increases, the amount of time taken by SSMD grows faster than that of MSMD. This gives clear indication that this latter method is faster than the former. In terms of scale, the MSMD method manages a link stream of 170 nodes and about 20 000 temporal edges in less than 3 seconds. Its counterpart takes more than 25 minutes for the same calculations.

Since MSMD is more scalable than SSMD, we generated a new set of link streams, again with the same process as before, with time instants drawn uniformly at random in the set $\{0, \ldots, 10\}$ while the duration of a temporal edge (t, uv) is drawn uniformly at random in the subset $\{0, \ldots, 10-t\}$. In that case, the size of Ω does not vary much. We let $|V| \in \{10, 20, \ldots, 190, 200\}$. The results are summarized in Figure 7.5(b). We fitted, with the statistical software R [175], a linear model on the runtime of MSMD as function of $|A_\Omega|$ in order to extrapolate the runtime of this method for larger values of link stream parameters. Since the number of nodes and event times are low compared to the number of temporal edges, the trend is linear in the number of temporal edges. The fit is reasonable and this is sufficient to illustrate the scaling trend. Extrapolating, we obtain the values shown with the blue line. The trend does not suggest the method is at this point scalable to big link streams as in general the sizes of V and Ω would also grow to be much larger than in this experiment.

7.3.3 MSMD against MSMD_γ on synthetic link streams with varying densities

We ran algorithms MSMD and MSMD_γ side-by-side on synthetic link streams to see how well the latter performs against the former. It is expected that MSMD_γ would be faster. However, instead of looking at the trend on link streams with varying number of nodes, we investigated this trend against the density of the stream. Thus, we constructed link streams as before from

graphs $G(n,p)$, while varying the parameter p instead of n. The density of the resulting link stream varies with p. The number of nodes is fixed to 200 and temporal edges have positive integer durations, to allow for durations while keeping the cost of computation low.

Experimentally, the density of each snapshot of the link stream affects the performance of MSMD_γ against MSMD. This can be seen in Figure 7.6: as the value of p increases, after a threshold around $p \approx 0.63$, MSMD_γ takes longer to complete its task than MSMD. This suggests that when many temporal edges appear at the same time, it becomes advantageous to compute all-pairs distances in the snapshots and reuse them on each connected components. Note, that these do not solve exactly the same problem since one works with $\gamma > 0$ while the other does not. However, if the choice of γ is a modeling parameter[4], this can be interesting to take into account. It is not clear why the running time of MSMD is decreasing for low probabilities, $p \in (0, 0.25)$. Recall that the connectivity threshold of $G(n,p)$ is $\frac{\log n}{n}$. Evaluating this adjusted threshold in the form of $\frac{\log |V||\Omega|}{|V||\Omega|} \approx 0.035$ does not explain why the two lines intersect.

Remark 7.15. *In link streams, whether temporal edges have durations or not and whether those durations are integers or real numbers can slow down our methods as durations influence the sizes of Ω and A_Ω. Running times with real temporal edge durations can sometimes be prohibitive, so we focused our experiments on integer durations.*

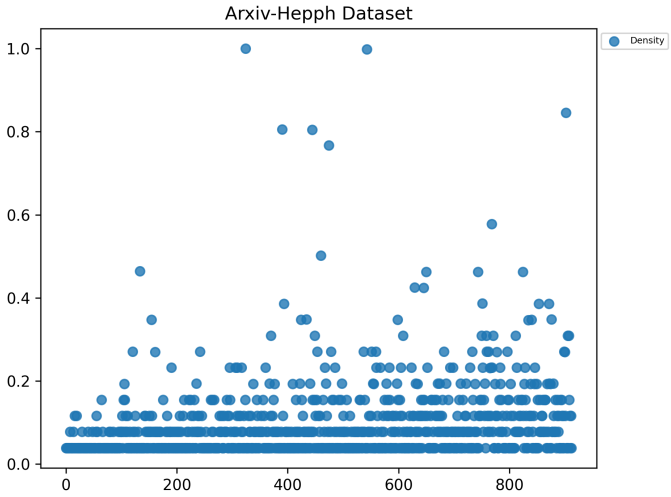
7.3.4 Comparing algorithms SSMD_γ and MSMD_γ on real datasets

Algorithm MSMD_γ can terminate in reasonable time on some datasets, as opposed to MSMD. In order to probe this method further, we tested how long it would take for this method to terminate on some datasets against SSMD_γ.

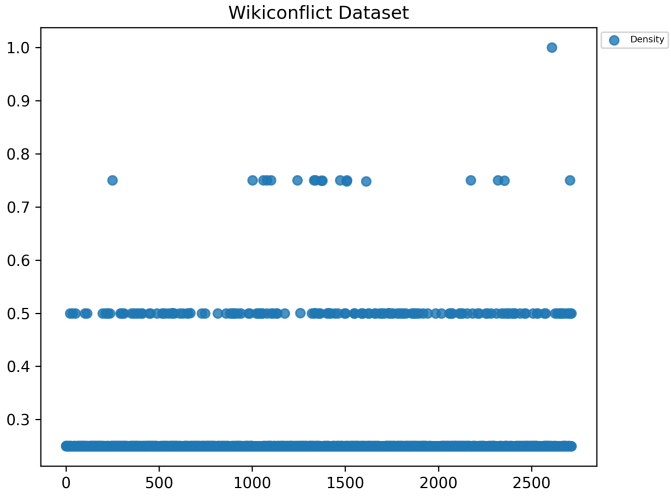
We compared both methods on selected datasets on which SSMD_γ took less than 10 seconds to finish (as observed in Table 7.1). This way, we could expect it to finish in a decent amount of time from all sources. We ran SSMD_γ from all sources alongside MSMD_γ. For the same task, it is faster to run the

[4]The choice of whether γ should be positive or null depends on the application. For example, if one models the verbal interactions in minutes between individuals that meet face-to-face, then the transmission delay γ between any two individuals is negligeable.

specialized method MSMD_γ. However, note that both methods could finish in a short amount of time on real-world datasets, which is positive. Statistics on the running times of both methods are summarized in Table 7.4.



(a) Scatterplot of values from the Arxiv-Hepph dataset



(b) Scatterplot of values from the Wikiconflict dataset

Figure 7.4: Visualization of the temporal evolution of the density on sublink streams of two datasets, Arxiv-Hepph and Wikiconflict. Each sublink stream was constructed by sampling 0.1% of the dataset. We evaluated the density of each snapshot on each sublink stream. Values are normalized to $[0, 1]$.

Dataset	Wu et al.	$SSMD_\gamma$	Runtime ratio
facebook-wosn	16.40	12.90	0.8
contact	1.00	1.16	1.2
lkml person	54.20	18.80	0.3
delicious ut	5190.00	8740.00	1.7
movielens	2.49	1.20	0.5
dnc	1.12	1.35	1.2
enron	23.10	22.40	1.0
munmun twitter	174.00	196.00	1.1
lastfm band	103.00	124.00	1.2
elec	3.77	3.43	0.9
epinions-rating	137.00	75.60	0.6
flickr-growth	1400.00	2180.00	1.6
dblp	1.18	0.36	0.3
sociopatterns hyper	0.28	0.25	0.9
digg	3.67	1.62	0.4
prosper loans	33.10	27.30	0.8
sociopatterns infect	0.47	0.37	0.8
delicious ui	6950.00	2200.00	0.3
mit	14.30	13.20	0.9
wikiconflict	37.70	40.80	1.1
slashdot-threads	9.16	6.18	0.7
lastfm song	197.00	91.20	0.5
arxiv-hepph	56.70	57.90	1.0
youtube-growth	518.00	299.00	0.6

Table 7.1: Runtime comparisons (in seconds) between $SSMD_\gamma$ and Wu et al.'s method. Datasets with runtime less than 10 seconds are in bold.

Dataset	$\lvert V\rvert$	$\lvert\Omega\rvert$	$\lvert A_\Omega\rvert$
facebook-wosn	63731	204914	817035
contact	274	15662	28244
lkml person	337509	624757	1565683
delicious ut	4512099	1583	301186579
movielens	16528	34535	95580
dnc	1891	10176	39264
enron	87273	178721	1148072
munmun twitter	530418	175218	4664605
lastfm band	174077	1058994	19150868
elec	7118	90741	103675
epinions-rating	755760	501	13668320
flickr-growth	2302925	134	33140017
dblp	12590	30	49759
sociopatterns hyper	113	973	20818
digg	30398	9125	87627
prosper loans	89269	1259	3394979
sociopatterns infect	410	223	17298
delicious ui	25221771	1583	301186579
mit	96	33452	1086404
wikiconflict	116836	215982	2917785
slashdot-threads	51083	67327	140778
lastfm song	1084620	1058994	19150868
arxiv-hepph	28093	2337	4596803
youtube-growth	3223585	203	9375374

Table 7.2: Values of the parameters extracted from the datsets used in the comparison between SSMD_γ and Wu et al.'s method. Datasets with runtime less than 10 seconds in Table 7.1 are in bold.

Method	Min	1st Q.	Median	Mean	3rd Q.	Max
$SSMD_\gamma$ (s)	0.25	1.55	20.6	588.0	99.4	8740
Wu et al. (s)	0.28	3.38	28.1	622.0	146.0	6950
Ratios	0.31	0.53	0.85	0.84	1.09	1.68

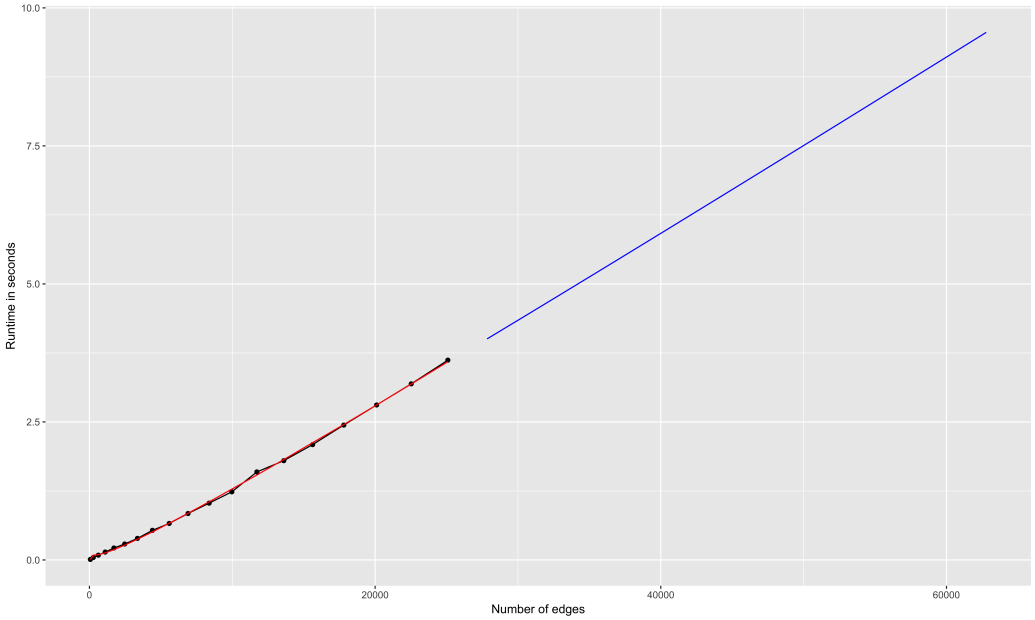
Table 7.3: Summary statistics of running times (in seconds) between algorithms $SSMD_\gamma$ and [208], with Q. standing for quartile. The ratio goes above 1 only at about the 3rd quartile. Outliers still exist.

Method	Min	1st Q.	Median	Mean	3rd Q.	Max
$SSMD_\gamma$ (s)	1.10	2.89	3.77	48.80	71.0	216.0
$MSMD_\gamma$ (s)	0.30	2.16	3.97	12.90	24.6	27.6

Table 7.4: Summary statistics of the running times (in seconds) between algorithms $SSMD_\gamma$ and $MSMD_\gamma$. In practice, $SSMD_\gamma$ is faster on some datasets than its counterpart. However, statistically $MSMD_\gamma$ is better suited to compute all metrics.

$\|V\|$	$\|A_\Omega\|$	MSMD (s)	SSMD (s)
100	6814	0.77	209.97
110	8284	0.98	341.77
120	9908	1.15	444.65
130	11654	1.30	555.58
140	13554	1.58	785.46
150	15564	1.77	952.33
160	17744	2.09	1371.26
170	20060	2.43	1586.48

(a) Comparisons between algorithms SSMD and MSMD. Temporal edges have zero duration. Number of event times $|\Omega|$ is 11, $p = 0.7$ and times are in seconds.



(b) Runtimes of MSMD in seconds as a function of A_Ω. Temporal edges have positive integer durations. The black dots and line are results, red line is statistically fitted and blue line is the prediction. Since $|V|$ and $|\Omega|$ are kept relatively low, the running times increase linearly with $|A_\Omega|$.

Figure 7.5: Runtimes (in seconds) of algorithms SSMD and MSMD on synthetic link streams. Link streams are generated randomly with fixed seed.

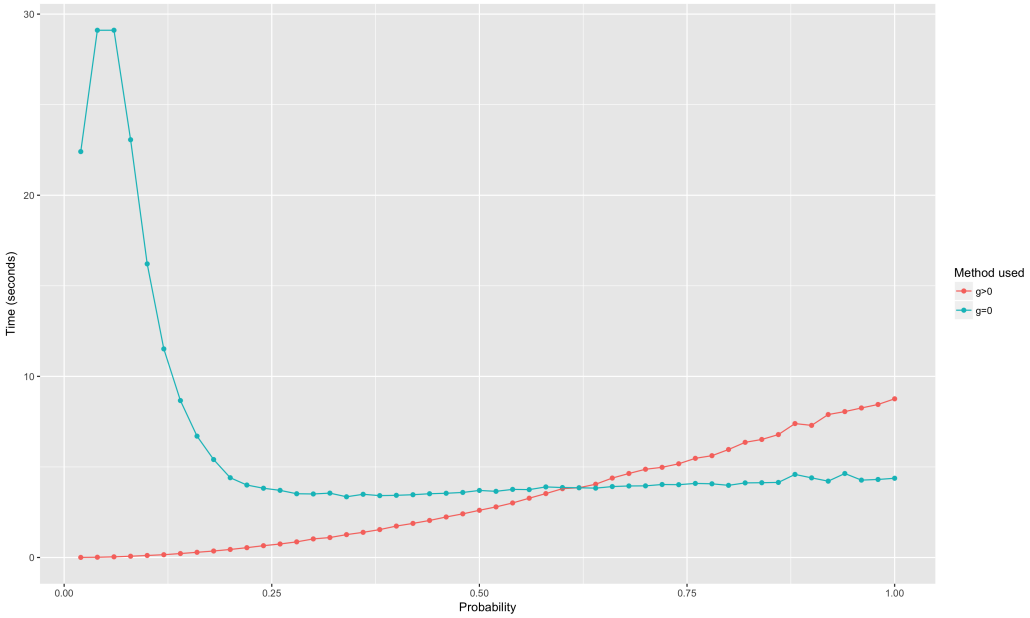


Figure 7.6: Runing time (in seconds) of MSMD_γ (in red ▬) and MSMD (in blue ▬) on synthetic link streams generated with $G(200, p)$ with varying values of p. As p increases, so does the density of every snapshots, favouring MSMD over MSMD_γ. Temporal edges have positive integer duration.

Chapter 8

Conclusion

This work is about properties of temporal networks, with a focus on the game of Cops and Robber that takes place over periodic graphs. As such, we think this work is interesting given the young literature on temporal networks and the few studies that use classical graph theoretical approaches on those structures.

8.1 Cops and Robber Game on Periodic Graphs

8.1.1 Summary

We showed that allowing a graph to change over time generates new challenges when studying the game of Cops and Robber. We presented many results to inform researchers of those challenges. We also exhibited results on periodic graphs that generalize their counterparts on static graphs, which shows what type of extra assumptions can be used to extend them.

In Chapter 4, we presented a novel characterization of copwin periodic graphs that works on the original structure, when represented as an *arena*. This provides a faster algorithm to determine if a periodic graph is copwin, as well as a more explanable output. Researchers are often interested in why a graph might not be copwin. Theorem 4.3 and Theorem 4.6 provide a similar justification for periodic graphs: if an augmented arena $\mathcal{A}$ of an arena $\mathcal{D}$ has no shadow corner, yet has no anchored star, then $\mathcal{D}$ cannot be copwin.

In Chapter 5 and Chapter 6 we provided both examples and general results on the cop numbers of periodic graphs. Both chapters go in tandem

and results in Chapter 5 helped guide our results in Chapter 6. In Chapter 5, one major result is Table 5.1 of (a, b, c)-copwin existence results and one might naturally want to extend this to larger values. We mentioned in the introduction that Theorem 6.4 is a major result of Chapter 6. We also showed some lower bounds on $c_{\circledcirc}(G)$, notably Proposition 6.9 and Theorem 6.2. We think these results might become valuable when studying larger periodic graphs by building larger periodic graphs from smaller ones.

8.1.2 Future Work

This work is a first foray into the analysis of the game of Cops and Robber on Periodic Graphs. One major avenue of research this opens up is the determination of cop numbers of specific classes of *periodic graphs*. This is common on undirected graphs. Recall for example that we know the cop numbers of planar graphs [6], outerplanar graphs [60] and hypercubes [153]. We gave partial answers to this question by focusing on $c_{\circledcirc}(G)$: then, for example, every periodic graph with footprint G has cop number at most $\mathrm{tw}(G) + 1$ (Theorem 6.4).

In order to extend results such as Theorem 6.4 to classes of periodic graphs that are not purely defined by their footprints, one would have to come up with interesting classes of periodic graphs. This is a major hurdle we faced, the literature so far is sparse on classes of periodic graphs. Moreover, those classes that have been defined, such as those presented by Casteigts et al. [52], are often not well suited to the game of Cops and Robber so it is difficult to compute their cop numbers.

The class of periodic graphs with footprint G, which we investigated when computing $c_{\circledcirc}(G)$, is nevertheless vast and interesting. One confounding aspect of this class of periodic graphs is that, although a priori it looks like computing $c_{\circledcirc}(G)$ is akin to playing the game with imperfect information, whenever an element of this family is chosen the cops will be fully aware of it. This is similar to computing the maximum cop number of a family of random graphs: the structure is only known a posteriori. Furthermore, the value of $c_{\circledcirc}(G)$ says something about the nature of G. The fact that when G is outerplanar we have $c_{\circledcirc}(G) - c(G) \leq 1$ means that outerplanar graphs have particularly strong separation properties. To emphasize: when G is outerplanar, no matter what periodic sequence of subgraphs of G one takes, 3 cops can always capture the robber. This is in line with the type of strategies we used to prove that $c_{\circledcirc}(G) \leq 3$. One might think of those strategies as

stubborn: they tell the cops where to go on G and the cops make their moves as their incident edges become available. The 3-copwin strategy on planar graphs, recall [6], are *dynamic* because they involve moving the cops, possibly at every turn, to guard isometric paths. We suspect that those strategies do not resist to the changing nature of periodic graphs. That is, it is likely that $c_{\circledcirc}(G) - c(G)$ grows larger with n when G is planar, although we still have no proof of this. This suggests the following line of reasoning. So far, when computing $c_{\circledcirc}(G)$ we have been looking at properties of G to understand properties of the class of periodic graphs with footprint G. However, one could also seek to understand properties of G from properties of the periodic graphs with footprint G. For example, if it turns out that $c_{\circledcirc}(G)$ is not bounded by a constant when G is planar, then one might conclude that the *only* winning strategies for the cops on G are dynamic. One could wonder what extra temporal assumptions on a class of periodic graph $\mathbb{G}_{\circledcirc}$ with planar footprints would be necessary so that $c(\mathcal{G})$ is bounded by a constant for every $\mathcal{G} \in \mathbb{G}_{\circledcirc}$. In the static case, placing a cop on a cutvertex always prevents the robber from moving from one connected component to another. We suggest extending this idea via "temporal cuts", which would require assumptions on how they are connected to each other over time.

Following Theorem 6.4, we can naturally wonder if it also holds that $c_{\circledcirc}(G) \geq \mathrm{tw}(G) + 1$. Since this looks difficult to prove, we conjecture that a weaker version of this statement holds. Let $\mathbb{G}$ be an infinite class of finite, connected graphs and $\mathbb{G}_{\circledcirc}$ be the infinite class of periodic graphs whose footprints belong to $\mathbb{G}$.

Conjecture 8.1. *$\mathbb{G}_{\circledcirc}$ has bounded cop number if and only if $\mathbb{G}$ has bounded treewidth.*

The connection between the cop number and the treewidth is also observed in some other cops and robber games such as the cops and fast robber [159] where, for example, $\frac{\mathrm{tw}(G)+1}{\Delta(G)+1} \leq c_\infty(G) \leq \mathrm{tw}(G) + 1$ for every graph G. We discussed the lack of adequate definition of "temporal treewidth" on temporal graphs in Subsection 2.1.4. Fluschnik et al. [88] surveyed possible candidates for such a definition and concluded that it might be more adequate to study "temporal treewidths" via cops and robber games, since some known width measures correspond to cop numbers, such as the original treewidth [184] and the DAG-width [31]. Thus, one might want to take inspiration from our work to carefully design a game of cops and robber that would highlight the "tree-like" structure of a temporal graph.

The discussion around Proposition 5.2 also leads one to wonder what would be the size of the smallest 3-copwin periodic graph and, possibly, the smallest 4-copwin one. Both Baird et al. [17] and Turcotte and Yvon [202] supplemented their theoretical results with algorithms to prove the order of the smallest 3-copwin and 4-copwin graphs, respectively. Thus, in order to determine if Proposition 5.2 presents the smallest 3-copwin periodic graph, one might want to use a computer search. This poses the problem of how to enumerate all periodic graphs with a specific footprint, if possible. Empty snapshots do not affect the cop number of a periodic graph, nor does, in some cases, changing the order of some snapshots. Thus it might be possible to extract a "compact" list of periodic graphs on a fixed footprint. This could also help with extending Table 5.1.

In Section 5.4, we explained that we can construct an at most k'-copwin periodic graph whose footprint is k-copwin for any integers $1 \leq k' \leq k$ when the period is not bounded. We did not explain how and it is not clear how to get an exact cop number k'. More interestingly is whether this can be achieved if one restricts the period to be depend at most logarithmically on the size of the footprint. We say logarithmically, because we suspect this is impossible if one wants to keep the period constant, the same way that k-copwin graphs with n vertices do not exist for every integers k and n. As a stepping stone in this direction, we suspect that it is impossible to create a copwin periodic graph $\mathcal{G}$ with period 2 whose footprint is the Robertson graph, which is the smallest 4-copwin graph [202]. On the same topic, we have not made use of the fact that our temporal graphs are periodic in all our results. Thus, the door is open to studying how the period affects some of our results. For example, if we focus on the class of periodic graphs $\mathbb{G}^2_{\circledcirc}$ with period 2, can we give a tighter upper bound on $c_{\circledcirc}(G)$ than $\mathrm{tw}(G)+1$ when $G \in \mathbb{G}^2_{\circledcirc}$? What about the generic construction in Proposition 5.28? We did not find ways to use the period in general, so this is an interesting and nontrivial path for further research.

On another note, we showed that $c_{\circledcirc}(G) \not\leq \gamma(G)$ in general (recall Proposition 3.10). From Proposition 5.28, we suspect that it is impossible to upper bound $c_{\circledcirc}(G)$, in general, by any *polynomial* function of $\gamma(G)$ even though $\gamma(G)$ is a trivial upper bound on the cop number of many cops and robber games on G.

In light of our results, one notices that the absence of a good notion of retractions of periodic graphs has been making it difficult for us to derive stronger results. We used the theorem of Berarducci and Intrigila [27] to

the effect that $c(H) \leq c(G)$ whenever H is a retract of G a few times, for example in Proposition 5.28, in order to bound the cop numbers of G and $G_{\max}$ in a periodic graph $\mathcal{G}$. However, when it came to $\mathcal{G}$ itself, we had to come up with other arguments. We often relied on the contrapositive of Proposition 3.4 to show that $c(\mathcal{G}) > k$ for some k. Yet, in some results like in Proposition 5.2 we had to devise an ad-hoc argument to show that $c(\mathcal{G}) \geq 3$. Although one might naturally want to generalize the construction in this proof to hypercubes of higher dimensions, it is not clear how to generalize the argument we used. Thus, a result of the form: $c(f(\mathcal{G})) \leq c(\mathcal{G})$ for some function f such that $f(\mathcal{G}) \subset \mathcal{G}$ would be greatly valuable.

On the algorithmic side we are finalizing the work on an extension of Algorithm 1 for $k > 1$ cops.

A common way to study algorithmic problems on temporal networks is through time-windows, often of some fixed length Δ (see [138]). For example, for any integer $\Delta \geq 1$, we can define a Δ-clique[1] in a periodic graph $\mathcal{G}$ as a pair $(X, [b, e])$ such that $\Delta = e - b$ and the subgraph $\bigcup_{t=b}^{e} G_t[X]$ is a clique. When $\Delta = 1$, Δ-cliques are clearly copwin in $\mathcal{G}$. However, we can construct a clique with a large value of Δ whose cop number scales with the number of nodes. Thus, we can wonder about the values of Δ for which the cop number of a Δ-clique attains certain values. We could also study the family $\mathbb{G}_{\textcircled{\scriptsize t}}^{\Delta}$ of periodic graphs $\mathcal{G}$ for which $(0, \ldots, p-1)$ can be written as a disjoint union of intervals $[b, e]$ of length Δ and such that every subsequence $(G_b, \ldots, G_e)$ either contains a Δ-clique or is an independent set. One might be able to get accurate estimates on the cop numbers of $\mathbb{G}_{\textcircled{\scriptsize t}}^{\Delta}$.

Finally, recall from Subsection 2.1.5 and Subsection 2.2.1 that there exists multiple other problems of mobile agents and pursuit-evasion games. We extended one specific game to the temporal setting, which opens the door to studying many of the other problems on temporal graphs.

8.2 Link Stream

8.2.1 Summary

In Chapter 7, we presented different algorithms to compute metrics between pairs of event nodes. As opposed to similar known algorithms, those methods

[1]The term comes from Viard et al. [204] who defined this in link streams. Our definition is slightly different for simplicity.

return all metrics at once in a single pass over the dataset. Moreover, the starting and arrival times of shortest journeys are returned, which is valuable information to compute, for example, the betweenness centrality of temporal nodes.

Algorithm SSMD works from a fixed source and is suitable when not all pairwise metrics are required. Our experiments show that SSMD_γ is comparable to the state of the art method to compute distances from a source node to all other nodes. These results improve on previous ones [189] and make use of a more efficient data structure. Even though some experiments are advantageous to our methods, we do not claim they are in general faster than the state of the art. However for the task at hand of computing all metrics at once on a link stream, we think we can fairly conclude that our methods are usable in practice. The experiment in Section 8.2 was designed to illustrate if SSMD_γ could be used in real-world setting. Thus, comparing it to another shortest journey method is relevant since it does not do significantly more operations that this type of algorithms.

In practice, MSMD has finished its task faster than its counterpart on synthetic link streams. Since the link streams used were smaller than what we would expect from real-world instances, we extrapolated the running times produced by MSMD. At this point, scalability is an issue, when $\gamma = 0$, and we could not expect to run this method on realistic link streams and obtain results in a reasonable amount the time. Thus, in order to speed up the computation time, we suggest studying how to lessen the amount of operations in either methods by skipping some temporal nodes and extrapolating the distances. Also, finding ways not to have to recompute the connected components and the all-pairs distances methods at every time would be helpful in improving the efficiency of both methods.

Both algorithms SSMD_γ and MSMD_γ, when $\gamma > 0$, could finish their tasks on some datasets in a decent amount of time. Algorithm MSMD_γ took less than 30 seconds to finish all tasks, even on datasets of almost 100 000 temporal edges. Thus, in that case, realistic datasets of decent size could be handled by our methods.

8.2.2 Future Work

In light of our experiments as well as the recent literature (see for example Bentert et al. [26]), it would be interesting to have a lower bound on the complexity of computing our metrics. Since shortest journey methods

in temporal networks seem to be based on the same type of arguments as shortest path algorithms in graphs, it might also be relevant to deduce a lower bound in term of the latter. That is, how much slower is it to compute distances in a link stream as opposed to computing distances in a graph? Can this complexity be expressed in terms of natural parameters of the link stream such as V, Ω and A_Ω?

Aside from scalability, another limitation of this study lies in the ordering of the objective functions we chose to optimize. Namely, we compute lengths of shortest fastest journeys. If one were to require lengths of fastest shortest journeys, our methods would need to be redesigned. Moreover, the multitude of possible combinations of optimal journeys to compute (foremost journeys, shortest foremost journeys, etc.) is not all considered in this work. Brunelli et al. [42] tackled this limitation by building a more general framework for optimal journeys. We believe our methods can be modified to compute some other types of journeys combining temporal and structural information hierarchically, such as shortest foremost journeys. In turn, those journeys can be used to compute other centralities than the betweenness centrality or to investigate different topics such as reachability. In Algorithm 4, for instance, whenever we consider a temporal edge (t, uv) with $u \neq s$, we have access to the last arrival time a_u from the source at time s_u. If we instead considered the *first* such arrival time, then we could decide if the journey from (s_u, s) to (t, v) that involves the temporal edge (t, uv) is *restless* (see Casteigts et al. [56]) or not and update the relevant dictionaries accordingly, say of *restless* reachability triples.

This work serves, in part, as a preprocessing to compute the betweenness centrality on link streams of Latapy et al. [144], which is done in [192]. Other centrality measures exist on graphs that use shortest paths, such as the closeness (see [164]). Thus, other centrality measures can be defined on link streams, via fastest journeys, shortest journeys, shortest fastest journeys, and other notions of optimal journeys. It follows that our work could be extended to compute other centralities measures as well.

Appendices

A Tables and Algorithms

Table A.1: List of neighbourhoods of the periodic graph shown in Figure 5.5

(t, u)	$N_t[u, \mathcal{G}]$		
(0, 0)	0	7	2
(0, 1)	1	3	4
(0, 2)	2	0	6
(0, 3)	3	5	1
(0, 4)	4	1	8
(0, 5)	5	6	3
(0, 6)	6	2	5
(0, 7)	7	0	
(0, 8)	8	4	
(1, 0)	0	3	1
(1, 1)	1	0	5
(1, 2)	2	8	
(1, 3)	3	6	0
(1, 4)	4	6	
(1, 5)	5	1	7
(1, 6)	6	4	3
(1, 7)	7	5	8
(1, 8)	8	7	2
(2, 0)	0	6	7
(2, 1)	1	2	5
(2, 2)	2	7	1
(2, 3)	3	4	
(2, 4)	4	5	3
(2, 5)	5	1	4
(2, 6)	6	8	0
(2, 7)	7	0	2
(2, 8)	8	6	
(3, 0)	0	3	2
(3, 1)	1	6	4
(3, 2)	2	0	6
(3, 3)	3	8	0
(3, 4)	4	1	7
(3, 5)	5	8	
(3, 6)	6	2	1
(3, 7)	7	4	
(3, 8)	8	5	3
(4, 0)	0	7	8
(4, 1)	1	3	2
(4, 2)	2	1	7
(4, 3)	3	5	1
(4, 4)	4	6	

(t, u)	$N_t[u, \mathcal{D}]$		
(4, 5)	5	6	3
(4, 6)	6	4	5
(4, 7)	7	2	0
(4, 8)	8	0	
(5, 0)	0	7	1
(5, 1)	1	0	6
(5, 2)	2	3	8
(5, 3)	3	2	
(5, 4)	4	8	5
(5, 5)	5	4	7
(5, 6)	6	1	
(5, 7)	7	5	0
(5, 8)	8	2	4
(6, 0)	0	2	1
(6, 1)	1	0	
(6, 2)	2	5	0
(6, 3)	3	7	8
(6, 4)	4	6	
(6, 5)	5	8	2
(6, 6)	6	4	7
(6, 7)	7	6	3
(6, 8)	8	3	5
(7, 0)	0	5	
(7, 1)	1	3	2
(7, 2)	2	1	7
(7, 3)	3	6	1
(7, 4)	4	7	8
(7, 5)	5	8	0
(7, 6)	6	3	
(7, 7)	7	2	4
(7, 8)	8	4	5
(8, 0)	0	8	
(8, 1)	1	6	5
(8, 2)	2	3	7
(8, 3)	3	4	2
(8, 4)	4	5	3
(8, 5)	5	1	4
(8, 6)	6	1	
(8, 7)	7	2	8
(8, 8)	8	7	0

Algorithm 2: SSMD *sf*-metric

Input: $L = (T, V, A)$ a link stream, Ω the set of event times, (t_s, s) a source event node

Output: Dictionaries d, f of *sf*-metrics and latencies from (t_s, s) to all other event nodes, set of dictionaries R_v for each $v \in V$

$f, d \leftarrow$ create dictionaries
for $v \in V$ **do** $\mathrm{R}_v \leftarrow$ create dictionary
for $t \in \mathrm{Sorted}(\{t_0 \in \Omega \mid t_0 \geq t_s\})$ **do**
    for $C \in \mathrm{connected\_components}(G_t)$ **do**
        $H \leftarrow G_t.\mathrm{induced\_subgraph}(C)$
        $d' \leftarrow \mathrm{all\_pairs\_distances}(H)$
        $D \leftarrow \{\}$
        if $s \in C$ **then** $D.\mathrm{insert}(-t, 0, s)$
        else
            $s_v \leftarrow \max_{u \in C} \mathrm{R}_u.\mathrm{last}()$
            for $v \in C$ **do**
                D.insert all $(-s_v, d_v, v)$ such that
                $(a_v, d_v) \in \mathrm{R}_v[s_v]$ for the largest a_v
        for $(s_u, d_u, u) \in \mathrm{Sorted}(D)$ **do**
            $U \leftarrow \{v \in C \mid \exists a_v : (a_v, d_u) \in \mathrm{R}_v[-s_u]\}$
            if $u = s$ **then** $U \leftarrow \{s\}$
            for $w \in C$ **do**
                $(\_, d_*) \leftarrow \mathrm{R}_w[s_u].\mathrm{last}()$
                $d_{\min} \leftarrow \min(d_u + \min_{u \in U} d'[(u, w)], d_*)$
                $\mathrm{R}_w[-s_u]$.remove all (t, d_0) s.t. $d_0 > d_{\min}$
                $\mathrm{R}_w[-s_u].\mathrm{insert}(t, d_{\min})$
                $f^*_w \leftarrow \min_{(t_0, w) \in f} f[(t_0, w)]$
                $f[(t, w)] \leftarrow \min(t + s_u, f^*_w)$
                $d[(t, w)] \leftarrow \min\ d_0$ s.t. $s_0 \in R_w$, $(a_0, d_0) \in R_w[s_0]$ and $a_0 - s_0 = f[(t, w)]$
return $d, f, \{\mathrm{R}_v \mid v \in V\}$

Algorithm 3: MSMD *sf*-metric

Input: $L = (T, V, A)$ a link stream, Ω the set of event times
Output: F a dictionary of latencies, D^0 a dictionary of reachability triples, D a dictionary of *sf*-metrics

for $u, v \in V$ **do** $SA_{uv}, F_{uv}, D_{uv}, D^0_{uv} \leftarrow$ create sorted dictionaries
for $t \in \Omega$ **do**
  $t^- \leftarrow$ last time of Ω before t
  for $C \in$ connected_components(G_t) **do**
    $H \leftarrow G_t$.induced_subgraph(C)
    $d_C \leftarrow$ all_pairs_distances(H)
    for $u, v \in C$ **do**
      SA_{uv}.insert(t, t)
      $D^0_{uv}[t]$.insert$(t, t, d_C[u, v])$
      $F_{uv}[t]$.insert(0)
    for $u \in C, v \in V \setminus C$ **do**
      $C_v \leftarrow$ conn. component of G_t containing v
      $s_v \leftarrow \max_{w \in C_v, (s,a) \in SA_{uw}}(s)$
      SA_{uv}.insert(s_v, t)
      $d_{\min} \leftarrow \infty$
      for $w \in C_v$, **do**
        $(\_, \_, d_w) \leftarrow D^0_{uw}[t^-]$.last()
        $d_{\min} \leftarrow \min(d_{\min}, d_w + d_C[w, v])$
      $D^0_{uv}[t]$.insert$(s_v, t, d_{\min})$
      $l_{uv} \leftarrow \min_{(s,a) \in SA_{uv}}(a - s)$
      $l = \min(l_{uv}, F_{uv}[t^-])$
      $F_{uv}[t] \leftarrow l$
      $(s^*, a^*) \leftarrow$ pair $(s, a) \in SA_{uv}$ s.t. $a - s = l$
      $D_{uv}[t][s^*] \leftarrow d^*$ s.t. $(s^*, a^*, d^*) \in D^0_{uv}[t]$
return F, D^0, D

Algorithm 4: SSMD *sf*-metric with $\gamma > 0$ (SSMD_γ)

Input: $L = (T, V, A)$ a link stream, Ω the set of event times, (t_s, s) a source event node

Output: Dictionaries d, f of *sf*-metrics and latencies from (t_s, s) to all other event nodes, set of dictionaries $\{\mathrm{R}_v \,|\, v \in V\}$

```
d, f, ← create dictionaries
for v ∈ V do R_v ← create dictionary
for (t, uv) ∈ Sorted(A_Ω) s.t. t ≥ t_s do
    if u = s then R_s[t].insert(t, 0)
    if R_u ≠ ∅ and dur(t, u, v) ≥ γ then
        s_u ← R_u.last()
        (a_u, d_u) ← R_u[s_u].last()
        if s_u exists then
            d_v ← d_u + 1
            if R_v[s_u] does not contain (t', d') s.t. t' ≤ t + γ and
              d' < d_v then R_v[s_u].insert(t + γ, d_v)
        f_v ← t − s_u
        if f[v] ≠ ∅ then
            (_, f'_v) ← f[v].last()
            if f'_v < f_v then f_v ← f'_v
        f[v].add(t, f_v)
        d_fas ← min d_0 s.t. (a_0, d_0) ∈ R_v[s_u] and a_0 − s_u = f_v
        if d_fas exists then d[v].add(t, d_fas)
return d, f, {R_v | v ∈ V}
```

Algorithm 5: MSMD *sf*-metric with $\gamma > 0$ (MSMD_γ)

Input: $L = (T, V, A)$ a link stream, Ω the set of event times
Output: F a dictionary of latencies, D^0 a dictionary of reachability triples, D a dictionary of *sf*-metrics

for $u, v \in V$ **do** $SA_{uv}, F_{uv}, D_{uv}, D^0_{uv} \leftarrow$ create sorted dictionaries
for $(t, u, v) \in \text{Sorted}(A_\Omega)$ *s.t.* $\text{dur}(t, u, v) \geq \gamma$ **do**
    $SA_{uv}[t].\text{insert}(t, t + \gamma)$
    $D^0_{uv}[t].\text{insert}(t, t + \gamma, 1)$
    $F_{uv}[t].\text{insert}(0)$
    for $w \in V \setminus \{v\}$ **do**
        $s_w \leftarrow \max_{(s,a) \in SA_{wu}[t^-]}(s)$
        if s_w *exists* **then**
            ▷ `There is a path from` w `to` v `that starts on` $s_w < t$
            $SA_{wv}[t].\text{insert}(s_w, t + \gamma)$
            $d_v \leftarrow D^0_{wv}[t^-].\text{last}()$
            $d_u \leftarrow D^0_{wu}[t^-].\text{last}()$
            if $d_v > d_u + 1$ **then** $D^0_{wv}[t].\text{insert}(s_w, t + \gamma, d_u + 1)$
            else $D^0_{wv}[t].\text{insert}(s_w, t + \gamma, d_v)$
        $l_{wv} \leftarrow \min_{(s,a) \in SA_{wv}[t]}(a - s)$
        $l \leftarrow \min(l_{wv}, F_{wv}[t^-])$
        $F_{wv}[t] \leftarrow l$
        $(s^*, a^*) \leftarrow$ pair $(s, a) \in SA_{wv}[t]$ s.t. $a - s = l$
        $D_{wv}[t][s^*] \leftarrow d^*$ s.t. $(s^*, a^*, d^*) \in D^0_{wv}[t]$
return F, D^0, D

B Datasets

The following datasets were used in the experiments. More documentation can be found online [143]. Datasets are part of the KONECT project [142]. Descriptions below are found in the directories of the datasets.

1. `arxiv-hepph` A co-citation network from the website arXiv `arxiv.org` of scientific papers. Papers are taken from the high enery physics phenomenology (hep-ph) section. Two papers are linked if they are both cited by another paper, with the timestamp indicating the date of the latter's publication.

2. `contact` A network of human contacts. It describes the Haggle network, a project funded by the European Union. Each edge describes a contact between two persons measured by carried wireless devices.

3. `dblp` A citation network. Extracted from the website `dblp.uni-trier.de/` of scientific publications. Nodes are publications and two publications are linked if one cites the other.

4. `delicious ui` A user-url network from the website `https://del.icio.us` (now deprecated). Edges connect users to the urls they tagged.

5. `delicious ut` An interaction network from the same website. Edges connected users to the tags they used.

6. `digg` A communication network. Extracted from the website `digg.com`. Edges connect two users when one replied to the other.

7. `dnc` A communication network. Extracted from the 2016 Democratic National Committee email leak of the american Democratic Party. Edges connect two people if one sent an email to the other.

8. `elec` An online contact network. Represents admin elections in the English Wikipedia. Edges connect two users if one voted for or against the other.

9. `enron` A communication network. Describes email exchanges in the former company Enron.

10. `epinions-ratings` A ratings network. The website seems deprecated. Each edge connects a user and a product they rated.

11. `facebook-wosn` A social network. Data is extracted from a portion of the website Facebook `facebook.com`. Edges connect users that are friends on the website.

12. `flickr-growth` A social network. Edges describe friendship connections on the website `flickr.com`.

13. `lastfm band` An interaction network between users and bands. From the website `last.fm`. Edges connect users to bands they listened to.

14. `lastfm song` An interaction network between users and songs listened. From the same website as above. Edges connect users to songs they listened to.

15. `lkml person` An interaction network of people on the Linux Kernel Mailing List (lkml). Edges connect people to threads in the mailing list they contributed to.

16. `mit` A human contact network. Edges connect 100 students when they had contact with each other.

17. `movielens` An interaction network. Extracted from the website `http://movielens.umn.edu/`. Edges connect users to the tags they used.

18. `munmun twitter` An interaction network of users and tags. Extracted from the website `twitter.com`. Edges connect users to the tags they used in tweets.

19. `prosper loans` An interaction network. Extracted from the website `prosper.com`. Edges connect people (lenders) to other people (borrowers) to whom they lent money.

20. `slashdot-threads` A comunication network. Extracted from the website `https://slashdot.org/`. Edges connect users when one replied to another.

21. `sociopatterns hyper` A human contact network. Edges represent face-to-face contacts of more than 20 seconds.

22. `sociopatterns infect` A human contact network. Edges represent face-to-face contacts of more than 20 seconds.

23. `wikiconflict` A network of online contacts. Extracted from the english Wikipedia `https://en.wikipedia.org/wiki/Main_Page`. Each edge connects users to other users with whom they are in editing conflicts.

24. `youtube-growth` A social network. Extracted from the website `youtube.com`. Edges connect users with their friends on the platform.

Bibliography

[1] *11th Workshop on GRAph Searching, Theory and Applications (GRASTA)*. 2023. URL: https://grasta23.bici.events/home.

[2] *54th Southeastern International Conference on Combinatorics, Graph Theory & Computing*. 2023. URL: https://www.math.fau.edu/combinatorics/index.php.

[3] Eric Aaron, Danny Krizanc, and Elliot Meyerson. "DMVP: foremost waypoint coverage of time-varying graphs". In: *International Workshop on Graph-Theoretic Concepts in Computer Science*. Springer. 2014, pp. 29–41.

[4] Eric Aaron, Danny Krizanc, and Elliot Meyerson. "Multi-robot foremost coverage of time-varying graphs". In: *International Symposium on Algorithms and Experiments for Sensor Systems, Wireless Networks and Distributed Robotics*. Springer. 2014, pp. 22–38.

[5] Ankush Agarwalla, John Augustine, William K Moses Jr, Sankar K Madhav, and Arvind Krishna Sridhar. "Deterministic dispersion of mobile robots in dynamic rings". In: *Proceedings of the 19th International Conference on Distributed Computing and Networking*. 2018, pp. 1–4.

[6] Martin Aigner and Michael Fromme. "A game of cops and robbers". In: *Discrete Applied Mathematics* 8.1 (1984), pp. 1–12.

[7] Eleni C Akrida, George B Mertzios, and Paul G Spirakis. "The temporal explorer who returns to the base". In: *J. Comput. Syst. Sci.* 120 (2018), pp. 179–193.

[8] Eleni C Akrida, George B Mertzios, Paul G Spirakis, and Viktor Zamaraev. "Temporal vertex cover with a sliding time window". In: *Journal of Computer and System Sciences* 107 (2020), pp. 108–123.

[9] Lyudmil Aleksandrov and Hristo N. Djidjev. “Improved Bounds on the Size of Separators of Toroidal Graphs”. In: *Optimal Algorithms*. 1989.

[10] Ahmad Alsayed and Desmond J. Higham. “Betweenness in time dependent networks”. In: *Chaos, Solitons and Fractals* 72 (2015), pp. 35–48.

[11] Ernst Althaus, Benjamin Merlin Bumpus, James Fairbanks, and Daniel Rosiak. *Compositional Algorithms on Compositional Data: Deciding Sheaves on Presheaves*. 2023. arXiv: 2302.05575 [cs.CC].

[12] Thomas Andreae. “Note on a pursuit game played on graphs”. In: *Discrete Applied Mathematics* 9.2 (1984), pp. 111–115.

[13] Richard P Anstee and Martin Farber. “On bridged graphs and cop-win graphs”. In: *Journal of Combinatorial Theory, Series B* 44.1 (1988), pp. 22–28.

[14] John Augustine, Gopal Pandurangan, and Peter Robinson. “Fast byzantine agreement in dynamic networks”. In: *Proceedings of the 2013 ACM symposium on Principles of distributed computing*. 2013, pp. 74–83.

[15] Baruch Awerbuch and Shimon Even. “Efficient and reliable broadcast is achievable in an eventually connected network”. In: *Proceedings of the third annual ACM symposium on Principles of distributed computing*. 1984, pp. 278–281.

[16] Fan Bai and Ahmed Helmy. “A survey of mobility models”. In: *Wireless Adhoc Networks. University of Southern California, USA* 206 (2004), p. 147.

[17] William Baird, Andrew Beveridge, Anthony Bonato, Paolo Codenotti, Aaron Maurer, John McCauley, and Silviya Valeva. “On the minimum order of k-cop win graphs”. In: *Contributions to Discrete Mathematics* 9.1 (2014).

[18] Stefan Balev, Juan Luis Jiménez Laredo, Ioannis Lamprou, Yoann Pigné, and Eric Sanlaville. “Cops and Robbers on Dynamic Graphs: Offline and Online Case”. In: *27th International Colloquium on Structural Information and Communication Complexity (SIROCCO 2020)* (2020).

[19] Paul Balister, Béla Bollobás, Bhargav Narayanan, and Amy Shaw. "Catching a fast robber on the grid". In: *Journal of Combinatorial Theory, Series A* 152 (2017), pp. 341–352.

[20] Albert-Laszlo Barabasi. "The origin of bursts and heavy tails in human dynamics". In: *Nature* 435.7039 (2005), pp. 207–211.

[21] Albert-László Barabási. *Linked: The new science of networks*. 2003.

[22] Albert-László Barabási. *Network science*. Cambridge University Press, 2016.

[23] Lali Barriere, Paola Flocchini, Pierre Fraigniaud, and Nicola Santoro. "Capture of an intruder by mobile agents". In: *Proceedings of the fourteenth annual ACM symposium on Parallel algorithms and architectures*. 2002, pp. 200–209.

[24] Julien Baste, Binh-Minh Bui-Xuan, and Antoine Roux. "Temporal matching". In: *Theoretical Computer Science* 806 (2020), pp. 184–196.

[25] Hervé Baumann, Pierluigi Crescenzi, and Pierre Fraigniaud. "Parsimonious flooding in dynamic graphs". In: *Proceedings of the 28th ACM symposium on Principles of distributed computing*. 2009, pp. 260–269.

[26] Matthias Bentert, Anne-Sophie Himmel, André Nichterlein, and Rolf Niedermeier. "Efficient computation of optimal temporal walks under waiting-time constraints". In: *Applied Network Science* 5.1 (2020), pp. 1–26.

[27] Alessandro Berarducci and Benedetto Intrigila. "On the cop number of a graph". In: *Advances in Applied Mathematics* 14.4 (1993), pp. 389–403.

[28] Elwyn R Berlekamp, John H Conway, and Richard K Guy. *Winning Ways for Your Mathematical Plays*. Vol. 1,2,3 and 4. A K Peters Ltd., 2001-2004.

[29] Kenneth A Berman. "Vulnerability of scheduled networks and a generalization of Menger's theorem". In: *Networks: An International Journal* 28.3 (1996), pp. 125–134.

[30] Dietmar Berwanger. "Graph games with perfect information". 2013.

[31] Dietmar Berwanger, Anuj Dawar, Paul Hunter, and Stephan Kreutzer. "DAG-width and parity games". In: *STACS 2006: 23rd Annual Symposium on Theoretical Aspects of Computer Science, Marseille, France, February 23-25, 2006. Proceedings 23*. Springer. 2006, pp. 524–536.

[32] Martin Biely, Peter Robinson, and Ulrich Schmid. "Agreement in directed dynamic networks". In: *International Colloquium on Structural Information and Communication Complexity*. Springer. 2012, pp. 73–84.

[33] Hans L Bodlaender. "A linear-time algorithm for finding tree-decompositions of small treewidth". In: *SIAM Journal on computing* 25.6 (1996), pp. 1305–1317.

[34] Anthony Bonato. *An Invitation to Pursuit-Evasion Games and Graph Theory*. Vol. 97. American Mathematical Society, 2022.

[35] Anthony Bonato and Andrea Burgess. "Cops and Robbers on graphs based on designs". In: *Journal of Combinatorial Designs* 21.9 (2013), pp. 404–418.

[36] Anthony Bonato, Nancy E Clarke, Stephen Finbow, Shannon Fitzpatrick, and Margaret-Ellen Messinger. "A note on bounds for the cop number using tree decompositions". In: *arXiv preprint arXiv:1308.2839* (2013).

[37] Anthony Bonato and Gary MacGillivray. "Characterizations and algorithms for generalized Cops and Robbers games". In: *Contributions Discret. Math.* 12 (2017).

[38] Anthony Bonato and Richard J. Nowakowski. *The Game of Cops and Robbers on Graphs*. American Mathematical Society, 2011.

[39] Prosenjit Bose, Jean-Lou De Carufel, and Thomas Shermer. "Pursuit-Evasion in Graphs: Zombies, Lazy Zombies and a Survivor". In: *33rd International Symposium on Algorithms and Computation (ISAAC 2022)*. Ed. by Sang Won Bae and Heejin Park. Vol. 248. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022, 56:1–56:13.

[40] Marjorie Bournat, Swan Dubois, and Franck Petit. "Computability of perpetual exploration in highly dynamic rings". In: *IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*. IEEE. 2017, pp. 794–804.

[41] Peter Bradshaw, Seyyed Aliasghar Hosseini, and Jérémie Turcotte. "Cops and robbers on directed and undirected abelian Cayley graphs". In: *European Journal of Combinatorics* 97 (2021), p. 103383.

[42] Filippo Brunelli, Pierluigi Crescenzi, and Laurent Viennot. "On computing Pareto optimal paths in weighted time-dependent networks". In: *Information Processing Letters* 168 (2021), pp. 1–12.

[43] Filippo Brunelli and Laurent Viennot. "Computing Temporal Reachability Under Waiting-Time Constraints in Linear Time". In: *2nd Symposium on Algorithmic Foundations of Dynamic Networks (SAND 2023)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik. 2023.

[44] Binh-Minh Bui-Xuan, Afonso Ferreira, and Aubin Jarry. "Computing shortest, fastest, and foremost journeys in dynamic networks". In: *International Journal of Foundations of Computer Science* 14.02 (2003), pp. 267–285.

[45] Benjamin Merlin Bumpus. "Generalizing graph decompositions". PhD thesis. University of Glasgow, 2021.

[46] Benjamin Merlin Bumpus, James Fairbanks, Martti Karvonen, Wilmer Leal, and Frédéric Simard. *Towards a Unified Theory of Time-Varying Data*. 2024. arXiv: `2402.00206 [math.CT]`.

[47] Benjamin Merlin Bumpus and Zoltan A Kocsis. "Spined categories: generalizing tree-width beyond graphs". In: *European Journal of Combinatorics* 114 (2023), p. 103794.

[48] Benjamin Merlin Bumpus, Zoltan A. Kocsis, and Jade Edenstar Master. *Structured Decompositions: Structural and Algorithmic Compositionality*. 2023. arXiv: `2207.06091 [math.CT]`.

[49] Benjamin Merlin Bumpus and Kitty Meeks. "Edge exploration of temporal graphs". In: *Algorithmica* 85.3 (2023), pp. 688–716.

[50] Andrea C Burgess, Rosalind A Cameron, Nancy E Clarke, Peter Danziger, Stephen Finbow, Caleb W Jones, and David A Pike. "Cops that surround a robber". In: *Discrete Applied Mathematics* 285 (2020), pp. 552–566.

[51] Johannes Carmesin. "Local 2-separators". In: *Journal of Combinatorial Theory, Series B* 156 (2022), pp. 101–144.

[52] Arnaud Casteigts. “A Journey through Dynamic Networks (with Excursions)”. Habilitation à diriger des recherches. 2018.

[53] Arnaud Casteigts, Paola Flocchini, Bernard Mans, and Nicola Santoro. “Measuring temporal lags in delay-tolerant networks”. In: *IEEE Transactions on Computers* 63.2 (2014), pp. 397–410.

[54] Arnaud Casteigts, Paola Flocchini, Bernard Mans, and Nicola Santoro. “Shortest, fastest, and foremost broadcast in dynamic networks”. In: *International Journal of Foundations of Computer Science* 26.4 (2015), pp. 499–522.

[55] Arnaud Casteigts, Paola Flocchini, Walter Quattrociocchi, and Nicola Santoro. “Time-varying graphs and dynamic networks”. In: *International Journal of Parallel, Emergent and Distributed Systems* 27.5 (2012), pp. 387–408.

[56] Arnaud Casteigts, Anne-Sophie Himmel, Hendrik Molter, and Philipp Zschoche. “Finding Temporal Paths Under Waiting Time Constraints”. In: *31st International Symposium on Algorithms and Computation (ISAAC 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik. 2020.

[57] Arnaud Casteigts, Anne-Sophie Himmel, Hendrik Molter, and Philipp Zschoche. “Finding temporal paths under waiting time constraints”. In: *Algorithmica* 83.9 (2021), pp. 2754–2802.

[58] Arnaud Casteigts, Joseph G. Peters, and Jason Schoeters. “Temporal cliques admit sparse spanners”. In: *Leibniz International Proceedings in Informatics, LIPIcs* 132.134 (2019), pp. 1–14.

[59] Maria Chudnovsky, Sergey Norin, Paul Seymour, and Jérémie Turcotte. “Cops and robbers on P_5-free graphs”. In: (2023). arXiv: `2301.13175 [math.CO]`.

[60] Nancy E. Clarke. “Constrained cops and robber”. PhD thesis. Dalhousie University, 2002. 104, 2002.

[61] Nancy E. Clarke and Richard J. Nowakowski. “Cops, robber and traps”. In: *Utilitas Mathematica* 60 (2001), pp. 91–98.

[62] Benoît Colson, Patrice Marcotte, and Gilles Savard. “An overview of bilevel optimization”. In: *Annals of Operations Research* 153.1 (2007), pp. 235–256.

[63] Sandip Das, Harmender Gahlawat, Uma kant Sahoo, and Sagnik Sen. "Cops and Robber on some families of oriented graphs". In: *Theoretical Computer Science* 888 (2021), pp. 31–40.

[64] Jean-Lou De Carufel, Paola Flocchini, Nicola Santoro, and Frédéric Simard. *Copnumbers of periodic graphs*. 2023. arXiv: 2310.13616 [math.CO].

[65] Jean-Lou De Carufel, Paola Flocchini, Nicola Santoro, and Frédéric Simard. "Cops & Robber on Periodic Temporal Graphs: Characterization and Improved Bounds". In: *30th International Colloquium, SIROCCO 2023, Alcalá de Henares, Spain, June 6–9, 2023, Proceedings*. Ed. by Sergio Rajsbaum, Alkida Balliu, Joshua J. Daymude, and Dennis Olivetti. Springer Cham, 2023, pp. 386–405.

[66] Xavier Défago, Maria Potop-Butucaru, and Sébastien Tixeuil. "Fault-tolerant mobile robots". In: *Distributed Computing by Mobile Entities: Current Research in Moving and Computing* (2019), pp. 234–251.

[67] Giuseppe Antonio Di Luna. *Mobile Agents on Dynamic Graphs*. Ed. by Paola Flocchini, Giuseppe Prencipe, and Nicola Santoro. Cham: Springer International Publishing, 2019, pp. 549–584.

[68] Giuseppe Antonio Di Luna, Stefan Dobrev, Paola Flocchini, and Nicola Santoro. "Distributed exploration of dynamic rings". In: *Distributed Computing* 33 (2020), pp. 41–67.

[69] Giuseppe Antonio Di Luna, Stefan Dobrev, Paola Flocchini, and Nicola Santoro. "Live exploration of dynamic rings". In: *36th IEEE International Conference on Distributed Computing Systems*. 2016, pp. 570–579.

[70] Giuseppe Antonio Di Luna, Paola Flocchini, Linda Pagli, Giuseppe Prencipe, Nicola Santoro, and Giovanni Viglietta. "Gathering in dynamic rings". In: *24th International Colloquium, SIROCCO 2017, Porquerolles, France, June 19-22, 2017, Revised Selected Papers*. Ed. by Shantanu Das and Sebastien Tixeuil. Springer. 2017, pp. 339–355.

[71] Reinhard Diestel. *Graph Theory - Diestel*. 2000.

[72] Stefan Dobrev, Paola Flocchini, Giuseppe Prencipe, and Nicola Santoro. "Mobile search for a black hole in an anonymous ring". In: *15th International conference Distributed Computing, DISC 2001*. Vol. 2180. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, 2001, pp. 166–179.

[73] Rodney G Downey and Michael R Fellows. *Fundamentals of Parameterized Complexity*. 2013, p. 763.

[74] Ran Duan and Le Zhang. "Faster randomized worst-case update time for dynamic subgraph connectivity". In: *Workshop on Algorithms and Data Structures*. Springer. 2017, pp. 337–348.

[75] Gabriel L. Duarte, Mateus de Oliveira Oliveira, and Uéverton S. Souza. "Co-Degeneracy and Co-Treewidth: Using the Complement to Solve Dense Instances". In: *46th International Symposium on Mathematical Foundations of Computer Science, MFCS 2021, August 23-27, 2021, Tallinn, Estonia*. Ed. by Filippo Bonchi and Simon J. Puglisi. Vol. 202. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021, 42:1–42:17.

[76] Vida Dujmović, Pat Morin, and David R Wood. "Layered separators in minor-closed graph classes with applications". In: *Journal of Combinatorial Theory, Series B* 127 (2017), pp. 111–147.

[77] John A Ellis, Ivan Hal Sudborough, and Jonathan S Turner. "The vertex separation and search number of a graph". In: *Information and computation* 113.1 (1994), pp. 50–79.

[78] David Eppstein, Zvi Galil, and Giuseppe F Italiano. "Dynamic graph algorithms". In: *Algorithms and theory of computation handbook* 1 (1999), pp. 9–1.

[79] Thomas Erlebach, Michael Hoffmann, and Frank Kammer. "On temporal graph exploration". In: *Journal of Computer and System Sciences* 119 (2021), pp. 1–18.

[80] Thomas Erlebach and Jakob T Spooner. "A Game of Cops and Robbers on Graphs with Periodic Edge-Connectivity". In: *46th International Conference on Current Trends in Theory and Practice of Informatics (SOFSEM)*. 2020, pp. 64–75.

[81] Thomas Erlebach and Jakob T Spooner. "Faster exploration of degree-bounded temporal graphs". In: *43rd International Symposium on Mathematical Foundations of Computer Science (MFCS)*. 2018, pp. 1–13.

[82] William Evans, Paul Hunter, and Mohammad Ali Safari. *D-width and cops and robbers*. Tech. rep. 2007.

[83] Farhan A Faruqi. *Differential game theory with applications to missiles and autonomous systems guidance*. John Wiley & Sons, 2017.

[84] Afonso Ferreira. "Building a reference combinatorial model for MANETs". In: *IEEE Network* 18.5 (2004), pp. 24–29.

[85] Afonso Ferreira and Laurent Viennot. *A Note on Models, Algorithms, and Data Structures for Dynamic Communication Networks*. Research Report RR-4403. INRIA, 2002.

[86] Paola Flocchini, Bernard Mans, and Nicola Santoro. "On the exploration of time-varying networks". In: *Theoretical Computer Science* 469 (2013), pp. 53–68.

[87] Paola Flocchini, Giuseppe Prencipe, and Nicola Santoro. *Distributed Computing by Mobile Entities*. Ed. by Paola Flocchini, Giuseppe Prencipe, and Nicola Santoro. Vol. 11340. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2019.

[88] Till Fluschnik, Hendrik Molter, Rolf Niedermeier, Malte Renken, and Philipp Zschoche. "As time goes by: reflections on treewidth for temporal graphs". In: *Treewidth, Kernels, and Algorithms*. Springer, 2020, pp. 49–77.

[89] Fedor V Fomin, Petr A Golovach, Jan Kratochvil, Nicolas Nisse, and Karol Suchan. "Pursuing a fast robber on a graph". In: *Theoretical Computer Science* 411.7-9 (2010), pp. 1167–1181.

[90] Fedor V. Fomin and Dimitrios M. Thilikos. "An annotated bibliography on guaranteed graph searching". In: *Theoretical Computer Science* 399.3 (2008), pp. 236–245.

[91] Santo Fortunato and Darko Hric. "Community detection in networks: A user guide". In: *Physics Reports* 659 (2016), pp. 1–44.

[92] Peter Frankl. "Cops and robbers in graphs with large girth and Cayley graphs". In: *Discrete Applied Mathematics* 17.3 (1987), pp. 301–305.

[93] Linton C Freeman. "Centrality in Social Networks". In: *Social Networks* 1.1968 (1978), pp. 215–239.

[94] Alan Frieze, Michael Krivelevich, and Po-Shen Loh. "Variations on cops and robbers". In: *Journal of Graph Theory* 69.4 (2012), pp. 383–402.

[95] Harmender Gahlawat, Zin Mar Myint, and Sagnik Sen. "Cops and robber on variants of retracts and subdivisions of oriented graphs". In: (2023). arXiv: 2307.00584 [math.CO].

[96] Harmender Gahlawat and Meirav Zehavi. "Parameterized analysis of the cops and robber game". In: *48th International Symposium on Mathematical Foundations of Computer Science (MFCS 2023)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik. 2023.

[97] Robert Ganian, Petr Hliněnỳ, Joachim Kneis, Daniel Meister, Jan Obdržálek, Peter Rossmanith, and Somnath Sikdar. "Are there any good digraph width measures?" In: *Journal of Combinatorial Theory, Series B* 116 (2016), pp. 250–286.

[98] Ziyuan Gao and Boting Yang. "The one-cop-moves game on planar graphs". In: *Journal of Combinatorial Optimization* 42 (2021), pp. 442–475.

[99] Agelos Georgakopoulos. *Compact metric spaces with infinite cop number*. 2023. arXiv: 2309.03757 [math.CO].

[100] Hugo Gimbert and Florian Horn. "Simple Stochastic Games with Few Random Vertices Are Easy to Solve". In: *Foundations of Software Science and Computational Structures*. Ed. by Roberto Amadio. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 5–19.

[101] Petr A Golovach. "A topological invariant in pursuit problems". In: *Differentsial'nye Uravneniya* 25.6 (1989), pp. 923–929.

[102] Petr A Golovach. "Equivalence of two formalizations of a search problem on graphs". In: *Vestnik Leningradskogo Universiteta Seriya Matematika Mekhanika Astronomiya* 1 (1989), pp. 10–14.

[103] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. 2016.

[104] Tsuyoshi Gotoh, Paola Flocchini, Toshimitsu Masuzawa, and Nicola Santoro. “Exploration of dynamic networks: tight bounds on the number of agents”. In: *Journal of Computer and System Sciences* 122 (2021), pp. 1–18.

[105] Tsuyoshi Gotoh, Yuichi Sudo, Fukuhito Ooshita, Hirotsugu Kakugawa, and Toshimitsu Masuzawa. “Group exploration of dynamic tori”. In: *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*. IEEE. 2018, pp. 775–785.

[106] Ilya Gromovikov, William B. Kinnersley, and Ben Seamone. “Fully active cops and robbers”. In: *Australasian Journal of Combinatorics* 76.2 (2020), pp. 248–265.

[107] Geňa Hahn and Gary MacGillivray. “A note on k-cop, l-robber games on graphs”. In: *Discrete Mathematics* 306.19-20 (2006), pp. 2492–2497.

[108] Geňa Hahn and Claude Tardif. “Graph homomorphisms: structure and symmetry”. In: *Graph Symmetry* (1997), pp. 107–166.

[109] Rudolf Halin. “S-functions for graphs”. In: *Journal of geometry* 8 (1976), pp. 171–186.

[110] Peter Hammerstein and Reinhard Selten. “Game theory and evolutionary biology”. In: *Handbook of game theory with economic applications* 2 (1994), pp. 929–993.

[111] Frank Harary and Gopal Gupta. “Dynamic graph models”. In: *Mathematical and Computer Modelling* 25.7 (1997), pp. 79–88.

[112] Daniel J Harvey and David R Wood. “Parameters tied to treewidth”. In: *Journal of Graph Theory* 84.4 (2017), pp. 364–385.

[113] Alan Hill. “Cops and robbers: theme and variations”. PhD thesis. Dalhousie University, 2008.

[114] Petter Holme and Jari Saramäki. *Temporal Networks*. Springer Berlin, Heidelberg, 2015.

[115] Petter Holme and Jari Saramäki. “Temporal networks”. In: *Physics Reports* 519.3 (2012), pp. 97–125.

[116] Seyyed Aliasghar Hosseini and Bojan Mohar. “Game of cops and robbers in oriented quotients of the integer grid”. In: *Discrete Mathematics* 341.2 (2018), pp. 439–450.

[117] Kristóf Huszár, Jonathan Spreer, and Uli Wagner. “On the treewidth of triangulated 3-manifolds”. In: *Journal of Computational Geometry* 10.2 (2019).

[118] Allen Ibiapina and Ana Silva. “Snapshot disjointness in temporal graphs”. In: (2023). arXiv: 2302.06653 [cs.DM].

[119] David Ilcinkas, Ralf Klasing, and Ahmed Mouhamadou Wade. “Exploration of Constantly Connected Dynamic Graphs Based on Cactuses”. In: *Structural Information and Communication Complexity. SIROCCO 2014*. Ed. by Magnús M. Halldórsson. Vol. 8576. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2014, pp. 250–262.

[120] David Ilcinkas and Ahmed Mouhamadou Wade. “Exploration of the T-interval-connected dynamic graphs: the case of the ring”. In: *International Colloquium on Structural Information and Communication Complexity*. Springer. 2013, pp. 13–23.

[121] David Ilcinkas and Ahmed Mouhamadou Wade. “On the Power of Waiting When Exploring Public Transportation Systems”. In: *Principles of Distributed Systems. OPODIS 2011*. Ed. by Antonio Fernàndez Anta, Giuseppe Lipari, and Matthieu Roy. Vol. 7109. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 451–464.

[122] David Ilcinkas and Ahmed Mouhamadou Wade. “On the power of waiting when exploring public transportation systems”. In: *Principles of Distributed Systems: 15th International Conference, OPODIS 2011, Toulouse, France, December 13-16, 2011. Proceedings 15*. Springer. 2011, pp. 451–464.

[123] Vesna Iršič, Bojan Mohar, and Alexandra Wesolek. “Cops and robber on hyperbolic manifolds”. In: *European Conference on Combinatorics, Graph Theory and Applications*. 12. 2023.

[124] Rufus Isaacs. *Differential games: a mathematical theory with applications to warfare and pursuit, control and optimization*. John Wiley & Sons Inc., New York, 1965.

[125] Bart MP Jansen, Jari JH De Kroon, and Michał Włodarczyk. "Vertex deletion parameterized by elimination distance and even less". In: *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*. 2021, pp. 1757–1769.

[126] R. Jathar, V. Yadav, and A. Gupta. "Using periodic contacts for efficient routing in delay tolerant networks". In: *Ad Hoc & Sensor Wireless Networks* 22.1,2 (2014), pp. 283–308.

[127] Hai Jin, Na Wang, Dongxiao Yu, Qiang-Sheng Hua, Xuanhua Shi, and Xia Xie. "Core maintenance in dynamic graphs: A parallel approach based on matching". In: *IEEE Transactions on Parallel and Distributed Systems* 29.11 (2018), pp. 2416–2428.

[128] D. Gage Jordan, Danica C. Slavish, Jessee Dietch, Brett Messman, Camilo Ruggero, Kimberly Kelly, and Daniel J. Taylor. "Investigating sleep, stress, and mood dynamics via temporal network analysis". In: *Sleep Medicine* 103 (2023), pp. 1–11.

[129] Gwenaël Joret, Marcin Kamiński, and Dirk Oliver Theis. "The cops and robber game on graphs with forbidden (induced) subgraphs". In: *Contributions to Discrete Mathematics* 5.2 (2010).

[130] Seyed Mehran Kazemi, Rishab Goel, Kshitij Jain, Ivan Kobyzev, Akshay Sethi, Peter Forsyth, and Pascal Poupart. "Representation learning for dynamic graphs: A survey". In: *The Journal of Machine Learning Research* 21.1 (2020), pp. 2648–2720.

[131] Athanasios Kehagias and Georgios Konstantinidis. *Cops and Robbers, Game Theory and Zermelo's Early Results*. 2014. arXiv: `1407.1647 [cs.DM]`.

[132] Athanasios Kehagias and Pawel Pralat. "Some remarks on cops and drunk robbers". In: *Theoretical Computer Science*. Vol. 463. 2012, pp. 133–147.

[133] David Kempe, Jon Kleinberg, and Amit Kumar. "Connectivity and inference problems for temporal networks". In: *Journal of Computer and System Sciences* 64.4 (2002), pp. 820–842.

[134] Devvrit Khatri, Natasha Komarov, Aaron Krim-Yee, Nithish Kumar, Ben Seamone, Virgélot Virgile, and AnQi Xu. "A study of cops and robbers in oriented graphs". In: (2018), pp. 1–23.

[135] Nancy G Kinnersley. "The vertex separation number of a graph equals its path-width". In: *Information Processing Letters* 42.6 (1992), pp. 345–350.

[136] William B. Kinnersley. "Cops and Robbers is EXPTIME-complete". en. In: *Journal of Combinatorial Theory, Series B* 111 (Mar. 2015), pp. 201–220.

[137] Lefteris M Kirousis and Christos H Papadimitriou. "Searching and pebbling". In: *Theoretical Computer Science* 47 (1986), pp. 205–218.

[138] Nina Klobas, George B Mertzios, and Paul G Spirakis. "Sliding into the Future: Investigating Sliding Windows in Temporal Graphs (Invited Talk)". In: *48th International Symposium on Mathematical Foundations of Computer Science (MFCS 2023)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik. 2023.

[139] Natasha Komarov and Peter Winkler. "Capturing the Drunk Robber on a Graph". In: *The Electronic Journal of Combinatorics* 21.3 (2014), p. 14.

[140] Vassilis Kostakos. "Temporal graphs". In: *Physica A: Statistical Mechanics and its Applications* 388.6 (2009), pp. 1007–1023.

[141] Fabian Kuhn, Nancy Lynch, and Rotem Oshman. "Distributed computation in dynamic networks". In: *42nd ACM Symposium on Theory of Computing*. 2010, pp. 513–522.

[142] Jérôme Kunegis. "Konect: the koblenz network collection". In: *Proceedings of the 22nd International Conference on World Wide Web*. Acm. 2013, pp. 1343–1350.

[143] Jérôme Kunegis. *The KONECT Project*. `http://konect.cc/`. Accessed: 21-10-2019. 2019.

[144] Matthieu Latapy, Tiphaine Viard, and Clémence Magnien. "Stream graphs and link streams for the modeling of interactions over time". In: *Social Network Analysis and Mining* 8.1 (2018), p. 61.

[145] Mo Li, Junchang Xin, Zhiqiong Wang, and Huilin Liu. "Accelerating Minimum Temporal Paths Query Based on Dynamic Programming". In: *International Conference on Advanced Data Mining and Applications*. Springer. 2019, pp. 48–62.

[146] Xiannuan Liang and Yang Xiao. “Game theory for network security”. In: *IEEE Communications Surveys & Tutorials* 15.1 (2012), pp. 472–486.

[147] Richard J. Lipton and Robert E. Tarjan. “A Separator Theorem for Planar Graphs”. In: *Siam Journal on Applied Mathematics* 36 (1977), pp. 177–189.

[148] Po-Shen Loh and Siyoung Oh. “Cops and Robbers on Planar-Directed Graphs”. In: *Journal of Graph Theory* 86.3 (2017), pp. 329–340.

[149] Giuseppe Antonio Di Luna. “Mobile Agents on Dynamic Graphs”. In: *Distributed Computing by Mobile Entities*. 2019.

[150] Qi Luo, Dongxiao Yu, Xiuzhen Cheng, Zhipeng Cai, Jiguo Yu, and Weifeng Lv. “Batch processing for truss maintenance in large dynamic graphs”. In: *IEEE Transactions on Computational Social Systems* 7.6 (2020), pp. 1435–1446.

[151] Yao Ma and Jiliang Tang. *Deep learning on graphs*. Cambridge University Press, 2021.

[152] Zijuan Ma, Jingbo Zhao, Huilin Chen, Yanqiang Tao, Yifan Zhang, Fang Fan, et al. “Temporal Network of Depressive Symptoms across College Students with Distinct Depressive Trajectories during the COVID-19 Pandemic”. In: *Depression and Anxiety* 2023 (2023).

[153] Malaz Maamoun and Henry Meyniel. “On a game of policemen and robber”. In: *Discrete Applied Mathematics* 17.3 (1987), pp. 307–309.

[154] Kaveh Madani. “Game theory and water resources”. In: *Journal of hydrology* 381.3-4 (2010), pp. 225–238.

[155] Naércio Magaia, Alexandre P. Francisco, Paulo Pereira, and Miguel Correia. “Betweenness centrality in Delay Tolerant Networks: A survey”. In: *Ad Hoc Networks* 33 (2015), pp. 284–305.

[156] Patrick Maillé, Evangelos Markakis, Maurizio Naldi, George D Stamoulis, and Bruno Tuffin. “Sponsored search auctions: An overview of research with emphasis on game theoretic aspects”. In: *Electronic Commerce Research* 12 (2012), pp. 265–300.

[157] Andrea Marino and Ana Silva. “Königsberg sightseeing: Eulerian walks in temporal graphs”. In: *International Workshop on Combinatorial Algorithms*. Springer. 2021, pp. 485–500.

[158] Djibril Mboup, Cherif Diallo, and Hocine Cherifi. "Temporal networks based on human mobility models: A comparative analysis with real-world networks". In: *IEEE Access* 10 (2022), pp. 5912–5935.

[159] Abbas Mehrabian. "Cops and robber game with a fast robber". Master's Thesis. University of Waterloo, 2011.

[160] Andrew Mellor. "The temporal event graph". In: *Journal of Complex Networks* 6.4 (Oct. 2017), pp. 639–659.

[161] Othon Michail and Paul G Spirakis. "Traveling salesman problems in temporal graphs". In: *Theoretical Computer Science* 634 (2016), pp. 1–23.

[162] Bojan Mohar. "Min-max theorem for the game of Cops and Robber on geodesic spaces". In: (2021). arXiv: `2112.03018 [math.CO]`.

[163] Nils Morawietz, Carolin Rehs, and Mathias Weller. "A timecop's work is harder than you think". In: *45th International Symposium on Mathematical Foundations of Computer Science (MFCS 2020)*. 2020.

[164] Mark Newman. *Networks*. Oxford university press, 2018.

[165] Vincenzo Nicosia, John Tang, Cecilia Mascolo, Mirco Musolesi, Giovanni Russo, and Vito Latora. "Graph metrics for temporal networks". In: *Temporal networks*. Ed. by Petter Holme and Jari Saramäki. Springer Berlin Heidelberg, 2013, pp. 15–40.

[166] Nicolas Nisse. "Network decontamination". In: *Distributed Computing by Mobile Entities: Current Research in Moving and Computing*. Springer, 2019, pp. 516–548.

[167] Richard J. Nowakowski and Peter Winkler. "Vertex-to-vertex pursuit in a graph". In: *Discrete Mathematics* 43.2-3 (1983), pp. 235–239.

[168] R. O'Dell and R. Wattenhofer. "Information dissemination in highly dynamic graphs". In: *Joint Workshop on Foundations of Mobile Computing*. 2005, pp. 104–110.

[169] James B. Orlin. "The Complexity of Dynamic Languages and Dynamic Optimization Problems". In: *Proceedings of the Thirteenth Annual ACM Symposium on Theory of Computing*. Stoc '81. Milwaukee, Wisconsin, USA: Association for Computing Machinery, 1981, pp. 218–227.

[170] Jan Petr, Julien Portier, and Leo Versteegen. “A faster algorithm for Cops and Robbers”. In: *Discrete Applied Mathematics* 320 (2022), pp. 11–14.

[171] Pawel Pralat. “When does a random graph have constant cop number?” In: *Australas. J Comb.* 46 (2010), pp. 285–296.

[172] Alain Quilliot. “Jeux et points fixes sur les graphes”. Thèse de 3ème cycle. Université de Paris VI, France, 1978.

[173] 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.

[174] Marc Quincampoix. “Differential games”. In: *Complex Social and Behavioral Systems*. Ed. by Marilda Sotomayor, David Pérez-Castrillo, and Filippo Castiglione. Springer, 2020, pp. 305–315.

[175] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria, 2013.

[176] Emily Riehl. *Category theory in context*. Courier Dover Publications, 2017.

[177] Neil Robertson and Paul D. Seymour. “Graph minors. II. Algorithmic aspects of tree-width”. In: *Journal of algorithms* 7.3 (1986), pp. 309–322.

[178] Joel J.P.C. Rodrigues. *Advances in delay-tolerant networks (DTNs): Architecture and enhanced performance*. Woodhead Publishing, 2021.

[179] Joel J.P.C. Rodrigues and Vasco N.G.J. Soares. “An introduction to delay and disruption-tolerant networks (DTNs)”. In: *Advances in Delay-Tolerant Networks (DTNs)*. Elsevier, 2021, pp. 1–21.

[180] Daniel Rosiak. *Sheaf theory through examples*. MIT Press, 2022.

[181] Emanuele Rossi, Ben Chamberlain, Fabrizio Frasca, Davide Eynard, Federico Monti, and Michael Bronstein. *Temporal graph networks for deep learning on dynamic graphs*. 2020. arXiv: `2006.10637 [cs.LG]`.

[182] Tim Roughgarden. “Algorithmic game theory”. In: *Communications of the ACM* 53.7 (2010), pp. 78–86.

[183] Nicola Santoro. *Design and Analysis of Distributed Algorithms*. Ed. by 2007 Hoboken, N.J. : Wiley-Interscience. 2007.

[184] Paul D Seymour and Robin Thomas. “Graph searching and a min-max theorem for tree-width”. In: *Journal of Combinatorial Theory, Series B* 58.1 (1993), pp. 22–33.

[185] Farhad Shahrokhi. “New representation results for planar graphs”. In: (2015). arXiv: `1502.06175 [math.CO]`.

[186] Claude Shannon. “Presentation of a maze-solving machine”. In: *8th Conference of the Josiah Macy Jr. Found. (Cybernetics)* (1951), pp. 173–180.

[187] Yossi Shiloach and Shimon Even. “An on-line edge-deletion problem”. In: *Journal of the ACM (JACM)* 28.1 (1981), pp. 1–4.

[188] Frédéric Simard. “Evaluating metrics in link streams”. In: *Social Network Analysis and Mining* 11.1 (2021), pp. 1–16.

[189] Frédéric Simard. “On computing distances and latencies in Link Streams”. In: *2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. IEEE. 2019, pp. 394–397.

[190] Frédéric Simard. *SSMD and MSMD repository*. `https://bitbucket.org/simfr404/linkstreams_cpp/src/master/`. Accessed: March 26, 2024. 2019.

[191] Frédéric Simard, Josée Desharnais, and François Laviolette. “General Cops and Robbers Games with randomness”. In: *Theoretical Computer Science* (2021).

[192] Frédéric Simard, Clémence Magnien, and Matthieu Latapy. “Computing Betweenness Centrality in Link Streams”. In: *Journal of Graph Algorithms and Applications* 27.3 (2023), pp. 195–217.

[193] Frédéric Simard, Michael Morin, Claude-Guy Quimper, François Laviolette, and Josée Desharnais. “Bounding an Optimal Search Path with a Game of Cop and Robber on Graphs”. In: *Principles and Practice of Constraint Programming: 21st International Conference, CP 2015, Cork, Ireland, August 31–September 4, 2015, Proceedings*. Vol. 9255. Springer Science Business Media, 2015, pp. 403–418.

[194] Vaidy Sivaraman. “An application of the Gyárfás path argument”. In: *Discrete Mathematics* 342.8 (2019), pp. 2306–2307.

[195] Veronika Slivova. “Cops and robber game on directed complete graphs”. Bachelor’s Thesis. Charles University in Prague, 2015.

[196] Steven H Strogatz. “Exploring complex networks”. In: *Nature* 410.6825 (2001), pp. 268–276.

[197] Kazuo Sugihara and Ichiro Suzuki. “Distributed algorithms for formation of geometric patterns with many mobile robots”. In: *Journal of robotic systems* 13.3 (1996), pp. 127–139.

[198] Steven Tadelis. *Game theory: an introduction*. Princeton university press, 2013.

[199] John Tang, Mirco Musolesi, Cecilia Mascolo, Vito Latora, and Vincenzo Nicosia. “Analysing Information Flows and Key Mediators through Temporal Centrality Metrics”. In: *Proceedings of the 3rd Workshop on Social Network Systems (SNS ’10)*. Paris, France: Acm, 2010.

[200] Suhas Thejaswi, Juho Lauri, and Aristides Gionis. *Restless reachability problems in temporal graphs*. 2021. arXiv: `2010.08423 [cs.DS]`.

[201] Szymon Toruńczyk. “Flip-width: Cops and robber on dense graphs”. In: *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE. 2023, pp. 663–700.

[202] Jérémie Turcotte and Samuel Yvon. “4-cop-win graphs have at least 19 vertices”. In: *Discrete Applied Mathematics* 301 (2021), pp. 74–98.

[203] Kirill Veselkov, Guadalupe Gonzalez, Shahad Aljifri, Dieter Galea, Reza Mirnezami, Jozef Youssef, Michael Bronstein, and Ivan Laponogov. “HyperFoods: Machine intelligent mapping of cancer-beating molecules in foods”. In: *Scientific reports* 9.1 (2019), p. 9237.

[204] Tiphaine Viard, Matthieu Latapy, and Clémence Magnien. “Computing maximal cliques in link streams”. In: *Theoretical Computer Science* 609 (2016), pp. 245–252.

[205] Zsolt Adam Wagner. “Cops and Robbers on diameter two graphs”. In: *Discrete Mathematics* 338.3 (2015), pp. 107–109.

[206] Jiakun Wang, Hao Yu, and Yun Li. “Research on the co-evolution of temporal networks structure and public opinion propagation”. In: *Journal of Information Science* (2022), p. 0.

[207] Klaus Wehmuth, Artur Ziviani, and Eric Fleury. “A unifying model for representing time-varying graphs”. In: *2nd IEEE International Conference on Data Science and Advanced Analytics, (DSAA)*. 2015, pp. 1–10.

[208] Huanhuan Wu, James Cheng, Silu Huang, Yiping Ke, Yi Lu, and Yanyan Xu. “Path Problems in Temporal Graphs”. In: *Proceedings of the VLDB Endowment* 7.9 (2014), pp. 721–732.

[209] Shun Wu, Fengchen Fu, Lei Wang, Minhang Yang, Shi Dong, Yongqing He, Qingqing Zhang, and Rong Guo. “Short-Term Regional Temperature Prediction Based on Deep Spatial and Temporal Networks”. In: *Atmosphere* 13.12 (2022).