Computational Complexity Aspects of Implicit Graph Representations

Von der Fakultät für Elektrotechnik und Informatik
der Gottfried Willhelm Leibniz Universität Hannover
zur Erlangung des Grades

Doktor der Naturwissenschaften
Dr. rer. nat.

genehmigte Dissertation
von

M. Sc. Maurice Chandoo

geboren am 27. Juni 1991 in Hannover

2017

Referent:	Heribert Vollmer, Leibniz Universität Hannover
Korreferent:	Johannes Köbler, Humboldt-Universität zu Berlin
Tag der Promotion:	20.07.2018

La clarté orne les pensées profondes.

Luc de Clapiers,
marquis de Vauvenargues

Acknowledgments

I thank my doctoral advisor Heribert Vollmer for not only having introduced me to the field of theoretical computer science but also for having done so with inspiring clarity and great emphasis on intuition. Moreover, I deeply thank him for granting me the liberty to pursue my own research.

Abstract

Implicit graph representations are immutable data structures for restricted classes of graphs such as planar graphs. A graph class has an implicit representation if the vertices of every graph in this class can be assigned short labels such that the adjacency of two vertices can be decided by an algorithm which gets the two labels of these vertices as input. A representation of a graph in that class is then given by the set of labels of its vertices. The algorithm which determines adjacency is only allowed to depend on the graph class. Such representations are attractive because they are space-efficient and in many cases also allow for constant-time edge queries. Therefore they outperform less specialized representations such as adjacency matrices or lists and are even optimal in an asymptotic sense.

In the first part of this thesis we investigate the limitations of such representations when constraining the complexity of an algorithm which decodes adjacency. First, we prove that imposing such computational constraints does indeed affect what graph classes have an implicit representation. Then we observe that the adjacency structure of almost all graph classes that are known to have an implicit representation can be described by formulas of first-order logic. The quantifier-free fragment of this logic can be characterized in terms of RAMs: a graph class can be expressed by a quantifier-free formula if and only if it has an implicit representation where edges can be queried in constant-time on a RAM without division. We provide two reduction notions for graph classes which reveal that trees and interval graphs are representative for certain fragments of this logic. We conclude this part by providing a big picture of the newly introduced classes and point out viable research directions.

In the second part we consider the tractability of algorithmic problems on graph classes with implicit representations. Intuitively, if a graph class has an implicit representation with very low complexity then it should have a simple adjacency structure. Therefore it seems plausible to expect certain algorithmic problems to be tractable on such graph classes. We consider how realistic it is to expect an algorithmic meta-theorem of the form "if a graph class X has an implicit representation with complexity Y then problem Z is tractable on X". Our considerations quickly reveal that even for the most humble choices of Y and various Z this is either impossible or leads to the frontiers of algorithmic research. We show that the complexity classes of graph classes introduced in the previous chapter can be interpreted as graph parameters and therefore can be considered within the framework of parameterized complexity. We embark on a case study where Z is the graph isomorphism problem and Y is the quantifier-free, four-variable fragment of first order logic with only the order predicate on the universe. This leads to a problem that has been studied independently and resisted classification for over two decades: the isomorphism problem for circular-arc (CA) graphs. We examine how a certain method, which we call flip trick, can be applied to this problem. We show that for a broad class of CA graphs the isomorphism problem reduces to the representation problem and as a consequence can be solved in polynomial-time.

Keywords: adjacency labeling schemes, descriptive complexity of graph properties, reductions for graph classes, CA graph isomorphism, pointer numbers

Zusammenfassung

Implizite Graphrepräsentationen sind statische Datenstrukturen für beschränkte Graphklassen wie zum Beispiel planare Graphen. Eine Graphklasse hat eine implizite Repräsentation, falls die Knoten jedes Graphen dieser Klasse mit kurzen Labels beschriftet werden können, sodass die Adjazenz zweier Knoten von einem Algorithmus entschieden werden kann, welcher die zwei Labels der Knoten als Eingabe erhält. Eine Repräsentation eines Graphen aus dieser Klasse besteht aus der Menge der Labels seiner Knoten. Der Algorithmus, welcher die Adjazenz entscheidet, darf nur von der Graphklasse abhängen. Solche Repräsentationen sind attraktiv, da sie speichereffizient sind und oftmals auch Kantenabfragen in konstanter Zeit zulassen. Deshalb übertreffen sie weniger spezialisierte Repräsentationen wie Adjazenzmatrizen oder -listen und sind sogar asymptotisch optimal.

Im ersten Teil dieser Arbeit untersuchen wir, welche Auswirkungen das Einschränken der Komplexität von Algorithmen, welche die Adjazenz decodieren, auf die Menge von Graphklassen, die eine implizite Repräsentation haben, hat. Es stellt sich heraus, dass die Menge der Graphklassen mit so einer Repräsentation tatsächlich von der gewählten Komplexität abhängt. Anschließend beobachten wir, dass die Adjazenzstruktur von fast allen Graphklassen, von denen man weiß, dass sie eine implizite Repräsentation haben, in Prädikatenlogik erster Stufe ausgedrückt werden kann. Das quantorenfreie Fragment dieser Logik kann wie folgt charaktersiert werden: eine Graphklasse kann genau dann durch eine quantorenfreie Formel erster Stufe ausgedrückt werden, wenn sie eine implizite Repräsentation hat, in der Kantenabfragen in konstanter Zeit auf einer RAM ohne Division durchgeführt werden können. Wir führen zwei Reduktionsbegriffe für Graphklassen ein, welche es uns ermöglichen zu zeigen, dass Bäume und Intervalgraphen für bestimmte Fragmente dieser Logik repräsentativ sind. Im letzten Teil fassen wir unsere Ergebnisse zusammen und stellen die verschiedenen, neu eingeführten Klassen und deren Beziehungen in einem Schaubild dar.

Im zweiten Teil beschäftigen wir uns mit der Komplexität algorithmischer Probleme auf Klassen von Graphen mit impliziten Repräsentationen. Intuitiv gesehen sollte eine Graphklasse mit einer impliziten Repräsentation von geringer Komplexität eine ebenso simple Adjazenzstruktur haben. Daher erscheint es plausibel zu erwarten, dass bestimmte algorithmische Probleme effizient auf solchen Graphklassen lösbar sind. Wir untersuchen die Frage, ob sich ein algorithmisches Metatheorem der Form „wenn eine Graphklasse X eine implizite Repräsentation mit Komplexität Y hat, dann ist Problem Z effizient auf X lösbar“ beweisen lässt. Es stellt sich schnell heraus, dass selbst für die bescheidenste Wahl von Y und verschiedene Z dies entweder unmöglich ist oder uns an die Grenzen der Forschung in der Algorithmik führt. Daher führen wir eine Fallstudie durch, wobei Z das Graphenisomorphieproblem ist und Y ein spezielles Fragment der Prädikatenlogik. Dies führt uns zum Isomorphieproblem für Kreisbogengraphen, welches seit mehr als zwei Jahrzenten trotz beachtlicher Anstrengungen nicht klassifiziert werden konnte. Wir schauen uns an, wie eine bestimmte Methode (Flip Trick) auf dieses Problem angewandt werden kann. Es stellt sich heraus, dass für eine große Klasse von Kreisbogengraphen das Isomorphieproblem auf das Repräsentationsproblem reduzierbar ist und somit in Polynomialzeit gelöst werden kann.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Finding efficient representations for various kinds of data is among the most fundamental tasks in computer science. Depending on the type of data and its intended purpose there are various constraints that have to be taken into consideration. The simplest scenario is that of archiving data. In this case the main concern is to use as few bits as possible to store the data, which can be achieved by a compression scheme. A drawback of compression schemes is that every time a computation needs to be performed on the data it has to be decompressed beforehand. While this is a reasonable limitation in the context of an archive it might not be if one expects to regularly perform computations on the data. To find a more suitable representation in that case it has to be specified what kind of operations need to be performed on the data. Then the goal is to find a representation which does not consume much space and where these operations can be performed quickly. A representation that realizes these requirements for a certain type of data and a prescribed set of operations is called a data structure.

An important type of data are relational structures. In this thesis we focus on graphs, which are a ubiquitous special case of relational structures. The data structures that we are interested in are only equipped with one operation: determine whether two given vertices are adjacent. We call this operation an edge query. If one wants to store arbitrary graphs then adjacency matrices are optimal in the following sense. First, querying an edge can be performed in constant-time in the RAM model. Secondly, adjacency matrices require n^2 bits to represent a graph on n vertices. This is asymptotically optimal because there are at least 2^{cn^2} graphs on n vertices for sufficiently large n and some $c > 0$. But, what if one only wants to represent a certain class of graphs such as interval graphs? An interval graph is a graph where every vertex can be assigned to a closed interval on the real line such that two vertices are adjacent if and only if their corresponding intervals intersect. It can be shown that there are at most $2^{\mathcal{O}(n \log n)}$ interval graphs on n vertices. Therefore an adjacency matrix is space-inefficient for such graphs. An optimal representation for interval graphs can be obtained as follows. For a given interval graph with n vertices consider its interval model, i.e. the set of n intervals that are associated with the vertices. Enumerate the endpoints of the intervals from left to right and label each vertex with the two endpoints of its interval. See Figure 1.1 for an example. Observe that the set of labels of the vertices are a representation of the graph because the interval model can be reconstructed from it. Moreover, to see whether two vertices are adjacent it suffices to inspect only their labels. A label requires $\log(4n^2) \leq 4 \log n$ bits and therefore an interval graph on n vertices can be stored using at most $4n \log n$ bits, which is asymptotically optimal. The idea behind this representation for interval graphs can be generalized as follows.

Let $\mathcal{C}$ be a graph class that has $2^{\Theta(nf(n))}$ graphs on n vertices for some function $f \in o(n)$; if f is not sublinear in n then adjacency matrices are already optimal. We want to find a

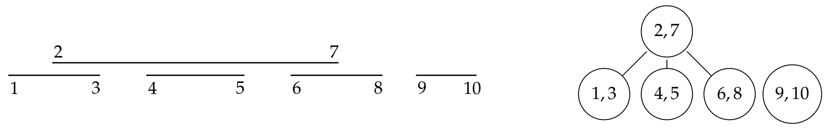


FIGURE 1.1: Interval model and the resulting labeling of the interval graph

decision algorithm A called label decoder algorithm and a constant $c \in \mathbb{N}$ called label length such that the following holds. For every graph G in $\mathcal{C}$ with n vertices there exists a function ℓ which maps the vertices of G to binary strings of length $cf(n)$ such that for all vertices $u \neq v$ of G it holds that u and v are adjacent iff A accepts the input $(\ell(u), \ell(v))$. The tuple (A, c) is called an implicit representation (or labeling scheme) for $\mathcal{C}$ and ℓ is called a labeling of G. Observe that the runtime of the label decoder A is the time required to perform an edge query. Due to the length of the labels ($cf(n)$) such a representation is asymptotically space-optimal. In this thesis we restrict ourselves to the case where $f(n) = \log n$ and call graph classes with at most $2^{\mathcal{O}(n \log n)}$ graphs on n vertices small. A lot of natural graph classes are small as we shall see.

The concept of implicit representations has been introduced in [Mul88] and [KNR92]. Arguably the most basic research question in this field is what graph classes admit an implicit representation. While quite a lot graph classes are known to have an implicit representation there is an absence of negative results. More precisely, for any graph class that satisfies a weak uniformity condition (which is implied by being hereditary) there has been no proof that this graph class does not have an implicit representation, even when assuming strong computational constraints on the label decoder algorithm. Stated differently, so far it was not known whether the computational aspect even matters with respect to what graph classes have an implicit representation. We demonstrate that there is indeed a complexity hierarchy for implicit representations. However, the graph classes used to demonstrate these separations are far removed from any natural graph class, which is unsatisfactory. Therefore we consider what algorithmic resources have been employed to show that graph classes have implicit representations. We observe that for many graph classes their simplest implicit representation is essentially a slight variation of the one used for interval graphs. This is formalized by the statement that interval graphs are complete for a class called $\mathsf{GFO}(<)$. This class contains almost all graph classes that are currently known to have an implicit representation. The first-order logic formalism that underlies the definition of $\mathsf{GFO}(<)$ supplies us with many other interesting sets of graph classes with implicit representations. A particularly notable one is $\mathsf{GFO}_{\mathrm{qf}}$. The class $\mathsf{GFO}_{\mathrm{qf}}$ contains $\mathsf{GFO}(<)$ and can be alternatively described as the set of graph classes that have an implicit representation with a constant-time label decoder algorithm. Therefore we deem it to be a particularly interesting task to find a small and hereditary graph class that is not in $\mathsf{GFO}_{\mathrm{qf}}$. Three prominent examples of small and hereditary graph classes not known to have implicit representations can be located in a generalization of $\mathsf{GFO}_{\mathrm{qf}}$ named $\mathsf{PBS}(\mathbb{N})$. Informally, $\mathsf{PBS}(\mathbb{N})$ is the class that one obtains from $\mathsf{GFO}_{\mathrm{qf}}$ when allowing arbitrarily long vertex labels. We introduce these and other complexity classes of labeling schemes as a way to study the limitations of implicit representations. Our considerations suggest that classically studied complexity classes such as P or even AC^0 are not suitable in that regard.

Another important problem in the context of implicit representations is to find a representation for a given graph. Consider an implicit representation (A, c) where A is the label decoder algorithm and c the label length. Let G be a graph on n vertices. We call a function ℓ which maps the vertices of G to strings of length $c \log n$ a representation of G with respect to (A, c) if for all pairs of vertices $u \neq v$ in G it holds that they are adjacent iff A accepts the input $(\ell(u), \ell(v))$. The representation problem for (A, c) is to find a representation of G with respect to (A, c) under the promise that it exists. The canonical representation problem for (A, c) has the additional requirement that whenever two graphs G, H are isomorphic then the images of their computed representations must coincide. For certain geometrical intersection graph classes such as interval graphs their implicit representation directly corresponds to their geometrical representation. As a consequence, certain choices for A and c lead to problems that haven been studied independently such as the (canonical) representation problem for interval graphs. Circular-arc (CA) graphs are a generalization of interval graphs where the vertices are mapped to arcs on the unit circle instead of intervals on the real line. Even though the descriptive complexity of their implicit representations are very similar, the tractability of the canonical representation problem for CA graphs remains an open question whereas the one for interval graphs is known to be solvable in logspace [Kö+11]. Intrigued by this discrepancy, we investigate a certain approach to finding canonical representation for CA graphs in Chapter 4.

Publications

The results of Chapter 3 and the beginning of Chapter 4 are based on the conference article [Cha16b] and its journal version [Cha17a], which has not been published yet. A novel contribution in this thesis is the characterization of the class $\mathsf{GFO}_{\mathrm{qf}}$ in terms of constant-time RAMs. The remainder of the thesis which deals with the canonical representation problem for circular-arc graphs is based on the conference article [Cha16a] and its journal version [Cha17b]. The content presented in Section 4.7 is novel as well and has not been published previously.

Chapter 2

Preliminaries

2.1 General Notation and Terminology

Let $\mathbb{N} = \{1, 2, \dots\}$ be the set of natural numbers and $\mathbb{N}_0$ is $\mathbb{N} \cup \{0\}$. For $n \in \mathbb{N}$ let $[n] = \{1, 2, \dots, n\}$ and let $[n]_0 = [n] \cup \{0\}$. We say two sets A, B overlap, in symbols $A \between B$, if $A \cap B$, $A \setminus B$ and $B \setminus A$ are non-empty. When we say $\log n$ we mean $\lceil \log_2 n \rceil$. Let $\exp(n) = 2^n$ and let $\exp^0(n) = n$ and $\exp^i(n) = \exp(\exp^{i-1}(n))$ for all $i \geq 1$. For a function f we write $\mathrm{Im}(f)$ and $\mathrm{dom}(f)$ to denote its image and domain, respectively. Let A be a family of sets over some ground set U, i.e. $A \subseteq \mathcal{P}(U)$. The set A is called Helly if for all subsets A' of A it holds that the overall intersection of A' is non-empty ($\cap_{x \in A'} x \neq \emptyset$) whenever the pairwise intersection of A' is non-empty (for all $x, y \in A'$ it holds that $x \cap y \neq \emptyset$). Let f be a k^2-ary Boolean function and let $A = (a_{i,j})_{i,j \in [k]}$ be a $(k \times k)$-matrix over $\{0, 1\}$ for some $k \in \mathbb{N}$. We write $f(A)$ to mean $f(a_{1,1}, a_{1,2}, \dots, a_{1,k}, a_{2,1}, \dots, a_{k,k})$, i.e. plugging in the values of A going from left to right and top to bottom. For two graphs G, H we write $G \cong H$ to denote that they are isomorphic and $G \subseteq H$ to mean that G is an induced subgraph of H. For a graph G and a subset of its vertices V' we write $G[V']$ to mean the subgraph of G which is induced by V'. We write $\overline{G}$ to denote the edge-complement of a graph G. The graph K_n denotes the complete graph on n vertices for $n \in \mathbb{N}$. We speak of G as unlabeled graph to emphasize that we talk about the isomorphism class of G rather than a specific adjacency matrix of G. A graph class is a set of finite and unlabeled graphs, i.e. it is closed under isomorphism. For a graph class $\mathcal{C}$ and $n \in \mathbb{N}$ we write $\mathcal{C}_n$ to denote the set of all graphs in $\mathcal{C}$ with n vertices. Similarly, we write $\mathcal{C}_{\leq n}$ to denote all graphs in $\mathcal{C}$ with at most n vertices. For a set of graph classes $\mathbb{A}$ we write $[\mathbb{A}]_{\subseteq}$ to denote its closure under subsets, i.e. $\{\mathcal{C} \subseteq \mathcal{D} \mid \mathcal{D} \in \mathbb{A}\}$.

2.2 Complexity Theory

We use the term complexity class informally to mean a countable set of languages with computational restrictions. Unless specified otherwise we consider languages over the binary alphabet $\{0, 1\}$. Let ALL denote the set of all languages. Let R denote the set of decidable languages. For a function $t \colon \mathbb{N} \to \mathbb{N}$ let $\mathsf{TIME}(t)$ denote the set of languages that can be decided by a deterministic Turing machine in time t. For a set of functions T where every $t \in T$ has signature $\mathbb{N} \to \mathbb{N}$ let $\mathsf{TIME}(T) = \cup_{t \in T} \mathsf{TIME}(t)$. Let P be the set of languages that can be decided in polynomial time, i.e. $\mathsf{P} = \mathsf{TIME}(n^{\mathcal{O}(1)})$. NP is the set of languages that can be decided in polynomial time by a non-deterministic Turing machine. PSPACE is the set of languages that can be decided in polynomial space by a deterministic Turing machine. The class $k\mathsf{EXP}$ is defined as $\mathsf{TIME}(\exp^k(n^{\mathcal{O}(1)}))$ for $k \geq 0$. This means $0\mathsf{EXP} = \mathsf{P}$. If $k = 1$ we simply write EXP. For a language L let NP^L denote the set of languages that can be decided by a non-deterministic Turing machine in polynomial time with an oracle for L; this means the Turing machine can decide oracle queries of the form '$x \in L$?' in constant-time (the length of x must be polynomially bounded in terms of the input length). For a set of languages A let NP^{A} be $\cup_{L \in \mathsf{A}} \mathsf{NP}^{L}$. The complexity class Σ_k^{P} is inductively defined as $\Sigma_1^{\mathsf{P}} = \mathsf{NP}$ and $\Sigma_{k+1}^{\mathsf{P}} = \mathsf{NP}^{\Sigma_k^{\mathsf{P}}}$ for $k \geq 1$. The polynomial hierarchy PH is $\cup_{k \in \mathbb{N}} \Sigma_k^{\mathsf{P}}$.

A logspace transducer is a deterministic Turing machine M with a read-only input tape, a work tape and a write-only output tape. The work tape is only allowed to use at most $\mathcal{O}(\log n)$ cells where n denotes the input length. To write onto the output tape M has a designated state called output state with the following semantic. If M enters the output state then the symbol in the current cell of the work tape is written to the current cell of the output tape and the head on the output tape is moved one cell to the right. Other than that, M cannot write or move the head on the output tape. This means as soon as something is written to the output tape it cannot be modified afterwards. Let Σ and Γ be the input and work alphabet of M respectively. Then M computes a function $f_M \colon \Sigma^* \to \Gamma^*$. We say a (partial) function f is computed by a logspace transducer M if $f(x) = f_M(x)$ whenever $f(x)$ is defined. We call f logspace-computable if there exists a logspace transducer M which computes f. The class of logspace-computable functions is closed under composition. Let f be a function which maps words over some alphabet to words over some other alphabet. We say that the length of f is polynomially bounded if $|f(x)|$ is polynomially bounded by $|x|$. Only functions whose length is polynomially bounded can be logspace-computable since the runtime of a logspace transducer is polynomially bounded. A language is in logspace if its characteristic function is logspace-computable. Given two functions f and g we say f is logspace-reducible to g if it can be shown that f is logspace-computable under the premise that g is logspace-computable. Intuitively, this means that a logspace transducer which computes g can be used as a subroutine when constructing a logspace transducer for f. Even though it is in general not possible to write $g(x)$ for some word x to the work tape because it might be too long, it suffices to compute single bits of $g(x)$ 'on the fly' as they are needed. Analogously, given three functions f, g, h we say f is logspace-reducible to g and h if f can be shown to be logspace-computable under the premise that g and h are logspace-computable. For two functions f and g we say that they are logspace-equivalent if f is logspace-reducible to g and vice versa. We remark that this is a more general definition than what is usually termed logspace reduction. However, since we only use logspace reductions as a tool to design logspace algorithms in a structured manner the level of detail of this definition suffices.

In the following we briefly define Boolean circuits and the two complexity classes AC^0 and TC^0 (we consider the logspace-uniform variant of these classes here). We consider a (Boolean) circuit C with n input bits $x_1, \ldots, x_n$ to be a directed acyclic graph (DAG) where all vertices with in-degree zero are called input gates and are labeled with $x_1, \ldots, x_n$ and the other vertices are labeled with conjunction '$\wedge$', disjunction '$\vee$' and negation '$\neg$'. Additionally, C has a single designated output gate which allows us to naturally interpret C as an n-ary Boolean function f_C (this natural interpretation requires that every negation has in-degree one). For $\vec{a} \in \{0,1\}^n$ we say $C(\vec{a})$ is the value $f_C(\vec{a})$ computed by C on input $\vec{a}$. The size of a circuit C is the number of non-input gates and the depth of C is the length of a longest directed path in the underlying DAG of C. We say a family of circuits $(C_n)_{n \in \mathbb{N}}$ where C_n has n input gates decides a language $L \subseteq \{0,1\}^*$ if $x \in L$ iff $C_n(x) = 1$ for all $x \in \{0,1\}^n$ and $n \in \mathbb{N}$. We say a family of circuits $(C_n)_{n \in \mathbb{N}}$ is a family of AC^0-circuits if there exist $c, d \in \mathbb{N}$ such that the size of C_n is at most n^c and the depth of C_n is at most d

for all $n \in \mathbb{N}$ (polynomial size and constant depth). Suppose that besides '$\wedge$', '$\vee$' and '$\neg$' we allow non-input gates to be labeled with 'MAJ' which has the following interpretation. If a vertex/gate z is labeled with 'MAJ' and has in-neighbors $y_1, \ldots, y_k$ then z outputs one iff the majority of $y_1, \ldots, y_k$ outputs one. We say a family of circuits is a family of TC^0-circuits if it satisfies the same condition as a family of AC^0-circuits but can additionally use majority gates. Unlike a Turing machine a family of circuits is an infinite object. In order to guarantee that such a family has a finite description a so-called uniformity condition must be imposed. A family of circuits $(C_n)_{n \in \mathbb{N}}$ is logspace-uniform if there exists a logspace transducer M such that on input n in unary it outputs a reasonable representation of C_n. Let AC^0 (TC^0) be the set of languages that can be decided by a logspace-uniform family of AC^0-circuits (TC^0-circuits). It holds that TC^0 is a subset of logspace.

Let P denote a predicate such as being polynomial-time computable and let L and L' be languages over the alphabets Σ and Δ, respectively. We say L is P many-one reducible to L' if there exists a P function $f \colon \Sigma^* \to \Delta^*$ such that $x \in L$ iff $f(x) \in L'$ for all $x \in \Sigma^*$. The complexity class AC^0 can be lifted to a class of functional problems by allowing circuits to have more than one output gate. This allows us to talk about AC^0 many-one reducibility.

2.3 First-Order Logic

Let $\mathcal{N}$ be the structure that has $\mathbb{N}_0$ as universe equipped with the order relation '$<$' and addition '$+$' and multiplication '$\times$' as functions. For $n \geq 1$ let $\mathcal{N}_n$ be the structure that has $[n]_0 = \{0, 1, \ldots, n\}$ as universe, the order relation '$<$' and addition as well as multiplication defined as:

$$+(x,y) = \begin{cases} x+y & ,\text{if } x+y \leq n \\ 0 & ,\text{if } x+y > n \end{cases}, \quad \times(x,y) = \begin{cases} xy & ,\text{if } xy \leq n \\ 0 & ,\text{if } xy > n \end{cases}$$

For $\sigma \subseteq \{<, +, \times\}$ let $\mathsf{FO}_k(\sigma)$ be the set of first-order formulas with Boolean connectives $\neg, \vee, \wedge$, quantifiers $\exists, \forall$ and k free variables using only equality and the relation and function symbols from σ. If $\sigma = \{<, +, \times\}$ we simply write FO_k and if $\sigma = \emptyset$ we write $\mathsf{FO}_k(=)$. A formula is called an atom if it contains no Boolean connectives and no quantifiers. For a formula φ with a atoms let us call the a-ary Boolean function that results from replacing every atom in φ by a proposition underlying Boolean function of φ. Let $\mathrm{Vars}(\varphi)$ be the set of free variables in φ. Given $\varphi \in \mathsf{FO}_k(\sigma)$, $\mathrm{Vars}(\varphi) = (x_1, \ldots, x_k)$ and an assignment $a_1, \ldots, a_k \in [n]_0$ we write $\mathcal{N}_n, (a_1, \ldots, a_k) \models \varphi$ if the interpretation $\mathcal{N}_n, (a_1, \ldots, a_k)$ satisfies φ under the semantics of first-order logic.

Let φ be a formula in FO_k. We define the bounded model checking problem for φ as follows. On input $a_1, \ldots, a_k, n \in \mathbb{N}$ with $a_i \in [n]_0$ for $i \in [k]$ decide whether $\mathcal{N}_n, (a_1, \ldots, a_k) \models \varphi$. We assume that the input is encoded in binary.

2.4 Graph Theory

We consider an undirected graph to be a special case of a directed graph with symmetric edge relation. For the first part of this work we say graph to mean a directed graph, unless specified otherwise. Starting from Section 4.2 we say graph to mean an undirected graph. Most of the time this should be clear from the context, i.e. when we write $\{u, v\}$ as opposed to (u, v) for an edge then this implies that we are looking at an undirected graph. At certain points there might be more than one sensible interpretation. For example, suppose there are two sets of graph classes $\mathbb{A}$ and $\mathbb{B}$ with $\mathbb{A} \subseteq \mathbb{B}$ and $\mathbb{A}$ only contains undirected graph classes whereas $\mathbb{B}$ also contains directed graph classes. It trivially follows that $\mathbb{A} \subsetneq \mathbb{B}$. However, the more interesting question is whether $\mathbb{B}$ also contains an undirected graph class which is not in $\mathbb{A}$. If we do not know this we shall not consider $\mathbb{A}$ to be a proper subset of $\mathbb{B}$. We write $\mathcal{G}$ to denote the set of all graphs or the set of all undirected graphs, depending on the context.

2.4.1 Neighborhoods

For a directed graph G and a vertex u let $N_{\text{in}}(u)$ and $N_{\text{out}}(u)$ denote the in- and out-neighbors of u in G. The following definitions are meant w.r.t. an undirected graph G. For a vertex v of G we define its open neighborhood $N(v)$ as the set of vertices which are adjacent to v and its closed neighborhood $N[v] = N(v) \cup \{v\}$. A vertex v is called universal if $N[v] = V(G)$. A vertex is called isolated if it has no neighbors. We will use two different notions of twins. In Chapter 3 we call a pair of vertices twins if $N(u) \setminus \{v\} = N(v) \setminus \{u\}$. In Chapter 4 we call a pair of vertices twins if $N[u] = N[v]$. The latter notion is more restrictive in the sense that every equivalence class must be a clique whereas in the former case it can either be a clique or an independent set. A graph G is twin-free if for every pair of distinct vertices it holds that they are not twins. A twin class is an equivalence class of the twin relation. For two subsets of vertices S, S' with $S' \subseteq S$ we define the exclusive neighborhood $N_S(S')$ as all vertices $v \in V(G) \setminus S$ such that v is adjacent to all vertices in S' and to none in $S \setminus S'$.

2.4.2 Isomorphism and Invariance

We call a bijective function τ which maps the vertices of a graph G to some set V' a relabeling of G and $\tau(G)$ denotes the graph obtained after relabeling the vertices of G according to τ. Let G and H be two graphs and let $X \subseteq V(G)$ and $Y \subseteq V(H)$. We say X and Y are in the same orbit, in symbols $X \sim_{\text{orb}} Y$, if there exists an isomorphism π from G to H such that $\pi(X) = Y$. Let f be a function which maps a graph along with a subset of its vertex set to a binary string, i.e. $f(G, X) \in \{0,1\}^*$ and $X \subseteq V(G)$. We call f an invariant for a graph class $\mathcal{C}$ if $f(G, X) = f(H, Y)$ whenever $X \sim_{\text{orb}} Y$ and $G, H \in \mathcal{C}$. Let us call a function f which maps a graph G to a set of subsets of its vertices, i.e. $f(G) \subseteq \mathcal{P}(V(G))$, a vertex set selector. For example, the function that maps a graph to the set of its cliques is a vertex set selector. The characteristic function χ_f of a vertex set selector f is defined as $\chi_f(G, X) = 1 \Leftrightarrow X \in f(G)$. We say a vertex set selector f is invariant for a graph class $\mathcal{C}$ if its characteristic function χ_f is an invariant for $\mathcal{C}$. We call f globally invariant if χ_f is an invariant for all graphs. Intuitively, a vertex set selector f is invariant for $\mathcal{C}$ if a graph $G \in \mathcal{C}$ can be arbitrarily relabeled and f still returns the 'same' vertex sets as before w.r.t. $\sim_{\text{orb}}$.

2.4.3 Graph Classes, Graph Parameters and Graph Class Properties

Graph Class Properties. Let $\mathcal{C}$ be a graph class. $\mathcal{C}$ is small if $|\mathcal{C}_n| \in n^{\mathcal{O}(n)} = 2^{\mathcal{O}(n \log n)}$ (in the literature this is also called factorial speed of growth). $\mathcal{C}$ is tiny if there exists a $c < \frac{1}{2}$ such that $|\mathcal{C}_n| \leq n^{cn}$ for all sufficiently large n. $\mathcal{C}$ is hereditary if it is closed under taking induced subgraphs, i.e. if G is in $\mathcal{C}$ then every induced subgraph of G must be in $\mathcal{C}$. We write $[\mathcal{C}]_{\subseteq}$ to denote the hereditary closure of $\mathcal{C}$, i.e. $[\mathcal{C}]_{\subseteq}$ is the set of graphs that occur as induced subgraph of some graph in $\mathcal{C}$. $\mathcal{C}$ is sparse if there exists a $c \in \mathbb{N}$ such that every graph in $\mathcal{C}_n$ has at most cn edges for all $n \in \mathbb{N}$. $\mathcal{C}$ is uniformly sparse if it is a subset of a hereditary and sparse graph class. Stated differently, $\mathcal{C}$ is uniformly sparse iff it is in $[\text{Sparse} \cap \text{Hereditary}]_{\subseteq}$. For example, planar graphs are uniformly sparse. $\mathcal{C}$ is inflatable if for every graph G in $\mathcal{C}$ with n vertices there exists a graph H in $\mathcal{C}$ with m vertices which contains G as induced subgraph for all $n < m \in \mathbb{N}$. Every graph class that is closed under adding isolated vertices is inflatable. $\mathcal{C}$ is self-universal if for every finite subset X of $\mathcal{C}$ there exists a graph G in $\mathcal{C}$ which contains every graph in X as induced subgraph. Every graph class that is closed under disjoint union is self-universal.

Fact 2.1. *$\mathcal{C}$ is hereditary and self-universal iff there exists a (possibly infinite) graph G such that a graph H is in $\mathcal{C}$ iff H is an induced subgraph of G.*

Proof. If $\mathcal{C}$ is finite this equivalence is trivial. Suppose that $\mathcal{C}$ is infinite. If $\mathcal{C}$ is hereditary and self-universal we can build G inductively as follows. Let G_1 be the single vertex graph. Let G_{i+1} be a graph in $\mathcal{C}$ which contains all graphs with at most $i+1$ vertices in $\mathcal{C}$ and G_i for all $i \geq 1$. It can be assumed that G_i has vertex set $[n_i]$ where $n_i = |V(G_i)|$ and that the identity function on $[n_i]$ witnesses that G_i is an induced subgraph of G_{i+1} for all $i \geq 1$. Then G has vertex set $\mathbb{N}$ and $E(G) = \cup_{i \in \mathbb{N}} E(G_i)$. It is not difficult to see that H is in $\mathcal{C}$ iff H is an induced subgraph of G. The other direction is trivial. □

It follows that every infinite, hereditary, self-universal graph class $\mathcal{C}$ is inflatable. An undirected graph H is called a minor of an undirected graph G if H can be obtained from G by deleting vertices and edges, and contracting edges (merging two adjacent vertices into one vertex which inherits the neighbors of the two old vertices). $\mathcal{C}$ is minor-closed if every graph that occurs as minor of some graph in $\mathcal{C}$ is in $\mathcal{C}$ as well. For a graph G we call a graph class where G does not occur as minor G-minor free. For a graph class $\mathcal{C}$ let $\text{MF}(\mathcal{C})$ denote the set of graph classes that are G-minor free for some G in $\mathcal{C}$. Unlike the previous properties being minor-closed only applies to undirected graph classes.

Geometrical Intersection Graph Classes. Let $\mathcal{F}$ be a family of sets over some ground set. For a finite multisubset X of $\mathcal{F}$ (X can contain the same element more than once) the intersection graph $G(X)$ of X has X as vertex set and two vertices $u, v \in X$ are adjacent iff u and v have non-empty intersection. The set of graphs $G(X)$ where X is a finite multisubset of $\mathcal{F}$ is called intersection graph class of $\mathcal{F}$. Similarly, the set of graphs $G(X)$ where X is a finite subset of $\mathcal{F}$ is called unique intersection graph class of $\mathcal{F}$. A graph class $\mathcal{C}$ is called (unique) intersection graph class if there exists a family of sets $\mathcal{F}$ such that $\mathcal{C}$ is the (unique) intersection graph class of $\mathcal{F}$. Due to the symmetry of intersection only undirected graph classes can be intersection graph classes. An undirected graph class is a unique intersection graph class iff it is hereditary and self-universal due to Fact 2.1.

Interval graphs are the (unique) intersection graph class of intervals on the real line. Circular-arc (CA) graphs are the intersection graph class of arcs on the unit circle. They are a generalization of interval graphs. k-interval graphs are the intersection graph class of the union of k intervals on the real line for $k \in \mathbb{N}$. The minimal $k \in \mathbb{N}$ such that a given graph G is a k-interval graph is called the interval number of G. The boxicity of a graph G is the minimal $k \in \mathbb{N}$ such that it can be described as intersection graph of k-dimensional axis-aligned boxes. A graph has boxicity 1 iff it is an interval graph. A disk graph is the intersection graph of disks in the plane. A graph is a k-ball graph if it is the intersection graph of k-dimensional balls embedded in $\mathbb{R}^k$. Disk graphs are 2-ball graphs. A graph is a kd-line segment graph if it is the intersection graph of line segments embedded in $\mathbb{R}^k$. We call 2d-line segment graphs just line segment graphs. A chord of a circle C is a line segment whose endpoints lie on C. A graph is a circle graph if it is the intersection graph of chords of a circle. Clearly, circle graphs are a subset of line segment graphs.

For a family of sets $\mathcal{F}$ let P be a predicate on finite subsets of $\mathcal{F}$. Let us call the set of graphs $G(X)$ where X is a finite (multi)subset of $\mathcal{F}$ and X satisfies P the P-intersection graph class of $\mathcal{F}$. Commonly considered choices for P include being proper (there are no two elements x, y in X such that $x \subseteq y$), being unit (every element of X has unit size; this requires $\mathcal{F}$ to have some size measure) and being Helly (see Section 2.1 for the definition). For example, proper interval graphs are the proper-intersection graph class of intervals on the real line (no interval is contained in another one). This should make it clear what we mean by, for instance, unit disk graphs or Helly CA graphs.

Other Graph Classes and Parameters. A graph parameter λ is a function which maps graphs to natural numbers. We say a graph class $\mathcal{C}$ is bounded by λ if there exists a $c \in \mathbb{N}$ such that $\lambda(G) \leq c$ for all $G \in \mathcal{C}$. For example, planar graphs have bounded chromatic number. We sometimes view a graph parameter λ as the set of graph classes $\mathcal{C}(\lambda)$ which are bounded by it. We say two parameters λ and μ are equivalent if they bound the same set of graph classes, i.e. $\mathcal{C}(\lambda) = \mathcal{C}(\mu)$.

A graph G is called a k-dot product graph if each vertex u of G can be assigned k real numbers $u_1, \ldots, u_k$ such that two vertices u, v of G are adjacent iff the dot product of $(u_1, \ldots, u_k)$ and $(v_1, \ldots, v_k)$ is at least one ($\Sigma_{i=1}^{k} u_i \cdot v_i \geq 1$) for $k \in \mathbb{N}$ [Fid+98]. The least $k \in \mathbb{N}$ such that a given graph G is a k-dot product graph is called dot product dimension of G. The twin index of an undirected graph G is its number of twin classes (here by twin we mean $N(u) \setminus \{v\} = N(v) \setminus \{u\}$). The intersection number of a graph G is the least $k \in \mathbb{N}$ such that there exists a family of sets $\mathcal{F}$ over a ground set U with k elements with $|\mathcal{F}| = |V(G)|$ and G is isomorphic to the intersection graph of $\mathcal{F}$. The degeneracy of a graph G is the least $k \in \mathbb{N}$ such that every induced subgraph of G contains a vertex of degree at most k. For example, every forest has degeneracy 1 because it either has a leaf or every vertex is isolated. The arboricity of a graph G is the least $k \in \mathbb{N}$ such that there are k forests $F_1, \ldots, F_k$ with the same vertex set as G such that $E(G) = \cup_{i \in [k]} E(F_i)$. The thickness of a graph G is the least $k \in \mathbb{N}$ such that there are k planar graphs $H_1, \ldots, H_k$ with the same vertex set as G such that $E(G) = \cup_{i \in [k]} E(H_i)$. It is well-known that arboricity, thickness and degeneracy are equivalent; they bound exactly the set of uniformly sparse graph classes. We remark that we will also talk about the parameters tree-width and clique-width. Since we do not require their definition in our proofs we omit them.

2.4.4 CA Models, Representations and Matrices

An arc is a connected and closed set of points on the (unit) circle. A CA model is a set of arcs $\mathcal{A} = \{A_1, \ldots, A_n\}$. Let $p \neq p'$ be two points on the circle. Then the arc A specified by $[p, p']$ is given by the part of the circle that is traversed when starting from p going in

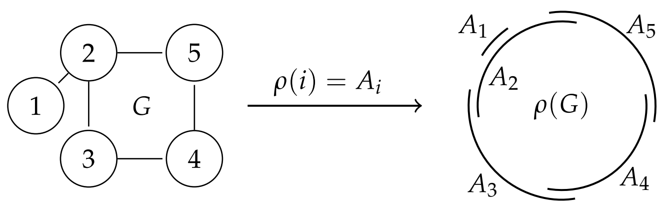


FIGURE 2.1: A CA graph and a representation of it

clockwise direction until p' is reached. We say that p is the left and p' the right endpoint of A and write $l(\cdot), r(\cdot)$ to denote the left and right endpoint of an arc in general. If $A = [p, p']$ then the arc obtained by swapping the endpoints $\overline{A} = [p', p]$ covers exactly the opposite part of the circle plus the endpoints. We say $\overline{A}$ is obtained by flipping A. In our context, we are only interested in the intersection structure of a CA model and thus only the relative position of the endpoints to each other matter. All endpoints can w.l.o.g. be assumed to be pairwise different and no arc covers the full circle. Under these assumptions, a CA model $\mathcal{A}$ with n arcs can be described as a unique string as follows. Pick an arbitrary arc $A \in \mathcal{A}$ and relabel the arcs with $1, \dots, n$ in order of appearance of their left endpoints when traversing the circle clockwise starting from the left endpoint of A. Then write down the endpoints in order of appearance when traversing the circle clockwise starting from the left endpoint of A. Do this for every arc and pick the lexicographically smallest resulting string as representation for $\mathcal{A}$. For example, the smallest such string for the CA model in Figure 2.1 would result from choosing A_1: $(l(1), r(1), l(2), r(5), l(3), r(2), \dots)$. Let $\text{str}(\mathcal{A})$ denote this smallest string representation. For a CA model $\mathcal{A}$ let $\mathcal{A}^{\text{r}}$ be the CA model obtained after reversing the order of its endpoints. Observe that reversing the endpoints does not affect the intersection structure of a CA model. Therefore we consider two CA models $\mathcal{A}$ and $\mathcal{B}$ to be equal if $\text{str}(\mathcal{A}) = \text{str}(\mathcal{B})$ or $\text{str}(\mathcal{A}^{\text{r}}) = \text{str}(\mathcal{B})$.

Let G be a graph and $\rho = (\mathcal{A}, f)$ consists of a CA model $\mathcal{A}$ and a bijective mapping f from the vertices of G to the arcs in $\mathcal{A}$. Then ρ is called a CA representation of G if for all $u \neq v \in V(G)$ it holds that $\{u, v\} \in E(G) \Leftrightarrow f(u) \cap f(v) \neq \emptyset$. We write $\rho(x)$ to mean the arc $f(x)$ corresponding to the vertex x, $\rho(G)$ for the CA model $\mathcal{A}$ and for a subset $V' \subseteq V(G)$ let $\rho[V'] = \{\rho(v) \mid v \in V'\}$. A graph is a CA graph if it has a CA representation.

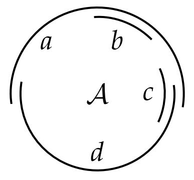
We say a CA model $\mathcal{A}$ has a hole if there exists a point on the circle which isn't contained by any arc in $\mathcal{A}$. Every such CA model can be understood as interval model (a set of intervals on the real line) by straightening the arcs. Conversely, every interval model can be seen as CA model by bending the intervals. Therefore a graph is an interval graph iff it admits a CA representation with a hole.

A family of sets $\mathcal{F}$ over some ground set is called Helly if for all subsets $\mathcal{F}'$ of $\mathcal{F}$ such that all elements in $\mathcal{F}'$ intersect pairwise it holds that $\cap_{A \in \mathcal{F}'} A$ is non-empty. A CA graph G is called Helly (HCA graph) if it has a CA representation ρ with a Helly CA model $\rho(G)$. This is the case iff for all inclusion-maximal cliques C in G it holds that the overall intersection of C in ρ is non-empty, i.e. $\bigcap_{v \in C} \rho(v) \neq \emptyset$. Every interval model has the Helly property and therefore every interval graph is a Helly CA graph.

The intersection type of two circular arcs A and B can be one of the following five types:

- `di`: A and B are disjoint — $A \cap B = \emptyset$
- `cs`: A contains B — $B \subset A$
- `cd`: A is contained by B — $A \subset B$
- `cc`: A and B jointly cover the circle (circle cover) — $A \between B$ and $A \cup B =$ whole circle
- `ov`: A and B overlap — $A \between B$ and $A \cup B \neq$ whole circle

Using these types we can associate a matrix with every CA model. An intersection matrix is a square matrix with entries $\{\texttt{cc}, \texttt{cd}, \texttt{cs}, \texttt{di}, \texttt{ov}\}$. Given a CA model $\mathcal{A}$ we define its intersection matrix $\mu_{\mathcal{A}}$ such that $(\mu_{\mathcal{A}})_{A,B} \in \{\texttt{cc}, \texttt{cd}, \texttt{cs}, \texttt{di}, \texttt{ov}\}$ reflects the intersection type of the arcs $A \neq B \in \mathcal{A}$. An intersection matrix μ is called a CA (interval) matrix if it is the intersection matrix of some CA model (with a hole). See Figure 2.2 for an example of a



$\mu_{\mathcal{A}}$	a	b	c	d
a	-	`cs`	`ov`	`cc`
b	`cd`	-	`di`	`di`
c	`ov`	`di`	-	`ov`
d	`cc`	`di`	`ov`	-

FIGURE 2.2: A CA model $\mathcal{A}$ and its intersection matrix $\mu_{\mathcal{A}}$

CA model and the CA matrix which it induces. Given an intersection matrix μ and two distinct vertices u, v of μ we sometimes write $u \; \alpha \; v$ instead of $\mu_{u,v} = \alpha$ if μ is clear from the context. Also, we will talk about an intersection matrix μ as if it were a graph and call its indices vertices. We consider two vertices u, v of μ to be adjacent if they do not have a `di`-entry in μ. For a vertex u of an intersection matrix μ and $\alpha \in \{\texttt{cc}, \texttt{cd}, \texttt{cs}, \texttt{di}, \texttt{ov}\}$ we write $N^\alpha(u)$ to denote the set of vertices v of μ such that $\mu_{u,v} = \alpha$; we also call a vertex in $N^\alpha(v)$ an α-neighbor of v.

2.5 Labeling Schemes

We use the terms implicit representation and labeling scheme interchangeably.

Definition 2.2. *A labeling scheme is a tuple $S = (F, c)$ where $F \subseteq \{0,1\}^* \times \{0,1\}^*$ is called label decoder and $c \in \mathbb{N}$ is the label length. A graph G on n vertices is in the class of graphs spanned by S, denoted by $G \in gr(S)$, if there exists a labeling $\ell \colon V(G) \to \{0,1\}^{c \log n}$ such that for all $u \neq v \in V(G)$:*

$$(u, v) \in E(G) \Leftrightarrow (\ell(u), \ell(v)) \in F$$

We say a graph class $\mathcal{C}$ is represented by (or has) a labeling scheme S if $\mathcal{C} \subseteq gr(S)$.

The labeling scheme for interval graphs that we have seen in the introduction can be formalized as follows. We define the label decoder F_{Intv} such that (x_1x_2, y_1y_2) is in F_{Intv} iff neither x_2 is (lexicographically) smaller than y_1 nor y_2 is smaller than x_1 for all $x_1, x_2, y_1, y_2 \in \{0,1\}^m$ and $m \in \mathbb{N}$. If x_1, x_2, y_1, y_2 are interpreted as natural numbers and we assume that $x_1 \leq x_2$ and $y_1 \leq y_2$ then the label decoder says that neither of the two intervals $[x_1, x_2]$, $[y_1, y_2]$ ends before the other one starts; this means that they must intersect. The label length in this case is $c = 4$ because two numbers from $[2n]$ can be encoded using $4 \log n$ bits for $n \geq 2$. The labeling scheme $(F_{\text{Intv}}, 4)$ represents interval graphs.

Labeling schemes are intimately connected to polynomial universal graphs. A graph class $\mathcal{C}$ has polynomial universal graphs if there exists a family of graphs $(G_n)_{n \in \mathbb{N}}$ such that every graph in $\mathcal{C}$ with n vertices occurs as induced subgraph in G_n for all $n \in \mathbb{N}$ and $|V(G_n)|$ is polynomially bounded in n. A graph class has a labeling scheme iff it has polynomial universal graphs. If a graph class $\mathcal{C}$ has a labeling scheme (F, c) then it has polynomial universal graphs $(G_n)_{n \in \mathbb{N}}$ where G_n has vertex set $\{0,1\}^{c \log n}$ and the edges are determined by the label decoder F. Conversely, if $\mathcal{C}$ has polynomial universal graphs $(G_n)_{n \in \mathbb{N}}$ with $|V(G_n)| \leq n^c$ for some $c \in \mathbb{N}$ then this can be used to construct the following labeling scheme $S = (F, c+1)$ for $\mathcal{C}$. The label decoder F is defined as follows. It holds that $(x_1x_2, y_1y_2) \in F$ iff $(x_1, y_1) \in E(G_n)$ where $\text{bin}(n) = x_2$ for all $x_1, y_1 \in \{0,1\}^{cm}$ and $x_2, y_2 \in \{0,1\}^m$ for all $m \in \mathbb{N}$. It remains to argue that S represents $\mathcal{C}$. Given a graph $H \in \mathcal{C}$ with n vertices we know that it is an induced subgraph of G_n which is witnessed by some injective function $\pi \colon V(H) \to V(G_n)$. Since $|V(G_n)| \leq n^c$ we can assume w.l.o.g. that $V(G_n)$ is a subset of $\{0,1\}^{c \log n}$. The labeling which assigns each vertex u of H the label $\pi(u)\text{bin}(n)$ shows that H is in $\text{gr}(S)$. The number of vertices is part of the label because it ensures that for all $n \neq m \in \mathbb{N}$ the graphs G_n and G_m are encoded in disjoint subsets of the label decoder F.

Definition 2.3. *A language $L \subseteq \{0,1\}^*$ induces the following label decoder F_L. For all $x, y \in \{0,1\}^*$ it holds that $(x,y) \in F_L$ iff $xy \in L$ and $|x| = |y|$. For a set of languages* A *we say that a graph class $\mathcal{C}$ is in* GA *if there exists a language $L \in \mathsf{A}$ and $c \in \mathbb{N}$ such that $\mathcal{C}$ is represented by the labeling scheme (F_L, c).*

We say a labeling scheme $S = (F, c)$ is in GA if there exists a language L in A such that $F = F_L$.

For every set of languages A the set of graph classes GA is trivially closed under subsets. Also, if A is closed under complement then GA is closed under edge-complement. A graph class has an implicit representation iff it is in GALL (no computational complexity constraint).

It is not difficult to see that there is a language L in AC^0 such that $F_L = F_{\text{Intv}}$ and therefore interval graphs are in GAC^0. It is also an easy exercise to show that forests, circle graphs and all graph classes with bounded interval number, arboricity or boxicity are in GAC^0.

Chapter 3

A Complexity Theory for Implicit Representations

The concept of labeling schemes, or implicit representations, was introduced in [Mul88], and independently in [KNR92]. In this chapter we shall deal with the question of what graph classes admit such a representation when subjected to computational constraints. For many graph classes it has been established that they possess an implicit representation. To the best of our knowledge, if a graph class is known to have a labeling scheme then this is due to one of the following four reasons[1]. (I) The graph class is uniformly sparse. Every such graph class has a quite simple labeling scheme, which we introduce in Section 3.2. Graph classes which fall into this category include all proper minor-closed graph classes such as planar graphs or the ones with bounded tree-width. (II) The graph class is an intersection graph class and its intersection model can be directly translated into a labeling scheme. Examples in this category include k-interval graphs, circle graphs and permutation graphs. (III) The graph class has a tree-based decomposition which can be encoded in a labeling scheme. An example of this are graph classes with bounded clique-width as we shall see in Section 3.2. The last reason is trivial but important to be aware of. (IV) The graph class is a subset of a graph class which falls into one of the previous three categories.

If one asks what graph classes do not have a labeling scheme then nothing but the following observation has been known so far, which is based on the fact that non-small graph classes cannot have a labeling scheme[2].

Fact 3.1. *Let $\mathcal{C}$ be a graph class. For $k \in \mathbb{N}$ let $[\mathcal{C}]^k_{\subseteq}$ denote the set of graphs with n vertices which occur as induced subgraph of some graph in $\mathcal{C}$ with at most n^k vertices for all $n \in \mathbb{N}$. If $\mathcal{C}$ is in* GALL *then $[\mathcal{C}]^k_{\subseteq}$ is in* GALL *for all $k \in \mathbb{N}$.*

Proof. Assume that $\mathcal{C}$ is in GALL via the polynomial universal graphs $(G_n)_{n \in \mathbb{N}}$. This means that there exists a $d \in \mathbb{N}$ such that $|V(G_n)| \leq n^d$ holds for all $n \in \mathbb{N}$. Let $k \in \mathbb{N}$. It holds that $[\mathcal{C}]^k_{\subseteq}$ has polynomial universal graphs $(H_n)_{n \in \mathbb{N}}$ where H_n is the disjoint union of $G_n, \ldots, G_{n^k}$. Observe that $|V(H_n)| = |V(G_n)| + \cdots + |V(G_{n^k})| \leq \sum_{i=n}^{n^k} i^d$ which is polynomially bounded in n due to Faulhaber's formula. □

The classes $[\mathcal{C}]^k_{\subseteq}$ can be seen as polynomially bounded closure under induced subgraphs. If it can be shown for some $k \in \mathbb{N}$ that $\mathcal{C}_k$ is not small then this implies that $\mathcal{C}$ is

[1] Any statement which tries to summarize the current state of knowledge, or what is unknown, runs into danger of being incorrect simply because the author is not aware of every relevant result. In the following we omit phrases such as 'to the best of our knowledge' since we trust the reader to recognize that kind of statement and mentally add this qualifier.

[2] The concept of labeling schemes can be generalized to graph classes of all sizes. However, here we shall restrict ourselves to small graph classes since they already provide us with a rich playground.

not in GALL. For example, consider the set of graphs $\mathcal{C}$ which have not more edges than vertices. Every graph G with n vertices occurs as induced subgraph of some graph of $\mathcal{C}$ with at most n^2 vertices because G can have at most n^2 edges. Stated differently, the set of all graphs is a subset of $[\mathcal{C}]^2_\subseteq$ and thus $\mathcal{C}$ cannot be in GALL. This is a rather dull reason which only tells us that certain graph classes are non-small classes in disguise. To exclude such disguised graph classes one can consider only hereditary graph classes $\mathcal{C}$ because in that case $\mathcal{C} = [\mathcal{C}]_\subseteq = \cup_{k\in\mathbb{N}}[\mathcal{C}]^k_\subseteq$. In the following we sometimes say natural instead of hereditary because most graph classes of interest are hereditary, or their hereditary closure is considered to be natural. Kannan, Naor and Rudich asked whether every small and hereditary graph class has a labeling scheme with a polynomial-time computable label decoder [KNR92]. This was restated as conjecture by Spinrad [Spi03, p. 19]:

Conjecture 3.2 (Implicit Graph Conjecture)**.** *Every small and hereditary graph class is in* GP*.*

We remark that it is not even known whether every small and hereditary graph class is in GALL, which is a purely graph-theoretical question. Therefore a proof of this conjecture would at least require major improvements of our understanding of the set of small and hereditary graph classes. Disproving this conjecture would require us to comprehend the power of polynomial-time computable label decoders. However, until now the impact of computational constraints on labeling schemes has been uncharted territory. For example, it was not even known whether GAC^0 is a strict subset of GALL. In Section 3.1 we show that there is a strict complexity hierarchy starting from exponential time label decoders and thus we minimize the gap of knowledge to the question of whether $\mathsf{GAC}^0 = \mathsf{GEXP}$. We are not even aware of a graph class that could serve as a candidate to separate these two classes. Stated differently, all natural graph classes which are known to have a labeling scheme so far can be found in GAC^0.

We think that there currently is no realistic route to resolving the implicit graph conjecture. However, we believe that the broader question which underlies this conjecture deserves attention. Namely, how does the computational complexity of label decoders affect what natural graph classes can be represented? In the following, we aim to develop formal concepts (classes and reductions) that allow for a meaningful study of this question.

3.1 Hierarchy of Labeling Schemes

When labeling schemes were first introduced by Muller in [Mul88] the label decoder was required to be computable. Clearly, this is a reasonable restriction since otherwise it would be impossible to query edges in a labeling scheme with an undecidable label decoder. Taking this consideration a step further, in order for a labeling scheme to be practical querying an edge should be a quick operation, i.e. at least sublinear with respect to the number of vertices. Kannan et al acknowledged this by stating in their definition of an implicit representation that the label decoder must be computable in polynomial time [KNR92]. As a consequence querying an edge in such a labeling scheme takes only polylogarithmic time.

Observe that there can be different labeling schemes that represent the same graph class (just as there are different Turing machines deciding the same language). Therefore one might ask whether every graph class that is represented by some labeling scheme with an undecidable label decoder can also be represented by a labeling scheme with a decidable label decoder. Similarly, can every labeling scheme with a decidable label decoder be replaced by one that has a polynomial-time decidable label decoder? The latter question is equivalent to asking whether Muller's definition coincides with that of Kannan et al. Spinrad remarked that it is not known whether these definitions are equivalent and no difference could be observed on the graph classes considered so far [Spi03, p. 22]. In our terminology the former question can be phrased as $\mathsf{GALL} \overset{?}{=} \mathsf{GR}$ and the latter as $\mathsf{GR} \overset{?}{=} \mathsf{GP}$. We resolve these two questions by proving an analogue of the time hierarchy from classical complexity which shows that all of these three classes are distinct.

We say a function $t\colon \mathbb{N} \to \mathbb{N}$ is time-constructible if there exists a Turing machine that halts after exactly $t(n)$ steps for every input of length n and all $n \in \mathbb{N}$.

Theorem 3.3. *For every time-constructible function $t\colon \mathbb{N} \to \mathbb{N}$ it holds that $\mathsf{GTIME}(t(n)) \subsetneq \mathsf{GTIME}(\exp^2(n) \cdot t(n))$.*

A less general variant of this theorem was proved independently by Rotbart and Simonsen using Kolmogorov complexity [Rot16, p. 64].

Note that separations in the classical context do not necessarily extend to this setting, i.e. given two sets of languages $\mathsf{A} \subsetneq \mathsf{A}'$ it must not be the case that $\mathsf{GA} \subsetneq \mathsf{GA}'$. Therefore the previous theorem does not directly follow from the original time hierarchy theorem.

Corollary 3.4. $\mathsf{GEXP} \subsetneq \mathsf{G2EXP} \subsetneq \cdots \subsetneq \mathsf{GR} \subsetneq \mathsf{GALL}$.

Proof. We explain why $\mathsf{GEXP} \subsetneq \mathsf{G2EXP}$ follows from Theorem 3.3. The same argument shows that $\mathsf{G}k\mathsf{EXP} \subsetneq \mathsf{G}(k+1)\mathsf{EXP}$ for all $k \geq 1$. Recall that EXP equals the infinite union of $\mathsf{TIME}(\exp(n^c))$ over all $c \in \mathbb{N}$ and therefore Theorem 3.3 cannot be applied directly. However, $\mathsf{EXP} \subseteq \mathsf{TIME}(\exp(1.5^n))$ and $\mathsf{GTIME}(\exp(1.5^n)) \subsetneq \mathsf{G2EXP}$ does follow from Theorem 3.3 and therefore $\mathsf{GEXP} \subsetneq \mathsf{G2EXP}$ holds. As a consequence $\mathsf{G}k\mathsf{EXP}$ is a strict subset of GR for every $k \geq 0$. That GR is a strict susbet of GALL follows from the fact that its diagonalization graph class $\mathcal{C}_{\mathsf{R}}$ is not in GR (see Lemma 3.8) and every diagonalization graph class has a labeling scheme and therefore lies in GALL (see Definition 3.9 and the subsequent paragraph). □

The basic idea behind the proof of Theorem 3.3 is the following diagonalization argument. Let $\mathsf{A} = \{F_1, F_2, \dots\}$ be a set of label decoders. Then a labeling scheme in GA can be seen as pair of natural numbers, one for the label decoder and one for the label length. Let $\tau : \mathbb{N} \to \mathbb{N}^2$ be a surjective function and $S_{\tau(x)} = (F_y, z)$ with $\tau(x) = (y, z)$. It follows that for every labeling scheme S in GA there exists an $x \in \mathbb{N}$ such that $S = S_{\tau(x)}$. The following graph class cannot be in GA:

$$G \in \mathcal{C}_{\mathsf{A}} \Leftrightarrow G \text{ is the smallest graph on } n \text{ vertices s.t. } G \notin \mathrm{gr}(S_{\tau(n)})$$

where smallest is meant w.r.t. some order such as the lexicographical one. Note that the order must be for unlabeled graphs. However, an order for labeled graphs can be easily adapted to unlabeled ones. Assume $\mathcal{C}_{\mathsf{A}}$ is in GA via the labeling scheme S. There exists an $n \in \mathbb{N}$ such that $S = S_{\tau(n)}$ and it follows that $\mathcal{C}_{\mathsf{A}}$ contains a graph on n vertices that cannot be in S per definition, contradiction. Then it remains to show that $\mathcal{C}_{\mathsf{A}}$ is in the class that we wish to separate from GA.

For the remainder of this section we formalize this idea in three steps. First, we state the requirements for a pairing function τ and show that such a function exists. We continue by arguing that the diagonalization graph class $\mathcal{C}_{\mathsf{A}}$ is not contained GA. In the last step we construct a label decoder for $\mathcal{C}_{\mathsf{TIME}(t(n))}$ and show that it can be computed in time $\exp^2(n) \cdot t(n)$.

Definition 3.5. *A surjective function $\tau : \mathbb{N} \to \mathbb{N}^2$ is an admissible pairing if all of the following holds:*

1. $|\tau^{-1}(y,z)|$ *is infinite for all* $y, z \in \mathbb{N}$
2. $y, z \leq \log x$ *for all* $x \geq 1$ *and* $\tau(x) = (y,z)$
3. $\tau(x)$ *is undefined if* x *is not a power of two*
4. τ *is computable in polynomial time given its input in unary.*

Note, that a graph on n vertices has labels of the same length as a graph on m vertices whenever $\log n = \log m$ (rounded up). The third condition prevents this from happening, i.e. for all $G \neq H \in \mathcal{C}_{\mathsf{A}}$ it holds that their vertices are assigned labels of different length.

Lemma 3.6. *There exists an admissible pairing function.*

Proof. Consider the function $\tau(x) = (y,z)$ iff $x = \exp(2^y \cdot 3^z \cdot 5^w)$ for some $w \geq 0$. The first condition of Definition 3.5 holds because for every $w \in \mathbb{N}$ there exists an x with $\tau(x) = (y,z)$. For the second condition assume that there exists an $x \geq 1$ and $\tau(x) = (y,z)$ such that $y > \log x$. This cannot be the case because then $x < 2^y$ which contradicts $\log x = 2^y 3^z 5^w$. The same applies to z. The third condition is obvious. For the fourth condition observe that on input 1^x it suffices to consider y, z, w between 0 and $\log(\log x)$ such that $\log x = 2^y 3^z 5^w$. □

Definition 3.7. *Let* $\mathsf{A} = \{L_1, L_2, \ldots\}$ *be a countable set of languages,* $\prec$ *an order on unlabeled graphs and* τ *an admissible pairing. For* $n \in \mathbb{N}$ *and* $\tau(n) = (y,z)$ *let* $S_{\tau(n)}$ *be* (F_{L_y}, z)*. The diagonalization graph class of* A *is defined as:*

$$\mathcal{C}_{\mathsf{A}} = \bigcup_{n \in \mathrm{dom}(\tau)} \left\{ G \in \mathcal{G}_n \;\middle|\; G \text{ is the smallest graph w.r.t. } \prec \text{ not in } gr(S_{\tau(n)}) \right\}$$

where $\mathcal{G}_n$ *denotes the set of all graphs on* n *vertices.*

When we consider the diagonalization graph class of a set of languages we assume the lexicographical order for $\prec$ and the function given in the proof of Lemma 3.6 for τ.

Lemma 3.8. *For every countable set of languages* A *it holds that* $\mathcal{C}_{\mathsf{A}} \notin \mathsf{GA}$*.*

Proof. As argued in the paragraph after Theorem 3.3 it holds that for any labeling scheme S in GA there exists a graph G that is in $\mathcal{C}_{\mathsf{A}}$ but not in $\mathrm{gr}(S)$ and thus this lemma holds. If the labeling scheme S is in GA then there exists an $n \in \mathbb{N}$ such that $S = S_{\tau(n)}$ where $S_{\tau(n)} = (F_y, z)$, $\tau(n) = (y,z)$ and $\mathsf{A} = \{F_1, F_2, \ldots\}$. Therefore there must be a graph G on n vertices in $\mathcal{C}_{\mathsf{A}}$ which is not in $\mathrm{gr}(S)$. Observe that this argument is not quite correct. It only works if $\mathrm{gr}(S)$ does not contain all graphs on n vertices, otherwise such a graph G does not exist. However, due to the fact that $|\tau^{-1}(y,z)|$ is infinite it follows that there exists an arbitrarily large $n \in \mathbb{N}$ such that $S = S_{\tau(n)}$. And since $\mathrm{gr}(S)$ is small it follows that it does not contain all graphs on n vertices for sufficiently large n. □

To show that $\mathcal{C}_{\mathsf{A}}$ is in some class GB we need to define a labeling scheme $S_{\mathsf{A}} = (F_{\mathsf{A}}, 1)$ that represents $\mathcal{C}_{\mathsf{A}}$ and consider the complexity of computing its label decoder.

Definition 3.9. *Let* $\mathcal{C}$ *be a graph class such that* $|\mathcal{C}_n| = 0$ *whenever* n *is not a power of two and* $|\mathcal{C}_n| \leq 1$ *whenever* n *is a power of two. For* $G \in \mathcal{C}$ *let* G_0 *denote the lexicographically smallest labeled graph with* $G_0 \cong G$ *and* $V(G_0) = \{0,1\}^m$*. We define the label decoder* $F_{\mathcal{C}}$ *as follows. For every* $m \in \mathbb{N}$ *such that there exists* $G \in \mathcal{C}$ *on* 2^m *vertices and for all* $x, y \in \{0,1\}^m$ *let*

$$(x,y) \in F_{\mathcal{C}} \Leftrightarrow (x,y) \in E(G_0)$$

Observe that the diagonalization graph class $\mathcal{C}_{\mathsf{A}}$ of some set of languages A satisfies the prerequisite of the previous definition and the labeling scheme $(F_{\mathcal{C}_{\mathsf{A}}}, 1)$ represents $\mathcal{C}_{\mathsf{A}}$. Instead of $F_{\mathcal{C}_{\mathsf{A}}}$ we simply write F_{A} for the label decoder.

Up to this point the exact correspondence between $y \in \mathbb{N}$ and the label decoder F_y was not important since we only required the set of label decoders A to be countable. To show that the label decoder $F_{\mathsf{TIME}(t(n))}$ can be computed in time $\exp^2(n) \cdot t(n)$ it is important that the label decoder F_y for a given $y \in \mathbb{N}$ can be effectively computed.

Lemma 3.10. *For every time-constructible $t\colon \mathbb{N} \to \mathbb{N}$ there exists a mapping $f\colon \mathbb{N} \to \mathsf{ALL}$ such that $\mathrm{Im}(f) = \mathsf{TIME}(t(n))$. Additionally, on input $x \in \mathbb{N}$ in unary and $w \in \{0,1\}^*$ the question $w \in f(x)$ can be decided in time $n^{\mathcal{O}(1)} \cdot t(|w|)$ with $n = |w| + x$.*

Proof. Let $p\colon \mathbb{N} \to \mathbb{N}^2$ be a surjective function such that $y, z \leq x$ for all x in the domain of p and given x in unary $p(x)$ can be computed in polynomial time. For example, $p(x) = (y,z) \Leftrightarrow x = 2^y 3^z$. Then the desired mapping $f\colon \mathbb{N} \to \mathsf{ALL}$ can be constructed from p as follows. If $p(x) = (y,z)$ then $f(x)$ is the language that is decided by the Turing machine M_y when running at most $z \cdot t(|w|)$ steps on input w. If $p(x)$ is undefined then $f(x)$ shall be the empty language. Clearly, $\mathrm{Im}(f) = \mathsf{TIME}(t(n))$ because a language L is in $\mathsf{TIME}(t(n))$ iff there exists a Turing machine M and a $c \in \mathbb{N}$ such that M decides L and runs in time $c \cdot t(|w|)$. For the second part we construct a universal Turing machine with the required time bound. On input $x \in \mathbb{N}$ in unary and $w \in \{0,1\}^*$ it computes $p(x) = (y,z)$. If $p(x)$ is undefined it rejects (this corresponds to recognizing the empty language). Otherwise, it simulates M_y on input word w for $z \cdot t(|w|)$ steps. Due to the fact that t is time-constructible it is possible to run a counter during the simulation of M_y in order to not exceed the $z \cdot t(|w|)$ steps. The input length is $n := x + |w|$. The simulation can be run in time $n^{\mathcal{O}(1)} \cdot z \cdot t(|w|)$. Since $z \leq x \leq n$ the desired time bound follows. □

Lemma 3.11. *For every time-constructible function $t\colon \mathbb{N} \to \mathbb{N}$ it holds that the label decoder $F_{\mathsf{TIME}(t(n))}$ can be computed in $\mathsf{TIME}(\exp^2(n) \cdot t(n))$.*

Proof. On input xy with $x, y \in \{0,1\}^m$ and $m \geq 1$ compute $\tau(2^m) = (y,z)$. If it is undefined then reject. Otherwise there is a labeling scheme $S_{\tau(2^m)} = (F_y, z)$ and we need to compute the smallest graph G_0 on 2^m vertices such that $G_0 \notin \mathrm{gr}(S_{\tau(2^m)})$. If G_0 exists we assume that its vertex set is $\{0,1\}^m$ and accept iff $(x,y) \in E(G_0)$. If it does not exist then reject.

The graph G_0 can be computed as follows. Iterate over all labeled graphs H with 2^m vertices in order and over all functions $\ell\colon V(H) \to \{0,1\}^{zm}$. Check if $H \in \mathrm{gr}(S_{\tau(2^m)})$ by checking for every pair of vertices $u \neq v \in V(H)$ if $(u,v) \in E(H) \Leftrightarrow (\ell(u), \ell(v)) \in F_y$. If this condition fails for all labelings ℓ then $G_0 = H$. To query the label decoder F_y we apply the previous Lemma 3.10.

Let us consider the time requirement w.r.t. m. To compute $\tau(2^m)$ we write down 2^m in unary and compute τ in polynomial time w.r.t. 2^m which is in the order $2^{\mathcal{O}(m)}$. To compute G_0 there are four nested loops. The first one goes over all labeled graphs on 2^m vertices which is bounded by $\exp(\exp(m)^2) = \exp^2(2m)$. The second loop considers all possible labelings ℓ of which there can be at most $\exp(zm)^{\exp(m)} = \exp(\exp(m)zm) \leq \exp^2(zm^2)$. It holds that $y, z \leq \log(2^m) = m$ due to the second condition of Definition 3.5 and therefore $\exp^2(zm^2) \leq \exp^2(m^3)$. The other two loops go over all vertices of H of which there are 2^m. Due to Lemma 3.10 the time required to compute $(\ell(u), \ell(v)) \in F_y$ is $y^{\mathcal{O}(1)} \cdot t(2m) = m^{\mathcal{O}(1)} \cdot t(2m)$. In total this means the algorithm runs in time $\mathcal{O}(\exp^2(m^{\mathcal{O}(1)}) \cdot t(2m))$ which is the required time bound since the input length is $2m$. □

Lemma 3.8 states that $\mathcal{C}_{\mathsf{TIME}(t(n))} \notin \mathsf{GTIME}(t(n))$ and from Lemma 3.11 it follows that $\mathcal{C}_{\mathsf{TIME}(t(n))} \in \mathsf{GTIME}(\exp^2(n) \cdot t(n))$ therefore proving Theorem 3.3. Notice, this argument fails to separate GP from GEXP because the runtime to compute the label decoder F_{P} is at least double exponential due to the first two loops mentioned in the proof of Lemma 3.11. Also, we remark that if the closure under induced subgraphs of the diagonalization graph class of P is small then the implicit graph conjecture is false.

3.2 Expressiveness of Primitive Labeling Schemes

To understand the limitations of labeling schemes it is reasonable to start with very simple ones first and then gradually increase the complexity. In this section we present two such simple families of labeling schemes and explain how they relate to other well-known sets of graph classes. Interestingly, these two families of labeling schemes can be seen as generalizations of uniformly sparse graph classes and graph classes with bounded degree. Therefore they are already able to represent many graph classes that are of theoretical and practical importance. For this section we assume all graphs and graph classes to be undirected. This also means that for sets of graph classes such as GAC^0 we only consider its restriction to undirected graph classes.

In [Spi03, p. 20] Spinrad describes a labeling scheme for every sparse and hereditary graph class. A graph class $\mathcal{C}$ is sparse and hereditary iff it has bounded degeneracy. Therefore there exists a constant c such that every graph $G \in \mathcal{C}$ has a vertex with degree at most c. The following labeling scheme represents $\mathcal{C}$. Given a graph G from $\mathcal{C}$ assign each vertex a unique identifier $1, \ldots, n$. Choose a vertex v in G with at most c neighbors. Store the identifier of v along with the identifiers of its c neighbors in the label of v. Delete the vertex v from G and repeat this process. Since $\mathcal{C}$ is hereditary it follows that G without v also has a vertex of degree at most c. Two vertices u, v with labels $u_0, u_1, \ldots, u_c$ and $v_0, v_1, \ldots, v_c$ are adjacent iff $u_0 \in \{v_1, \ldots, v_c\}$ or $v_0 \in \{u_1, \ldots, u_c\}$. For every $c \in \mathbb{N}$ this construction yields a labeling scheme S_c. Let us call the smallest number c such that a graph can be represented by S_c its or-pointer number. This can be further generalized and leads to the following four graph parameters.

Definition 3.12 (Pointer Numbers). *The (bijective) and/or-pointer number of a graph G with n vertices is the least $k \in \mathbb{N}$ such that there exist a (bijective) function $\ell_{\mathrm{id}} \colon V(G) \to [n]$ and a function $\ell \colon V(G) \to [n]^k$ for which it holds that $\{u, v\} \in E(G)$ iff $\ell_{\mathrm{id}}(u) \in \ell(v)$ and/or $\ell_{\mathrm{id}}(v) \in \ell(u)$ for all $u \neq v \in V(G)$.*

The bijectiveness constraint can be understood as restriction on the possible labelings that are allowed, i.e. the id of each vertex must be unique. In the bijective case the function $\ell_{\mathrm{id}} \colon V(G) \to [n]$ becomes obsolete and the function ℓ can be understood as a mapping from $V(G)$ to $V(G)^k$. Notice how this constraint is satisfied in the case of the labeling described in the previous paragraph.

Fact 3.13. *The bijective or-pointer number and degeneracy are equivalent.*

Proof. "$\Rightarrow$": Let G have bijective or-pointer number at most k. It holds that G has degeneracy at most $2k$. Every induced subgraph of G has bijective or-pointer number at most k. Additionally, every graph with bijective or-pointer number at most k can have at most kn edges which implies that such a graph must have a vertex with degree at most $2k$.

"$\Leftarrow$": Let G have degeneracy at most k. Then G has bijective or-pointer number at most k due to the labeling described in the first paragraph of this subsection. □

Fact 3.14. *The bijective and-pointer number of a graph equals its maximum degree.*

Fact 3.15. *Planar graphs have unbounded and-pointer number.*

Proof. Consider the graph G_k shown in Figure 3.1. We show that for every $l \in \mathbb{N}$ there exists a $k \in \mathbb{N}$ such that the and-pointer number of G_k is larger than l. For a given l let $k = l^2 + 1$. For the sake of contradiction, assume that G_k has and-pointer number l via (ℓ_{id}, ℓ). This means $\ell(\cdot)$ has at most l elements. For two vertices u, v let us say that they are equivalent if $\ell_{\text{id}}(u) = \ell_{\text{id}}(v)$. Since x is adjacent to x_i it holds that $\ell_{\text{id}}(x_i) \in \ell(x)$ for all $i \in [k]$. Therefore $\{x_1, \dots, x_k\}$ consists of at most l equivalence classes. The same holds for $\{y_1, \dots, y_k\}$. Due to the pigeonhole principle it follows that there are $r := \lceil \frac{k}{l} \rceil$ vertices $a_1, \dots, a_r \in \{x_1, \dots, x_k\}$ that are equivalent. For a_i let b_i denote the vertex in $\{y_1, \dots, y_k\}$ that is adjacent to a_i. For all $i \neq j \in [r]$ it holds that b_i and b_j are not equivalent, otherwise G_k would contain $K_{2,2}$. This implies that $\{y_1, \dots, y_k\}$ must consist of at least r different equivalence classes. However, $\{y_1, \dots, y_k\}$ consists of at most l different classes and $r = \lceil \frac{k}{l} \rceil = \lceil \frac{l^2+1}{l} \rceil = l + 1$. Therefore the and-pointer number of G_k must be larger than l. □

We remark that every forest has and-pointer number at most two. See Figure 3.2 for an example of how to label a tree; the number left of the bar is the id of the vertex.

In comparison, it seems not quite as simple to prove that a small graph class has unbounded or-pointer number. The following observation might be helpful in that regard. Consider a graph G with a k-or-pointer representation (ℓ_{id}, ℓ). Let c denote the number of unique ids, i.e. the cardinality of the image of ℓ_{id}. There exists an induced subgraph of G with c vertices which has bijective or-pointer number at most k and therefore this subgraph can have at most kc edges. Informally, if a graph has many edges then in any k-or-pointer representation of this graph there cannot be many unique ids. As a consequence the structure of such a graph is quite constrained. Therefore we suspect that the edge-complement of some sparse graph class such as planar graphs has unbounded or-pointer number.

In Figure 3.3 we give an overview of the relation of GAC^0 and the pointer numbers to other sets of graph classes defined in terms of graph class properties and graph parameters. A graph parameter in this figure is interpreted as the set of graph classes that are bounded by it. For example, a well-known result by Robertson and Seymour states that a graph class $\mathcal{C}$ has bounded tree-width iff there exists a planar graph G such that no graph in $\mathcal{C}$ has G as minor, i.e. $\mathcal{C} \in \text{MF}(\text{Planar})$. The remainder of this section explains the relations of the classes shown in Figure 3.3 going from top to bottom.

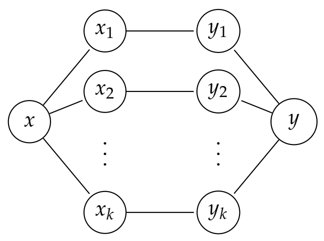


FIGURE 3.1: Planar graphs G_k with unbounded and-pointer number

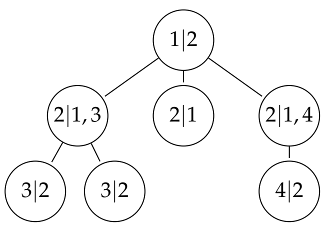


FIGURE 3.2: And-pointer labeling of a tree

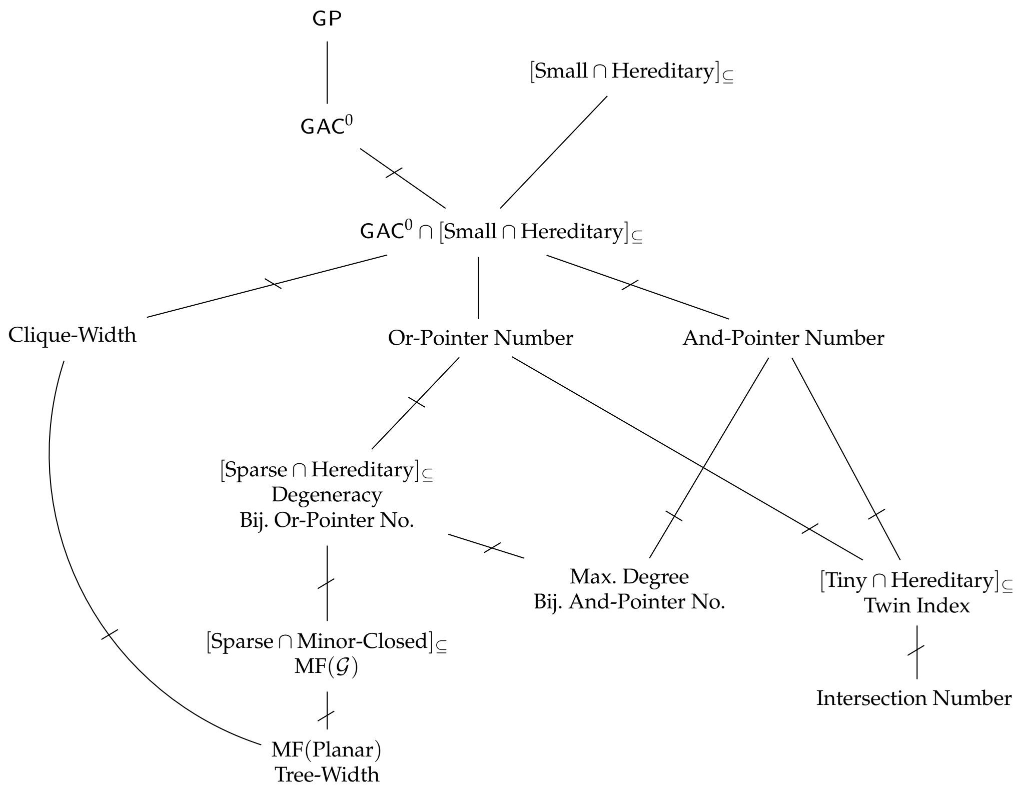

GP
[Small ∩ Hereditary]⊆
GAC0
GAC0 ∩ [Small ∩ Hereditary]⊆
Clique-Width
Or-Pointer Number
And-Pointer Number
[Sparse ∩ Hereditary]⊆ Degeneracy Bij. Or-Pointer No.
Max. Degree Bij. And-Pointer No.
[Tiny ∩ Hereditary]⊆ Twin Index
[Sparse ∩ Minor-Closed]⊆ MF(𝒢)
Intersection Number
MF(Planar) Tree-Width

FIGURE 3.3: Various sets of graph classes and their relation to labeling schemes

The question of whether $[\text{Small} \cap \text{Hereditary}]_{\subseteq}$ is a subset of GP is equivalent to the implicit graph conjecture. Therefore it is also unknown whether $[\text{Small} \cap \text{Hereditary}]_{\subseteq}$ is a subset of GAC^0 since this would imply the implicit graph conjecture. However, it can be shown that the converse does not hold, i.e. GAC^0 is not a subset of $[\text{Small} \cap \text{Hereditary}]_{\subseteq}$. From a graph-theoretical point of view being hereditary is the weakest form of uniformity condition that can be imposed on a graph class in order for it to have some meaningful structure. Therefore there probably is no elegant graph-theoretical characterization of GAC^0. This arbitrariness might also be one of the reasons why it is so difficult to analyze GAC^0 and its supersets.

Theorem 3.16. $\mathsf{GAC}^0 \not\subseteq [\text{Small} \cap \text{Hereditary}]_{\subseteq}$.

Proof. For a graph class $\mathcal{C}$ let $[\mathcal{C}]_{\subseteq}$ denote its closure under induced subgraphs. Observe that if a graph class $\mathcal{C}$ is in $[\text{Small} \cap \text{Hereditary}]_{\subseteq}$ then $[\mathcal{C}]_{\subseteq}$ is in $[\text{Small} \cap \text{Hereditary}]_{\subseteq}$ as well. To prove the above statement we show that (I) $\mathsf{GR} \not\subseteq [\text{Small} \cap \text{Hereditary}]_{\subseteq}$ and (II) for every graph class $\mathcal{C} \in \mathsf{GR}$ there exists a graph class $\mathcal{D}$ in GAC^0 with $[\mathcal{C}]_{\subseteq} \subseteq [\mathcal{D}]_{\subseteq}$.

For (I) we construct a labeling scheme $S = (F, 1)$ where F is computable and $[\text{gr}(S)]_{\subseteq}$ is the class of all graphs. Let $f \colon \mathbb{N} \to \mathcal{G}$ be a computable function such that:

1. the image of f equals the set of all graphs $\mathcal{G}$
2. $|V(f(n))| \leq n$ for all $n \in \mathbb{N}$
3. if n is not a power of two then $f(n)$ is the graph with a single vertex

For $n \geq 2$ let G_n be the lexicographically smallest labeled graph that is isomorphic to $f(n)$ and $V(G_n) \subseteq \{0,1\}^{\log n}$. We define the label decoder F as

$$(x,y) \in F \Leftrightarrow (x,y) \in E(G_{2^m})$$

for all $x, y \in \{0,1\}^m$ and $m \in \mathbb{N}$. For every graph G with at least two vertices there exists an $m \in \mathbb{N}$ such that G is isomorphic to $f(2^m)$ and therefore G occurs as induced subgraph of the graph with 2^m vertices in $\mathrm{gr}(S)$.

We show that (II) holds due to a padding argument. Given a labeling scheme $S = (F, c)$ in GR and a function $p \colon \mathbb{N} \to \mathbb{N}$ we define the labeling scheme $S_p = (F_p, c)$ as

$$(x,y) \in F \Leftrightarrow (xx', yy') \in F_p$$

for all $m \in \mathbb{N}, x, y \in \{0,1\}^{cm}, x', y' \in \{0,1\}^{cp(m)}$. To see that $[\mathrm{gr}(S)]_\subseteq \subseteq [\mathrm{gr}(S_p)]_\subseteq$ consider a graph G that is in $\mathrm{gr}(S)$ via a labeling $\ell \colon V(G) \to \{0,1\}^{c \log n}$. We show that there is a graph G_0 in $\mathrm{gr}(S_p)$ on $n_0 = 2^{\log n + p(\log n)}$ vertices such that G is an induced subgraph of G_0. Let us assume that $V(G) \subseteq V(G_0)$. The partial labeling $\ell' \colon V(G_0) \to \{0,1\}^{c(\log n + p(\log n))}$ with $\ell'(u) = \ell(u)0^{cp(\log n)}$ for all $u \in V(G)$ shows that G is an induced subgraph of G_0. It remains to argue that for every computable label decoder F there exists a padding function p such that F_p is computable by a family of logspace-uniform AC^0-circuits. Let $t \colon \mathbb{N} \to \mathbb{N}$ be the runtime of a Turing machine which computes F. The idea is to choose p sufficiently large in terms of t such that the logspace transducer which computes the circuit family can precompute the satisfying assignments for the circuit and then compile them into a DNF. Notice that membership in F_p only depends on a small part of the input bits and therefore the DNF is only polynomial w.r.t. the input size and thus can be directly encoded into the circuit. □

The or- and and-pointer numbers are hereditary because deleting vertices does not increase them. Also, the labeling schemes behind these numbers can be computed in GAC^0 and therefore the pointer numbers are contained in the intersection of these two classes. The and-pointer number is strictly contained in the intersection because planar graphs have unbounded and-pointer number but bounded or-pointer number.

Fact 3.17 ([Spi03, p. 165 f.]). *Every graph class with bounded clique-width is in* GAC^0.

Proof. A subset of vertices S of a graph G is called a k-module if it can be partitioned into k parts $S_1, \ldots, S_k$ such that S_i is a module in $G[(V \setminus S) \cup S_i]$, i.e. the vertices in S_i are indistinguishable to vertices of $V \setminus S$, for $k \in \mathbb{N}$. A k-module S of G is called balanced if S contains at least one third and at most two thirds of the vertices of G. Spinrad asserts that every graph with clique-width k has a balanced k-module. Therefore given a graph G with clique-width k one can construct a binary tree $T(G)$ by recursively finding a balanced k-module S and putting the vertices of S in the left node and the vertices of $V(G) \setminus S$ in the right node. The root node contains every vertex of G. Since every child node in $T(G)$ only has at most two thirds of the vertices of its parent node it follows that the tree has depth $\mathcal{O}(\log n)$. Spinrad constructs the following labeling scheme by using this tree. Given a graph G with clique width k a vertex v of G is labeled as follows. There is a path in $T(G)$ from the root node to the leaf node which contains v. Let this path be $x_1, \ldots, x_c$ where x_1 is the root node and x_c is the leaf. For each $1 \leq i < c$ the following information is stored in the label of v. For x_i let S^i be the balanced k-module stored in the left child node of x_i which can be partitioned into $S^i_1, \ldots, S^i_k$. The first bit of v for the i-th level denotes whether

v is in the left child of x_i. If v is in the left child of x_i then this means v is in the balanced k-module S^i and one also stores the index j such that $v \in S^i_j$ for $1 \leq j \leq k$. If v is in the right child of x_i then one stores the subset of X of $\{1, \ldots, k\}$ such that v is adjacent to the modules S^i_j for all $j \in X$. Each level only requires a constant number of bits. To check whether two vertices u, v of G are adjacent one has to find the first level of $T(G)$ such that u and v are placed in different nodes. Assume that u is in the left subtree and v in the right one. Then u and v are adjacent iff the index j of the part of the balanced k-module that u is contained in is part of the subset X of v for this level. It is not difficult to construct a constant-depth circuit which computes this label decoder. □

Every uniformly sparse graph class has bounded degeneracy and thus bounded or-pointer number due to Fact 3.13. Since the family of complete graphs has bounded or-pointer number but is not (uniformly) sparse this inclusion is strict.

Fact 3.18. *Every graph class with bounded twin index has bounded or- and and-pointer number.*

Proof. Let $\mathcal{C}$ be a graph class with bounded twin index k. This means a graph G in $\mathcal{C}$ has at most k twin classes $V_1, \ldots, V_k$. The following labeling of the vertices in G shows that G has or- and and-pointer number at most k. Given a vertex v in G let its id be the index of the twin class, i.e. $\ell_{\text{id}}(v) = i$ such that $v \in V_i$. Then let $\ell(v)$ be the subset of $[k]$ such that $j \in \ell(v)$ iff the twin class V_j is adjacent to the twin class of v or $j = \ell_{\text{id}}(v)$ and V_j is a clique. □

The class of square grid graphs has unbounded twin index but bounded degree. Therefore the twin index is strictly contained in the pointer numbers.

Fact 3.19. *A graph class is in* $[\text{Tiny} \cap \text{Hereditary}]_{\subseteq}$ *iff it has bounded twin index.*

Proof. "$\Rightarrow$": This direction is proved in [SZ94].

"$\Leftarrow$": Let $\mathcal{C}$ be the set of graphs with twin index at most k for a $k \in \mathbb{N}$. Clearly, $\mathcal{C}$ is hereditary. It remains to show that $\mathcal{C}$ is tiny. Recall that a graph class $\mathcal{C}$ is tiny if there exists a $c < \frac{1}{2}$ such that $|\mathcal{C}_n| \leq n^{cn}$ for all sufficiently large n. A graph G on n vertices in $\mathcal{C}$ with twin index $1 \leq i \leq k$ is determined by the following choices. Choose an unordered partition of n into i parts $p_1, p_2, \ldots, p_i \in [n]$, which means $p_1 + \cdots + p_i = n$ and $p_1 \leq p_2 \leq \cdots \leq p_i$. Let $P_{n,i}$ denote the number of such partitions. This partition tells us that the first p_1 vertices of G are a twin class, the next p_2 vertices of G are a twin class and so on. For every twin class one has to choose whether it is a clique or an independent set (2^i possibilities). It remains to choose how the i twin classes interact ($|\mathcal{G}_i| \leq 2^{i^2}$ possibilities).

$$\sum_{i=1}^{k} 2^{i^2+i} \cdot P_{n,i} \leq \sum_{i=1}^{k} 2^{i^2+i} \cdot 2^n \leq k \cdot 2^{k^2+k} \cdot 2^n \leq k \cdot 2^{n+k^2} \leq 2^{\frac{1}{3}n \log n} = n^{\frac{1}{3}n}$$

These inequalities are meant to hold for fixed k and sufficiently large n. The first inequality holds due to the fact that the partition number $P_n = \sum_{i=1}^{n} P_{n,i}$ is in $\mathcal{O}(2^n)$, which follows from the asymptotic formula for P_n by Hardy and Ramanujan. □

To see that the intersection number is bounded by the twin index observe that a graph with intersection number k can have at most 2^k different twin classes because the twin class of a vertex is determined by a subset of k elements. To see that this inclusion is strict consider the class of complete bipartite graphs which have twin index two but unbounded intersection number.

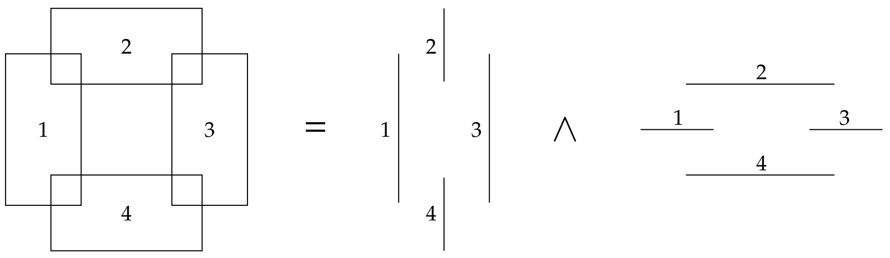


FIGURE 3.4: Axis-aligned rectangle intersection graph as conjunction of two interval graphs

3.3 Reductions Between Graph Classes

The concept of reduction is vital to complexity theory as it enables one to formally compare the complexity of problems as opposed to just treating each problem separately. In our context we want to say a graph class $\mathcal{C}$ reduces to a graph class $\mathcal{D}$ if the adjacency of graphs in $\mathcal{C}$ can be expressed in terms of adjacency of graphs in $\mathcal{D}$. It should satisfy the following closure property: if $\mathcal{D}$ has a labeling scheme and $\mathcal{C}$ is reducible to $\mathcal{D}$ then $\mathcal{C}$ has a labeling scheme as well. This closure should also hold when the complexity of label decoders is restricted, i.e. if $\mathcal{D}$ is in GP and $\mathcal{C}$ reduces to $\mathcal{D}$ then $\mathcal{C}$ is in GP. Such a reduction notion makes it possible to compare graph classes for which no labeling schemes are known to see whether there might be a common obstacle to designing a labeling scheme. We introduce two kinds of reduction that satisfy this closure property. Before we formally define and examine them let us first explain the intuition behind them.

For the first reduction type, called algebraic reduction, the idea is to express the adjacency of a graph G on vertex set V in terms of graphs $H_1, \ldots, H_k$ which also have vertex set V and a k-ary Boolean function f. Two vertices u, v in G are adjacent iff $f(x_1, \ldots, x_k) = 1$ where x_i denotes whether u and v are adjacent in H_i. For example, every axis-aligned rectangle intersection graph can be expressed as the conjunction of two interval graphs as shown in Figure 3.4, i.e. two boxes intersect iff both of their corresponding intervals intersect. Therefore we say axis-aligned rectangle intersection graphs reduce to interval graphs. We call this type of reduction algebraic because it is build upon an interpretation of Boolean functions as functions on graph classes. The resulting algebra on graph classes inherits some of the properties of its Boolean ancestor. For instance, the negation of a graph class $\mathcal{C}$ equals its edge-complement co-$\mathcal{C}$ in this interpretation. Thus negating a graph class twice is an involution, i.e. $\neg\neg\mathcal{C} = \mathcal{C}$. Additionally, this algebra gives a unifying terminology for concepts such as arboricity, thickness and boxicity. We remark that a similar but in a sense less general notion called locally bounded coverings of graphs has been used in [LMZ12] and [Atm+15]. They say a set of graphs $H_1, \ldots, H_k$ is a covering of a graph G if $V(G) = \cup_{i=1}^{k} V(H_i)$ and $E(G) = \cup_{i=1}^{k} E(H_i)$. If we assume that $V(H_i) = V(G)$ then in our terminology G is the disjunction of $H_1, \ldots, H_k$. They also use this as a tool to prove the existence of implicit representations [Atm+15, Lem. 4].

The second kind of reduction is called subgraph reduction. While it is technically more tedious to define than the algebraic variant, the underlying intuition is just as simple. Instead of expressing the adjacency of a graph G using a sequence of graphs as before, we do this in terms of a single, larger graph H. Informally, every vertex of G is assigned to a constant-sized subgraph of H and two vertices u, v of G are adjacent if their combined (labeled) subgraph in H satisfies some condition. To illustrate this let us consider how the adjacency of k-interval graphs can be expressed in terms of interval graphs in this sense. Let G be a k-interval graph with n vertices. This means there exists an interval model

$M(G)$ of G with kn intervals. Let H be the interval graph with kn vertices that is induced by the intervals of $M(G)$. Then assign each vertex u of G the k vertices $u_1, \dots, u_k$ in H that correspond to its k intervals. Two vertices u, v in G are adjacent iff there exist $i, j \in [k]$ such that u_i and v_j are adjacent in H. It is not clear whether k-interval graphs can also be reduced to interval graphs in the algebraic sense. We show that algebraic reductions are a special case of subgraph reductions under certain circumstances.

An important application of reductions is to demonstrate that certain problems are representative (complete) for certain complexity classes. The notion of completeness is also applicable to our setting and it is natural to ask what a complete graph class for GP looks like. We show that no hereditary graph class can be complete for GAC^0 or any superset thereof with respect to algebraic or subgraph reductions. In fact, it might very well be the case that there do not even exist complete graph classes for GP and GAC^0 at all. On the upside, in the next section we introduce classes of labeling schemes defined in terms of first-order logic for which completeness results under both types of reductions can be shown.

3.3.1 Algebraic Reductions

Definition 3.20. *Let f be a k-ary Boolean function and $H_1, \dots, H_k$ are graphs with the same vertex set V and $k \geq 0$. We define $f(H_1, \dots, H_k)$ to be the graph with vertex set V and an edge (u, v) iff $f(x_1, \dots, x_k) = 1$ where $x_i = [\![(u, v) \in E(H_i)]\!]$ for all $u \neq v \in V$.*

The constant Boolean functions 0 and 1 define the empty and complete graph, respectively. The negation of a graph is its edge-complement.

Definition 3.21. *Let f be a k-ary Boolean function and $\mathcal{C}_1, \dots, \mathcal{C}_k$ are graph classes and $k \geq 0$. We define $f(\mathcal{C}_1, \dots, \mathcal{C}_k)$ to be the graph class that contains every graph G such that there exist $(H_1, \dots, H_k) \in \mathcal{C}_1 \times \cdots \times \mathcal{C}_k$ with $G = f(H_1, \dots, H_k)$ assuming that G and $H_1, \dots, H_k$ all have the same vertex set.*

Under the graph class interpretation the constant Boolean functions 0 and 1 define the class of empty and complete graphs, respectively. The negation of a graph class $\mathcal{C}$ is the edge-complement co-$\mathcal{C}$. A graph G has arboricity at most k iff $G \in \bigvee_{i=1}^{k}$ Forest. Similarly, G has thickness at most k iff $G \in \bigvee_{i=1}^{k}$ Planar and boxicity at most k iff $G \in \bigwedge_{i=1}^{k}$ Interval.

Suppose you are given two Boolean formulas F_1, F_2 with k variables. Due to the previous definition we can naturally interpret F_i as a function f_i which maps k graph classes to a graph class (each subformula of F_i evaluates to a graph class). The following statement shows that F_1 and F_2 are logically equivalent iff $f_1 = f_2$.

Lemma 3.22 (Compositional Equivalence). *Given Boolean functions $f, g, h_1, \dots, h_l$ where $f, h_1, \dots, h_l$ have arity k and g has arity l such that $f(\vec{x}) = g(h_1(\vec{x}), \dots, h_l(\vec{x}))$ for all $\vec{x} \in \{0,1\}^k$. Then for all sequences of graph classes $\vec{\mathcal{C}} = (\mathcal{C}_1, \dots, \mathcal{C}_k)$ it holds that*

$$f(\vec{\mathcal{C}}) = g(h_1(\vec{\mathcal{C}}), \dots, h_l(\vec{\mathcal{C}}))$$

Proof. We show that for all sequences of graphs $\vec{H} = (H_1, \dots, H_k)$ it holds that $f(\vec{H}) = g(h_1(\vec{H}), \dots, h_l(\vec{H}))$. From that it directly follows that $f(\vec{\mathcal{C}}) = g(h_1(\vec{\mathcal{C}}), \dots, h_l(\vec{\mathcal{C}}))$. Let $G_f = f(\vec{H})$ and $G_g = g(h_1(\vec{H}), \dots, h_l(\vec{H}))$. It holds that

$$(u, v) \in E(G_f) \Leftrightarrow f(\vec{x}) = 1 \Leftrightarrow g(h_1(\vec{x}), \dots, h_l(\vec{x})) = 1 \Leftrightarrow (u, v) \in E(G_g)$$

with $\vec{x} = (x_1, \ldots, x_k)$ and $x_i = [\![(u,v) \in E(H_i)]\!]$ for $i \in [k]$. □

Given a Boolean function f of arbitrary arity it is trivial to construct a unary Boolean function g such that $f(x, \ldots, x) = g(x)$ for all $x \in \{0,1\}$. This is not possible under the graph class interpretation. For instance, Forest $\vee$ Forest $\neq f$(Forest) for any of the four unary Boolean functions f. Therefore disjunction is not idempotent and one can show that neither is conjunction. However, this algebra does inherit some of the properties of its Boolean ancestor. For instance, disjunction and conjunction are associative and commutative, negation is an involution and De Morgan's laws apply. More generally, all laws of Boolean algebra where every variable occurs at most once on each side apply to this algebra as well. Boolean algebra is a special case of this algebra on graph classes if one restricts the universe to the class of complete graphs $\{K_n \mid n \in \mathbb{N}\}$ and the edge-complement of it.

Definition 3.23 (Algebraic Reduction)**.** *Let F be a set of Boolean functions and $\mathcal{C}, \mathcal{D}$ are graph classes. We say $\mathcal{C} \leq_F \mathcal{D}$ if $\mathcal{C} \subseteq f(\mathcal{D}, \ldots, \mathcal{D})$ for some $f \in F$. We write $[\mathcal{D}]_F$ to denote the set of graph classes reducible to $\mathcal{D}$ w.r.t. $\leq_F$.*

In order for $\leq_F$ to be reflexive and transitive the following additional requirement has to be made. A set of Boolean functions is a Boolean clone if it is closed under composition and contains all projection functions $\pi^i_k(x_1, \ldots, x_k) = x_i$ for $1 \leq i \leq k$ and $k \in \mathbb{N}$. For a set of Boolean functions F let us write $[F]_{\text{clone}}$ to denote the closure of F and all projections functions under composition. Alternatively, one can think of $[F]_{\text{clone}}$ as all Boolean functions that can be expressed by Boolean formulas which use functions from F as connectives. For example, $[\neg, \wedge]_{\text{clone}}$ is the set of all Boolean functions BF.

Lemma 3.24. *If F is a Boolean clone then $\leq_F$ is reflexive and transitive.*

Proof. Reflexivity directly follows from the identity function which is a projection function and therefore contained in F. For transitivity let $\mathcal{C} \leq_F \mathcal{D}$ via a k-ary $f \in F$ and $\mathcal{D} \leq_F \mathcal{E}$ via an l-ary $g \in F$. We show that $\mathcal{C} \leq_F \mathcal{E}$ via the kl-ary Boolean function $h(\vec{x_1}, \ldots, \vec{x_k}) = f(g(\vec{x_1}), \ldots, g(\vec{x_k}))$ where $\vec{x_i}$ denotes a sequence of l variables. Since h is the composition of f and g it follows that h is in F. Consider a graph G in $\mathcal{C}$. There are graphs $H_1, \ldots, H_k \in \mathcal{D}$ such that $G = f(H_1, \ldots, H_k)$. Since $H_i \in \mathcal{D}$ there are graphs $I^i_1, \ldots, I^i_l \in \mathcal{E}$ such that $H_i = g(I^i_1, \ldots, I^i_l)$ for all $i \in [k]$. Therefore $G = h(\vec{I^1}, \ldots, \vec{I^k}) = f(g(\vec{I^1}), \ldots, g(\vec{I^k}))$ with $\vec{I^i} = (I^i_1, \ldots, I^i_l)$ for $i \in [k]$. □

We say a set of graph classes $\mathbb{A}$ is closed under a k-ary Boolean function f if for all $\mathcal{C}_1, \ldots, \mathcal{C}_k \in \mathbb{A}$ it holds that $f(\mathcal{C}_1, \ldots, \mathcal{C}_k) \in \mathbb{A}$. We say $\mathbb{A}$ is closed under $\leq_F$ for a set of Boolean functions F if $\mathcal{C} \leq_F \mathcal{D}$ and $\mathcal{D} \in \mathbb{A}$ implies $\mathcal{C} \in \mathbb{A}$.

Lemma 3.25. *Let F be a Boolean clone and let B be a set of Boolean functions with $F = [B]_{\text{clone}}$. If a set of graph classes $\mathbb{A}$ is closed under subsets and f for every $f \in B$ then $\mathbb{A}$ is closed under $\leq_F$.*

Proof. Let $\mathbb{A}$ be closed under subsets and every Boolean function in B. We need to argue that $\mathbb{A}$ is closed under f for every $f \in F$. From that and the closure under subsets it follows that $\mathbb{A}$ is closed under $\leq_F$.

Observe that if $\mathbb{A}$ is closed under some Boolean functions then it is also closed under the composition of these functions for the following reason. Let $f, g, h_1, \ldots, h_l$ be Boolean functions where $f, h_1, \ldots, h_l$ have arity k and g has arity l and f is the composition of g with $h_1, \ldots, h_l$. Let $\mathbb{A}$ be closed under g and $h_1, \ldots, h_l$. Given $\vec{\mathcal{D}} \in \mathbb{A}^k$ it holds that $f(\vec{\mathcal{D}}) = g(h_1(\vec{\mathcal{D}}), \ldots, h_l(\vec{\mathcal{D}}))$ due to the compositional equivalence from Lemma 3.22.

Since $\mathbb{A}$ is closed under h_i it follows that $\mathcal{D}_i := h_i(\vec{\mathcal{D}})$ is in $\mathbb{A}$ for all $i \in [l]$. Therefore $g(\mathcal{D}_1, \ldots, \mathcal{D}_l)$ is in $\mathbb{A}$ as well and hence $\mathbb{A}$ is closed under f. Since every function in F can be expressed as composition of functions from B and projections and $\mathbb{A}$ is closed under every function from B and projections it follows that it is closed under every function from F. □

If $\mathbb{A}$ is closed under union then the implication in Lemma 3.25 becomes an equivalence.

Corollary 3.26. *A set of graph classes is closed under $\leq_{\mathrm{BF}}$ if it is closed under subsets, negation and conjunction. Moreover, for a set of graph classes $\mathbb{A}$ that is closed under union it holds that $\mathbb{A}$ is closed under $\leq_{\mathrm{BF}}$ iff it is closed under subsets, negation and conjunction.*

Lemma 3.27. *The classes* $\mathsf{GAC}^0, \mathsf{GL}, \mathsf{GP}, \mathsf{GEXP}, \mathsf{GR}$ *and* GALL *are closed under* $\leq_{\mathrm{BF}}$.

Proof. If a set of languages A is closed under complement then GA is closed under negation. To see that the above mentioned classes are closed under conjunction, consider the following construction. Given two labeling schemes $S_1 = (F_1, c_1)$ and $S_2 = (F_2, c_2)$ let the labeling scheme $S_3 = (F_3, c_1 + c_2)$ with $(x_1x_2, y_1y_2) \in F_3 \Leftrightarrow (x_1, y_1) \in F_1 \wedge (x_2, y_2) \in F_2$ and $\frac{|x_i|}{|x|} = \frac{|y_i|}{|y|} = \frac{c_i}{c}$ for $i \in [2]$. It holds that $\mathrm{gr}(S_1) \wedge \mathrm{gr}(S_2) \subseteq \mathrm{gr}(S_3)$ and it is simple to see that this construction works for all of the above complexity classes. □

Lemma 3.28. *The class* $[\text{Small} \cap \text{Hereditary}]_{\subseteq}$ *is closed under* $\leq_{\mathrm{BF}}$.

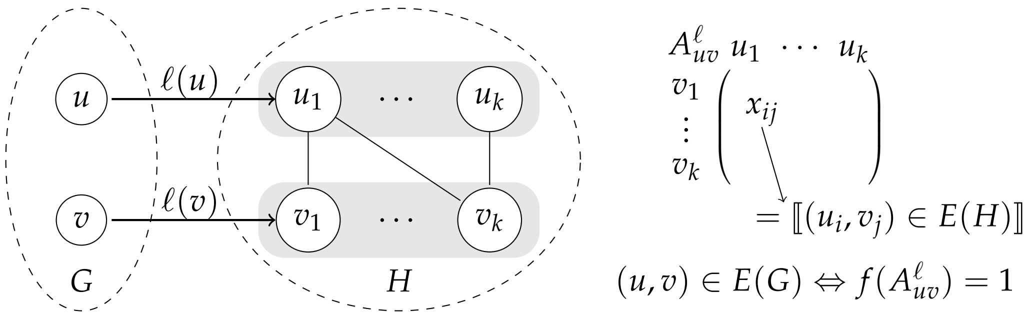
Proof. If a graph class is small and hereditary then so is its edge-complement and therefore we have closure under negation. It remains to show that this set of graph classes is closed under conjunction. Let $\mathcal{C}, \mathcal{D}$ be small and hereditary graph classes. A graph in $\mathcal{C} \wedge \mathcal{D}$ is determined by choosing a graph in $\mathcal{C}$ and $\mathcal{D}$ and therefore $|(\mathcal{C} \wedge \mathcal{D})_n| \leq |\mathcal{C}_n| \cdot |\mathcal{D}_n| \in n^{\mathcal{O}(n)}$ since $\mathcal{C}$ and $\mathcal{D}$ are small. Given a graph $G \in \mathcal{C} \wedge \mathcal{D}$. Let $G = H_1 \wedge H_2$ with $H_1 \in \mathcal{C}, H_2 \in \mathcal{D}$. It holds that $G[V'] = H_1[V'] \wedge H_2[V']$ for all vertex subsets V' of G. Since $\mathcal{C}$ and $\mathcal{D}$ are hereditary it follows that every induced subgraph of G is in $\mathcal{C} \wedge \mathcal{D}$. □

Let us say a graph class $\mathcal{C}$ is $\leq_{\mathrm{BF}}$-complete for a set of graph classes $\mathbb{A}$ if $\mathcal{C}$ is in $\mathbb{A}$ and $\mathcal{D} \leq_{\mathrm{BF}} \mathcal{C}$ for every $\mathcal{D} \in \mathbb{A}$. Alternatively, one can also say $\mathcal{C}$ is complete for $\mathbb{A}$ if $[\mathcal{C}]_{\leq_{\mathrm{BF}}} = \mathbb{A}$. Observe that a directed graph class is not $\leq_{\mathrm{BF}}$-reducible to an undirected one. Since classes such as GP and GAC^0 contain directed graph classes it trivially follows that no undirected graph class can be complete for them. If we assume that only undirected graph classes can be small and hereditary then it trivially holds that GAC^0 and its supersets do not have a $\leq_{\mathrm{BF}}$-complete small and hereditary graph class. To make this more interesting we can either drop this assumption or we can restrict GAC^0 to undirected graph classes. Irregardless of the choice, the following statement holds.

Fact 3.29. *There exists no small, hereditary graph class that is $\leq_{\mathrm{BF}}$-complete for* GAC^0, *or a superset thereof.*

Proof. Assume there exists a small, hereditary graph class $\mathcal{C}$ that is complete for GAC^0 with respect to $\leq_{\mathrm{BF}}$. Since $\mathcal{C}$ is in $[\text{Small} \cap \text{Hereditary}]_{\subseteq}$ and $[\text{Small} \cap \text{Hereditary}]_{\subseteq}$ is closed under $\leq_{\mathrm{BF}}$ it follows that every class reducible to $\mathcal{C}$ must be in $[\text{Small} \cap \text{Hereditary}]_{\subseteq}$ and thus $\mathsf{GAC}^0 \subseteq [\text{Small} \cap \text{Hereditary}]_{\subseteq}$. This contradicts Theorem 3.16. □

We conclude this subsection by showing that every uniformly sparse graph class is $\leq_{\mathrm{BF}}$-reducible to interval graphs. A graph class is uniformly sparse iff it has bounded arboricity. Stated differently, $\mathcal{C}$ is uniformly sparse iff there exists a $k \in \mathbb{N}$ such that $\mathcal{C} \subseteq \bigvee_{i=1}^{k} \text{Forest}$.



u
$\ell(u)$
u_1
$\cdots$
u_k
v
$\ell(v)$
v_1
v_k
G
H
A^ℓ_{uv} u_1 $\cdots$ u_k
v_1
v_k
x_{ij}
$= [\![(u_i, v_j) \in E(H)]\!]$
$(u,v) \in E(G) \Leftrightarrow f(A^\ell_{uv}) = 1$

FIGURE 3.5: Schematic of an (H, f)-representation of a graph G with labeling ℓ

It holds that every forest has boxicity at most 2, i.e. Forest $\subseteq$ Interval $\wedge$ Interval. Therefore if $\mathcal{C}$ is uniformly sparse then there exists a $k \in \mathbb{N}$ such that $\mathcal{C} \subseteq \bigvee_{i=1}^{k}$(Interval $\wedge$ Interval). Notice that this reduction does not require negation and therefore is a $\leq_{\mathrm{M}}$-reduction where M denotes the monotone clone $[\wedge, \vee]_{\text{clone}}$.

3.3.2 Subgraph Reductions

Given $k \geq 0$, a k^2-ary Boolean function f and a $(k \times k)$-matrix A over $\{0,1\}$. We write $f(A)$ to denote the value of f when plugging in the entries of A from left to right and top to bottom. We say f is diagonal if the value of f only depends on the k entries on the main diagonal of A. Given $k, l \geq 1$, a k^2-ary Boolean function f and an l^2-ary Boolean function g. We define the composition of f with g to be the $(kl)^2$-ary Boolean function

$$(f \circ g) \begin{pmatrix} B_{1,1} & \dots & B_{1,k} \\ \vdots & \ddots & \vdots \\ B_{k,1} & \dots & B_{k,k} \end{pmatrix} = f \begin{pmatrix} g(B_{1,1}) & \dots & g(B_{1,k}) \\ \vdots & \ddots & \vdots \\ g(B_{k,1}) & \dots & g(B_{k,k}) \end{pmatrix}$$

where $B_{i,j}$ is a $(l \times l)$-matrix for $i, j \in [k]$.

Definition 3.30. *Given graphs G, H, $k \in \mathbb{N}$ and a k^2-ary Boolean function f. We say G has an (H, f)-representation if there exists an $\ell \colon V(G) \to V(H)^k$ such that for all $u \neq v \in V(G)$*

$$(u, v) \in E(G) \Leftrightarrow f(A^\ell_{uv}) = 1$$

with $A^\ell_{uv} = ([\![(\ell(u)_i, \ell(v)_j) \in E(H)]\!])_{i,j \in [k]}$.

Definition 3.31 (Subgraph Reduction)**.** *Given graph classes $\mathcal{C}, \mathcal{D}$. We say $\mathcal{C} \leq_{\text{sg}} \mathcal{D}$ if there exist $c, k \in \mathbb{N}$ and a k^2-ary Boolean function f such that for all $n \in \mathbb{N}$ and $G \in \mathcal{C}_n$ there exists an $H \in \mathcal{D}_{n^c}$ such that G has an (H, f)-representation. We say $\mathcal{C} \leq^{\text{diag}}_{\text{sg}} \mathcal{D}$ if this holds for a diagonal f. We write $[\mathcal{C}]_{\text{sg}}$ to denote the set of graph classes $\leq_{\text{sg}}$-reducible to $\mathcal{C}$.*

In the case of intersection graph classes the subgraph reduction can be simplified. Given two families of sets X and Y let $\mathcal{C}_X$ and $\mathcal{C}_Y$ denote the intersection graph classes that they induce. It holds that $\mathcal{C}_Y \leq_{\text{sg}} \mathcal{C}_X$ if there exists a $k \in \mathbb{N}$, a k^2-ary Boolean function f and a labeling $\ell \colon Y \to X^k$ such that for all $u \neq v \in Y$ it holds that $u \cap v \neq \emptyset \Leftrightarrow f(A^\ell_{uv}) = 1$ with $A^\ell_{u,v} = ([\![\ell(u)_i \cap \ell(v)_j \neq \emptyset]\!])_{i,j \in [k]}$. For instance, let X be the set of intervals on the real line and $Y = \left\{ \cup_{i=1}^{k} x_i \mid x_1, \dots, x_k \in X \right\}$ for some $k \in \mathbb{N}$. This means $\mathcal{C}_X$ is the set of interval graphs and $\mathcal{C}_Y$ is the set of k-interval graphs. To reduce $\mathcal{C}_Y$ to $\mathcal{C}_X$ we can choose the labeling $\ell(y) = (x_1, \dots, x_k)$ with $y \in Y$ and $x_1, \dots, x_k \in X$ such that $y = \cup_{i=1}^{k} x_i$ and f is the Boolean function that returns one if at least one of its arguments is one.

Let $\mathcal{C}, \mathcal{D}$ be two graph classes such that $\mathcal{C} \leq_{\text{sg}} \mathcal{D}$ via $c, k \in \mathbb{N}$ and a k^2-ary Boolean function f. In the definition of subgraph reductions it is required that every graph on n vertices in $\mathcal{C}$ must have an (H, f)-representation for a graph $H \in \mathcal{D}$ on exactly n^c vertices. The following lemma shows that if $\mathcal{D}$ satisfies some fairly weak conditions then this size restriction becomes obsolete.

We call a graph class $\mathcal{C}$ inflatable if for every graph G on n vertices in $\mathcal{C}$ and all $m > n$ there exists a graph on m vertices in $\mathcal{C}$ which contains G as induced subgraph.

Lemma 3.32. *Let $\mathcal{D}$ be a hereditary and inflatable graph class. It holds for all graph classes $\mathcal{C}$ that $\mathcal{C} \leq_{\text{sg}} \mathcal{D}$ iff there exists a $k \in \mathbb{N}$ and a k^2-Boolean function f such that every graph in $\mathcal{C}$ has an (H, f)-representation for some $H \in \mathcal{D}$.*

Proof. The direction "$\Rightarrow$" is clear. For the other direction consider a graph class $\mathcal{C}$ and let f be a k^2-ary Boolean function such that every graph in $\mathcal{C}$ has an (H, f)-representation for some $H \in \mathcal{D}$. We show that $\mathcal{C} \leq_{\text{sg}} \mathcal{D}$ via $c = k, k$ and f. Let G be a graph in $\mathcal{C}$ with n vertices. We show that there exists a graph H'' on n^k vertices in $\mathcal{D}$ such that G has an (H'', f)-representation. There exists a graph H in $\mathcal{D}$ such that G has an (H, f)-representation via a labeling $\ell \colon V(G) \to V(H)^k$. Let V' be the set of vertices of H that occur in the image of ℓ. Let H' be the induced subgraph of H on V'. It holds that H' has at most kn vertices and G has an (H', f)-representation via ℓ. Since $\mathcal{D}$ is hereditary it holds that H' is in it. Let H'' be a graph with n^k vertices in $\mathcal{D}$ which contains H' as induced subgraph. Since $\mathcal{D}$ is inflatable such a graph must exist. It follows that G has an (H'', f)-representation via ℓ. □

Lemma 3.33. $\leq_{\text{sg}}$ *and* $\leq_{\text{sg}}^{\text{diag}}$ *are reflexive and transitive.*

Proof. For reflexivity it suffices to show that $\mathcal{C} \leq_{\text{sg}}^{\text{diag}} \mathcal{C}$. This holds because every $G \in \mathcal{C}$ has a (G, f)-representation with $f(x) = x$ ($c = k = 1$) and f is diagonal.

For transitivity assume that $\mathcal{C} \leq_{\text{sg}} \mathcal{D}$ via $c, k \in \mathbb{N}$ and a k^2-ary Boolean function f and $\mathcal{D} \leq_{\text{sg}} \mathcal{E}$ via $d, l \in \mathbb{N}$ and an l^2-ary Boolean function g. We show that $\mathcal{C} \leq_{\text{sg}} \mathcal{E}$ via cd, kl and the $(kl)^2$-ary Boolean function $f \circ g$. Given $G \in \mathcal{C}_n$ we need to show that there exists an $I \in \mathcal{E}_{n^{cd}}$ such that G has an $(I, f \circ g)$-representation. Since $G \in \mathcal{C}_n$ there exist an $H \in \mathcal{D}_{n^c}$ such that G has an (H, f)-representation via a labeling $\ell \colon V(G) \to V(H)^k$. Also, there exists an $I \in \mathcal{E}_{n^{cd}}$ such that H has an (I, g)-representation via a labeling $\ell' \colon V(H) \to V(I)^l$. Then it can be verified that G has an $(I, f \circ g)$-representation due to the following labeling $\ell'' \colon V(G) \to V(I)^{kl}$. For $u \in V(G)$ let $\ell(u) = (u_1, \dots, u_k)$ and let $\ell'(u_i) = (u_{i,1}, \dots, u_{i,l})$ for $i \in [k]$. We define $\ell''(u)$ as $(u_{1,1}, \dots, u_{1,l}, u_{2,1}, \dots, u_{2,l}, \dots, u_{k,1}, \dots, u_{k,l})$. The same argument shows that $\leq_{\text{sg}}^{\text{diag}}$ is transitive because the composition of two diagonal Boolean functions yields a diagonal Boolean function. □

Lemma 3.34. *The class* GAC^0 *is closed under* $\leq_{\text{sg}}$.

Proof. Let $\mathcal{C} \leq_{\text{sg}} \mathcal{D}$ and let $\mathcal{D} \in \mathsf{GAC}^0$. We need to show that $\mathcal{C} \in \mathsf{GAC}^0$. Let $\mathcal{D} \in \mathsf{GAC}^0$ via the labeling scheme $S = (F, c)$ and $\mathcal{C} \leq_{\text{sg}} \mathcal{D}$ via $d, k \in \mathbb{N}$ and a k^2-ary Boolean function f. Then we claim that the labeling scheme $S' = (F', cdk)$ with

$$(x_1 \dots x_k, y_1 \dots y_k) \in F' \Leftrightarrow f \begin{pmatrix} [\![(x_1, y_1) \in F]\!] & [\![(x_1, y_2) \in F]\!] & \dots \\ \vdots & \ddots & \\ [\![(x_k, y_1) \in F]\!] & & [\![(x_k, y_k) \in F]\!] \end{pmatrix} = 1$$

for all $x_i, y_i \in \{0,1\}^{cdm}$ with $m \in \mathbb{N}$ and $i \in [k]$ represents $\mathcal{C}$ and F' can be computed in AC^0 since F can be computed in AC^0. We show how to label a given $G \in \mathcal{C}_n$. Since $\mathcal{C} \leq_{\mathrm{sg}} \mathcal{D}$ there exist an $H \in \mathcal{D}_{n^d}$ such that G has an (H,f)-representation via the labeling $\ell_G \colon V(G) \to V(H)^k$. Since $\mathcal{D}$ is represented by S there also exists a labeling $\ell_H \colon V(H) \to \{0,1\}^{c \log n^d}$ such that H is represented by S via ℓ_H. Then a vertex u of G can be labeled with $\ell_H(u_1) \dots \ell_H(u_k)$ where u_i is the i-th component of $\ell_G(u)$. □

Corollary 3.35. *The classes* $\mathsf{GL}, \mathsf{GP}, \mathsf{GEXP}, \mathsf{GR}$ *and* GALL *are closed under* $\leq_{\mathrm{sg}}$.

Proof. Observe that for all these complexity classes the same argument as the one given for GAC^0 in Lemma 3.34 works. □

Lemma 3.36. *Let $\mathcal{C}, \mathcal{D}$ be graph classes and $\mathcal{D}$ is in* $[\text{Small} \cap \text{Hereditary}]_{\subseteq}$. *If $\mathcal{C} \leq_{\mathrm{sg}} \mathcal{D}$ holds then $\mathcal{C}$ is small.*

Proof. Let $\mathcal{C} \leq_{\mathrm{sg}} \mathcal{D}$ via $c, k \in \mathbb{N}$ and a k^2-ary Boolean function f and $\mathcal{D}$ is small and hereditary. A graph G with n vertices in $\mathcal{C}$ is determined by a graph H with n^c vertices in $\mathcal{D}$ and a labeling $\ell \colon V(G) \to V(H)^k$. Observe that the adjacency relation of G only depends on an induced subgraph H' of H with at most kn vertices (every vertex of H that occurs in the image of ℓ). Since $\mathcal{D}$ is small and hereditary the number of options for H' is limited by

$$\sum_{i=1}^{kn} |\mathcal{D}_i| \leq \sum_{i=1}^{kn} i^{\mathcal{O}(i)} \leq kn \cdot kn^{\mathcal{O}(kn)} \leq n^{\mathcal{O}(n)}$$

The number of different labelings $\ell \colon V(G) \to V(H')^k$ is limited by $(kn)^{kn} \leq n^{\mathcal{O}(n)}$. Therefore $\mathcal{C}$ is small. □

Lemma 3.37. *The class* $[\text{Small} \cap \text{Hereditary}]_{\subseteq}$ *is closed under* $\leq_{\mathrm{sg}}$.

Proof. Let $\mathcal{C}, \mathcal{D}$ be graph classes such that $\mathcal{C} \leq_{\mathrm{sg}} \mathcal{D}$ via $c, k \in \mathbb{N}$ and a k^2-ary Boolean function f and $\mathcal{D}$ is in $[\text{Small} \cap \text{Hereditary}]_{\subseteq}$. For the sake of contradiction let us assume that $\mathcal{C}$ is not a subset of a small and hereditary graph class. Let $\mathcal{C}'$ be the closure of $\mathcal{C}$ under induced subgraphs. The class $\mathcal{C}'$ cannot be small since this would imply that $\mathcal{C}$ is in $[\text{Small} \cap \text{Hereditary}]_{\subseteq}$. We argue that there exists a finite graph class $\mathcal{E}$ such that $\mathcal{C}' \setminus \mathcal{E} \leq_{\mathrm{sg}} \mathcal{D}$ via c, k and f. From the previous lemma it follows that $\mathcal{C}' \setminus \mathcal{E}$ is small. Since $\mathcal{E}$ is only finite it follows that $\mathcal{C}'$ is also small, which is a contradiction. Let n_0 be the smallest number such that $kn_0 \leq n_0^c$ and $\mathcal{E}$ is the class of all graphs with at most n_0 vertices. Let $G' \in \mathcal{C}' \setminus \mathcal{E}$ with n' vertices. We need to show that there exists a graph $H' \in \mathcal{D}$ on $(n')^c$ vertices such that G' has an (H', f)-representation. There exists a $G \in \mathcal{C}$ on n vertices such that G' is an induced subgraph of G and an $H \in \mathcal{D}_{n^c}$ such that G has an (H, f)-representation via a labeling $\ell \colon V(G) \to V(H)^k$. Therefore G' has an (H, f)-representation via the labeling $\ell' \colon V(G') \to V(H)^k$ defined as the restriction of ℓ to vertices from G'. Since G' is an induced subgraph of G it holds that $n' \leq n$ and therefore $(n')^c \leq n^c$. Additionally, since $G \notin \mathcal{E}$ it holds that $kn' \leq (n')^c$. To construct the desired graph $H' \in \mathcal{D}$ with $(n')^c$ vertices such that G' has an (H', f)-representation one can delete vertices from H until only $(n')^c$ vertices remain but every vertex in the image of ℓ' is still in H'. □

Corollary 3.38. *There exists no small, hereditary graph class that is complete for* GAC^0 *(or a superset thereof) with respect to* $\leq_{\mathrm{sg}}$.

Proof. GAC^0 is no subset of $[\text{Small} \cap \text{Hereditary}]_\subseteq$ due to Theorem 3.16. If a small and hereditary graph class is complete for GAC^0 w.r.t. $\leq_{\text{sg}}$ then this would imply that GAC^0 is a subset of $[\text{Small} \cap \text{Hereditary}]_\subseteq$ since $[\text{Small} \cap \text{Hereditary}]_\subseteq$ is closed under $\leq_{\text{sg}}$. □

We call a graph class $\mathcal{C}$ self-universal if for every finite subset of $\mathcal{C}$ there exists a graph in $\mathcal{C}$ that contains every graph of that finite subset as induced subgraph. For instance, every graph class that is closed under disjoint union is self-universal.

Lemma 3.39. *Let $\mathcal{D}$ be a self-universal and inflatable graph class. Then it holds that $\mathcal{C} \leq_{\text{BF}} \mathcal{D}$ implies $\mathcal{C} \leq_{\text{sg}}^{\text{diag}} \mathcal{D}$ for all graph classes $\mathcal{C}$.*

Proof. Since $\mathcal{C} \leq_{\text{BF}} \mathcal{D}$ there exists a k-ary Boolean function f for some $k \in \mathbb{N}$ such that $\mathcal{C} \subseteq f(\mathcal{D}, \ldots, \mathcal{D})$. Let f' be defined as the k^2-ary diagonal Boolean function $f'(A) = f(A_{1,1}, \ldots, A_{k,k})$. To show that $\mathcal{C} \leq_{\text{sg}}^{\text{diag}} \mathcal{D}$ we argue that for all $n \in \mathbb{N}$ every graph $G \in \mathcal{C}_n$ has an (H, f')-representation for some $H \in \mathcal{D}_{n^k}$. It holds that $G = f(H_1, \ldots, H_k)$ for some $H_1, \ldots, H_k \in \mathcal{D}_n$ and $G, H_1, \ldots, H_k$ all have the same vertex set V. Since $\mathcal{D}$ is self-universal and inflatable there exists a graph $H \in \mathcal{D}_{n^k}$ which contains $H_1, \ldots, H_k$ as induced subgraphs. Let $\pi_i \colon V \to V(H)$ be the witness that shows that H_i is an induced subgraph of H for $i \in [k]$. Then G has an (H, f)-representation via the labeling $\ell(u) = (\pi_1(u), \ldots, \pi_k(u))$. □

Fact 3.40. *For every intersection graph class $\mathcal{D}$ it holds that $\mathcal{C} \leq_{\text{BF}} \mathcal{D}$ iff $\mathcal{C} \leq_{\text{sg}}^{\text{diag}} \mathcal{D}$ for all graph classes $\mathcal{C}$.*

Proof. "$\Rightarrow$": Every intersection graph class is self-universal and inflatable. Therefore this direction follows from the previous lemma.

"$\Leftarrow$": Let $\mathcal{C} \leq_{\text{sg}}^{\text{diag}} \mathcal{D}$ via $c, k \in \mathbb{N}$ and a k^2-ary diagonal Boolean function f. Let g be the k-ary Boolean function which underlies f, i.e. $f(A) = g(A_{1,1}, \ldots, A_{k,k})$. We claim that $\mathcal{C} \leq_{\text{BF}} \mathcal{D}$ via g. This means for every $G \in \mathcal{C}$ we need to show that there exist $H_1, \ldots, H_k \in \mathcal{D}$ on vertex set $V(G)$ such that $G = g(H_1, \ldots, H_k)$. Given $G \in \mathcal{C}_n$ there exist an $H \in \mathcal{D}_{n^c}$ and a labeling $\ell \colon V(G) \to V(H)^k$ such that $(u, v) \in E(G)$ iff $g(x_1, \ldots, x_k) = 1$ with $x_i := [\![(\ell(u)_i, \ell(v)_i) \in E(H)]\!]$. Since $\mathcal{D}$ is an intersection graph class we can assume without loss of generality that no vertex of H occurs more than once in the image of ℓ (otherwise we could just clone that vertex). Let H_i be the induced subgraph of H that has $\{\ell(u)_i \mid u \in V(G)\}$ as vertex set. Since $\mathcal{D}$ is hereditary it follows that H_i is in $\mathcal{D}$ for all $i \in [k]$. □

3.4 Logical Labeling Schemes

Consider the labeling scheme for interval graphs that we have seen in the introduction. A vertex of an interval graph is labeled with two numbers which represent the endpoints of its interval. The label decoder for this labeling scheme can be expressed as FO formula $\varphi(x_1, x_2, y_1, y_2) \triangleq \neg(x_2 < y_1 \vee y_2 < x_1)$. Given two vertices u, v with labels $u_1, u_2, v_1, v_2 \in \mathbb{N}$ and $u_1 < u_2, v_1 < v_2$ it holds that $[u_1, u_2]$ and $[v_1, v_2]$ intersect iff $\varphi(u_1, u_2, v_1, v_2)$ holds. Since φ only describes the label decoder we need to specify the label length in order to extend this to a (logical) labeling scheme (φ, c). First, observe that a binary string of length $ck \log n$ can be interpreted as k numbers from $[n^c]$. k describes the number of variables per vertex and therefore is exactly half the number of free variables of φ (in this case $k = 2$). For an interval graph with n vertices it suffices to pick numbers from $[2n] \subseteq [n^2]$ which means $c = 2$. Therefore (φ, c) is a logical labeling scheme for interval graphs. Labeling schemes for the pointer numbers and many intersection graph classes can be expressed as logical labeling schemes. The only graph classes with a labeling scheme for which we do not know whether they also admit a logical labeling scheme are those that have bounded clique-width. Our motivation for introducing logical labeling schemes is that they strike a balance between expressiveness and being amenable to analysis, which does not seem to apply to GP or even GAC^0.

Quantifier-free logical labeling schemes are well-behaved from a graph-theoretical point of view in the sense that graph classes which can be expressed by such labeling schemes must be a subset of a small and hereditary graph class. This immediately separates them from classes such as GAC^0. We call the set of graph classes with a quantifier-free logical labeling scheme $\mathsf{GFO}_{\mathrm{qf}}$. The class $\mathsf{GFO}_{\mathrm{qf}}$ can be characterized in terms of constant-time RAMs and in terms of what we call polynomial-Boolean systems. Most candidates for the implicit graph conjecture such as line segment graphs or k-dot product graphs (small and hereditary graph classes not known to have a labeling scheme) can be described by polynomial-Boolean systems. A particularly interesting observation is that if $\mathsf{GFO}_{\mathrm{qf}}$ has a hereditary $\leq_{\mathrm{BF}}$-complete graph class then all graph classes that can be described by a polynomial-Boolean system are also in $\mathsf{GFO}_{\mathrm{qf}}$ and therefore would cease to be candidates for the implicit graph conjecture. Additionally, we show that algebraic reductions are intimately related to quantifier-free logical labeling schemes and give examples of complete graph classes for certain fragments.

3.4.1 Definition and Basic Properties

Definition 3.41. *A (quantifier-free, atomic) logical labeling scheme is a tuple $S = (\varphi, c)$ with a (quantifier-free, atomic) formula $\varphi \in \mathsf{FO}_{2k}$ and $c, k \in \mathbb{N}$. A graph G is in $gr(S)$ if there exists a labeling $\ell \colon V(G) \to [n^c]_0^k$ such that $(u,v) \in E(G) \Leftrightarrow \mathcal{N}_{n^c}, (\ell(u), \ell(v)) \models \varphi$ for all $u \neq v \in V(G)$.*

Definition 3.42. *Let $\sigma \subseteq \{<, +, \times\}$. A graph class $\mathcal{C}$ is in $\mathsf{GFO}(\sigma)$ if there exists a logical labeling scheme (φ, c) with $\varphi \in \mathsf{FO}_{2k}(\sigma)$ and $c, k \in \mathbb{N}$ such that $\mathcal{C} \subseteq gr(\varphi, c)$. Let $\mathsf{GFO}_{\mathrm{qf}}(\sigma)$ denote the quantifier-free analogue.*

We say a logical labeling scheme $S = (\varphi, c)$ is in $\mathsf{GFO}_{(\mathrm{qf})}(\sigma)$ if φ is a (quantifier-free) formula in $\mathsf{FO}(\sigma)$.

Lemma 3.43. *Let $\sigma \subseteq \{<, +, \times\}$ and let A be a complexity class that is closed under AC^0 many-one reductions. If the bounded model checking problem for every (quantifier-free) formula in $\mathsf{FO}(\sigma)$ can be decided in A then $\mathsf{GFO}_{(\mathrm{qf})}(\sigma) \subseteq \mathsf{GA}$.*

Proof. We show that the above statement holds for $\sigma = \{<, +, \times\}$ and formulas with quantifiers. From this the result for all restricted classes of formulas follows.

Let us assume that the bounded model checking problem for every formula in FO can be decided in A. Consider a logical labeling scheme $S = (\varphi, c)$ with $2k$ variables. Let G be a graph that is in $\mathrm{gr}(S)$ via a labeling $\ell \colon V(G) \to [n^c]_0^k$. We can translate the labeling ℓ into a binary encoded one $\ell' \colon V(G) \to \{0,1\}^{kc' \log n}$ where $c' = c + 1$ and a block of $c' \log n$ bits represents a number in $[n^c]_0$. For all $u \neq v \in V(G)$ it holds that $(u,v) \in E(G) \Leftrightarrow \mathcal{N}_{n^c}, (\ell(u), \ell(v)) \models \varphi \Leftrightarrow (\ell'(u), \ell'(v), \mathrm{bin}(n^c))$ is a positive instance of the bounded model checking problem for φ. This almost gives us a labeling scheme which shows that $\mathrm{gr}(S)$ is in GA. The problem, however, is that $\mathrm{bin}(n^c)$ is not part of the labeling and the formal definition of GA does not allow us to use any input but the labels. To solve this we append $\text{bin}(n^c)$ to the labeling. Consider a labeling $\ell''\colon V(G) \to \{0,1\}^{(k+1)c' \log n}$ such that $\ell''(u) = \ell'(u)\text{bin}(n^c)$. The last $c' \log n$ bits of $\ell''(u)$ encode n^c. Now, the following labeling scheme $S' = (F, (k+1)c')$ shows that $\text{gr}(S)$ is in GA. The label decoder is defined as $(x_1 \dots x_{k+1}, y_1 \dots y_{k+1}) \in F$ iff $(x_1, \dots, x_k, y_1, \dots, y_k, x_{k+1})$ is a positive instance of the bounded model checking problem for φ for all $x_i, y_i \in \{0,1\}^{c'm}$, $i \in [k+1]$ and $m \in \mathbb{N}$. The label decoder F can be decided in A since it is closed under AC^0 many-one reductions and thus we can transform the input of the label decoder into an instance of the bounded model checking problem. □

Theorem 3.44. $\text{GFO}_{\text{qf}}(<,+) \subsetneq \text{GAC}^0$, $\text{GFO}_{\text{qf}} \subsetneq \text{GTC}^0$ *and* $\text{GFO} \subseteq \text{GPH}$.

Proof. Due to the previous lemma we can prove these inclusions by showing that the bounded model checking problem for formulas in $\text{FO}_{\text{qf}}(<,+)$, FO_{qf} and FO can be solved in AC^0, TC^0 and PH, respectively. First, let us explain why the bounded model checking problem for formulas in FO_{qf} can be solved in TC^0. The naive approach to model-check a quantifier-free formula φ is as follows: evaluate the terms of φ (expressions involving addition and multiplication of the free variables (the input)), then evaluate the atomic formulas which means comparing numbers and finally compute the underlying Boolean function of φ. Since the order relation, addition and multiplication can be computed in TC^0 (see [Vol99]) and φ is fixed this naive approach can be realized by a family of TC^0-circuits (for addition and multiplication one has to additionally handle the overflow condition). Since order and addition can be computed in AC^0 it follows that the bounded model checking problem for formulas in $\text{FO}_{\text{qf}}(<,+)$ is in AC^0. In the case of formulas with quantifiers we can use the non-determinism of PH to evaluate them. Observe that the number of bits that need to be guessed are only linear w.r.t. the input size. Therefore we can conclude that $\text{GFO}_{\text{qf}}(<,+) \subseteq \text{GAC}^0$, $\text{GFO}_{\text{qf}} \subseteq \text{GTC}^0$ and $\text{GFO} \subseteq \text{GPH}$. The strictness of the first two inclusions is a consequence of the fact that $\text{GAC}^0 \not\subseteq [\text{Small} \cap \text{Hereditary}]_{\subseteq}$ (see Theorem 3.16) and $\text{GFO}_{\text{qf}} \subseteq [\text{Small} \cap \text{Hereditary}]_{\subseteq}$ which is proved later (see Theorem 3.72 and Corollary 3.77). □

Lemma 3.45. $\text{GFO}(\sigma)$ *and* $\text{GFO}_{\text{qf}}(\sigma)$ *are closed under* $\leq_{\text{BF}}$ *for all* $\sigma \subseteq \{<,+,\times\}$.

Proof. Let $\mathcal{C} \leq_{\text{BF}} \mathcal{D}$ via a k-ary Boolean function f and $\mathcal{D}$ is in $\text{GFO}_{(\text{qf})}(\sigma)$ via a logical labeling scheme $S = (\varphi, c)$ with $2l$ variables. We construct a logical labeling scheme $S' = (\psi, c)$ with $2kl$ variables which shows that $\mathcal{C}$ is in $\text{GFO}_{(\text{qf})}(\sigma)$. Let ψ have variables $x_{i,j}, y_{i,j}$ for $i \in [k]$ and $j \in [l]$ and let us write $\vec{x}_i$ for $x_{i,1}, \dots, x_{i,l}$. Let $\psi(\vec{x}_1, \dots, \vec{x}_k, \vec{y}_1, \dots, \vec{y}_k)$ be defined as $f(\varphi(\vec{x}_1, \vec{y}_1), \dots, \varphi(\vec{x}_k, \vec{y}_k))$. We claim that $\mathcal{C} \subseteq \text{gr}(S')$. Consider a graph $G \in \mathcal{C}$ with vertex set V. There exist k graphs $H_1, \dots, H_k \in \mathcal{D}$ with vertex set V such that $G = f(H_1, \dots, H_k)$. Since H_i is in $\mathcal{D}$ it is also in $\text{gr}(S)$ via a labeling $\ell_i\colon V \to [n^c]_0^l$ for $i \in [k]$. Consider the labeling $\ell\colon V \to [n^c]_0^{kl}$ with $\ell(u) = (\ell_1(u), \dots, \ell_k(u))$ for all $u \in V$. It is easy to verify that G is in $\text{gr}(S')$ via ℓ. □

Lemma 3.46. $\text{GFO}(\sigma)$ *and* $\text{GFO}_{\text{qf}}(\sigma)$ *are closed under* $\leq_{\text{sg}}$ *for all* $\sigma \subseteq \{<,+,\times\}$.

Proof. Let $\mathcal{C} \leq_{\text{sg}} \mathcal{D}$ via $c, k \in \mathbb{N}$ and a k^2-ary Boolean function f and $\mathcal{D}$ is in $\text{GFO}_{(\text{qf})}(\sigma)$ via the labeling scheme $S = (\varphi, d)$ and φ has $2l$ variables. Let ψ be a formula with $2kl$ variables $\vec{x}_1, \dots, \vec{x}_k, \vec{y}_1, \dots, \vec{y}_k$ where $\vec{x}_i$ and $\vec{y}_i$ are sequences of l variables. We define ψ as $f(A)$ where A is a $(k \times k)$-matrix with $A_{ij} = \varphi(\vec{x}_i, \vec{y}_j)$ for $i, j \in [k]$. We claim that

$\mathcal{C} \subseteq \mathrm{gr}(\psi, cd)$. Given a graph $G \in \mathcal{C}$ on n vertices there exists a graph $H \in \mathcal{D}$ on n^c vertices such that G has an (H, f)-representation via a labeling $\ell_G \colon V(G) \to V(H)^k$. Also, $H \in \mathrm{gr}(S)$ via a labeling $\ell_H \colon V(H) \to [n^{cd}]_0^l$. Let $\ell \colon V(G) \to [n^{cd}]_0^{kl}$ be defined as follows. Given $u \in V(G)$ let $\ell_G(u) = (u_1, \ldots, u_k)$ and $\ell_H(u_i) = (u_{i,1}, \ldots, u_{i,l})$ for $i \in [k]$. We define $\ell(u)$ as $(u_{1,1}, \ldots, u_{1,l}, u_{2,1}, \ldots, u_{2,l}, \ldots, u_{k,1}, \ldots, u_{k,l})$. It can be verified that G is in $\mathrm{gr}(\psi, cd)$ via the labeling ℓ. No new atoms or quantifiers are introduced in ψ compared to φ and thus it remains in the same class of formulas. $\square$

In a logical labeling scheme $S = (\varphi, c)$ the label length c determines the size of the universe ($\mathcal{N}_{n^c}$) which is used to interpret φ for graphs on n vertices. For quantifier-free formulas the universe size only affects at what point the overflow condition of addition and multiplication applies. Since φ is known a priori one can always choose c sufficiently large in order to ensure that no overflow occurs for all labelings of a predetermined size. Is it possible to exploit the overflow condition to express a graph class that would not be expressible without it? We show that for certain fragments this is not the case. Moreover, for these fragments the formula can be assumed to be always interpreted over $\mathcal{N}$ irregardless of the number of vertices of the graph.

Definition 3.47. *Given a logical labeling scheme $S = (\varphi, c)$. A graph G is in $gr_\infty(S)$ if there exists a labeling $\ell \colon V(G) \to [n^c]_0^k$ such that $(u, v) \in E(G) \Leftrightarrow \mathcal{N}, (\ell(u), \ell(v)) \models \varphi$ for all $u \neq v \in V(G)$.*

Lemma 3.48. *Let $\sigma = \emptyset$, or $\sigma \subseteq \{<, +, \times\}$ and '$<$' is in σ. A graph class $\mathcal{C}$ is in $\mathsf{GFO}_{\mathrm{qf}}(\sigma)$ iff there exists a logical labeling scheme S in $\mathsf{GFO}_{\mathrm{qf}}(\sigma)$ such that $\mathcal{C} \subseteq gr_\infty(S)$.*

Proof. In the case that $\sigma = \emptyset$ it is easy to check that for every logical labeling scheme S in $\mathsf{GFO}_{\mathrm{qf}}(=)$ it holds that $\mathrm{gr}(S) = \mathrm{gr}_\infty(S)$ and thus the above claim holds. Therefore let us consider the case where '$<$' is in σ.

"$\Rightarrow$": Let $\mathcal{C}$ be a graph class that is in $\mathsf{GFO}_{\mathrm{qf}}(\sigma)$ via a logical labeling scheme $S = (\varphi, c)$, i.e. $\mathcal{C} \subseteq \mathrm{gr}(S)$. We construct a logical labeling scheme $S' = (\psi, c)$ such that $\mathcal{C} \subseteq \mathrm{gr}_\infty(S')$ and S' is in $\mathsf{GFO}_{\mathrm{qf}}(\sigma)$. We assume w.l.o.g. that we have access to the constants $c_0 = 0$ and $c_1 = n^c$ in ψ. The constants can be realized by adding two variables to each vertex which are promised to contain the value of the constants for the considered labelings. We build ψ from φ such that the overflow checks are incorporated into the propositional part of ψ. To do this we replace each atomic subformula A of φ by a guarded one A'. We demonstrate how to do this based on the following example. Let $A(x_1, x_2, y_1, y_2)$ be the atomic formula $\times(+(x_1, y_2), x_2) < +(x_2, y_1)$. We convert A into A' by checking whether an overflow occurs at each subterm bottom-up. Let $a \to b$ denote propositional implication which is shorthand for $\neg a \vee b$. Then A' is the following formula (order of operation is implied by indentation and reading a propositional formula of the form $\varphi \to \alpha \wedge \neg\varphi \to \beta$ as "if φ

then α else β").

$$\begin{array}{l}
c_1 < +(x_1,y_2) \rightarrow \\
\quad c_1 < \times(c_0,x_2) \rightarrow \\
\quad\quad c_1 < +(x_2,y_1) \rightarrow \\
\quad\quad\quad c_0 < c_0 \\
\quad\quad \wedge \neg c_1 < +(x_2,y_1) \rightarrow \\
\quad\quad\quad c_0 < +(x_1,y_1) \\
\quad \wedge \neg c_1 < \times(c_0,x_2) \rightarrow \\
\quad\quad c_1 < +(x_2,y_1) \rightarrow \\
\quad\quad\quad \times(c_0,x_2) < c_0 \\
\quad\quad \wedge \neg c_1 < +(x_2,y_1) \rightarrow \\
\quad\quad\quad \times(c_0,x_2) < +(x_2,y_1) \\
\wedge \neg c_1 < +(x_1,y_2) \rightarrow \\
\quad c_1 < \times(+(x_1,y_2),x_2) \rightarrow \\
\quad\quad \vdots
\end{array}$$

The correctness of this transformation follows from showing that $\mathcal{N}_{n^c}, \vec{a} \models A$ iff $\mathcal{N}, \vec{a} \models A'$ for all $\vec{a} \in [n^c]_0^4$.

"$\Leftarrow$": Let $\mathcal{C}$ be a graph class and $S = (\varphi, c)$ is a logical labeling scheme in $\mathsf{GFO}_{\mathrm{qf}}(\sigma)$ such that $\mathcal{C} \subseteq \mathrm{gr}_\infty(S)$. The maximal value that results from evaluating any term in φ must be polynomially bounded, i.e. there exists a $d \in \mathbb{N}$ such that the largest value produced while evaluating φ for a graph with n vertices does not exceed n^{cd}. Therefore $\mathrm{gr}_\infty(\varphi, c) \subseteq \mathrm{gr}(\varphi, cd)$ and $\mathcal{C} \in \mathsf{GFO}_{\mathrm{qf}}(\sigma)$. □

Fact 3.49. *Let $\sigma = \emptyset$, or $\sigma \subseteq \{<, +, \times\}$ and '$<$' is in σ. $\mathsf{GFO}(\sigma)$ and $\mathsf{GFO}_{\mathrm{qf}}(\sigma)$ are closed under union.*

Proof. First, we argue why showing closure under union for $\mathsf{GFO}_{(\mathrm{qf})}(\sigma)$ reduces to proving that ($\star$) for every labeling scheme $S = (\varphi, c)$ in $\mathsf{GFO}_{(\mathrm{qf})}(\sigma)$ there exists a labeling scheme $S' = (\varphi', c+1)$ with $\mathrm{gr}(S) \subseteq \mathrm{gr}(S')$. Let $\mathcal{C}, \mathcal{D}$ be in $\mathsf{GFO}_{(\mathrm{qf})}(\sigma)$ via labeling schemes $S_1 = (\varphi_1, c_1)$ and $S_2 = (\varphi_2, c_2)$. Due to ($\star$) we can assume w.l.o.g. that $c_1 = c_2 = c$. Let $2k_i$ be the number of free variables of φ_i for $i \in [2]$. Furthermore, we assume w.l.o.g. that $\mathrm{gr}(S_1)$ and $\mathrm{gr}(S_2)$ contain all empty graphs $\overline{K_n}$ on n vertices. Let $S = (\psi, c)$ with $\psi(\vec{x_1}, \vec{x_2}, \vec{y_1}, \vec{y_2}) \triangleq \varphi_1(\vec{x_1}, \vec{y_1}) \vee \varphi_2(\vec{x_2}, \vec{y_2})$ where $\vec{x_i}, \vec{y_i}$ are sequences of k_i variables for $i \in [2]$. It holds that S is in $\mathsf{GFO}_{\mathrm{qf}}(\sigma)$. To see that $\mathcal{C} \cup \mathcal{D} \subseteq \mathrm{gr}(S)$ holds consider a graph G on n vertices in $\mathcal{C}$. We can combine a labeling which shows that G is in $\mathrm{gr}(S_1)$ with a labeling that shows that $\overline{K_n}$ is in $\mathrm{gr}(S_2)$ to get a labeling which shows that G is in $\mathrm{gr}(S)$ because $G \vee \overline{K_n} = G$. The correctness relies on the fact that the labeling schemes S, S_1, S_2 are all interpreted over the same universe $\mathcal{N}_{n^c}$ for all graphs with n vertices and $n \in \mathbb{N}$.

Next, let us explain why ($\star$) holds. If $\sigma = \emptyset$ then ($\star$) obviously holds for $\mathsf{GFO}_{\mathrm{qf}}(=)$. Since $\mathsf{GFO}_{\mathrm{qf}}(=) = \mathsf{GFO}(=)$ as we shall see later (Fact 3.53) this also applies to $\mathsf{GFO}(=)$. Therefore we consider the case where '$<$' is in σ. Let $S = (\varphi, c)$ be a labeling scheme with $2k$ variables in $\mathsf{GFO}_{(\mathrm{qf})}(\sigma)$. We assume that φ is in prenex normal form and it has $q \geq 0$ quantified variables, i.e. $\varphi(\vec{x}) \triangleq Q_1 z_1 \ldots Q_q z_q \psi(\vec{x}, \vec{z})$ with $Q_i \in \{\forall, \exists\}$ for $i \in [q]$ and $\vec{z} = (z_1, \ldots, z_q)$ and ψ is a quantifier-free formula. Let ψ' be the formula that is obtained from ψ by incorporating the overflow checks into the propositional part (the same construction that is used in the proof of Lemma 3.48). Let $\varphi'(\vec{x})$ be a FO formula which is equivalent to $(Q_1 z_1 \leq c_1) \dots (Q_q z_q \leq c_1) : \psi(\vec{x}, \vec{z})$ where c_1 is a constant representing the value n^c. It holds that $\mathrm{gr}(S) \subseteq \mathrm{gr}(\varphi', c+1)$. $\square$

Lemma 3.50. *Let $\sigma \subseteq \{<, +, \times\}$. Every graph class in $\mathsf{GFO}_{(\mathrm{qf})}(\sigma)$ can be represented by a logical labeling scheme $S = (\varphi, c)$ in $\mathsf{GFO}_{(\mathrm{qf})}(\sigma)$ such that every free variable of φ occurs in at most one atom of φ.*

Proof. Let $\mathcal{C}$ be in $\mathsf{GFO}_{(\mathrm{qf})}(\sigma)$ via a logical labeling scheme $S = (\varphi, c)$. Suppose φ has free variables $x_1, \dots, x_k, y_1, \dots, y_k$ and atoms $A_1, \dots, A_a$. Let ψ be a formula with $2ak$ free variables $x_{i,j}, y_{i,j}$ with $i \in [k], j \in [a]$ defined as follows. The formula ψ is obtained by renaming every variable x_i and y_i that occurs in the atom A_j of φ to $x_{i,j}$ and $y_{i,j}$ for all $i \in [k], j \in [a]$. To show that $\mathrm{gr}(\varphi, c) \subseteq \mathrm{gr}(\psi, c)$ consider a graph G which is in $\mathrm{gr}(\varphi, c)$ via a labeling $\ell \colon V(G) \to [n^c]_0^k$. The following labeling $\ell' \colon V(G) \to [n^c]_0^{ak}$ shows that G is in $\mathrm{gr}(\psi, c)$. Given a vertex $u \in V(G)$ let $\ell(u) = (u_1, \dots, u_k)$. We define $\ell'(u)$ as $(\underbrace{u_1, \dots, u_1}_{a \text{ times}}, \dots, \underbrace{u_k, \dots, u_k}_{a \text{ times}})$. $\square$

Theorem 3.51 (Algebraic Interpretation)**.** *Let $\sigma = \emptyset$, or $\sigma \subseteq \{<, +, \times\}$ and '$<$' is in σ. A graph class $\mathcal{C}$ is in $\mathsf{GFO}_{\mathrm{qf}}(\sigma)$ iff there exist an $a \in \mathbb{N}$, atomic labeling schemes $S_1, \dots, S_a$ in $\mathsf{GFO}_{\mathrm{qf}}(\sigma)$ and an a-ary Boolean function f such that $\mathcal{C} \subseteq f(gr_\infty(S_1), \dots, gr_\infty(S_a))$.*

Proof. "$\Rightarrow$": Let $\mathcal{C}$ be in $\mathsf{GFO}_{\mathrm{qf}}(\sigma)$. Due to Lemma 3.48 there exists a a logical labeling scheme $S = (\varphi, c)$ in $\mathsf{GFO}_{\mathrm{qf}}(\sigma)$ such that $\mathcal{C} \subseteq \mathrm{gr}_\infty(S)$. Let $A_1, \dots, A_a$ be the atoms of φ and f is the underlying a-ary Boolean function of φ. Let φ have $2ak$ variables $x_{i,j}, y_{i,j}$ with $i \in [a]$ and $j \in [k]$. Furthermore, let the set of variables used in A_i be a subset of $\{x_{i,j}, y_{i,j} \mid j \in [k]\}$ for $i \in [a]$. This means that the variables that occur in A_i and A_j are disjoint for all $i \neq j \in [a]$, which can be assumed w.l.o.g. due to the previous lemma. We claim that $\mathcal{C} \subseteq f(\mathrm{gr}_\infty(A_1, c), \dots, \mathrm{gr}_\infty(A_a, c))$. For a graph $G \in \mathcal{C}$ there exist labelings $\ell_i \colon V(G) \to [n^c]_0^k$ for each $i \in [a]$ such that

$$(u, v) \in E(G) \Leftrightarrow f(x_1, \dots, x_a) = 1 \text{ with } x_i := [\![\mathcal{N}, (\ell_i(u), \ell_i(v)) \models A_i]\!]$$

for all $u \neq v \in V(G)$. Let H_i be the graph with the same vertex set as G and there is an edge $(u, v) \in E(H_i)$ iff $\mathcal{N}, (\ell_i(u), \ell_i(v)) \models A_i$ for all $i \in [a]$. It holds that $G = f(H_1, \dots, H_a)$ and $H_i \in \mathrm{gr}_\infty(A_i, c)$ via ℓ_i.

"$\Leftarrow$": Since $\mathsf{GFO}_{\mathrm{qf}}(\sigma)$ is closed under union (Fact 3.49) it holds that $\mathcal{D} = \bigcup_{i=1}^{a} \mathrm{gr}_\infty(S_i)$ is in $\mathsf{GFO}_{\mathrm{qf}}(\sigma)$. Additionally, $\mathcal{C} \leq_{\mathrm{BF}} \mathcal{D}$ via the a-ary Boolean function f because $\mathcal{C} \subseteq f(\mathcal{D}, \dots, \mathcal{D})$. It follows that $\mathcal{C}$ is in $\mathsf{GFO}_{\mathrm{qf}}(\sigma)$ due to closure under $\leq_{\mathrm{BF}}$. $\square$

Theorem 3.52. $\mathsf{GFO}_{\mathrm{qf}}(<) = \mathsf{GFO}_{\mathrm{qf}}(<, +) = \mathsf{GFO}_{\mathrm{qf}}(<, \times)$.

Proof. To prove that $\mathsf{GFO}_{\mathrm{qf}}(<, \alpha)$ is a subset of $\mathsf{GFO}_{\mathrm{qf}}(<)$ for $\alpha \in \{+, \times\}$ we argue that it suffices to show that for every atomic labeling scheme S in $\mathsf{GFO}_{\mathrm{qf}}(<, \alpha)$ it holds that $\mathrm{gr}_\infty(S) \in \mathsf{GFO}_{\mathrm{qf}}(<)$. Given a graph class $\mathcal{C} \in \mathsf{GFO}_{\mathrm{qf}}(<, \alpha)$, there exist atomic labeling schemes $S_1, \dots, S_a$ in $\mathsf{GFO}_{\mathrm{qf}}(<, \alpha)$ and an a-ary Boolean function f such that $\mathcal{C} \subseteq f(\mathrm{gr}_\infty(S_1), \dots, \mathrm{gr}_\infty(S_a))$ because of Theorem 3.51. By assumption it holds that $\mathrm{gr}_\infty(S_1), \dots, \mathrm{gr}_\infty(S_a)$ are in $\mathsf{GFO}_{\mathrm{qf}}(<)$ and therefore $\mathcal{D} = \bigcup_{i=1}^{k} \mathrm{gr}_\infty(S_i)$ is in $\mathsf{GFO}_{\mathrm{qf}}(<)$

due to closure under union. Then $\mathcal{C} \leq_{\mathrm{BF}} \mathcal{D}$ via f and due to closure under $\leq_{\mathrm{BF}}$ it follows that $\mathcal{C} \in \mathsf{GFO}_{\mathrm{qf}}(<)$.

Let $S = (\varphi, c)$ be an atomic labeling scheme in $\mathsf{GFO}_{\mathrm{qf}}(<, \alpha)$. We argue that $\mathrm{gr}_\infty(S)$ is in $\mathsf{GFO}_{\mathrm{qf}}(<)$. Using $\mathrm{gr}_\infty(S)$ instead of $\mathrm{gr}(S)$ allows us to assume that addition and multiplication are associative. Let φ have variables $x_1, \ldots, x_k, y_1, \ldots, y_k$. The idea is to rearrange the (in)equation such that the variables $x_1, \ldots, x_k$ are on one side of the (in)equation and $y_1, \ldots, y_k$ are on the other side. This allows us to precompute the required values in the labeling of the new labeling scheme which does not use α. Let us show how this works in detail when α is '$+$' and φ uses '$<$'. In that case φ is a linear inequation and can be written as

$$\sum_{i=1}^{k} a_i x_i + b_i y_i < \sum_{i=1}^{k} c_i x_i + d_i y_i$$

for certain $a_i, b_i, c_i, d_i \in \mathbb{N}_0$ for $i \in [k]$. This can be rewritten as:

$$\underbrace{\sum_{i=1}^{k} (a_i - c_i) x_i}_{l_n(x_1, \ldots, x_k)} < \underbrace{\sum_{i=1}^{k} (d_i - b_i) y_i}_{r_n(y_1, \ldots, y_k)}$$

For $n \in \mathbb{N}$ let l_n, r_n be the functions induced by the left-hand and right-hand expression with signature $l_n, r_n \colon [n^c]_0^k \to \mathbb{R}$. Let E_n be the union of the image of l_n and the image of r_n. Let $E_n = \{e_1, \ldots, e_{z_n}\}$ for some $z_n \in \mathbb{N}$ and $e_i < e_j$ for all $i < j$ with $i, j \in [z_n]$. It holds for all $n \in \mathbb{N}$, $\vec{a}, \vec{b} \in [n^c]_0^k$ and $e_i = l_n(\vec{a}), e_j = r_n(\vec{b})$ that

$$\mathcal{N}, (\vec{a}, \vec{b}) \models \varphi \Leftrightarrow l_n(\vec{a}) < r_n(\vec{b}) \Leftrightarrow e_i < e_j \Leftrightarrow i < j$$

We claim that for the labeling scheme $S' = (\psi, c')$ where $\psi(x_1, x_2, y_1, y_2)$ is $x_1 < y_2$ and $c' \in \mathbb{N}$ is chosen sufficiently large, it holds that $\mathrm{gr}_\infty(S) \subseteq \mathrm{gr}_\infty(S')$. Consider a graph G on n vertices that is in $\mathrm{gr}_\infty(S)$ via a labeling $\ell \colon V(G) \to [n^c]_0^k$. We construct a labeling $\ell' \colon V(G) \to [n^{c'}]_0^2$ which shows that G is in $\mathrm{gr}_\infty(S')$. For $u \in V(G)$ let $\ell'(u) = (i, j)$ with $e_i = l_n(\ell(u))$ and $e_j = r_n(\ell(v))$. For all $u \neq v \in V(G)$ it holds that

$$\begin{aligned} (u, v) \in E(G) &\Leftrightarrow \mathcal{N}, (\ell(u), \ell(v)) \models \varphi \\ &\Leftrightarrow l_n(\ell(u)) < r_n(\ell(v)) \\ &\Leftrightarrow \ell'(u)_1 < \ell'(v)_2 \\ &\Leftrightarrow \mathcal{N}, (\ell'(u), \ell'(v)) \models \psi \end{aligned}$$

□

Fact 3.53. $\mathsf{GFO}_{\mathrm{qf}}(=) = \mathsf{GFO}(=)$.

Proof. Let $\mathcal{C}$ be in $\mathsf{GFO}(=)$ via a labeling scheme $S = (\varphi, c)$ with $2k$ variables, i.e. $\mathcal{C} \subseteq \mathrm{gr}(S)$. Observe that $\mathcal{N}_n, \vec{a} \models \varphi$ iff $\mathcal{N}, \vec{a} \models \varphi$ for all $n > r$ and $\vec{a} \in [n]_0^{2k}$ where r denotes the number of free and quantified variables in φ. This means $\mathcal{C}_{>r} \subseteq \mathrm{gr}_\infty(S)$. Let ψ be a quantifier-free formula in $\mathsf{FO}_{2k}(=)$ such that $\mathcal{N}, \vec{a} \models \varphi$ iff $\mathcal{N}, \vec{a} \models \psi$ for all $\vec{a} \in \mathbb{N}_0^{2k}$. The existence of ψ can be proved by quantifier elimination. It follows that $\mathcal{C}_{>r} \subseteq \mathrm{gr}_\infty(\psi, c)$ and therefore $\mathcal{C}_{>r} \in \mathsf{GFO}_{\mathrm{qf}}(=)$. Since $\mathsf{GFO}_{\mathrm{qf}}(=)$ is closed under union and every finite graph class is in $\mathsf{GFO}_{\mathrm{qf}}(=)$ it follows that $\mathcal{C}$ is in $\mathsf{GFO}_{\mathrm{qf}}(=)$. □

Fact 3.54. $\mathsf{GFO}_{\mathrm{qf}}(<) = \mathsf{GFO}(<)$.

Proof. Observe that $\mathsf{FO}(<)$ has no quantifier-elimination in the sense that there is no quantifier-free formula in $\mathsf{FO}(<)$ which is equivalent to $\exists z\, x < z \wedge z < y$ where x, y are free variables. Instead, we show that ($\star$) for every formula φ in $\mathsf{FO}_k(<)$ there exists a quantifier-free formula ψ in $\mathsf{FO}_k(<, +)$ such that φ and ψ are equivalent w.r.t. $\mathcal{N}_n$ for all $n \in \mathbb{N}$. It immediately follows that $\mathsf{GFO}(<) \subseteq \mathsf{GFO}_{\mathrm{qf}}(<, +)$. Since $\mathsf{GFO}_{\mathrm{qf}}(<) = \mathsf{GFO}_{\mathrm{qf}}(<, +)$ (Theorem 3.52) it holds that $\mathsf{GFO}(<) = \mathsf{GFO}_{\mathrm{qf}}(<)$.

Now, let us argue why ($\star$) holds. For every formula in $\mathsf{FO}(<)$ it can be assumed w.l.o.g. that it contains no negation since $\neg x = y$ is equivalent to $x < y \vee y < x$ and $\neg x < y$ is equivalent to $x = y \vee y < x$. To prove that every formula in $\mathsf{FO}(<)$ has a quantifier-free equivalent in $\mathsf{FO}(<, +)$ it suffices to show that every formula φ of the form $\exists z\, C$ where C is a conjunction of atoms from $\mathsf{FO}(<)$ has a quantifier-free equivalent ψ in $\mathsf{FO}(<, +)$ (see [Smo91, p. 310]). We assume that ψ can use the constants c_0, c_1, c_m which represent 0, 1 and the maximal value in the universe, respectively. If C is unsatisfiable then a quantifier-free equivalent of φ is the negation of a tautology. Therefore we assume that C is satisfiable. The conjunctive clause C can be seen as a directed acyclic graph D_C. The equality atoms in C induce a partition of the variables in C; let the vertex set of D_C be that partition. For two vertices U, V in D_C there is an edge (U, V) if there exist variables $x \in U, y \in V$ such that $x < y$ is a literal in C. Let Z be the vertex of D_C which contains the quantified variable z. Assume that Z contains another variable $x \neq z$. In that case a quantifier-free equivalent ψ of φ can be obtained by renaming every occurrence of z in C to x and removing the quantifier. If Z contains only z we can proceed as follows. We assume that z occurs in at least one literal of C since otherwise it could be trivially removed. This implies that Z is not an isolated vertex in D_C. If Z has in-degree zero then z can be replaced by the constant c_0. Similarly, if Z has out-degree zero then z can be replaced by the constant c_m. If Z has neither in-degree nor out-degree zero then ψ can be constructed from φ as follows. For all in-neighbors X of Z, out-neighbors Y of Z and variables $x \in X, y \in Y$ append '$\wedge\, x + c_1 < y \wedge x \neq c_m$' to ψ. Then remove the quantifier and every atom containing z from ψ. The atom $x + c_1 < y$ ensures that the difference between x and y is at least two, which was previously expressed by saying that there exists a value z between x and y. A problem occurs when $x + c_1$ evaluates to zero because x is assigned the maximal value of the universe due to the overflow condition. To prevent this we add the atom $x \neq c_m$. More formally, it can be checked that $\mathcal{N}_n, \vec{a} \models \varphi$ iff $\mathcal{N}_n, \vec{a} \models \psi$ for all $n \in \mathbb{N}$. □

We remark that quantifier-free labeling schemes in $\mathsf{GFO}(<)$ are solely determined by their formula in the following sense. Given such a formula φ with $2k$ variables it holds that $\mathrm{gr}(\varphi, k) = \cup_{i \in \mathbb{N}} \mathrm{gr}(\varphi, i)$ ([Cha16b, Lem. 20]).

3.4.2 Complete Graph Classes

Corollary 3.55. *Let $\sigma = \emptyset$, or $\sigma \subseteq \{<, +, \times\}$ and '$<$' is in σ. A graph class $\mathcal{D}$ is $\leq_{\mathrm{BF}}$-complete for $\mathsf{GFO}_{\mathrm{qf}}(\sigma)$ iff $\mathcal{D}$ is in $\mathsf{GFO}_{\mathrm{qf}}(\sigma)$ and $gr_\infty(S) \leq_{\mathrm{BF}} \mathcal{D}$ holds for all atomic labeling schemes S in $\mathsf{GFO}_{\mathrm{qf}}(\sigma)$.*

Proof. "$\Rightarrow$": If $\mathcal{D}$ is $\leq_{\mathrm{BF}}$-complete for $\mathsf{GFO}_{\mathrm{qf}}(\sigma)$ then every graph class in $\mathsf{GFO}_{\mathrm{qf}}(\sigma)$ is $\leq_{\mathrm{BF}}$-reducible to $\mathcal{D}$. From Lemma 3.48 it follows that $\mathrm{gr}_\infty(S)$ is in $\mathsf{GFO}_{\mathrm{qf}}(\sigma)$ for every atomic labeling scheme S in $\mathsf{GFO}_{\mathrm{qf}}(\sigma)$.

"$\Leftarrow$": Suppose that $\mathrm{gr}_\infty(S) \leq_{\mathrm{BF}} \mathcal{D}$ holds for all atomic labeling schemes S in $\mathsf{GFO}_{\mathrm{qf}}(\sigma)$. From Theorem 3.51 it follows that if a graph class $\mathcal{C}$ is in $\mathsf{GFO}_{\mathrm{qf}}(\sigma)$ then there exist atomic labeling schemes $S_1, \dots, S_a$ and an a-ary Boolean function f in $\mathsf{GFO}_{\mathrm{qf}}(\sigma)$ such that $\mathcal{C} \subseteq f(\mathrm{gr}_\infty(S_1), \dots, \mathrm{gr}_\infty(S_a))$. There exist Boolean functions $g_1, \dots, g_a$ such that $\mathrm{gr}_\infty(S_i) \subseteq g_i(\mathcal{D}, \dots, \mathcal{D})$ for all $i \in [a]$. Therefore $\mathcal{C} \subseteq f(g_1(\mathcal{D}, \dots, \mathcal{D}), \dots, g_a(\mathcal{D}, \dots, \mathcal{D}))$ and thus $\mathcal{C} \leq_{\mathrm{BF}} \mathcal{D}$. □

Definition 3.56. *A directed graph G is dichotomic if for all $u, v \in V(G)$ and $\alpha \in \{\mathrm{in}, \mathrm{out}\}$ it holds that $N_\alpha(u) \cap N_\alpha(v) = \emptyset$ or $N_\alpha(u) = N_\alpha(v)$.*

Observe that every directed forest is dichotomic. Every vertex in a forest has in-degree at most one and therefore $N_{\mathrm{in}}(u) = N_{\mathrm{in}}(v)$ or $N_{\mathrm{in}}(u) \cap N_{\mathrm{in}}(v) = \emptyset$ for all $u \neq v \in V(G)$. Additionally, the out-neighborhoods of every distinct pair of vertices are disjoint because every node has a unique parent.

Lemma 3.57. *There exists an atomic labeling scheme S in $\mathsf{GFO}(=)$ such that $gr(S)$ is exactly the class of dichotomic graphs.*

Proof. Let $S = (\varphi, 1)$ with $\varphi(x_1, x_2, y_1, y_2) \triangleq x_1 = y_2$.

First, we argue that every dichotomic graph is in $\mathrm{gr}(S)$. Given a dichotomic graph G with n vertices. Let $\sim$ be the equivalence relation on $V(G)$ such that $u \sim v$ if u and v have identical out-neighborhoods. Let $V_1, \dots, V_k$ be the equivalence classes of $\sim$. We write V'_i to denote the out-neighbors of the vertices in V_i for $i \in [k]$. Let V'_0 be the set of vertices which have in-degree zero. It holds that $V'_0, V'_1, \dots, V'_k$ is a partition of $V(G)$ (with possibly some empty sets) since G is dichotomic. The following labeling $\ell\colon V(G) \to [n]_0^2$ shows that G is in $\mathrm{gr}(S)$. For $u \in V(G)$ let $\ell(u) = (u_1, u_2)$ with $u_1, u_2 \in [k]_0$ such that $u \in V_{u_1}$ and $u \in V'_{u_2}$. Since $k \leq n$ this is a valid labeling.

For the other direction let G be a graph that is in $\mathrm{gr}(S)$ via the labeling $\ell\colon V(G) \to [n]_0^2$. Consider two vertices u, v of G. Let $\ell(u) = (u_1, u_2)$ and $\ell(v) = (v_1, v_2)$. If $u_1 = v_1$ then they have identical out-neighborhoods. If $u_1 \neq v_1$ then they have disjoint out-neighborhoods. The same applies to the in-neighborhoods and u_2, v_2. Therefore G is dichotomic. □

Theorem 3.58. *Dichotomic graphs are $\leq_{\mathrm{BF}}$-complete for $\mathsf{GFO}(=)$.*

Proof. From the previous lemma it follows that dichotomic graphs are in $\mathsf{GFO}(=)$. For the hardness we have to argue that for every atomic labeling scheme S in $\mathsf{GFO}(=)$ it holds that $\mathrm{gr}_\infty(S)$ is $\leq_{\mathrm{BF}}$-reducible to dichotomic graphs (see Corollary 3.55). Let $S = (\varphi, c)$ be an atomic labeling scheme in $\mathsf{GFO}(=)$ with $2k$ variables $x_1, \dots, x_k, y_1, \dots, y_k$ for some $c, k \in \mathbb{N}$. The formula φ must be one of the following:

1. $x_a = x_b$ for some $a, b \in [k]$
2. $y_a = y_b$ for some $a, b \in [k]$
3. $x_a = y_b$ for some $a, b \in [k]$

It is simple to check that every graph in $\mathrm{gr}(S)$ is dichotomic for the first two cases. It remains to deal with the third case. Given a graph G that is in $\mathrm{gr}(S)$ via a labeling $\ell\colon V(G) \to [n^c]_0^k$. We construct a labeling $\ell'\colon V(G) \to [n]_0^2$ such that $(u, v) \in E(G)$ iff $\ell'(u)_1 = \ell'(v)_2$ for all $u \neq v \in V(G)$. A graph is dichotomic iff it has such a labeling ℓ' (see the proof of Lemma 3.57). Let $V(G) = \{v_1, \dots, v_n\}$ and $\ell(v_i) = (v_i^1, \dots, v_i^k)$ for $i \in [n]$. Observe that only the a-th and b-th component of the labeling ℓ are relevant because the other components are never considered. For a set $Z \subseteq \mathbb{N}$ and $z \in Z$ let $\mathrm{ord}(z, Z)$ denote the number of elements smaller than z in Z plus one, e.g. $\mathrm{ord}(0, \{0, 3, 4\}) = 1$. Let $A = \{v_1^a, \dots, v_n^a\}$. Given $i \in [n]$ we define $\ell'(v_i)$ as $(\mathrm{ord}(v_i^a, A), v'_i)$ with $v'_i = 0$ if v_i^b is not in A and $v'_i = \mathrm{ord}(v^b_i, A)$ otherwise. Correctness follows from the fact that $v^a_i = v^b_j$ iff $\ell'(v_i)_1 = \ell'(v_j)_2$ for all $i, j \in [n]$ and that only numbers between 0 and n are used. □

Corollary 3.59. *Dichotomic graphs are $\leq_{\mathrm{sg}}$-complete for* $\mathsf{GFO}(=)$.

Proof. Lemma 3.39 states that for every self-universal and inflatable graph class $\mathcal{D}$ it holds that $\mathcal{C} \leq_{\mathrm{BF}} \mathcal{D}$ implies $\mathcal{C} \leq_{\mathrm{sg}} \mathcal{D}$. Since dichotomic graphs are closed under disjoint union and the graph with a single vertex is dichotomic it follows that this class is self-universal and inflatable. As a consequence dichotomic graphs are $\leq_{\mathrm{sg}}$-complete for $\mathsf{GFO}(=)$. □

Theorem 3.60. *Path graphs are $\leq_{\mathrm{sg}}$-complete for* $\mathsf{GFO}(=)$.

Proof. Dichotomic graphs are $\leq_{\mathrm{sg}}$-reducible to path graphs via $c = 3, k = 4$ and $f(A) = a_{1,4} \wedge a_{2,3}$ for $A = (a_{i,j})_{i,j \in [4]}$. For $n \in \mathbb{N}$ let P_n be the undirected path graph with n vertices. We assume that P_n has vertex set $\{0, \dots, n-1\}$ and two vertices are adjacent if their absolute difference is one. We need to show that every dichotomic graph G on n vertices has a (P_{n^3}, f)-representation via some labeling $\ell\colon V(G) \to [n^3 - 1]^4_0$. Since G is dichotomic there exists a labeling $\ell'\colon V(G) \to [n]^2_0$ such that $(u, v) \in E(G)$ iff $\ell'(u)_1 = \ell'(v)_2$ for all $u \neq v \in V(G)$ (see proof of Lemma 3.57). For $u \in V(G)$ let $\ell'(u) = (u_1, u_2)$; we define $\ell(u)$ as $(2u_1, 2u_1 + 1, 2u_2, 2u_2 + 1)$. Notice that the maximal value in ℓ is $2n + 1$ which is smaller than $n^3 - 1$ for all $n \geq 2$. For two vertices $u \neq v \in V(G)$ with $\ell'(u) = (u_1, u_2), \ell'(v) = (v_1, v_2)$ it holds that $(u, v) \in E(G)$ iff $u_1 = v_2$ iff $f(A^{\ell}_{uv}) = 1$. Therefore G has a (P_{n^3}, f)-representation via ℓ. □

Definition 3.61. *A directed graph G is a linear neighborhood graph if for all $u \neq v \in V(G)$ and $\alpha \in \{\mathrm{in}, \mathrm{out}\}$ it holds that $N_\alpha(u) \subseteq N_\alpha(v)$ or $N_\alpha(v) \subseteq N_\alpha(u)$.*

Lemma 3.62. *There exists an atomic labeling scheme S in* $\mathsf{GFO}(<)$ *such that $gr(S)$ is exactly the class of linear neighborhood graphs.*

Proof. Let $S = (\varphi, 1)$ with $\varphi(x_1, x_2, y_1, y_2) \triangleq x_1 < y_2$.

First, we show that every linear neighborhood graph is in $\mathrm{gr}(S)$. Given a linear neighborhood graph G with n vertices. Let $\sim$ be the equivalence relation on $V(G)$ such that $u \sim v$ if u and v have identical in-neighborhoods. Let V_0 be the set of vertices with in-degree zero. Let $V_1, \dots, V_k$ be the equivalence classes of $\sim$ except V_0 such that $N_{\mathrm{in}}(V_i) \subseteq N_{\mathrm{in}}(V_j)$ for all $1 \leq i < j \leq k$. Observe that $V_0, \dots, V_k$ is a partition of $V(G)$. The following labeling $\ell\colon V(G) \to [n]^2_0$ shows that G is in $\mathrm{gr}(S)$. For $u \in V(G)$ let $\ell(u) = (u_1, u_2)$ with $u \in V_{u_2}$ and u_1 is the minimal value such that $u \in N_{\mathrm{in}}(V_{u_1+1})$ ($u_1 = k$ if this minimum does not exist) for $u_1, u_2 \in \{0, \dots, k\}$. To see that this is correct let us consider an edge $(u, v) \in E(G)$ and $\ell(u) = (u_1, u_2), \ell(v) = (v_1, v_2)$. It holds that $u \in N_{\mathrm{in}}(v) = N_{\mathrm{in}}(V_{v_2})$. Since $u \in N_{\mathrm{in}}(V_{v_2})$ it follows that $u_1 + 1 \leq v_2$ and thus $u_1 < v_2$. Next, consider a non-edge $(u, v) \notin E(G)$. It holds that $u \notin N_{\mathrm{in}}(v) = N_{\mathrm{in}}(V_{v_2})$. Therefore $u_1 + 1 \geq v_2$ and thus $u_1 \not< v_2$.

For the other direction let G be a graph that is in $\mathrm{gr}(S)$ via a labeling $\ell\colon V(G) \to [n]^2_0$. We argue that G is a linear neighborhood graph. Given two vertices $u \neq v \in V(G)$ and $\ell(u) = (u_1, u_2), \ell(v) = (v_1, v_2)$. If $u_1 \leq v_1$ then $N_{\mathrm{out}}(v) \subseteq N_{\mathrm{out}}(u)$. If $u_1 \geq v_1$ then $N_{\mathrm{out}}(u) \subseteq N_{\mathrm{out}}(v)$. The same holds for u_2, v_2 and the in-neighborhoods of u and v. Therefore G is a linear neighborhood graph. □

Theorem 3.63. *Linear neighborhood graphs are $\leq_{\mathrm{BF}}$-complete for* $\mathsf{GFO}(<)$.

Proof. Membership follows from the previous lemma. For the hardness we have to show for every atomic labeling scheme $S = (\varphi, c)$ in $\mathsf{GFO}(<)$ that $\mathrm{gr}(S)$ is $\leq_{\mathrm{BF}}$-reducible to linear neighborhood graphs. If φ uses equality then it can be rewritten using order because $x = y$ iff $\neg(x < y \vee y < x)$. Therefore we assume that φ uses order. Let φ have variables x_1, x_2, y_1, y_2. Using more than two variables per vertex is useless as we have seen in the proof of Theorem 3.58. If φ is $x_i < x_j$ (or $y_i < y_j$) for $i, j \in [2]$ then it is trivial to see that $\mathrm{gr}(S)$ is a subset of linear neighborhood graphs. We assume w.l.o.g. that φ is $x_1 < y_2$. We show that $\mathrm{gr}(\varphi, c) \subseteq \mathrm{gr}(\varphi, 1)$ for all $c \in \mathbb{N}$. Since $\mathrm{gr}(\varphi, 1)$ are linear neighborhood graphs this concludes the hardness. Let G be a graph with n vertices that is in $\mathrm{gr}(\varphi, c)$ via a labeling $\ell \colon V(G) \to [n^c]_0^2$. We argue that there is a labeling $\ell' \colon V(G) \to [n]_0^2$, which shows that G is in $\mathrm{gr}(\varphi, 1)$. Let $V(G) = \{v_1, \dots, v_n\}$ and $\ell(v_i) = (v_i^1, v_i^2)$ for $i \in [n]$. Observe that only the relative order of the labels is relevant. Therefore the labels $v_1^2, \dots, v_n^2$ can be mapped to new labels $\bar{v}_1^2, \dots, \bar{v}_n^2 \subseteq \{1, \dots, n\}$ such that order is preserved, i.e. $v_i^2 < v_j^2$ iff $\bar{v}_i^2 < \bar{v}_j^2$ for all $i, j \in [n]$. Similarly, the labels $v_1^1, \dots, v_n^1$ can be mapped to new labels $\bar{v}_1^1, \dots, \bar{v}_n^1 \subseteq \{0, 1, \dots, n\}$ such that $v_i^1 < v_j^2$ iff $\bar{v}_i^1 < \bar{v}_j^2$ for all $i, j \in [n]$. □

Corollary 3.64. *Linear neighborhood graphs are $\leq_{\mathrm{sg}}$-complete for* $\mathsf{GFO}(<)$.

Proof. Same argument as in the proof of Corollary 3.59. □

Theorem 3.65. *The transitive closure of directed paths is $\leq_{\mathrm{sg}}$-complete for* $\mathsf{GFO}(<)$.

Proof. Let D_n denote the transitive closure of the directed path on n vertices. Let us assume that D_n has $\{0, \dots, n-1\}$ as vertex set and $(u, v) \in E(D_n)$ if $u < v$. It is clear from the definition that this graph class is in $\mathsf{GFO}(<)$.

We show that linear neighborhood graphs are $\leq_{\mathrm{sg}}$-reducible to this class via $c = 2, k = 2$ and $f(A) = a_{1,2}$ for $A = (a_{i,j})_{i,j \in [2]}$. Let G be a linear neighborhood graph with n vertices. Then there exists a labeling $\ell \colon V(G) \to [n]_0^2$ such that $(u, v) \in E(G)$ iff $\ell(u)_1 < \ell(v)_2$. It holds that G has a (D_{n^2}, f)-representation via the same labeling ℓ. □

Theorem 3.66. *Interval graphs are $\leq_{\mathrm{sg}}$-complete for* $\mathsf{GFO}(<)$.

Proof. The transitive closure of directed paths is $\leq_{\mathrm{sg}}$-reducible to interval graphs via $c = 2$, $k = 2$, $f(A) = a_{2,1} \wedge \neg a_{1,2}$ for $A = (a_{i,j})_{i,j \in [2]}$. We show that for all $n \in \mathbb{N}$ there exists an interval graph H on n^2 vertices such that D_n has an (H, f)-representation. Let $\mathcal{I}$ denote the set of intervals on the real line. The following function $\ell \colon V(D_n) \to \mathcal{I}^2$ is a labeling for D_n. For $u \in V(D_n) = [n-1]_0$ let $\ell(u) = ([0, u], [u, u])$. The image of ℓ defines an interval graph H' with $2n \leq n^2$ vertices. Let H be an interval graph with n^2 vertices which contains H' as induced subgraph. For two vertices $u \neq v \in V(D_n)$ it holds that

$$(u, v) \in E(D_n) \Leftrightarrow u < v \Leftrightarrow [u, u] \cap [0, v] \neq \emptyset \wedge [0, u] \cap [v, v] = \emptyset \Leftrightarrow f(A_{uv}^{\ell}) = 1$$

and therefore D_n has an (H, f)-representation via ℓ. □

3.4.3 Polynomial-Boolean Systems

In the beginning, we defined a labeling scheme independently of a model of computation. The label decoder was just a binary relation over words. In the case of logical labeling schemes we neglected this separation by identifying label decoders with logical formulas. It would have been more hygienic to say that a logical formula φ with $2k$ variables computes (or represents) a label decoder $F_\varphi \subseteq \mathbb{N}_0^{2k}$. However, a subtle difference between logical labeling schemes and classical ones is that the label length c in a logical labeling scheme also influences how the formula is interpreted whereas in classical labeling schemes the label length does not affect how a Turing machine which computes the label decoder is executed. In the case of quantifier-free logical labeling schemes this dependence can be removed as we have shown in Lemma 3.48. In this section we consider a generalization of $\mathsf{GFO}_{\mathrm{qf}}$ where the restriction on the label length is dropped. Observe that a quantifier-free logical labeling scheme can be seen as a Boolean combination of polynomial inequations. We formalize this by what we call polynomial-Boolean systems. We consider a polynomial to be an expression over a set of variables that only involves addition and multiplication. We also consider the constant zero to be a polynomial.

Definition 3.67. *A polynomial-Boolean system (PBS) with k variables is a tuple $R = ((p_1, \dots, p_l), f)$ where $p_1, \dots, p_l$ are polynomials with k variables and f is an l^2-ary Boolean function and $k, l \in \mathbb{N}$. Given $\mathbb{X} \in \{\mathbb{N}_0, \mathbb{Q}, \mathbb{R}\}$ the PBS R induces a k-ary relation $F_R^{\mathbb{X}}$ over $\mathbb{X}$ which is defined as*

$$(a_1, \dots, a_k) \in F_R^{\mathbb{X}} \Leftrightarrow f(x_{1,1}, \dots, x_{l,l}) = 1 \text{ with } x_{i,j} = [\![p_i(a_1, \dots, a_k) < p_j(a_1, \dots, a_k)]\!]$$

for all $a_1, \dots, a_k \in \mathbb{X}$ and $i, j \in [l]$.

Definition 3.68. *Let $\mathbb{X} \in \{\mathbb{N}_0, \mathbb{Q}, \mathbb{R}\}$. For $k \in \mathbb{N}$ and a relation $F \subseteq \mathbb{X}^{2k}$ let $gr(F)$ be the following set of graphs. A graph G is in $gr(F)$ if there exists a labeling $\ell \colon V(G) \to \mathbb{X}^k$ such that $(u, v) \in E(G) \Leftrightarrow (\ell(u), \ell(v)) \in F$ for all $u \neq v \in V(G)$. A graph class $\mathcal{C}$ is in* $\mathsf{PBS}(\mathbb{X})$ *if there exists a PBS R such that $\mathcal{C} \subseteq gr(F_R^{\mathbb{X}})$.*

Spinrad briefly mentions what appears to be a non-uniform variant of $\mathsf{PBS}(\mathbb{R})$, which he calls Warren representable [Spi03, p. 55].

Fact 3.69. *The class of kd-line segment graphs, k-ball graphs and k-dot product graphs are in* $\mathsf{PBS}(\mathbb{Q})$ *for all $k \in \mathbb{N}$.*

Proof. It is intuitively clear from the definitions of these graph classes that they lie in $\mathsf{PBS}(\mathbb{R})$. For example, in a line segment graph each vertex can be assigned four real numbers which represent the two endpoints of the line segment of that vertex. It remains to verify that a Boolean combination of the results of polynomial inequations suffices to determine whether two line segments intersect. To see that these graph classes are in $\mathsf{PBS}(\mathbb{Q})$ we make the following observation. If a graph class $\mathcal{C}$ is in $\mathsf{PBS}(\mathbb{R})$ via a PBS R and for every graph G in $\mathcal{C}$ there exists a rational labeling ℓ of G that shows that G is in $\mathrm{gr}(F_R^{\mathbb{R}})$ then $\mathcal{C}$ is in $\mathsf{PBS}(\mathbb{Q})$ via R. For k-dot product graphs it is shown in [Fid+98, Proposition 3] that rational labelings suffice. For kd-line segment graphs and k-ball graphs a perturbation argument shows that rational coordinates suffice as well. □

For $k \geq 2$ it is unknown whether the graph classes mentioned in the previous fact even have a labeling scheme.

Lemma 3.70. *Let $\mathbb{X} \in \{\mathbb{N}_0, \mathbb{Q}, \mathbb{R}\}$.* $\mathsf{PBS}(\mathbb{X})$ *is closed under $\leq_{\mathrm{BF}}$ and $\leq_{\mathrm{sg}}$.*

Proof. For $\leq_{\mathrm{BF}}$ it suffices to check that $\mathsf{PBS}(\mathbb{X})$ is closed under susbets, negation and conjunction and then apply Corollary 3.26. For $\leq_{\mathrm{sg}}$ consider the following argument. Let $\mathcal{C} \leq_{\mathrm{sg}} \mathcal{D}$ via $c, k \in \mathbb{N}$ and a k^2-ary Boolean function f. Let $\mathcal{D} \in \mathsf{PBS}(\mathbb{X})$ via a PBS $R = ((p_1, \dots, p_l), g)$ with $2m$ variables and $l, m \in \mathbb{N}$. To avoid technical clutter we just outline what a PBS R' must look like such that $\mathcal{C} \subseteq \mathrm{gr}(F_{R'}^{\mathbb{X}})$ (and thus $\mathcal{C} \in \mathsf{PBS}(\mathbb{X})$). For a graph $G \in \mathcal{C}$ with n vertices there exists a graph $H \in \mathcal{D}$ with n^c vertices such that G has an (H, f)-representation via a labeling $\ell_G \colon V(G) \to V(H)^k$. Since $\mathcal{D} \in \mathsf{PBS}(\mathbb{X})$ it follows that H is in $\mathrm{gr}(F_R^{\mathbb{X}})$ via a labeling $\ell_H \colon V(H) \to \mathbb{X}^m$. By combining ℓ_G and ℓ_H we get a labeling $\ell \colon V(G) \to \mathbb{X}^{km}$. Intuitively, the labeling ℓ provides us with all the information required to determine adjacency in G. More specifically, one can construct a PBS R' with $2km$ variables from R and f such that $G \in \mathrm{gr}(F_{R'}^{\mathbb{X}})$ via ℓ. □

Theorem 3.71. $\mathsf{PBS}(\mathbb{N}_0) = \mathsf{PBS}(\mathbb{Q})$.

Proof. It is clear that $\mathsf{PBS}(\mathbb{N}_0) \subseteq \mathsf{PBS}(\mathbb{Q})$. Let $\mathbb{Q}_+ = \{n \in \mathbb{Q} \mid n \geq 0\}$. For the other direction we show that $\mathsf{PBS}(\mathbb{Q}) \subseteq \mathsf{PBS}(\mathbb{Q}_+)$ and $\mathsf{PBS}(\mathbb{Q}_+) \subseteq \mathsf{PBS}(\mathbb{N}_0)$.

Let $\mathcal{C} \in \mathsf{PBS}(\mathbb{Q})$ via a PBS $R = ((p_1, \ldots, p_l), f)$ with $2k$ variables. We outline a PBS R' such that $\mathcal{C}$ is in $\mathsf{PBS}(\mathbb{Q}_+)$ via R'. This construction relies on the following observation. Given $a \in \mathbb{Q}$ let $|a|$ denote its absolute value and $\mathrm{sign}(a)$ equals -1 if a is negative and 1 otherwise. For $n \in \mathbb{N}$ and a vector $\vec{a} \in \mathbb{Q}^n$ let $|\vec{a}| = (|a_1|, \ldots, |a_n|)$ and $\mathrm{sign}(\vec{a}) = (\mathrm{sign}(a_1), \ldots, \mathrm{sign}(a_n))$. For all polynomial functions $p, q \colon \mathbb{Q}^n \to \mathbb{Q}$ and sign patterns $\vec{s} \in \{-1, 1\}^n$ there exist polynomial functions $p', q' \colon \mathbb{Q}_+^n \to \mathbb{Q}_+$ such that for all $\vec{a} \in \mathbb{Q}^n$ with $\mathrm{sign}(\vec{a}) = \vec{s}$ it holds that $p(\vec{a}) < q(\vec{a})$ iff $p'(|\vec{a}|) < q'(|\vec{a}|)$. For example, consider the polynomials $p(x, y, z) = x^2y^3z + y$ and $q(x, y, z) = z$ and the sign pattern $(-1, 1, -1)$ for (x, y, z). If we only consider inputs with this sign pattern then it holds that $p(x, y, z) < q(x, y, z)$ iff $\underbrace{|y| + |z|}_{p'} < \underbrace{|x|^2|y|^3|z|}_{q'}$. For each variable in R we have two variables in R'. The first one is used to store the absolute value of the original variable and the second one encodes the sign. Let G be a graph that is in $\mathrm{gr}(F_R^{\mathbb{Q}})$ via a labeling $\ell \colon V(G) \to \mathbb{Q}^k$. Then we derive a labeling $\ell' \colon V(G) \to \mathbb{Q}_+^{2k}$ from ℓ as follows. Given $u \in V(G)$ let $\ell(u) = (u_1, \ldots, u_k)$. We set $\ell'(u) = (|u_1|, u'_1, \ldots, |u_k|, u'_k)$ where $u'_i = |u_i|$ if u_i is negative and any other non-negative value if u_i is positive. This allows us to infer the sign pattern and absolute values of the original labeling ℓ from ℓ'. The PBS R' is constructed such that $G \in \mathrm{gr}(F_{R'}^{\mathbb{Q}_+})$ via ℓ'. Suppose we are given two vertices $u \neq v \in V(G)$. Then the adjacency of u and v depends on the results of $p_i(\ell(u), \ell(v)) < p_j(\ell(u), \ell(v))$ for $i, j \in [l]$. We emulate the inequation $p_i(\ell(u), \ell(v)) < p_j(\ell(u), \ell(v))$ in R' by $p'(|\ell(u)|, |\ell(v)|) < q'(|\ell(u)|, |\ell(v)|)$ where p' and q' depend on p_i, p_j and the sign pattern of $\ell(u), \ell(v)$. Stated differently, for every pair $i, j \in [l]$ and every sign pattern $s \in \{-1, 1\}^{2k}$ there is a pair of polynomials in R' and additionally R' has the $2k$ identity polynomials to decode the signs.

To see that $\mathsf{PBS}(\mathbb{Q}_+) \subseteq \mathsf{PBS}(\mathbb{N}_0)$ it suffices to make the following observation. Given two polynomial functions $p, q \colon \mathbb{Q}_+^k \to \mathbb{Q}_+$ there exist two polynomial functions $p', q' \colon \mathbb{N}_0^{2k} \to \mathbb{N}_0$ such that for all $\vec{a} = (\frac{a_1}{b_1}, \ldots, \frac{a_k}{b_k}) \in \mathbb{Q}_+^k$ it holds that $p(\vec{a}) < q(\vec{a})$ iff $p'(a_1, b_1, \ldots, a_k, b_k) < q'(a_1, b_1, \ldots, a_k, b_k)$. The functions p' and q' can be obtained from the inequation $p < q$ by multiplication with the denominators. Using this observation a PBS R with $2k$ variables can be translated into a PBS R' with $4k$ variables such that $\mathrm{gr}(F_R^{\mathbb{Q}_+}) \subseteq \mathrm{gr}(F_{R'}^{\mathbb{N}_0})$. □

Theorem 3.72. $\mathsf{PBS}(\mathbb{R}) \subseteq [\text{Small} \cap \text{Hereditary}]_{\subseteq}$.

Proof. Let $R = ((p_1, \ldots, p_l), f)$ be a PBS with $2k$ variables. We show that $\mathrm{gr}(F_R^{\mathbb{R}})$ is small and hereditary. From that it follows that $\mathsf{PBS}(\mathbb{R})$ is a subset of $[\text{Small} \cap \text{Hereditary}]_{\subseteq}$. Let G be a graph that is in $\mathrm{gr}(F_R^{\mathbb{R}})$ via a labeling $\ell \colon V(G) \to \mathbb{R}^k$. An induced subgraph of G on vertex set $V' \subseteq V(G)$ is in $\mathrm{gr}(F_R^{\mathbb{R}})$ via the labeling ℓ restricted to V'. Thus $\mathrm{gr}(F_R^{\mathbb{R}})$ is hereditary.

It remains to argue that $\mathrm{gr}(F_R^{\mathbb{R}})$ is small. We do so by applying Warren's theorem [Spi03, p. 55], which can be stated as follows. Let $\mathcal{E} = (E_1, \dots, E_m)$ be a sequence of polynomial inequations over variables $x_1, \dots, x_n$. More specifically, the inequations are assumed to be of the form $p(x_1, \dots, x_n) < q(x_1, \dots, x_n)$ where p, q are polynomials. Also, let d denote the maximum degree that occurs in any of these inequations. The sequence $\mathcal{E}$ can be understood as a function from $\mathbb{R}^n$ to $\{0,1\}^m$ in the following sense. Given $\vec{a} \in \mathbb{R}^n$ let $\mathcal{E}(\vec{a}) = (e_1, \dots, e_m)$ with $e_i = 1$ iff the inequation $E_i(\vec{a})$ holds. An element of the image of $\mathcal{E}$ is called a sign pattern. Warren's theorem states that the cardinality of the image of $\mathcal{E}$ (or equivalently, the number of sign patterns of $\mathcal{E}$) is at most $\left(\frac{cdm}{n}\right)^n$ where c is some constant.

We show that the number of graphs on n vertices in $\mathrm{gr}(F_R^{\mathbb{R}})$ is bounded by the number of sign patterns of a certain sequence of equations $\mathcal{E}_{R,n}$. Consider a graph G on n vertices that is in $\mathrm{gr}(F_R^{\mathbb{R}})$ via a labeling $\ell \colon V(G) \to \mathbb{R}^k$. The presence of the edge (u,v) in G is determined by the result of l^2 polynomial inequations. Therefore G is determined by the result of a sequence of l^2n^2 polynomial inequations. These inequations use kn variables x_u^i with $u \in V(G)$ and $i \in [k]$. Let d denote the maximum degree over the polynomials $p_1, \dots, p_l$. This means $\mathcal{E}_{R,n}$ has kn variables, l^2n^2 equations and maximum degree d. Thus a graph on n vertices in $\mathrm{gr}(F_R^{\mathbb{R}})$ is determined by a sign pattern of $\mathcal{E}_{R,n}$. As a consequence there are at most $\left(\frac{cdl^2n^2}{kn}\right)^{kn} \in n^{\mathcal{O}(n)}$ graphs on n vertices in $\mathrm{gr}(F_R^{\mathbb{R}})$ (c, d, k, l are constants). □

Definition 3.73. *Let $c, k \in \mathbb{N}$ and $F \subseteq \mathbb{N}_0^{2k}$. We say a graph G is in $gr(F, c)$ if there exists a labeling $\ell \colon V(G) \to [n^c]_0^k$ such that $(u,v) \in E(G) \Leftrightarrow (\ell(u), \ell(v)) \in F$ for all $u \neq v \in V(G)$. We say a graph class $\mathcal{C}$ is in* $\mathsf{BoundedPBS}(\mathbb{N}_0)$ *if there exists a PBS R and $c \in \mathbb{N}$ such that $\mathcal{C} \subseteq gr(F_R^{\mathbb{N}_0}, c)$.*

Definition 3.74. *For $m \in \mathbb{N}$ let $\mathbb{Q}_m = \left\{ \frac{sa}{b} \mid a, b \in [m], s \in \{-1, 0, 1\} \right\}$. Let $F \subseteq \mathbb{Q}^{2k}$ and $c, k \in \mathbb{N}$. We say a graph G is in $gr(F, c)$ if there exists a labeling $\ell \colon V(G) \to (\mathbb{Q}_{n^c})^k$ such that $(u,v) \in E(G) \Leftrightarrow (\ell(u), \ell(v)) \in F$ for all $u \neq v \in V(G)$. A graph class $\mathcal{C}$ is in* $\mathsf{BoundedPBS}(\mathbb{Q})$ *if there exists a PBS R and $c \in \mathbb{N}$ such that $\mathcal{C} \subseteq gr(F_R^{\mathbb{Q}}, c)$.*

Theorem 3.75. $\mathsf{BoundedPBS}(\mathbb{N}_0) = \mathsf{BoundedPBS}(\mathbb{Q})$.

Proof. Let $\mathcal{C}$ be in $\mathsf{BoundedPBS}(\mathbb{Q})$ via a PBS R and $c \in \mathbb{N}$. This means $\mathcal{C} \subseteq \mathrm{gr}(F_R^{\mathbb{Q}}, c)$. First, we construct a PBS R' such that $\mathrm{gr}(F_R^{\mathbb{Q}}, c) \subseteq \mathrm{gr}(F_{R'}^{\mathbb{Q}_+}, c)$. This is the same construction that is described in the proof of Theorem 3.71. One has to additionally check that the restriction on the labeling is not violated. More precisely, if a graph G on n vertices is in $\mathrm{gr}(F_R^{\mathbb{Q}}, c)$ via a labeling $\ell \colon V(G) \to \mathbb{Q}_{n^c}$ then the labeling ℓ' which is derived from ℓ to show that G is in $\mathrm{gr}(F_{R'}^{\mathbb{Q}_+}, c)$ must have the codomain $(\mathbb{Q}_+)_{n^c} = \left\{ \frac{a}{b} \mid a, b \in [n^c] \right\} \cup \{0\}$ (analogously). In the second step we construct a PBS R'' such that $\mathrm{gr}(F_{R'}^{\mathbb{Q}_+}, c) \subseteq \mathrm{gr}(F_{R''}^{\mathbb{N}_0}, c)$. The same construction as in the proof of Theorem 3.71 can be applied. The restriction on the labeling is not violated by this construction either. □

Theorem 3.76. $\mathsf{GFO_{qf}} = \mathsf{BoundedPBS}(\mathbb{N}_0)$.

Proof. "$\subseteq$": Let $\mathcal{C}$ be a graph class in $\mathsf{GFO_{qf}}$. From Lemma 3.48 it follows that there exists a logical labeling scheme $S = (\varphi, c)$ in $\mathsf{GFO_{qf}}$ such that $\mathcal{C} \subseteq \mathrm{gr}_\infty(S)$. This means the interpretation of each term in φ is identical to a polynomial function over $\mathbb{N}_0$. Therefore S directly translates to a PBS R over $\mathbb{N}_0$ where the polynomial functions are given by the terms of φ and the Boolean function of R is given by the Boolean function underlying φ. It is easy to check that $\mathrm{gr}_\infty(S) \subseteq \mathrm{gr}(F_R, c)$ and thus $\mathcal{C} \in \mathsf{BoundedPBS}(\mathbb{N}_0)$.

"$\supseteq$": Let $\mathcal{C}$ be a graph class that is in $\mathsf{BoundedPBS}(\mathbb{N}_0)$ via a PBS R with $2k$ variables and $c, k \in \mathbb{N}$. The PBS R can be translated into a quantifier-free formula φ with $2k$ variables in a straightforward fashion. For a sufficiently large $c' \in \mathbb{N}$ it holds that $\mathrm{gr}(F_R^{\mathbb{N}_0}, c)$ is a subset of $\mathrm{gr}(\varphi, c')$ and thus $\mathcal{C} \in \mathsf{GFO}_{\mathrm{qf}}$. An adequate choice of c' in terms of c and the maximum degree over all polynomials in R ensures that the interpretation of (φ, c') coincides with that of R for all labelings whose codomain is $[n^c]_0^k$ by preventing overflows. □

Corollary 3.77. $\mathsf{GFO}_{\mathrm{qf}} = \mathsf{BoundedPBS}(\mathbb{N}_0) = \mathsf{BoundedPBS}(\mathbb{Q}) \subseteq \mathsf{PBS}(\mathbb{N}_0) = \mathsf{PBS}(\mathbb{Q}) \subseteq \mathsf{PBS}(\mathbb{R}) \subseteq [\text{Small} \cap \text{Hereditary}]_\subseteq$.

Next, we show that if $\mathsf{GFO}(<)$ can be separated from $\mathsf{PBS}(\mathbb{N}_0)$ then this separation can be amplified to $\mathsf{GFO}(<) \neq \mathsf{GFO}_{\mathrm{qf}}$. To prove this we show that if $\mathsf{GFO}_{\mathrm{qf}}$ has a complete graph class w.r.t $\leq_{\mathrm{BF}}$ which is hereditary then $\mathsf{PBS}(\mathbb{N}_0)$ collapses to $\mathsf{GFO}_{\mathrm{qf}}$. A similar statement holds w.r.t. $\leq_{\mathrm{sg}}$-reductions.

Let us say a set of graph classes $\mathbb{A}$ is closed under hereditary closure if for all $\mathcal{C}$ in $\mathbb{A}$ it holds that its hereditary closure $[\mathcal{C}]_\subseteq$ is in $\mathbb{A}$. If a set of graph classes $\mathbb{A}$ is closed under hereditary closure and subsets then a graph class $\mathcal{C}$ is in $\mathbb{A}$ iff $[\mathcal{C}]_\subseteq$ is in $\mathbb{A}$. As a consequence it suffices to consider only hereditary graph classes when studying sets of graph classes that are closed under hereditary closure.

Theorem 3.78. *If* $\mathsf{GFO}_{\mathrm{qf}}$ *is closed under hereditary closure then* $\mathsf{GFO}_{\mathrm{qf}} = \mathsf{PBS}(\mathbb{N}_0)$.

Proof. Assume that $\mathsf{GFO}_{\mathrm{qf}}$ is closed under hereditary closure. We show that $\mathsf{PBS}(\mathbb{N}_0) \subseteq \mathsf{BoundedPBS}(\mathbb{N}_0)$. Let $\mathcal{C}$ be a graph class that is in $\mathsf{PBS}(\mathbb{N}_0)$ via a PBS R with $2k$ variables, i.e. $\mathcal{C} \subseteq \mathrm{gr}(F_R^{\mathbb{N}_0})$. It holds that $\mathcal{D} := \mathrm{gr}(F_R^{\mathbb{N}_0}, 1)$ is in $\mathsf{PBS}(\mathbb{N}_0)$. We claim that every graph in $\mathcal{C}$ occurs as induced subgraph of some graph in $\mathcal{D}$ and therefore $\mathcal{C} \subseteq [\mathcal{D}]_\subseteq$. Since $\mathsf{BoundedPBS}(\mathbb{N}_0) = \mathsf{GFO}_{\mathrm{qf}}$ is closed under hereditary closure by assumption it follows that $[\mathcal{D}]_\subseteq$ (and thus $\mathcal{C}$) is in $\mathsf{GFO}_{\mathrm{qf}}$. Let G be a graph with n vertices that is in $\mathcal{C}$. There exists a labeling $\ell\colon V(G) \to \mathbb{N}_0^k$ such that G is in $\mathrm{gr}(F_R^{\mathbb{N}_0})$ via ℓ. Let $z \in \mathbb{N}_0$ be the maximal value that occurs in the image of ℓ. Let H be some graph with $z + n$ vertices and $V(G) \subseteq V(H)$. The labeling ℓ is a partial labeling of H which shows that G is an induced subgraph of H. If one extends the labeling ℓ such that $\ell(u) = (0, \ldots, 0)$ for all $u \in V(H) \setminus V(G)$ then this shows that H is in $\mathrm{gr}(F_R^{\mathbb{N}_0}, 1) = \mathcal{D}$. □

Lemma 3.79. *Let* $\mathbb{A}$ *be a set of graph classes closed under* $\leq_{\mathrm{BF}}$. *If there exists a hereditary graph class that is* $\leq_{\mathrm{BF}}$*-complete for* $\mathbb{A}$ *then* $\mathbb{A}$ *is closed under hereditary closure.*

Proof. Let $\mathcal{C}$ be a hereditary graph class that is $\leq_{\mathrm{BF}}$-complete for $\mathbb{A}$. Let $\mathcal{D}$ be a graph class in $\mathbb{A}$. Since $\mathcal{C}$ is complete for $\mathbb{A}$ there exists a k-ary Boolean function f such that $\mathcal{D} \subseteq f(\mathcal{C}, \ldots, \mathcal{C})$. We show that every graph which occurs as induced subgraph of some graph in $\mathcal{D}$ is also in $f(\mathcal{C}, \ldots, \mathcal{C})$, i.e. $[\mathcal{D}]_\subseteq \subseteq f(\mathcal{C}, \ldots, \mathcal{C})$. Let G be a graph in $\mathcal{D}$ and let G' be an induced subgraph of G on vertex set $V' \subseteq V(G)$. There exist graphs $H_1, \ldots, H_k \in \mathcal{C}$ on vertex set $V(G)$ such that $G = f(H_1, \ldots, H_k)$. It follows that $G' = f(H'_1, \ldots, H'_k)$ where H'_i is the induced subgraph of H_i on vertex set V' for $i \in [k]$. Since $\mathcal{C}$ is hereditary it contains $H'_1, \ldots, H'_k$ and thus $G' \in f(\mathcal{C}, \ldots, \mathcal{C})$. Therefore $[\mathcal{D}]_\subseteq \subseteq f(\mathcal{C}, \ldots, \mathcal{C})$. Stated differently, $[\mathcal{D}]_\subseteq$ is $\leq_{\mathrm{BF}}$-reducible to $\mathcal{C}$ via f and thus must be in $\mathbb{A}$. □

Lemma 3.80. *Let* $\mathbb{A}$ *be a set of graph classes closed under* $\leq_{\mathrm{sg}}$. *If there exists a hereditary and inflatable graph class that is* $\leq_{\mathrm{sg}}$*-complete for* $\mathbb{A}$ *then* $\mathbb{A}$ *is closed under hereditary closure.*

Proof. Let $\mathcal{C}$ be a hereditary and inflatable graph class that is $\leq_{\text{sg}}$-complete for $\mathbb{A}$. We argue that if a graph class $\mathcal{D}$ is $\leq_{\text{sg}}$-reducible to $\mathcal{C}$ then $[\mathcal{D}]_{\subseteq}$ is also $\leq_{\text{sg}}$-reducible to $\mathcal{C}$. From that the above statement follows. Due to Lemma 3.32 it holds that $\mathcal{D}$ is $\leq_{\text{sg}}$-reducible to $\mathcal{C}$ iff there exist a $k \in \mathbb{N}$ and a k^2-ary Boolean function f such that for all graphs G in $\mathcal{D}$ there exists a graph H in $\mathcal{C}$ such that G has an (H, f)-representation. Observe that if G has an (H, f)-representation then every induced subgraph of G has an (H, f)-representation as well. Therefore $[\mathcal{D}]_{\subseteq}$ is $\leq_{\text{sg}}$-reducible to $\mathcal{C}$ via f. □

Corollary 3.81. *If* GFO_{qf} *has a* $\leq_{\text{BF}}$*-complete graph class that is hereditary then* $\mathsf{GFO}_{\text{qf}} = \mathsf{PBS}(\mathbb{N}_0)$*. If* GFO_{qf} *has an* $\leq_{\text{sg}}$*-complete graph class that is hereditary and inflatable then* $\mathsf{GFO}_{\text{qf}} = \mathsf{PBS}(\mathbb{N}_0)$*.*

Since $\mathsf{GFO}(<)$ has a hereditary graph class which is $\leq_{\text{BF}}$-complete, namely linear neighborhood graphs, it follows that $\mathsf{GFO}(<)$ must be a strict subset of GFO_{qf} unless $\mathsf{GFO}(<) = \mathsf{PBS}(\mathbb{N}_0)$.

We conclude this subsection by making two remarks about what a labeling scheme for line segment graphs must look like if it exists. From Fact 3.69 we know that line segment graphs are in $\mathsf{PBS}(\mathbb{N}_0)$. The PBS R that shows this does exactly encode the geometrical representation of this graph class. It suffices to use natural numbers to encode the endpoints of the line segments, but how large do these numbers have to be? More specifically, given a line segment graph with n vertices what is the minimal number of bits w.r.t. n required to encode the endpoints, over all possible representations. If a logarithmic number of bits $\mathcal{O}(\log n)$ would suffice then there exists a $c \in \mathbb{N}$ such that $\text{gr}(F_R^{\mathbb{N}_0}, c)$ contains all line segment graphs and thus this class is in GFO_{qf}. In [MM13] it was proved that at least an exponential number of bits is required and therefore such a c does not exist. Similarly, encoding the geometrical representation of disk graphs and k-dot product graphs also requires at least an exponential number of bits [MM13; KM12].

The previous consideration essentially says that the geometrical representation of line segment graphs does not provide us with a labeling scheme due to the label length restriction. Nonetheless, there still can be a labeling scheme S which represents line segment graphs. In fact, it is even conceivable that $\text{gr}(S)$ is exactly the set of line segment graphs. We explain, however, that under a certain complexity-theoretic assumption this is impossible because $\text{gr}(S)$ must contain much more than just line segment graphs. Consider the following computational problem: given a PBS R as input (the Boolean function is given by a propositional formula), decide whether $F_R^{\mathbb{R}}$ is non-empty. This is called the decision problem for the existential theory of the reals ETR and is known to be in PSPACE [Can88] and NP-hard. Since many interesting problems can be reduced to it and it seems to not coincide with machine-based complexity classes, it has been awarded the status of a complexity class. The eponymous complexity class is defined as the closure of ETR under polynomial-time many-one reductions. Observe that the recognition problem for line segment graphs lies in ETR. In fact, this problem is complete for it [KM94]. Therefore if we assume that $\mathsf{NP} \neq \mathsf{ETR}$ then for every labeling scheme S in GP that represents line segment graphs the recognition problem for line segment graphs on $\text{gr}(S)$ is not in NP. This follows from the observation that the recognition problem for $\text{gr}(S)$ is in NP. Therefore recognizing line segment graphs reduces to recognizing them on graphs from $\text{gr}(S)$.

3.4.4 Constant-Time RAMs

We show that GFO_{qf} also admits a characterization which is interesting from a practical point of view. A graph class is in GFO_{qf} iff it has a labeling scheme whose label decoder can be computed in constant-time on a RAM with addition, subtraction and multiplication. Additionally, we show that adding division (and possibly some simple bitwise-operations) does not increase the set of graph classes that can be expressed beyond GTC^0.

A RAM has an unlimited number of registers $r_0, r_1, r_2, \dots$ and each register can hold a non-negative integer. The register r_0 is called accumulator and is used to store the result of arithmetic operations. The RAM is equipped with the four arithmetic instructions addition '$r_0 \leftarrow r_i + r_j$', subtraction '$r_0 \leftarrow r_i - r_j$', multiplication '$r_0 \leftarrow r_i \times r_j$' and division '$r_0 \leftarrow r_i / r_j$'. In the case of subtraction if $r_i < r_j$ then r_0 is assigned the value 0. In the case of division r_0 contains $\lfloor \frac{r_i}{r_j} \rfloor$ if $r_j \neq 0$ and 0 otherwise. Additionally, it has a conditional jump instruction 'JZ c' where c refers to a line in the program. If the accumulator contains zero then the program continues execution from line c, otherwise the jump instruction is skipped. We call such an instruction a backwards jump if c refers to a line that occurs before that instruction. Furthermore, the RAM is capable of indirect addressing. It has a store operation $r[r_i] \leftarrow r_0$ where it stores the contents of the accumulator in the register r_{r_i} and a load operation $r_0 \leftarrow r[r_i]$ where it stores the contents of r_{r_i} in the accumulator. We call a sequence of such instructions a RAM program. A RAM program P has a prescribed number of inputs k. Let $x_1, \dots, x_k \in \mathbb{N}_0$ be an input of P. Before P is executed every register r_i is initialized with the value x_i for $i \in [k]$ and every other register is initialized with the value 0. We say two RAM programs with the same number of inputs are equivalent if for every input the accumulator of the RAM holds the same value after the execution of either of these programs.

In the case of RAM programs that run in constant-time backwards jumps and indirect addressing operations are syntactic sugar.

Lemma 3.82. *For every RAM program that runs in constant time there exists an equivalent RAM program that neither uses backwards jumps nor indirect addressing.*

Proof. Let P be a RAM program that runs at most t steps on every input for a constant $t \in \mathbb{N}$. A backward jump in P can be seen as a loop that repeats at most t times. Additionally, before the end of each iteration it checks if the accumulator is not zero and in that case breaks meaning that further iterations of the loop are aborted. By unrolling this loop and replacing the conditional break by a jump-if-not-zero the backward jump can be removed.

The indirect addressing operations can be replaced by mimicking an associative array as follows. Assume P has $d \leq t$ indirect store operations. Let $s_i^{\text{key}}, s_i^{\text{val}}$ for $i \in [d]$ be $2d$ 'fresh' registers in the sense that none of them is directly addressed in P. Let us say an s_i is uninitialized if $s_i^{\text{key}} = 0$. For an indirect store $r[r_c] \leftarrow r_0$ we do the following. Let x be the value of r_c at that point. If x equals the index of a register that is directly addressed in P then execute $r_x \leftarrow r_0$. Otherwise, if there exists an $i \in [d]$ such that $s_i^{\text{key}} = x$ then execute $s_i^{\text{val}} \leftarrow r_0$. If no such i exists take the first uninitialized s_j and execute $s_i^{\text{key}} \leftarrow r_c$ and $s_i^{\text{val}} \leftarrow r_0$. For an indirect load $r_0 \leftarrow r[r_c]$ we proceed similarly. Let x be the value of r_c at that point. If x equals the index of a register that is directly addressed in P then execute $r_0 \leftarrow r_x$. If x equals the value of some s_i^{key} then execute $r_0 \leftarrow s_i^{\text{val}}$ and else do $r_0 \leftarrow 0$. For correctness observe that our associative array can never run out of space, i.e. there will always exist an uninitialized s_i if we need one and at no point will two different entries of our array have the same key, i.e. $s_i^{\text{key}} \neq s_j^{\text{key}}$ for all $i \neq j \in [d]$. Also, the case distinctions can be realized by using only forward jumps. $\square$

```
1: $P_1$
2: JZ 8
3: $P_2$
4: JZ 7
5: $P_3$
6: JZ 8
7: $P_4$
8: $P_5$
```

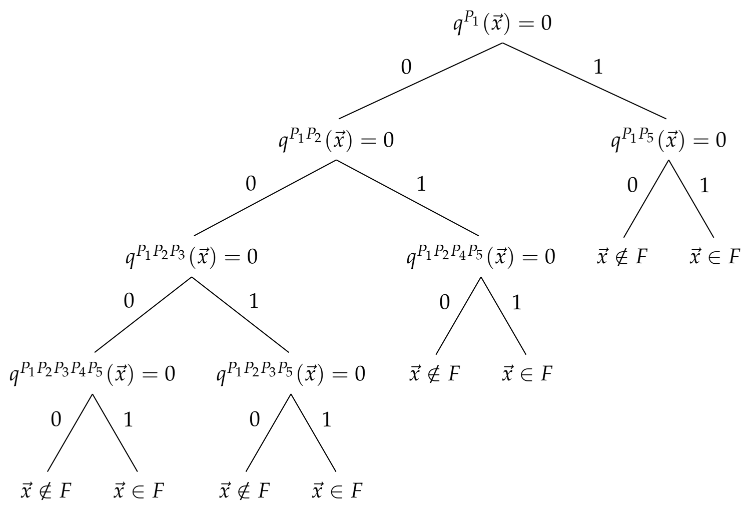

$q^{P_1}(\vec{x}) = 0$
0
1
$q^{P_1P_2}(\vec{x}) = 0$
$q^{P_1P_5}(\vec{x}) = 0$
$q^{P_1P_2P_3}(\vec{x}) = 0$
$q^{P_1P_2P_4P_5}(\vec{x}) = 0$
$\vec{x} \notin F$
$\vec{x} \in F$
$q^{P_1P_2P_3P_4P_5}(\vec{x}) = 0$
$q^{P_1P_2P_3P_5}(\vec{x}) = 0$

FIGURE 3.6: Converting a RAM program into a polynomial-Boolean system

Similar to a polynomial-Boolean system a RAM program P with k inputs can be interpreted to compute a relation $F_P \subseteq \mathbb{N}_0^k$ such that $\vec{x} \in F_P$ iff the accumulator does not contain the value 0 after executing the program P on input $\vec{x}$.

Theorem 3.83. *Given $k \in \mathbb{N}$ and $F \subseteq \mathbb{N}_0^k$. It holds that F can be computed by a RAM program without division in constant-time iff F is representable by a polynomial-Boolean system.*

Proof. "$\Rightarrow$": Let P be a RAM program with k inputs that does not use division, jumps or indirect addressing. It is not difficult to see that the value of the accumulator after executing P can be expressed as composition of addition, multiplication and cut-off subtraction using the input values and the constant zero. Let q^P denote the polynomial with k variables that corresponds to this composition when replacing cut-off subtraction by the regular one. For two RAM programs P_1, P_2 let $q^{P_1P_2}$ be the polynomial for the program that executes P_1 and then P_2.

Given a division-free RAM program P with k inputs which runs in constant time we show how to convert it into a polynomial-Boolean system which represents the same relation F over $\mathbb{N}^k$ that P computes. Due to Lemma 3.82 we can assume w.l.o.g. that P contains no backwards jumps or indirect addressing. For a subtraction '$x_0 \leftarrow x_i - x_j$' let us call 'if $x_i > x_j$ then $x_0 \leftarrow x_i - x_j$ else $x_0 \leftarrow 0$' its guarded version. The semantics of this guarded version can be expressed in a polynomial-Boolean system. Replace every subtraction in P by its guarded version. Observe that the first jump operation in P depends on the value of the accumulator at that point. Let P_1 be the jump-free program that occurs before that jump. Then the required value is given by the polynomial q^{P_1}. Then the first jump is taken iff the equation $q^{P_1} = 0$ holds. Assume that this first jump is taken and the portion of P that is executed after this jump and before the second jump is P_2. Then the second jump is taken iff $q^{P_1P_2} = 0$. Similarly, one can consider what happens if the first jump is not taken and construct a binary decision diagram in that fashion. An example of this given in Figure 3.6 where $\vec{x}$ denotes the input. The subprograms $P_1, \ldots, P_5$ do not contain jump instructions. An induction over the depth of the binary decision diagram shows that this works correctly.

"$\Leftarrow$": This is straightforward. To evaluate the polynomials it suffices to use addition and multiplication. Then a binary decision diagram of the Boolean function can be expressed in terms of conditional jumps. $\square$

Corollary 3.84. *A graph class $\mathcal{C}$ is in $\mathsf{GFO}_{\mathrm{qf}}$ iff it has a labeling scheme with a label decoder that can be computed in constant-time on a RAM without division.*

Proof. Due to Theorem 3.76 it holds that $\mathcal{C}$ is in $\mathsf{GFO}_{\mathrm{qf}}$ iff there exist $c, k \in \mathbb{N}$ and a relation $F \subseteq \mathbb{N}_0^k$ such that $\mathcal{C} \subseteq \mathrm{gr}(F, c)$ and F can be represented by a polynomial-Boolean system. From Theorem 3.83 it follows that F can be represented by polynomial-Boolean system iff it can be computed by a RAM program without division in constant-time. □

The division operation allows one to compute relations on a RAM in constant-time which cannot be represented by a polynomial-Boolean system. Therefore there could be a graph class which is not in $\mathsf{GFO}_{\mathrm{qf}}$ but has a labeling scheme with a label decoder that can be computed in constant-time on a RAM.

Fact 3.85. *The divisibility relation can be computed by a RAM program in constant-time but it cannot be represented by a polynomial-Boolean system.*

Proof. It holds that y divides x iff $\lfloor \frac{x-1}{y} \rfloor < \lfloor \frac{x}{y} \rfloor$ for $x \geq y > 0$. Therefore the divisibility relation can be computed by a RAM program that runs in constant time. To prove that this relation cannot be represented by a polynomial-Boolean system it suffices to show that even numbers cannot be represented by such a system. Every univariate polynomial is eventually monotone, i.e. after a certain point it will either never decrease or never increase. Therefore for every unary relation $F \subseteq \mathbb{N}_0$ representable by a polynomial system either F or its complement is finite. Since there are infinitely many even and odd numbers these two sets cannot be represented by a polynomial-Boolean system. □

Theorem 3.86. *Every graph class that has a labeling scheme with a label decoder that can be computed in constant-time on a RAM lies in GTC^0.*

Proof. For some $k \in \mathbb{N}$ let us say a relation $F \subseteq \mathbb{N}_0^k$ is computable in TC^0 if there exists a family of TC^0-circuits $(C_n)_{n \in \mathbb{N}}$ with the following property. The circuit C_n has kn input bits with k blocks $x_1, \ldots, x_k$ that each consists of n bits. The circuit C_n accepts the input $(x_1, \ldots, x_k)$ iff $(x_1, \ldots, x_k) \in F$ where x_i is interpreted as a non-negative integer encoded in binary. It suffices to argue that every relation that can be computed on a RAM in constant-time is computable in TC^0 as well. This can be shown in the same way that we converted a RAM program into a polynomial-Boolean system in the proof of Theorem 3.83. This works because all four arithmetic operations and comparing numbers ($<$) can be performed in TC^0 (for division see [Hes01], for multiplication see [Vol99]). □

3.5 Summary and Open Questions

In Figure 3.7 an overview of all the sets of graph classes that we have seen is given. We use this figure to summarize our results and point out interesting questions.

In the beginning we defined labeling schemes in terms of languages. This allowed us to restrict the complexity of labeling schemes in terms of classical complexity classes. For P and R the corresponding classes of labeling schemes GP and GR coincide with the definitions of labeling schemes that were given in [KNR92] and [Mul88], respectively. In Section 3.1 we established that GP is a proper subset of GR. This means that restricting the computational complexity of labeling schemes does indeed affect what set of graph classes can be represented. Moreover, there is a strict hierarchy of labeling schemes which somewhat resembles the situation known from classical complexity. The diagonalization

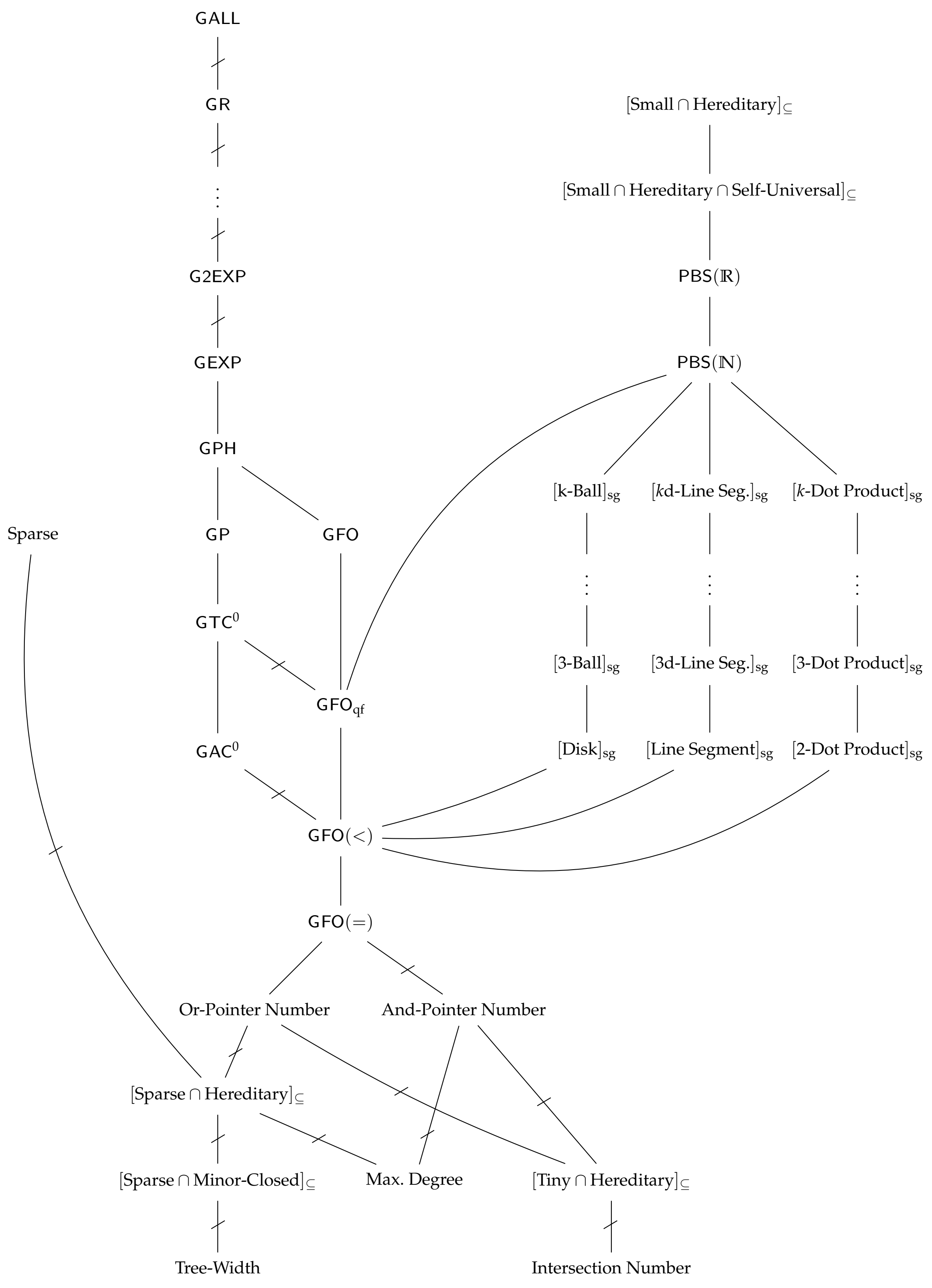


FIGURE 3.7: Landscape of small graph classes

argument used to prove this has a brute force aspect which requires double exponential time and therefore fails to separate classes below G2EXP. For example, it is not clear whether GP is a proper subset of GEXP even though classical complexity would suggest so. The graph classes used to show these separations are far removed from any natural graph class and as a consequence we do not learn anything about the complexity requirements of hereditary graph classes. It seems that not much else can be said about the classes between GAC^0 and GALL, which is supported by the fact that GAC^0 is not a subset of $[\text{Small} \cap \text{Hereditary}]_{\subseteq}$ (Theorem 3.16).

Interestingly though, every hereditary graph class with a labeling scheme that we are aware of is also in GAC^0. Motivated by this observation, we introduced logical labeling schemes which capture the algorithmic resources required to express many graph classes quite well. For example, interval graphs are complete for the fragment $\mathsf{GFO}(<)$ and trees are complete for the fragment $\mathsf{GFO}(=)$ w.r.t. $\leq_{\text{sg}}$-reductions. Every hereditary graph class with a labeling scheme which is mentioned in this thesis is also in $\mathsf{GFO}(<)$. The only exception for which we do not know whether this holds are graph classes with bounded clique-width. Such graph classes are in $[\text{Small} \cap \text{Hereditary} \cap \text{Self-Universal}]_{\subseteq}$. Can they also be represented by a polynomial-Boolean system?

When studying the limitations of labeling schemes we deem GFO_{qf} to be an important class for the following reasons. First, it is fairly robust and admits different characterizations in terms of polynomial-Boolean systems (Corollary 3.77) and constant-time RAMs (Corollary 3.84). Especially the RAM characterization is interesting from a practical perspective. Disproving that a graph class can be represented by a quantifier-free logical labeling scheme shows that it either has no constant-time labeling scheme or such a scheme requires division. Secondly, unlike GAC^0 and its supersets this class is well-behaved in the sense that it is a subset of $[\text{Small} \cap \text{Hereditary}]_{\subseteq}$. Stated differently, there are no wildly non-uniform graph classes in GFO_{qf}. This is one of the reasons why we believe that lower bounds against this class can be achieved. Therefore we suggest the following variant of the implicit graph conjecture:

Conjecture 3.87 (Weak[3] Implicit Graph Conjecture). *Every small and hereditary graph class is in GFO_{qf}, i.e. $\mathsf{GFO}_{\text{qf}} = [\text{Small} \cap \text{Hereditary}]_{\subseteq}$.*

Candidates for this conjecture obviously include the ones for the implicit graph conjecture such as k-ball graphs, kd-line segment graphs and k-dot product graphs for $k \geq 2$. Curiously, all of these graph classes contain interval graphs as a subset. In [Fid+98, Thm. 21] it is shown that interval graphs are 2-dot product graphs. Therefore the $\leq_{\text{sg}}$-closures of these graph classes contain $\mathsf{GFO}(<)$. Since $\mathsf{PBS}(\mathbb{N})$ is closed under $\leq_{\text{sg}}$-reductions and contains these graph classes (Fact 3.69) their $\leq_{\text{sg}}$-closures are subsets of $\mathsf{PBS}(\mathbb{N})$. Additionally, graphs with bounded clique-width are candidates for the weak implicit graph conjecture.

A question related to the weak implicit graph conjecture is whether GFO_{qf} has a complete graph class. While it is not difficult to find complete graph classes for $\mathsf{GFO}(<)$ and $\mathsf{GFO}(=)$ using the algebraic interpretation of quantifier-free logical labeling schemes (Theorem 3.51), it is not clear what a complete graph class for GFO_{qf} looks like or whether it even exists. In this regard we have shown that GFO_{qf} cannot have a $\leq_{\text{BF}}$-complete hereditary graph class unless $\mathsf{GFO}_{\text{qf}} = \mathsf{PBS}(\mathbb{N})$, and a similar statement for $\leq_{\text{sg}}$-reductions (Corollary 3.81). The question of whether GFO_{qf} and $\mathsf{PBS}(\mathbb{N})$ coincide can be understood as the question of whether the restriction on the label length in quantifier-free labeling schemes is significant. For instance, in the case of $\mathsf{GFO}_{\text{qf}}(<)$ this restriction can be dropped without

[3]We consider the strength of a conjecture to be its ability to resist refutation.

changing the class, i.e. vertices can be labeled with arbitrarily large numbers. If $\mathsf{GFO}_{\mathrm{qf}}$ can be separated from $\mathsf{PBS}(\mathbb{N})$ this does not only refute the weak implicit graph conjecture but also directly implies that $\mathsf{GFO}(<)$ is strictly included in $\mathsf{GFO}_{\mathrm{qf}}$ (see the paragraph after Corollary 3.81). We conjecture that all of the three inclusions from $\mathsf{GFO}(=)$ to $\mathsf{PBS}(\mathbb{N})$ are strict. As a side question we wonder whether GFO is included in $[\mathrm{Small} \cap \mathrm{Hereditary}]_{\subseteq}$, and if so how does it relate to polynomial-Boolean systems? This inclusion would also imply that GFO is a strict subset of GPH.

What are simple to verify criterions that allow us to conclude that a graph class $\mathcal{C}$ is in $\mathsf{GFO}_{\mathrm{qf}}$ without needing to find a labeling scheme? For the classes $\mathsf{GFO}(=)$ and $\mathsf{GFO}(<)$ it holds that they contain a graph class $\mathcal{C}$ iff they contain the hereditary closure of $\mathcal{C}$. For what finite sets of graphs X is the hereditary graph class defined by having X as forbidden induced subgraphs in $\mathsf{GFO}(<)$ or $\mathsf{GFO}(=)$?

The aim of structural graph theory is to find characteristics that are shared by certain sets of graph classes. For instance, the graph minor theorem does this for minor-closed graph classes by proving that every such class can be characterized by a finite set of forbidden minors. The class $[\mathrm{Tiny} \cap \mathrm{Hereditary}]_{\subseteq}$ is exactly the set of graph classes with bounded twin index (see Fact 3.19). Similarly, the class $[\mathrm{Sparse} \cap \mathrm{Hereditary}]_{\subseteq}$ can be characterized as the set of graph classes with bounded arboricity. Due to the simplicity of the parameters twin index and arboricity these are nice characterizations in the sense that they enhance our understanding of these sets of graph classes. What would be a nice characterization of $[\mathrm{Small} \cap \mathrm{Hereditary}]_{\subseteq}$? Notice that this class is substantially richer than the others and therefore the complexity of such a nice characterization should be expected to match this richness. The two reduction notions that we introduced seem to be adequate candidates for such a characterization. Does $[\mathrm{Small} \cap \mathrm{Hereditary}]_{\subseteq}$ have a complete graph class with respect to $\leq_{\mathrm{BF}}$- or $\leq_{\mathrm{sg}}$-reductions? Assume that $\mathcal{C}$ is such a complete graph class. This would mean that the adjacency structure of all other graph classes in $[\mathrm{Small} \cap \mathrm{Hereditary}]_{\subseteq}$ is just a Boolean combination of the adjacency of graphs in $\mathcal{C}$. The fact that forests are complete for $[\mathrm{Sparse} \cap \mathrm{Hereditary}]_{\subseteq}$ with respect to monotone algebraic reductions $\leq_{\mathrm{M}}$ (see the last paragraph of Subsection 3.3.1) serves as an example that algebraic reductions are indeed able to characterize a set of graph classes which is defined in terms of graph-theoretical properties. The following question is just a curiosity that pertains to this previous example. Let us say two graph classes are M-equivalent if they can be $\leq_{\mathrm{M}}$-reduced to each other. Furthermore, we say two graph classes are eventually M-equivalent if there exists an $n \in \mathbb{N}$ such that the restrictions of these graph classes to graphs with at least n vertices are M-equivalent. Does the eventually M-equivalence relation have finite index on sparse and hereditary graph classes?

It seems interesting and worthwhile to further investigate the properties of these reduction notions. An immediate question in that regard is whether there are two undirected graph classes $\mathcal{C}, \mathcal{D}$ such that $\mathcal{C}$ is $\leq_{\mathrm{sg}}$-reducible to $\mathcal{D}$ but not $\leq_{\mathrm{BF}}$-reducible. For example, k-interval graphs are $\leq_{\mathrm{sg}}$-reducible to interval graphs but it is not clear whether they are also $\leq_{\mathrm{BF}}$-reducible for $k \geq 2$. Also, are forests and interval graphs $\leq_{\mathrm{BF}}$-complete for $\mathsf{GFO}(=)$ and $\mathsf{GFO}(<)$ when restricted to undirected graph classes, respectively? Or is it the case that $\mathsf{GFO}(=)$ and $\mathsf{GFO}(<)$ restricted to undirected graph classes do not have $\leq_{\mathrm{BF}}$-complete graph classes? Another interesting task is to find two candidates for the implicit graph conjecture which can be reduced to each other. This would show that there must be a common obstacle to designing a labeling scheme for them.

Last but not least, there remain two open questions concerning the pointer numbers. Does every graph class with bounded and-pointer number also have bounded or-pointer number or are these two parameters incomparable? And, is there an undirected graph class in $\mathsf{GFO}(=)$ with unbounded or-pointer number (see the second paragraph after Fact 3.15)? In the next chapter we consider algorithmic questions regarding these two parameters.

Chapter 4

Algorithmic Properties of Graph Classes with Implicit Representations

We have seen that almost all graph classes known to have an implicit representation can be located in $\mathsf{GFO}(<)$. This includes algorithmically well-studied graph classes such as interval graphs and uniformly sparse graph classes. Due to the low descriptive complexity of $\mathsf{GFO}(<)$ it seems reasonable to suspect that the adjacency structure of graph classes from this class cannot be very complicated. This begs the question whether certain algorithmic problems might be tractable on every graph class in $\mathsf{GFO}(<)$. The same can be asked of other classes such as GAC^0 or $\mathsf{GFO}(=)$. In the first section we show how this can be naturally interpreted as a question in parameterized complexity. We then use this interpretation in conjunction with lower and upper bounds from the literature to identify interesting algorithmic questions.

After looking at various algorithmic problems in the first section we focus on the graph isomorphism problem. This is one of the few natural problems which is neither known to be in P nor to be NP-complete. A research direction that has seen substantial progress in recent years is the study of the isomorphism problem for restricted graph classes. Naturally, we ask ourselves how graph classes with implicit representations fit into this picture. The isomorphism problem for graphs of degeneracy at most two is already as hard as the general graph isomorphism problem. In Figure 4.1 a reduction is shown which maps a graph G with n vertices to a graph G' with $n + n^2$ vertices of degeneracy at most two such that the isomorphism class is preserved. To obtain G' from G copy the vertices of G and then connect n new vertices to each old vertex in G'. Then for every edge $\{u, v\}$ in G pick two vertices u', v' in G' such that u' is only adjacent to u and v' only to v and connect u' and v'. Since $\mathsf{GFO}(=)$ already contains this graph class there is no point in considering it without further restrictions. A natural restriction is to bound the number of variables per vertex. Three variables per vertex in $\mathsf{GFO}(=)$ already suffice to express graphs with degeneracy at most two and therefore this is an infeasible fragment. For graph classes that can be expressed with one variable per vertex in $\mathsf{GFO}(<)$ it is trivial to devise an isomorphism test that works in logspace. What about graph classes that can be expressed with two variables in $\mathsf{GFO}(<)$? Unlike the previous case this fragment already contains a couple of interesting graph classes such as forests and interval graphs. It is known that for

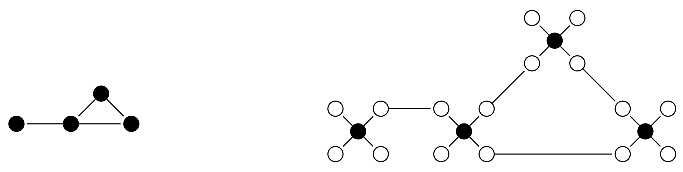


FIGURE 4.1: Example of the reduction for GI-hardness of 2-degenerate graphs

both classes isomorphism can be decided in logspace [Lin92; Kö+11]. A curious case is that of circular-arc (CA) graphs. While this graph class resides in the same fragment it is not even known whether isomorphism for this class can be decided in polynomial-time. The main part of this chapter is dedicated to investigating how this problem can be solved.

4.1 Parameterized Complexity of Sets of Graph Classes

A parameter κ in parameterized complexity is a total function which maps words over some alphabet Σ to natural numbers, i.e. $\kappa\colon \Sigma^* \to \mathbb{N}$. A parameterized problem is a tuple (L, κ) where L is a language and κ is a parameter, both over the same alphabet. This formalization of parameterized problems is used in the introductory textbook [FG06][1]. A graph parameter is a total function mapping unlabeled graphs to natural numbers and therefore can be regarded as special case of a parameter. For a parameter κ and $c \in \mathbb{N}$ let κ_c denote the set of words w with $\kappa(w) \leq c$. For example, tree-width$_c$ is the set of graphs with tree-width at most c.

To compare two parameters κ, τ over the same alphabet the following notion of boundedness is used: κ is upper bounded by τ, in symbols $\kappa \preccurlyeq \tau$, if there exists a function $f\colon \mathbb{N} \to \mathbb{N}$ such that $\kappa(w) \leq f(\tau(w))$ holds for all words w. If $\kappa \preccurlyeq \tau$ holds then (L, τ) is fpt-reducible to (L, κ) for every language L. The precise definition of fpt-reducibility is not relevant for us here; it can be found in [FG06]. It suffices to know that it can be seen as an analogon of polynomial-time many-one reducibility in classical complexity. Intuitively, designing a good algorithm for a problem using τ as parameter is not harder than doing this for κ. If $\kappa \preccurlyeq \tau$ and $\tau \preccurlyeq \kappa$ holds we say that κ and τ are equivalent. From the previous implication it follows that (L, κ) and (L, τ) are fpt-equivalent for all languages L whenever κ and τ are equivalent. As a consequence, it does not make a difference whether κ or τ is considered when assessing the complexity of a parameterized problem and thus it is more accurate to define a parameterized problem as a tuple $(L, \mathbb{K})$ where $\mathbb{K}$ denotes an equivalence class of parameters. In that sense parameterized complexity is no more about parameters than graph theory is about adjacency matrices. However, while it is self-evident that adjacency matrices represent graphs it is not so obvious what is represented by parameters. A different notion of boundedness helps to answer this question. Let us say a language L is bounded by a parameter κ if there exists a $c \in \mathbb{N}$ such that $L \subseteq \kappa_c$. Let us write $\mathbb{K}(\kappa)$ to denote the set of languages that are bounded by κ. We say κ is a subset of τ, in symbols $\kappa \subseteq \tau$, if $\mathbb{K}(\kappa) \subseteq \mathbb{K}(\tau)$. Stated differently, every language that is bounded by κ is bounded by τ as well. For example, the maximum degree is a subset of the clique number but not vice versa. In fact, the '$\subseteq$'-relation is just the inverse relation of '$\preccurlyeq$'.

Fact 4.1. *Let κ, τ be parameters over the same alphabet. It holds that $\kappa \preccurlyeq \tau$ iff $\tau \subseteq \kappa$.*

Proof. "$\Rightarrow$": Let $f\colon \mathbb{N} \to \mathbb{N}$ be a monotone function such that $\kappa(w) \leq f(\tau(w))$ for all words w. We show inductively that for every $i \in \mathbb{N}$ it holds that $\tau_i \subseteq \kappa_{f(i)}$. For the base case $i = 1$ it holds that $w \in \tau_1$ iff $\tau(w) = 1$. It follows that $\kappa(w) \leq f(1)$ and therefore $w \in \kappa_{f(1)}$. For the inductive step $i \to i+1$ it must be the case that w is either in $\tau_{i+1} \setminus \tau_i$ or in τ_i. If w is in τ_i then by induction hypothesis it holds that $w \in \kappa_{f(i)}$. Since f is monotone it follows that $w \in \kappa_{f(i+1)}$ as well. For the other case it holds that $\tau(w) = i + 1$ and therefore $\kappa(w) \leq f(i+1)$ which means $w \in \kappa_{f(i+1)}$.

[1] They make the additional requirement that a parameter must be polynomial-time computable, which we shall ignore here.

"$\Leftarrow$": Since $\tau \subseteq \kappa$ there exists a function $f\colon \mathbb{N} \to \mathbb{N}$ such that $\tau_i \subseteq \kappa_{f(i)}$ for all $i \in \mathbb{N}$. We show that $\kappa(w) \leq f(\tau(w))$ for all words w. Let $\tau(w) = k$ for some $k \in \mathbb{N}$. Then it holds that $w \in \tau_k$ and therefore $w \in \kappa_{f(k)}$ as well. This means $\kappa(w) \leq f(k) = f(\tau(w))$. □

Corollary 4.2. *Two parameters are equivalent iff they bound the same set of languages.*

Therefore the answer to the previous question is that a parameter represents a set of languages. However, not every set of languages can be interpreted as a parameter. We are only interested in sets of languages that can be represented by a parameter. To distinguish between such sets of languages and parameters let us call the former ones parameterizations.

Definition 4.3. *A set of languages $\mathbb{K}$ over an alphabet Σ is a parameterization if there exists a parameter κ over Σ such that $\mathbb{K} = \mathbb{K}(\kappa)$.*

Theorem 4.4. *A set of languages $\mathbb{K}$ over Σ is a parameterization iff the following holds:*

1. *$\mathbb{K}$ is closed under union*
2. *$\mathbb{K}$ contains $\{w\}$ for every word w over Σ*
3. *there exists a countable subset $\mathbb{K}'$ of $\mathbb{K}$ such that the closure of $\mathbb{K}'$ under subsets equals $\mathbb{K}$*

Proof. "$\Rightarrow$": Let $\mathbb{K}$ be a parameterization over Σ. This means there exists a parameter κ over Σ such that $\mathbb{K} = \mathbb{K}(\kappa)$. Clearly, $\mathbb{K}(\kappa)$ is closed under subsets. Let L, L' be languages over Σ which are both bounded by κ. This means $L \subseteq \kappa_i$ and $L' \subseteq \kappa_j$ for some $i, j \in \mathbb{N}$. We assume w.l.o.g. that $i \leq j$ and therefore $L \cup L' \subseteq \kappa_j$. Therefore $\mathbb{K}(\kappa)$ is closed under union. Since κ is total it follows that $\{w\}$ is in $\mathbb{K}(\kappa)$ for every word w. The countable subset $\mathbb{K}'$ of $\mathbb{K}(\kappa)$ such that the closure of $\mathbb{K}'$ under subsets equals $\mathbb{K}(\kappa)$ is given by $\{\kappa_1, \kappa_2, \dots\}$.

"$\Leftarrow$": Let $\mathbb{K}$ be a set of languages over Σ which satisfies the above three conditions. Observe that the third condition implies that $\mathbb{K}$ is closed under subsets. We construct a parameter κ over Σ such that $\mathbb{K} = \mathbb{K}(\kappa)$. Let $\mathbb{K}' = \{L_1, L_2, \dots\}$ be the countable subset of $\mathbb{K}$ whose closure under subsets equals $\mathbb{K}$. Let $L'_c = \bigcup_{i=1}^{c} L_i$. It holds that L'_c is in $\mathbb{K}$ for every $c \in \mathbb{N}$ because $\mathbb{K}$ is closed under union. Then $\kappa(w)$ being defined as the least k such that $w \in L'_k$ yields the required parameter. □

Therefore it is more accurate to understand a parameterized problem as a tuple $(L, \mathbb{K})$ where L is a language and $\mathbb{K}$ is a parameterization, both over the same alphabet. This alternative view on parameterized problems leads to an interesting different perspective on parameterized complexity which, however, we do not address here. The important observation in our context is the following. If we have a set of graph classes $\mathbb{A}$ which satisfies the conditions of Theorem 4.4 then asking about the parameterized complexity of some problem parameterized by $\mathbb{A}$ is a well-defined question. We show how Theorem 4.4 can be applied to classes such as GP.

Lemma 4.5. *For every countable set of languages A that contains all finite languages and for which GA is closed under union it holds that GA is a parameterization.*

Proof. We show that GA satisfies the three conditions of Theorem 4.4. GA is closed under union by assumption. Furthermore, every singleton graph class lies in GA. This can be shown by using a look-up table of finite size as label decoder. More precisely, for a graph G on n vertices a language L that contains only words of length $2\log n$ can be constructed such that $\{G\} \subseteq \mathrm{gr}(F_L, 1)$. Since A contains all finite languages it contains L as well. The

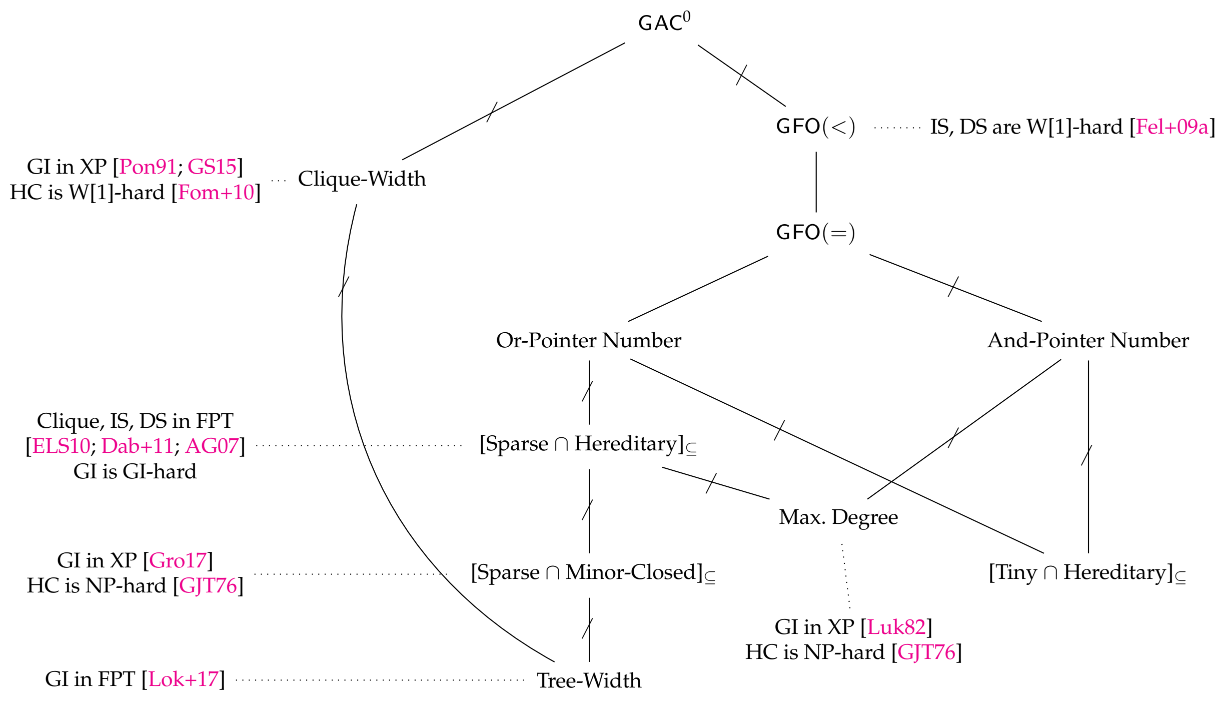


FIGURE 4.2: Parameterized complexity of graph classes with implicit representations

countable subset of GA such that its closure under subsets equals GA is given by the set of graph classes $\mathrm{gr}(S)$ for every labeling scheme S in GA. That this set is countable follows from the fact that there are only countably many labeling schemes S in GA since A is countable. □

Corollary 4.6. $\mathsf{GAC}^0, \mathsf{GL}, \mathsf{GP}, \mathsf{GNP}, \mathsf{GEXP}$ *and* GR *are parameterizations.*

It is also not difficult to show that logical classes such as $\mathsf{GFO}(=)$, $\mathsf{GFO}(<)$ and $\mathsf{GFO}_{\mathrm{qf}}$ are parameterizations. An interesting example of a set of graph classes which is no parameterization is the class GALL. It can be shown that there exists no countable subset of GALL whose closure under subsets equals GALL by a diagonalization argument.

For readers not familiar with parameterized complexity we give a rough description of the complexity classes mentioned in Figure 4.2. A parameterized problem $(L, \mathbb{K})$ is in FPT (fixed-parameter tractable) if there exists a $c \in \mathbb{N}$ such that for all $K \in \mathbb{K}$ it holds that L can be decided in $\mathsf{TIME}(n^c)$ if one only considers inputs from K. The multiplicative constant hidden in the big-oh can depend on K. A parameterized problem $(L, \mathbb{K})$ is in XP if for all $K \in \mathbb{K}$ it holds that L is in P if one only considers inputs from K. In contrast to FPT the degree of the polynomial that bounds the runtime is not fixed but can depend on K. The classes FPT and XP are regarded as the analogon of P and EXP in the parameterized world. If a parameterized problem is W[1]-hard then this can be seen as evidence that it is not in FPT.

In Figure 4.2 we assess the parameterized complexity of graph isomorphism (GI), Hamiltonian cycle (HC), clique, independent set (IS) and dominating set (DS) for structural parameterizations below GAC^0. The last three problems are additionally parameterized by the solution size which is part of the input. Let us give a brief explanation of this figure. The problems IS and DS are W[1]-hard for $\mathsf{GFO}(<)$ because in [Fel+09b] these problem are shown to be already W[1]-hard for unit 2-interval graphs which are contained in $\mathsf{GFO}(<)$. The statement that GI is GI-hard for sparse and hereditary graph classes means that there is a sparse and hereditary graph class for which GI is GI-hard; e.g. the class of graphs with degeneracy at most two as we have shown in the introduction of this chapter. An analogous interpretation is meant for the NP-hardness results of HC. It is interesting to note that in [GJT76] a stronger result is proven than what is shown in this figure: HC is already NP-hard on planar graphs with degree at most three. This implies that even when parameterizing by proper minor-closed graph classes with bounded degree it is hopeless to find XP-algorithms for HC. On the positive side, we found no results that indicate that clique is not in FPT on $\mathsf{GFO}(<)$ and DS or IS are not in FPT on $\mathsf{GFO}(=)$. Is the clique problem in FPT on $\mathsf{GFO}(<)$? Are IS and DS in FPT on $\mathsf{GFO}(=)$? Is GI in XP when parameterized by the and-pointer number? Instead of $\mathsf{GFO}(=)$ and $\mathsf{GFO}(<)$ one can also consider $[\text{Forest}]_{\text{BF}}$ and $[\text{Interval}]_{\text{BF}}$ as parameterizations for which it might be easier to find efficient algorithms. If unit 2-interval graphs are in $[\text{Interval}]_{\text{BF}}$ then IS and DS are W[1]-hard on $[\text{Interval}]_{\text{BF}}$ [Fel+09a].

4.2 The Isomorphism Problem for CA Graphs

Being a generalization of interval graphs—the archetype of geometrical intersection graphs—CA graphs are quite prominent as well and have been known for decades. Since then structural properties and algorithmic problems for this class have been thoroughly investigated with [Gav74] and [Tuc70] being two of the earliest works in this regard. In particular, finding characterizations of CA graphs and constructing a CA representation for a given CA graph have received a great deal of attention. Remarkably, finding a forbidden induced subgraph characterization of CA graphs is still an open problem. See [LS09] for a survey on this line of research and [CGS17] for one of the most recent results in that direction. It should also be mentioned that CA graphs are of practical relevance with applications arising in disciplines such as genetics and operations research. An explanation of the connection between genetics and interval graphs in layman's terms can be found in [WG86]. For a specialized account on this connection emphasizing circularity see [Sta67]. An example of how CA graphs can be used to model the problem of phasing traffic lights is given in [Gol04].

In the following we consider the canonical representation problem for CA graphs. The representation problem for CA graphs is as follows. Given a CA graph G as input we want to output a CA representation ρ_G of G. The canonical variant of this problem imposes the additional requirement that for every pair of isomorphic CA graphs G and H their representations ρ_G and ρ_H should have identical underlying sets of arcs, i.e. $\{\rho_G(v) \mid v \in V(G)\} = \{\rho_H(v) \mid v \in V(H)\}$. Notice that solving the representation problem for CA graphs implies solving the recognition problem for CA graphs, i.e. the question given a graph G is it a CA graph. Likewise, solving the canonical representation problem for CA graphs implies solving the isomorphism problem for CA graphs, i.e. deciding whether two given CA graphs are isomorphic.

Observe that every CA graph is a 2-interval graph because given a set of arcs one can cut the circle at some point and straighten the arcs. It is interesting to note that the isomorphism problem for interval graphs is logspace-complete [Kö+11] while the one for 2-interval graphs is already GI-complete (because every line graph is a 2-interval graph) and CA graphs lie inbetween these two classes.

While a polynomial-time algorithm for deciding isomorphism of interval graphs is known since 1976 due to Booth and Lueker this question still remains open for CA graphs. There have been two claimed polynomial-time algorithms for deciding isomorphism of CA graphs in [Wu83] and [Hsu95] which were shown to be incorrect in [Esc98] and

[Cur+13] respectively. For interval graphs even a linear-time algorithm for isomorphism is known [LB79]. A more recent result is that canonical interval representations for interval graphs can be computed in logspace and that this is optimal in the sense that recognition and deciding isomorphism for interval graphs is logspace-complete [Kö+11]. These two hardness results also carry over to the class of CA graphs. Furthermore, the isomorphism problem for proper CA graphs [KKV16] and Helly CA graphs [KKV13] have been shown to be decidable in logspace. It is also shown how to obtain canonical representations for these subclasses in logspace.

In the following sections we explain how the method used in [KKV13] to obtain canonical representations for Helly CA graphs can be adapted to CA graphs in general. Following this approach, canonical representations for CA graphs can be found by computing certain subsets of vertices called flip sets in an isomorphism-invariant manner. We introduce the class of uniform CA graphs for which this method yields canonical representations in polynomial-time. We then aim to isolate the instances of CA graphs which are difficult to handle with this method. We try to capture these hard instances by what we call restricted CA matrices and show that the canonical representation problem for CA graphs is logspace-reducible to that of restricted CA matrices. During this isolation process we find a subset of uniform CA graphs, namely Δ-uniform CA graphs, for which canonical representations can be computed in logspace. The Δ-uniform CA graphs contain Helly CA graphs and every CA graph without an induced 4-cycle. This generalizes the canonization result for Helly CA graphs given in [KKV13].

4.3 Normalized Representations

When trying to construct a CA representation for a CA graph G it is clear that whenever two vertices are non-adjacent their corresponding arcs must be disjoint in every CA representation of G. For two adjacent vertices the intersection type of their corresponding arcs might depend on the particular CA representation of G that one considers. Hsu has shown that this ambiguity can be removed as follows [Hsu95].

We adopt the notation of [KKV13].

Definition 4.7. *For a graph G we define its neighborhood matrix λ_G which is an intersection matrix as*

$$(\lambda_G)_{u,v} = \begin{cases} \mathtt{di} & \text{, if } \{u,v\} \notin E(G) \\ \mathtt{cd} & \text{, if } N[u] \subsetneq N[v] \\ \mathtt{cs} & \text{, if } N[v] \subsetneq N[u] \\ \mathtt{cc} & \text{, if } N[u] \between N[v] \text{ and } N[u] \cup N[v] = V(G) \\ & \quad \text{and } \forall w \in N[u] \setminus N[v] : N[w] \subset N[u] \\ & \quad \text{and } \forall w \in N[v] \setminus N[u] : N[w] \subset N[v] \\ \mathtt{ov} & \text{, otherwise} \end{cases}$$

for all $u \neq v \in V(G)$.

Let μ be an intersection matrix with vertex set V and let $\rho = (\mathcal{A}, f)$ where $\mathcal{A}$ is a CA model and f is a bijective mapping from V to $\mathcal{A}$. We say ρ is a CA representation of μ if f is an isomorphism from μ to the intersection matrix $\mu_{\mathcal{A}}$ of $\mathcal{A}$. We denote the set of such CA representations for μ with $\mathcal{N}(\mu)$. The representation problem for CA matrices is to compute a CA representation for a given CA matrix μ. The canonical representation problem for CA matrices is defined analogously to the canonical representation problem for CA graphs.

We say ρ is a normalized CA representation for a graph G if ρ is a CA representation for the neighborhood matrix λ_G of G. An example of a normalized representation can be seen in Figure 4.3; every non-labeled edge corresponds to an `ov`-entry in the neighborhood matrix. Let us denote the set of all normalized CA representations for G with $\mathcal{N}(G) = \mathcal{N}(\lambda_G)$.

Lemma 4.8 (Corollary 2.3. [Hsu95]). *Every twin-free CA graph G without a universal vertex has a normalized CA representation, that is $\mathcal{N}(G) \neq \emptyset$.*

Lemma 4.9. *The canonical representation problem for CA graphs is logspace-reducible to the canonical representation problem for vertex-colored twin-free CA graphs without a universal vertex.*

Proof. For a graph G let G_0 denote the induced subgraph of G that is obtained by removing all universal vertices from G and only taking one vertex from each twin-class and deleting the rest. Let c_0 be a coloring of G_0 which assigns each vertex the cardinality of its twin class in G. It holds that (G_0, c_0) and the number of universal vertices in G suffice to reconstruct G. Let G be a CA graph. Compute the graph (G_0, c_0). Since (G_0, c_0) is twin-free and without universal vertices we can compute a canonical representation ρ_0 for it. For a vertex v of G let v_0 denote the twin of v that occurs in G_0. A canonical representation of G is given by $v \mapsto \rho_0(v_0)$ for every non-universal vertex v of G and every universal vertex of G is represented by an arc which covers the whole circle. □

Therefore for our purposes it suffices to consider only twin-free graphs without universal vertices and a vertex-coloring.

From this point on we assume every graph to be twin-free and without a universal vertex unless explicitly stated otherwise. As a consequence we view CA graphs as a subset of CA matrices in the sense that the neighborhood matrix of every CA graph is a CA matrix.

McConnell [McC03] observed that the operation of flipping arcs in CA models has a counterpart in intersection matrices. He called this counterpart operation algebraic flips. Note that for two arcs A, B with intersection type $\alpha \in \{\texttt{cc}, \texttt{cd}, \texttt{cs}, \texttt{di}, \texttt{ov}\}$ the intersection type of $\overline{A}$ and B is solely determined by α. More precisely, the intersection type of $\overline{A}$ and B is $Z_{10}(\alpha)$ where Z_{10} is the function defined in Table 4.1. Similarly, the intersection type of A and $\overline{B}$ is given by $Z_{01}(\alpha)$. Using the functions Z_{ij} we can define the operation of flipping a set of vertices in an intersection matrix.

Definition 4.10. *Let μ be an intersection matrix with vertex set V and $X \subseteq V$. We define the intersection matrix $\mu^{(X)}$ obtained after flipping the vertices X in μ as*

$$\mu^{(X)}_{u,v} = Z_{ij}(\mu_{u,v}) \text{ with } i = 1 \text{ iff } u \in X \text{ and } j = 1 \text{ iff } v \in X$$

for all $u \neq v \in V$.

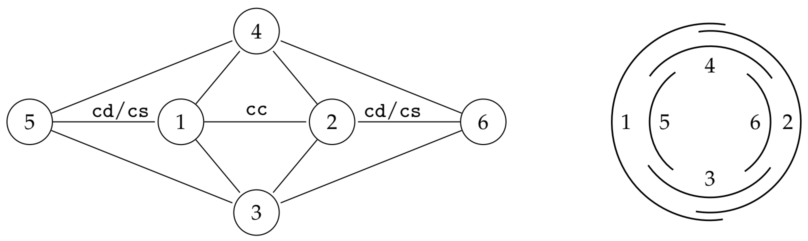


FIGURE 4.3: A CA graph and a normalized representation thereof

Since flipping the same set of arcs twice is an involution it follows that $(\mu^{(X)})^{(X)} = \mu$.

Definition 4.11. *Let V be a set of vertices, let $\mathcal{A}$ be a set of arcs and let ρ be a function that maps V to $\mathcal{A}$. Then $\rho^{(X)} \colon V \to \mathcal{A}$ for $X \subseteq V$ is defined as follows:*

$$\rho^{(X)}(v) = \begin{cases} \overline{\rho(v)} & \text{, if } v \in X \\ \rho(v) & \text{, if } v \notin X \end{cases}$$

Notice that flipping vertices in an intersection matrix is equivalent to flipping arcs in a CA representation in the following sense. Given an intersection matrix λ and a subset of its vertices X. It holds that $\rho \in \mathcal{N}(\lambda) \Leftrightarrow \rho^{(X)} \in \mathcal{N}(\lambda^{(X)})$. Also, it is not difficult to observe that flipping is an isomorphism-invariant operation in the sense that flipping sets of vertices which are in the same orbit lead to isomorphic intersection matrices.

4.4 Flip Trick

In this section we generalize the idea used by Köbler, Kuhnert and Verbitsky in [KKV13] to compute canonical representations for Helly CA graphs. They showed that finding canonical representations for Helly CA graphs can be reduced to finding canonical representations for vertex-colored interval matrices. We show that the idea behind this reduction also works for CA matrices in general. Recall that CA graphs can be seen as special case of CA matrices since the neighborhood matrix of every CA graph is a CA matrix. The converse does not hold, i.e. there exist CA matrices which are not expressible as the neighborhood matrix of a CA graph (for instance any CA matrix with only two vertices that are not disjoint). The key result here, which is used in the subsequent sections, is that finding canonical representations for CA matrices is logspace-reducible to the task of computing what we call an invariant flip set function.

McConnell showed in [McC03] that CA representations for CA graphs can be computed as follows. Given a CA graph G with neighborhood matrix λ one can compute a set of vertices X of G such that $\lambda^{(X)}$ is an interval matrix. We call such a set X a flip set. Then by computing an interval representation ρ for $\lambda^{(X)}$ and flipping back the arcs X in ρ one obtains a CA representation for λ and therefore for G as well [McC03]. We essentially use the same argument to obtain canonical CA representations.

Definition 4.12. *Let λ be a CA matrix. A subset of vertices X of λ is called a flip set if there exists a representation $\rho \in \mathcal{N}(\lambda)$ and a point x on the circle such that $v \in X$ iff $\rho(v)$ contains the point x.*

TABLE 4.1: Algebraic flip functions $Z_{xy} \colon \{\texttt{cc}, \texttt{cd}, \texttt{cs}, \texttt{di}, \texttt{ov}\} \to \{\texttt{cc}, \texttt{cd}, \texttt{cs}, \texttt{di}, \texttt{ov}\}$

$Z_{xy}(\alpha)$	cc	cd	cs	di	ov
Z_{00}	cc	cd	cs	di	ov
Z_{01}	cs	di	cc	cd	ov
Z_{10}	cd	cc	di	cs	ov
Z_{11}	di	cs	cd	cc	ov

The concept of flip sets has already been implicitly defined and used in both [McC03] and [KKV13]. They observed that $\lambda^{(X)}$ is an interval matrix whenever X is a flip set of a CA matrix λ. In fact, the other direction holds as well leading to the following characterization.

Lemma 4.13. *Let λ be a CA matrix and X is a subset of vertices of λ. It holds that X is a flip set iff $\lambda^{(X)}$ is an interval matrix.*

Proof. "$\Rightarrow$": Let X be a flip set of λ. Let $\rho \in \mathcal{N}(\lambda)$ be a witnessing representation of the fact that X is a flip set, i.e. there exists a point x on the circle such that every arc $\rho(v)$ with $v \in X$ contains x and every arc $\rho(v)$ with $v \notin X$ does not contain x. Consider the representation $\rho^{(X)} \in \mathcal{N}(\lambda^{(X)})$. It holds that no arc $\rho^{(X)}(v)$ with $v \in V(\lambda)$ contains the point x which implies that there is a hole in $\rho^{(X)}$ and thus $\lambda^{(X)}$ is an interval matrix.

"$\Leftarrow$": Let X be a subset of vertices of λ such that $\lambda^{(X)}$ is an interval matrix. We argue that X must be a flip set. Let $\rho \in \mathcal{N}(\lambda^{(X)})$ be a CA representation of $\lambda^{(X)}$ containing a hole at point x on the circle. Such a representation must exist since $\lambda^{(X)}$ is an interval matrix. This means the arc $\rho(v)$ does not contain the point x for every vertex $v \in V(\lambda)$. Consider the representation $\rho^{(X)} \in \mathcal{N}((\lambda^{(X)})^{(X)}) = \mathcal{N}(\lambda)$. Then it can be checked that $\rho^{(X)}(v)$ contains the point x iff v is in X and therefore X is a flip set with respect to λ. $\square$

We already mentioned that the canonical representation problem for vertex-colored interval matrices can be solved in logspace due to [KKV13]. However, since the theorem that we reference just states this result for uncolored interval matrices we shortly explain how to modify the proof to incorporate the coloring, which is a straightforward task for anyone familiar with the proof.

Theorem 4.14 ([KKV13, Thm. 5.5])**.** *The canonical representation problem for vertex-colored interval matrices can be solved in logspace.*

Proof. In Theorem 5.5 of [KKV13] it is stated that a canonical interval representation for an interval matrix can be found in logspace. To prove this they convert the input interval matrix λ into a colored tree $\mathbb{T}(\lambda)$ called Δ tree which is a complete invariant for interval matrices. The leafs of this tree correspond to the vertices of λ. By appending the color of a vertex from our vertex-colored interval matrix λ to the existing color of its corresponding leave node in the colored Δ tree $\mathbb{T}(\lambda)$ one obtains a complete invariant for vertex-colored interval matrices. Then by applying the same argument given in the proof of Theorem 5.5 one can also compute a canonical representation for a vertex-colored interval matrix using this slightly modified colored Δ tree. $\square$

A consequence of Lemma 4.13 and Theorem 4.14 is that flip sets can be recognized in logspace. Given an intersection matrix λ and a subset of vertices X of λ it suffices to check whether $\lambda^{(X)}$ is an interval matrix by trying to compute an interval representation.

Definition 4.15. *Let $\mathcal{C}$ be a class of CA matrices and f is a vertex set selector. The function f is called an invariant flip set function for $\mathcal{C}$ if the following conditions hold:*

1. *For every $\lambda \in \mathcal{C}$ there exists an $X \in f(\lambda)$ such that X is a flip set of λ*
2. *f is invariant for $\mathcal{C}$*

Recall that f is globally invariant if f is invariant for all intersection matrices.

Theorem 4.16. *Let $\mathcal{C}$ be a class of CA matrices. The canonical representation problem for vertex-colored $\mathcal{C}$ is logspace-reducible to the problem of computing an invariant flip set function for $\mathcal{C}$.*

Proof. Let f be an invariant flip set function for $\mathcal{C}$. Given a vertex-colored CA matrix (λ, c) with $\lambda \in \mathcal{C}$ a canonical representation can be computed as follows. For every flip set $X \in f(\lambda)$ we associate it with the colored interval matrix $I_X = (\lambda^{(X)}, c_X)$ where $c_X(v) = (c(v), \text{red})$ if v is in X and $(c(v), \text{blue})$ if v is not in X for all $v \in V(\lambda)$. For a colored interval matrix I let $\hat{\rho}_I$ denote a canonical representation of I. Such a canonical representation can be computed in logspace due to Theorem 4.14. Let $\hat{X}$ denote a flip set in $f(\lambda)$ such that the interval model of $\hat{\rho}_{I_{\hat{X}}}$ is lexicographically minimal, i.e. for all flip sets X in $f(\lambda)$ it holds that the model of $\hat{\rho}_{I_X}$ is not smaller than the model of $\hat{\rho}_{I_{\hat{X}}}$. Let $\hat{\rho}$ denote the CA representation that is obtained after flipping the red arcs in $\hat{\rho}_{I_{\hat{X}}}$. Since these are the arcs that were flipped to convert λ into I_X it holds that $\hat{\rho}$ is a representation of λ. To see that this leads to a canonical representation consider two isomorphic vertex-colored CA matrices (λ, c) and (μ, d) with $\lambda, \mu \in \mathcal{C}$ and $V(\lambda)$ and $V(\mu)$ are disjoint. Let $\mathcal{I}_\lambda$ be the set of colored interval matrices I_X for all flip sets $X \in f(\lambda)$, and the set $\mathcal{I}_\mu$ is defined analogously. Let $\mathcal{M}_\lambda$ be the set of interval models M such that there exists an $I \in \mathcal{I}_\lambda$ and M is the model underlying the canonical representation $\hat{\rho}_I$ of I. The set $\mathcal{M}_\mu$ is defined analogously. Since f is invariant it follows that for every $I \in \mathcal{I}_\lambda$ there exists an $I' \in \mathcal{I}_\mu$ such that I and I' are isomorphic, and vice versa. Since the models in $\mathcal{M}_\lambda$ and $\mathcal{M}_\mu$ only depend on the isomorphism type of the matrices in $\mathcal{I}_\lambda$ and $\mathcal{I}_\mu$ it follows that $\mathcal{M}_\lambda = \mathcal{M}_\mu$. The CA models which underlie the canonical representations of λ and μ are both derived from the smallest element in $\mathcal{M}_\lambda = \mathcal{M}_\mu$ and thus are identical. □

Suppose that there is a partition of the set of CA graphs into two classes $\mathcal{C}$ and $\mathcal{D}$ such that you can efficiently compute invariant flip set functions for both classes. One might be misled into thinking that this implies canonical representations for all CA graphs can be found efficiently. However, this is not the case unless the class $\mathcal{C}$ (or $\mathcal{D}$) can be efficiently recognized, or one of the two invariant flip set functions is globally invariant.

Lemma 4.17. *Let $\mathcal{C}$ and $\mathcal{D}$ be classes of CA matrices. The canonical representation problem for $\mathcal{C} \cup \mathcal{D}$ is logspace-reducible to the canonical representation problem for $\mathcal{C}$ and the problem of computing a globally invariant flip set function for $\mathcal{D}$.*

Proof. Let f be a globally invariant flip set function for $\mathcal{D}$. Let $\mathcal{D}'$ be the set of CA matrices λ such that $f(\lambda)$ contains a flip set. Clearly, $\mathcal{D}$ is a subset of $\mathcal{D}'$. It holds that $f(\lambda)$ contains a flip set iff $\lambda \in \mathcal{D}'$. Since f is globally invariant it follows that f is an invariant flip set function for $\mathcal{D}'$. To obtain a canonical representation for a matrix $\lambda \in \mathcal{C} \cup \mathcal{D}$ first compute $f(\lambda)$. If $f(\lambda)$ contains a flip set it holds that $\lambda \in \mathcal{D}'$ and therefore the output of f can be used to find a canonical representation for λ. If $f(\lambda)$ contains no flip set it must be the case that $\lambda \in \mathcal{C}$ and therefore the canonization algorithm for $\mathcal{C}$ can be applied. □

We conclude this section by restating the invariant flip set function that was used in [KKV13] to compute canonical representations for Helly CA graphs and explain why it is correct:

$$f_{\text{HCA}}(G) = \{N[u] \cap N[v] \mid u, v \in V(G)\}$$

In a Helly CA graph G every inclusion-maximal clique C of G is a flip set. To see why this holds let ρ be a representation of G with the Helly property. Since C is a clique this means every pair of arcs $\rho(u)$ and $\rho(v)$ with $u, v \in C$ intersects. By the Helly property it follows that the overall intersection $\bigcap_{v \in C} \rho(v)$ is non-empty. This means there exists a point x on the circle such that every arc $\rho(v)$ with $v \in C$ contains x. Assume there exists a vertex $w \in V(G) \setminus C$ such that $\rho(w)$ contains x. This means w must be adjacent to every vertex in C, which contradicts that C is inclusion-maximal. Hence C is a flip set.

In [KKV13, Thm. 3.2] it is shown that every Helly CA graph contains at least one inclusion-maximal clique which can be expressed as the common neighborhood of two vertices. Therefore $f_{\mathrm{HCA}}(G)$ returns at least one flip set for every Helly CA graph G. Also, it is trivial to see that f_{HCA} is globally invariant.

4.5 Uniform CA Graphs

We define the class of uniform CA graphs for which computing a particular invariant flip set function reduces to computing a representation. As a consequence, canonical representations for this class of CA graphs can be computed in polynomial-time. This is an interesting class for two reasons. First, it seems to capture the instances where it is easy to apply the flip trick. Secondly, its complement (within the CA graphs) is a rather exotic class of CA graphs with a quite particular structure. While the initial definition of uniformity makes it apparent why it suffices to find an arbitrary representation in order to obtain a canonical one, it is rather impractical when trying to understand what constitutes a uniform CA graph. We provide a more pleasant characterization of uniform CA graphs in terms of how certain triangles in a CA graph can be represented. This alternative characterization also reveals that every Helly CA graph is uniform. Additionally, we show that the canonical representation problem for uniform CA graphs is logspace-equivalent to what we call the non-Helly triangle representability problem. This problem is: given a CA graph G and a set T of three pairwise overlapping vertices as input, does there exist a representation ρ of G such that T covers the whole circle in ρ?

The following kind of flip set will lead us to uniform CA graphs when trying to compute canonical representations. Given a CA matrix λ recall that X is a flip set of λ if there exists a representation $\rho \in \mathcal{N}(\lambda)$ and a point x on the circle such that $x \in \rho(v)$ iff $v \in X$ for all vertices v of λ. We impose the additional restriction that x is not allowed to be an arbitrary point on the circle but instead has to be one of the endpoints in ρ.

Definition 4.18. *Let λ be a CA matrix and $u \in V(\lambda)$. A flip set X of λ is a u-flip set if there exists a representation $\rho \in \mathcal{N}(\lambda)$ and an endpoint x of $\rho(u)$ such that $v \in X$ iff $\rho(v)$ contains the point x.*

Clearly, every CA graph has a u-flip set for every vertex u. On the other hand, there are CA graphs that have flip sets which are not u-flip sets for any vertex u. For example, consider the cycle graph with $n \geq 4$ vertices. Every flip set that consists of exactly one vertex is not a u-flip set for any vertex u of the cycle graph.

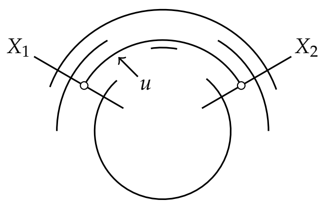


FIGURE 4.4: Exemplary u-flip sets X_1 and X_2

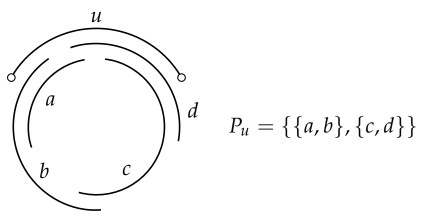


FIGURE 4.5: Example of a u-overlap partition P_u

Consider the following task: given a CA graph G and a vertex u, find a u-flip set of G. Clearly, no vertex v which is disjoint from u or contained by u belongs to X since in every representation the arc of v does not contain any of the two endpoints of the arc of u. Similarly, if a vertex v contains u or forms a circle cover with u then in every representation the arc of v contains both endpoints of u and therefore must be included in X. See Figure 4.4 for a schematic overview.

It remains to decide for the set of vertices $N^{\text{ov}}(u)$ that overlap with u whether they should be included in X. A vertex v which overlaps with u contains exactly one of the endpoints of u in any representation. Let x, y be two vertices that overlap with u. We say x and y overlap from the same side with u in ρ if $\rho(x)$ and $\rho(y)$ contain the same endpoint of $\rho(u)$. Evidently, this is an equivalence relation with respect to v and ρ which partitions $N^{\text{ov}}(u)$ into two parts, namely the part which contains the left endpoint and the one which contains the right endpoint. If X is a u-flip set then $X \cap N^{\text{ov}}(u)$ must be an equivalence class of the 'overlap from the same side with u in ρ'-relation for some $\rho \in \mathcal{N}(G)$.

Definition 4.19. *For a CA matrix λ and a vertex u of λ we say a partition Y of $N^{\text{ov}}(u)$ into two parts is a u-ov-partition if there exists a representation $\rho \in \mathcal{N}(\lambda)$ such that two vertices $x, y \in N^{\text{ov}}(u)$ are in the same part of Y iff $\rho(x)$ and $\rho(y)$ overlap from the same side with $\rho(u)$. We say ov-partition to mean an u-ov-partition for an arbitrary $u \in V(\lambda)$.*

In general, for a vertex u of a CA graph G there can be multiple u-ov-partitions. In fact, there are instances with exponentially many u-ov-partitions with respect to $|N^{\text{ov}}(u)|$. A trivial way of obtaining at least one u-ov-partition for every vertex u of a CA graph G is to compute an arbitrary representation $\rho \in \mathcal{N}(G)$. But the ov-partitions obtained by this method are not invariant and thus do not yield canonical representations. However, if one considers CA graphs where there is only one u-ov-partition for every vertex u then an arbitrary representation suffices.

Definition 4.20 (Uniform CA Graphs)**.** *A CA graph G is uniform if for every vertex u in G there exists exactly one u-ov-partition. This partition is denoted by $P_u = \{P_{u,1}, P_{u,2}\}$.*

Lemma 4.21. *The following mapping is an invariant flip set function for uniform CA graphs. Let G be a uniform CA graph.*

$$F_{\text{uniform}}(G) = \bigcup_{\substack{u \in V(G) \\ i \in \{1,2\}}} \{\{u\} \cup N^{\text{cd}}(u) \cup N^{\text{cc}}(u) \cup P_{u,i}\}$$

Proof. Let G be a uniform CA graph and X is in $F_{\text{uniform}}(G)$ with $X = \{u\} \cup N^{\text{cd}}(u) \cup N^{\text{cc}}(u) \cup P_{u,i}$ for some $u \in V(G)$ and $i \in \{1, 2\}$. It follows from Figure 4.4 and the definition of ov-partitions that X is a u-flip set. The invariance of $F_{\text{uniform}}(G)$ follows from the fact that the intersection type of two vertices as well as the property of being an ov-partition is independent of the vertex labels. □

We remark that the function F_{uniform} is undefined for non-uniform CA graphs since the sets $P_{u,1}$ and $P_{u,2}$ are not well-defined in that context.

Theorem 4.22. *Canonical representations for uniform CA graphs can be computed in polynomial-time.*

Proof. Let G be a uniform CA graph. Compute a normalized representation ρ of G and extract the u-ov-partition for each vertex u from ρ. Then compute $F_{\text{uniform}}(G)$ from

Lemma 4.21 to obtain a canonical CA representation for G. Since CA representations can be computed in polynomial-time (see for instance [McC03]) it follows that this procedure also works in polynomial-time. □

Considering that our definition of uniform CA graphs arose from the desire to compute invariant u-flip sets one might expect that these graphs are only a small special case of CA graphs. Surprisingly, quite the opposite is the case as we will see. We give an alternative definition of uniform CA graphs which gives a better intuition as to why many CA graphs are uniform.

Definition 4.23. *Let λ be a CA matrix. An* ov*-triangle T of λ is a set of three vertices that overlap pairwise, i.e. for all $u \neq v$ in T it holds that u* ov *v. An* ov*-triangle T is representable as non-Helly triangle (interval triangle) if there exists a representation $\rho \in \mathcal{N}(\lambda)$ such that the set of arcs $\{\rho(x) \mid x \in T\}$ does (not) cover the whole circle. Let $\mathcal{T}_{\text{NHT}}(\lambda)$ and $\mathcal{T}_{\text{IT}}(\lambda)$ denote the sets of* ov*-triangles representable as non-Helly triangles and interval triangles respectively.*

This definition also applies to CA graphs via their neighborhood matrix, i.e. $\mathcal{T}_{\text{IT}}(G) = \mathcal{T}_{\text{IT}}(\lambda)$ and $\mathcal{T}_{\text{NHT}}(G) = \mathcal{T}_{\text{NHT}}(\lambda)$ where λ is the neighborhood matrix of G. See Figure 4.6 for an example where the vertices u, x, z are represented as non-Helly triangle on the left and interval triangle on the right.

Recall that a set of arcs which intersect pairwise but have overall empty intersection is called non-Helly. Since three pairwise overlapping arcs that cover the whole circle have overall empty intersection we call such a set a non-Helly triangle. In fact, one can verify that this is the only non-Helly arrangement of three arcs. A complete list of inclusion-minimal non-Helly CA models can be found in [Joe+11, Corrollary 3.1].

Theorem 4.24. *A CA graph G is uniform iff $\mathcal{T}_{\text{IT}}(G) \cap \mathcal{T}_{\text{NHT}}(G) = \emptyset$.*

Proof. "$\Rightarrow$": Assume there exists a uniform CA graph G with $\mathcal{T}_{\text{IT}}(G) \cap \mathcal{T}_{\text{NHT}}(G) \neq \emptyset$. Let T be an ov-triangle in $\mathcal{T}_{\text{IT}}(G) \cap \mathcal{T}_{\text{NHT}}(G)$ and $T = \{x, y, z\}$. This means there exist two representations $\rho_I, \rho_N \in \mathcal{N}(G)$ such that T is represented as interval triangle in ρ_I and as non-Helly triangle in ρ_N. We assume w.l.o.g. that $\rho_I(y) \subset \rho_I(x) \cup \rho_I(z)$, i.e. y is placed in-between x and z in ρ_I. This means y and z must be in the same part of the unique x-ov-partition P_x. However, y and z do not contain the same endpoint of x in the representation ρ_N, which contradicts that G is uniform.

"$\Leftarrow$": Assume there exists a CA graph G with $\mathcal{T}_{\text{IT}}(G) \cap \mathcal{T}_{\text{NHT}}(G) = \emptyset$ that is not uniform. This means there exist a vertex u, two vertices $x, y \in N^{\text{ov}}(u)$ and two representations $\rho, \rho' \in \mathcal{N}(G)$ such that x and y overlap from the same side with u in ρ but not in ρ'. This implies that x and y must overlap and therefore $T = \{u, x, y\}$ is an ov-triangle. Notice that T must be represented as interval triangle in ρ because x and y both contain the same endpoint of u. It holds that T is represented as interval triangle in ρ' as well since otherwise $T \in \mathcal{T}_{\text{IT}}(G) \cap \mathcal{T}_{\text{NHT}}(G)$. Also, we assume w.l.o.g. that $\rho(y) \subset \rho(x) \cup \rho(u)$. Since u and y overlap it holds that $N[u] \setminus N[y] \neq \emptyset$. Due to ρ' it follows that $N[u] \setminus N[y] \subseteq N[u] \cap N[x]$. For a vertex $z \in N[u] \setminus N[y]$ to intersect with both u and x it is necessary that z overlaps

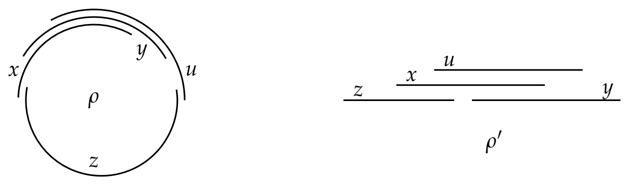


FIGURE 4.6: "$\Leftarrow$"-direction in the proof of Theorem 4.24

with u and x due to the representation ρ. It follows that $\{u,x,z\}$ is represented as non-Helly triangle in ρ. On the other hand, $\{u,x,z\}$ must be represented as interval triangle in ρ' and therefore $\mathcal{T}_{\mathrm{IT}}(G) \cap \mathcal{T}_{\mathrm{NHT}}(G) \neq \emptyset$, contradiction. See Figure 4.6 for a schematic overview of ρ and ρ'. □

Observe that if an ov-triangle T of G is representable as non-Helly triangle then this implies that T must have certain structural properties in G. For example, every vertex of G must be adjacent to at least one of the vertices in T since T covers the whole circle in some representation. Similarly, if T is representable as interval triangle this also implies some structural properties. For instance, there must be an $x \in T$ such that every vertex that is adjacent to x must also be adjacent to at least one other vertex in T. If an ov-triangle is representable as both non-Helly triangle and interval triangle then it must satisfy all of these structural properties at once. As a consequence such an ov-triangle must have a very particular structure which extends to the whole graph as we will see in the next section.

A CA graph is Helly if it has a Helly CA representation. In [Joe+11, Theorem 4.1] it is shown that every 'stable' representation of a Helly CA graph is Helly. Since every normalized representation has the 'stable' property it follows that a CA graph is Helly iff every normalized representation of it is Helly. If a CA graph G is Helly this implies that $\mathcal{T}_{\mathrm{NHT}}(G)$ is empty, and therefore every Helly CA graph is uniform.

A natural question to consider is the computational complexity of deciding whether an ov-triangle is representable as non-Helly triangle or interval triangle. Given a CA graph G and an ov-triangle T of G let us call the problem of deciding whether T is in $\mathcal{T}_{\mathrm{NHT}}(G)$ the non-Helly triangle representability problem. Analogously, deciding whether T is in $\mathcal{T}_{\mathrm{IT}}(G)$ is called the interval triangle representability problem. In the case of uniform CA graphs these two problems are complementary, i.e. an ov-triangle T is in $\mathcal{T}_{\mathrm{NHT}}(G)$ iff T is not in $\mathcal{T}_{\mathrm{IT}}(G)$. In the following, we show that solving either of these two problems for uniform CA graphs is logspace-equivalent to computing a canonical representation for uniform CA graphs.

Definition 4.25. *Let G be a CA graph and $T = \{u,v,w\}$ is an* ov*-triangle of G. We say v is amidst u and w if one of the following conditions holds:*

1. *$N_T(u)$ and $N_T(w)$ are non-empty*
2. *there exists a $z \in N_T(u,w)$ such that $\{u,w,z\} \in \mathcal{T}_{\mathrm{NHT}}(G)$*

Lemma 4.26. *Let G be a uniform CA graph and $T = \{u,v,w\}$ is an* ov*-triangle of G with $T \notin \mathcal{T}_{\mathrm{NHT}}(G)$. Then the following statements are equivalent:*

1. *v is amidst u and w*
2. *$\exists \rho \in \mathcal{N}(G) : \rho(v) \subset \rho(u) \cup \rho(w)$*
3. *$\forall \rho \in \mathcal{N}(G) : \rho(v) \subset \rho(u) \cup \rho(w)$*

Proof. "2 ⇒ 1": Let ρ be in $\mathcal{N}(G)$ such that $\rho(v) \subset \rho(u) \cup \rho(w)$ and assume that v is not amidst u, w. Since v overlaps with u and w it holds that $N[u] \setminus N[v]$ and $N[w] \setminus N[v]$ are non-empty. Because $N_T(u) = N_T(w) = \emptyset$ it must hold that $N_T(u,w) \neq \emptyset$. Let $z \in N_T(u,w)$. For z to intersect with u and w in ρ it must hold that $\{u,w,z\}$ is represented as non-Helly triangle in ρ. This contradicts the assumption that v is not amidst u, w.

"1 ⇒ 3": Let v be amidst u and w and assume that there exists a $\rho \in \mathcal{N}(G)$ such that $\rho(v) \not\subset \rho(u) \cup \rho(w)$. Since $T \notin \mathcal{T}_{\mathrm{NHT}}(G)$ and G is uniform it follows by Theorem 4.24

that T must be represented as interval triangle in every representation, which includes ρ. We assume w.l.o.g. that $\rho(w) \subset \rho(u) \cup \rho(v)$. From that it follows that $N_T(w)$ is empty and therefore there must be a $z \in N_T(u,w)$ such that $\{u,w,z\}$ is a non-Helly triangle in ρ, which is impossible.

"3 ⇒ 2": clear. □

Definition 4.27. *Let G be a CA graph and $u \in V(G)$. Let the binary relation $\sim_u$ on $N^{\mathtt{ov}}(u)$ be defined such that $x \sim_u y$ holds if one of the following holds:*

1. *$x = y$*
2. *$x \,\mathtt{cd}\, y$ or $x \,\mathtt{cs}\, y$*
3. *$x \,\mathtt{ov}\, y$, $\{u,x,y\} \notin \mathcal{T}_{\mathrm{NHT}}(G)$ and u is not amidst x and y*

Lemma 4.28. *For every uniform CA graph G and $u \in V(G)$ it holds that the partition induced by $\sim_u$ equals the unique u-$\mathtt{ov}$-partition P_u. Stated differently, $x \sim_u y$ iff x and y are in the same part of P_u.*

Proof. "⇒": Let $x \sim_u y$ and assume for the sake of contradiction that x and y are not in the same part of the u-$\mathtt{ov}$-partition. This means there exists a representation $\rho \in \mathcal{N}(G)$ such that $\rho(x)$ and $\rho(y)$ contain different endpoints of $\rho(u)$. This is only possible if x and y overlap. Since $\{u,x,y\} \notin \mathcal{T}_{\mathrm{NHT}}(G)$ this means $\{u,x,y\}$ must be represented as interval triangle in ρ. In order for $\rho(x)$ and $\rho(y)$ to contain different endpoints of $\rho(u)$ it must hold that $\rho(u) \subset \rho(x) \cup \rho(y)$, which implies that u is amidst x and y by Lemma 4.26. This contradicts $x \sim_u y$.

"⇐": Let x and y be in the same part of the u-$\mathtt{ov}$-partition and assume that $x \sim_u y$ does not hold. This implies that x and y must overlap and therefore $\{u,x,y\}$ form an $\mathtt{ov}$-triangle. For $x \sim_u y$ to not hold it must be either the case that $\{u,x,y\}$ is only representable as non-Helly triangle or u is amidst x and y. In both cases this contradicts x and y being in the same part of the u-$\mathtt{ov}$-partition. □

Theorem 4.29. *The representation, canonical representation, non-Helly triangle representability and interval triangle representability problem for uniform CA graphs are logspace-equivalent.*

Proof. The non-Helly triangle representability and interval triangle representability problem for uniform CA graphs are logspace-equivalent because they are complementary in the sense that an $\mathtt{ov}$-triangle is representable as non-Helly triangle iff it is not representable as interval triangle. This follows from the fact that an $\mathtt{ov}$-triangle can only be either represented as non-Helly triangle or interval triangle and these two possibilities are mutually exclusive in the case of uniform CA graphs. As a consequence these two problems are trivially reducible to the representation problem for uniform CA graphs. Given a uniform CA graph G, an $\mathtt{ov}$-triangle T of G and a representation $\rho \in \mathcal{N}(G)$ it holds that $T \in \mathcal{T}_{\mathrm{NHT}}(G)$ iff $T \notin \mathcal{T}_{\mathrm{IT}}(G)$ iff T is represented as non-Helly triangle in ρ.

The representation problem is obviously reducible to the canonical representation problem. Therefore it remains to show that the canonical representation problem for uniform CA graphs is reducible to the non-Helly triangle representability problem. To obtain a canonical representation for a uniform CA graph we can use the invariant flip set function given in Lemma 4.21. To compute this function we need to figure out the unique $\mathtt{ov}$-partitions for each vertex. By Lemma 4.28 this can be done by computing the equivalence relation $\sim_u$ for each vertex u. It can be verified that this relation is computable in logspace using queries of the form $T \in \mathcal{T}_{\mathrm{NHT}}(G)$. □

The isomorphism problem for CA graphs can be reduced to the one for non-uniform CA graphs in polynomial-time due to Theorem 4.22. However, a reduction from the canonical representation problem for CA graphs to the one for non-uniform CA graphs does not immediately follow from Theorem 4.22 unless uniform CA graphs can be recognized in polynomial-time. An alternative approach to construct such a reduction is to solve the non-Helly triangle representability problem for uniform CA graphs with an additional requirement.

Definition 4.30. *The globally invariant non-Helly triangle representability problem for uniform CA graphs is defined as follows. Let A be an algorithm that correctly decides the non-Helly triangle representability problem for uniform CA graphs. Let f_A be the function computed by A, i.e. for a graph G and an* ov*-triangle T of G it holds that $f_A(G,T) = 1$ iff A accepts (G,T). We say A decides the globally invariant non-Helly triangle representability problem for uniform CA graphs if f_A is an invariant* for all graphs. *Stated differently, the output of A must be independent of the vertex labels.*

Lemma 4.31. *The canonical representation problem for CA graphs is logspace-reducible to the globally invariant non-Helly triangle representability problem for uniform CA graphs and the canonical representation problem for vertex-colored non-uniform CA graphs.*

Proof. Suppose we are given an algorithm A which solves the globally invariant non-Helly triangle representability problem for uniform CA graphs. We argue that A can be used to compute a globally invariant flip set function for uniform CA graphs. From Lemma 4.17 it then follows that the canonical representation problem for CA graphs reduces to that for vertex-colored non-uniform CA graphs.

Given a CA graph G let $\Delta(G,A)$ be the set of ov-triangles T of G such that A accepts (G,T). If G is a uniform CA graph then $\Delta(G,A) = \mathcal{T}_{\text{NHT}}(G)$. Consider Definition 4.25 and 4.27 and suppose that each occurrence of $\mathcal{T}_{\text{NHT}}(G)$ is replaced by $\Delta(G,A)$. Let us call the new relation $\sim_u^A$. Clearly, in the case of uniform CA graphs $\sim_u$ and $\sim_u^A$ coincide. Next, consider the following variant of F_{uniform}:

$$F^A_{\text{uniform}}(G) = \bigcup_{\substack{u \in V(G) \\ X \in (N^{\text{ov}}(u)/\sim_u^A)}} \{\{u\} \cup N^{\text{cd}}(u) \cup N^{\text{cc}}(u) \cup X\}$$

where $(N^{\text{ov}}(u)/\sim_u^A)$ denotes the equivalence classes of $\sim_u^A$. If $\sim_u^A$ is not an equivalence relation let $(N^{\text{ov}}(u)/\sim_u^A) = \emptyset$. If G is a uniform CA graph then it follows from Lemma 4.28 that $F_{\text{uniform}}(G) = F^A_{\text{uniform}}(G)$. Therefore F^A_{uniform} is an invariant flip set function for uniform CA graphs. Additionally, it can be verified that F^A_{uniform} is globally invariant due to the fact that the answer of A is independent of the vertex labels. Also, the function F^A_{uniform} can be computed in logspace using queries of the form $T \in \Delta(G,A)$. Observe that $\Delta(G,A)$ only provides n^3 bits of information with $n = |V(G)|$ and therefore can be computed 'in a single query' by a functional oracle which outputs the n^3 bits of information. □

4.6 Non-Uniform CA Graphs and Restricted CA Matrices

In the first part of this section we examine the structure of non-uniform CA graphs. Every such graph must have two ov-triangles which have exactly one vertex in common and both are representable as interval triangle and as non-Helly triangle. This pair of ov-triangles enforces a particular structure in non-uniform CA graphs. In the second part we introduce restricted CA matrices, which try to partly capture this structure. Roughly speaking, restricted CA matrices can be seen as a generalization of the neighborhood matrices of non-uniform CA graphs. We pay the price of considering this more general class of structures in order to provide a logspace reduction from the canonical representation problem for CA graphs to that of restricted CA matrices.

Definition 4.32. *Given a CA graph G, an induced 4-cycle $C = (u, w, w', u')$ of G and $v \in V(G) \setminus C$. We say (C, v) is a non-uniformity witness of G if $\{u, v, w\}, \{u', v, w'\} \in \mathcal{T}_{\mathrm{IT}}(G) \cap \mathcal{T}_{\mathrm{NHT}}(G)$. We also simply call (C, v) a witness of G.*

Theorem 4.33. *A CA graph G is non-uniform iff G has a non-uniformity witness.*

Proof. "$\Rightarrow$": Let G be a non-uniform CA graph. Due to Theorem 4.24 there exists an ov-triangle T of G with $T \in \mathcal{T}_{\mathrm{IT}}(G) \cap \mathcal{T}_{\mathrm{NHT}}(G)$. Let $T = \{u, v, w\}$ and $\rho_{\mathrm{I}} \in \mathcal{N}(G)$ such that v is in-between u and w, i.e. $\rho_{\mathrm{I}}(v) \subset \rho_{\mathrm{I}}(u) \cup \rho_{\mathrm{I}}(w)$. First, we show that there exists an induced 4-cycle $C = (u, w, w', u')$ in G.

From the non-Helly triangle representation of T it follows that $N[u] \cup N[v] \cup N[w] = V(G)$. Since v is in-between u and w this means $N[u] \cup N[w] = V(G)$. It holds that u and w overlap. Therefore one of the conditions in the definition of the neighborhood matrix for u and w to form a circle cover must be violated. Let us assume w.l.o.g. that the violated condition is that there exists a $u' \in N[u] \setminus N[w]$ such that $N[u'] \not\subseteq N[u]$. This means u' must overlap with u and there exists a $w' \in N[u'] \setminus N[u]$. Since $w' \notin N[u]$ it follows from $N[u] \cup N[w] = V(G)$ that $w' \in N[w]$ and because w is disjoint from u', and because w' intersects with both u' and w it follows that w' overlaps with u' and w. Therefore $C = (u, w, w', u')$ is an induced 4-cycle in G.

It remains to show that $\{u', v, w'\}$ is an ov-triangle and that it is in both $\mathcal{T}_{\mathrm{IT}}(G)$ and $\mathcal{T}_{\mathrm{NHT}}(G)$. Consider the representation ρ_{I} from before. Assume for the sake of contradiction that v does not overlap with u'. Then due to ρ_{I} it must be the case that u' is disjoint from v and thus $u' \in N_T(u)$. However, due to fact that T is representable as non-Helly triangle this would imply that u' is contained by u, which is not the case. Therefore u' overlaps with v as the other intersections types are out of question. For the same reason w' overlaps with v and hence $T' = \{u', v, w'\}$ is an ov-triangle. Now, it can be verified that in every representation of G where T is a non-Helly triangle it follows that T' must be an interval triangle and vice versa. This concludes that T' is in $\mathcal{T}_{\mathrm{IT}}(G) \cap \mathcal{T}_{\mathrm{NHT}}(G)$.

"$\Leftarrow$": Follows directly from Theorem 4.24. □

In Figure 4.7 five non-uniform CA graphs and one uniform CA graph (X_4) are given by their CA models. We explain how to verify this claim. First, we have to check that every CA model is normalized. This means the graphs which are induced by these models must be twin-free and without a universal vertex. Additionally, the intersection types of the arcs must match the intersection types in the induced graph (or more precisely its neighborhood matrix). A quick way to determine whether two overlapping arcs also overlap in the graph is to check if they jointly occur in an induced n-cycle for some $n \geq 4$.

To see that the first five CA graphs are non-uniform we have to find an ov-triangle that is representable as both interval and non-Helly triangle. In the case of $\overline{3K_2}$ this ov-triangle can be $\{u, v, w\}$. In the given representation $\{u, v, w\}$ is represented as interval triangle. Observe that v and v' are in the same orbit and therefore the labels v and v' can be swapped in the representation. After swapping v and v' the ov-triangle $\{u, v, w\}$ is represented as non-Helly triangle. For the graph X_0 we can also choose the ov-triangle $\{u, v, w\}$. In this case there is an automorphism which swaps u with u' and w with w' and has the other vertices as fix-points. After changing the labels in the representation according to this

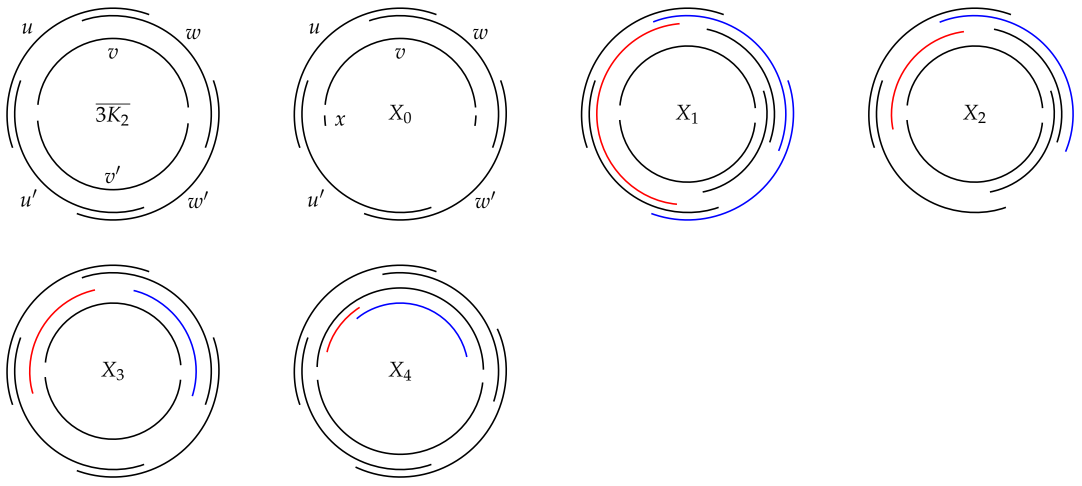


FIGURE 4.7: Examples of non-uniform CA graphs and one uniform CA graph X_4

automorphism it holds that $\{u, v, w\}$ is represented as non-Helly triangle. We remark that $\overline{3K_2}$ and X_0 are minimal in the sense that no induced subgraph of them is a non-uniform CA graph. Next, let us consider the graphs X_1 to X_3. Observe that the black arcs in each of these graphs form an induced $\overline{3K_2}$ subgraph. We assume that the black arcs are labeled with u, u', v, v', w, w' in the same way that the representation of $\overline{3K_2}$ is labeled. It holds that v and v' are in the same orbit in all of these four graphs because they have the same open neighborhood. Therefore $\{u, v, w\}$ is representable as both interval and non-Helly triangle due to the same argument that we made for $\overline{3K_2}$.

To show that X_4 is uniform we argue that it has a unique normalized representation, i.e. $|\mathcal{N}(X_4)| = 1$. Observe that this graph has a unique CA model. Additionally, it has no non-trivial automorphism (it is rigid). Therefore X_4 has a unique CA representation.

Fact 4.34. *Every non-uniform CA graph contains $\overline{3K_2}$ or X_0 as induced subgraph.*

Proof. Let G be a non-uniform CA graph. Due to Theorem 4.33 there exists a witness (C, v) of G with $C = (u, w, w', u')$. Since G does not contain a universal vertex it holds that $V(G) \setminus N[v]$ is non-empty. Due to the fact that $\{u, v, w\}$ and $\{u', v, w'\}$ can be represented as interval triangles it follows that $N_C(C \setminus \{x\}) \subseteq N[v]$ for all $x \in C$. Therefore $V(G) \setminus N[v] \subseteq N_C(C) \cup N_C(u, u') \cup N_C(w, w')$. Suppose there is a $v' \in N_C(C) \setminus N[v]$. Then the vertices of C along with v and v' form an induced $\overline{3K_2}$-subgraph of G. Assume that this is not the case, i.e. $N_C(C) \subseteq N[v]$. Since u and v overlap it must hold that $N[u] \setminus N[v] \neq \emptyset$. The only vertices that can be adjacent to $N[u]$ but not to $N[v]$ must be in $N_C(u, u')$ since $N_C(C) \subseteq N[v]$. Therefore there exists a vertex $x \in N_C(u, u')$ that is not adjacent to v. For the same reason there must be a vertex $y \in N_C(w, w')$ not adjacent to v because $N[w] \setminus N[v] \neq \emptyset$. The vertices of C along with v, x and y form an induced X_0-subgraph. □

Definition 4.35 (Restricted CA Matrix). *Let λ be a CA matrix. We say λ is a restricted CA matrix if it contains an induced 4-cycle $C = (u, w, w', u')$ called witness cycle such that:*

1. *$N_C(u, w)$, $N_C(u', w')$ and $N_C(x)$ are empty for every $x \in C$*
2. *For all $x \in N_C(C)$ it holds that x overlaps with all vertices in C*

	1				2				3		4		5		6		7
u	cs	ov	cs	ov	di	di	di	di	ov	ov	ov	cs	ov	ov	di	di	ov
w	di	di	di	di	cs	ov	cs	ov	di	di	ov	ov	ov	cs	ov	ov	ov
w'	di	di	di	di	cs	cs	ov	ov	ov	ov	di	di	ov	ov	ov	cs	ov
u'	cs	cs	ov	ov	di	di	di	di	ov	cs	ov	ov	di	di	ov	ov	ov

TABLE 4.2: Intersection types of restricted CA matrices with witness cycle (u, w, w', u)

Observe that the intersection matrix of every CA model that is shown in Figure 4.7 is a restricted CA matrix.

Fact 4.36. *Given an intersection matrix λ, vertices $x, y_1, \dots, y_k$ of λ and intersection types $\alpha_1, \dots, \alpha_k$, we say x is an $(\alpha_1, \dots, \alpha_k)$-neighbor of $(y_1, \dots, y_k)$ if $\lambda_{x,y_i} = \alpha_i$ for all $i \in [k]$. A CA matrix λ is restricted iff λ contains an induced 4-cycle $C = (u, w, w', u')$ such that for all vertices $x \in V(\lambda) \setminus C$ there exists a column $\overline{\alpha}$ in Table 4.2 such that x is a $\overline{\alpha}$-neighbor of C.*

Proof. We use the numbers in the table headline to refer to the different columns. For example, 2.3 refers to the third column from left in the second part of the table: $(\texttt{di}, \texttt{cs}, \texttt{ov}, \texttt{di})$.

"$\Rightarrow$": Let λ be a restricted CA matrix with witness cycle $C = (u, w, w', u')$. We need to show for every $x \in V(\lambda) \setminus C$ there exists a column $\overline{\alpha}$ in Table 4.2 such that x is a $\overline{\alpha}$-neighbor of C. Due to the definition of restricted CA matrices it must hold that x is in (exactly) one of the following seven sets: $N_C(C), N_C(u, u'), N_C(w, w')$ or $N_C(C \setminus \{z\})$ for a $z \in C$. If x is in $N_C(C)$ then x overlaps with every vertex of C by definition. This corresponds to the last column 7.1 of the table. If $x \in N_C(u, u')$ then x is disjoint from w and w'. In that case x is an $\overline{\alpha}$-neighbor of C where $\overline{\alpha}$ must be one of the four columns in part one of the table. For the same reason if $x \in N_C(w, w')$ then it is an $\overline{\alpha}$-neighbor of C where $\overline{\alpha}$ corresponds to one of the two columns in the second part of the table. If x is in $N_C(C \setminus \{w\})$ then x is disjoint from w and x overlaps with both u and w'. The intersection type between x and u' can be one of the following: x overlaps with u or x is contained by u or x contains u. The first two cases are covered by the third part of the table. However, if x contains u then there exists no corresponding column in the table since it does not have any cd-entries. This can be resolved by using the following observation: if x is in $N_C(C \setminus \{w\})$ and contains u' then (u, w, w', x) is a witness cycle of λ as well. As a consequence we can assume without loss of generality that a witness cycle C of λ can be chosen such that there exists no $x \in N_C(C \setminus \{w\})$ which contains u'. The same argument applies to the remaining three cases $x \in N_C(C \setminus \{z\})$ with $z \in \{u, u', w'\}$.

"$\Leftarrow$": clear. □

In the remainder of this section we prove that the canonical representation problem for CA graphs is logspace-reducible to the canonical representation problem for vertex-colored restricted CA matrices. The proof outline looks as follows. First, we define a subset of uniform CA graphs, namely Δ-uniform CA graphs, for which the globally invariant non-Helly triangle representability problem can be solved in logspace. Therefore the canonical representation problem for CA graphs is logspace-reducible to that of CA graphs which are not Δ-uniform. This reduction follows from a slightly modified version of Lemma 4.31. Then we show that the neighborhood matrix of a non-Δ-uniform CA graph can be converted into a vertex-colored restricted CA matrix by flipping 'long' arcs. By coloring the flipped arcs the isomorphism type is preserved.

Definition 4.37. *For a graph G we define Δ_G as the following set of* ov*-triangles (see Definition 4.23). An* ov*-triangle T of G is in Δ_G if there exist three pairwise different vertices u, v, w in T such that the following holds:*

1. *$N[u] \cup N[v] \cup N[w] = V(G)$*
2. *For all $z \in T$ it holds that if a vertex $x \in N_T(z)$ then $x \mathtt{cd}\, z$*
3. *If there exist u', w' such that (u, w, w', u') is an induced 4-cycle and v overlaps with u' and w' then $N[v] \subseteq N[u'] \cup N[w']$*

Definition 4.38. *A CA graph G is Δ-uniform if $\Delta_G \cap \mathcal{T}_{\mathrm{IT}}(G) = \emptyset$.*

Let us explain the intuition behind these two definitions. The set Δ_G approximates $\mathcal{T}_{\mathrm{NHT}}(G)$. More precisely, whenever an $\mathtt{ov}$-triangle $T = \{u, v, w\}$ is in $\mathcal{T}_{\mathrm{NHT}}(G)$ this implies that T satisfies certain constraints such as for example $N[u] \cup N[v] \cup N[w] = V(G)$. The set Δ_G consists of three such constraints. Therefore if an $\mathtt{ov}$-triangle is representable as non-Helly triangle it must also be in Δ_G, i.e. $\mathcal{T}_{\mathrm{NHT}}(G) \subseteq \Delta_G$. The Δ-uniform CA graphs can be alternatively seen as the subset of uniform CA graphs where the constraints of Δ_G suffice to characterize $\mathcal{T}_{\mathrm{NHT}}(G)$, i.e. $\Delta_G = \mathcal{T}_{\mathrm{NHT}}(G)$.

Lemma 4.39. *For every graph G it holds that $\mathcal{T}_{\mathrm{NHT}}(G) \subseteq \Delta_G$. If G is a Δ-uniform CA graph then $\mathcal{T}_{\mathrm{NHT}}(G) = \Delta_G$.*

Proof. For the first claim consider a graph G. If G is not a CA graph then $\mathcal{T}_{\mathrm{NHT}}(G) = \emptyset$. Therefore we can assume that G is a CA graph. Given an $\mathtt{ov}$-triangle $T \in \mathcal{T}_{\mathrm{NHT}}(G)$ we show that it must be in Δ_G. Let $\rho \in \mathcal{N}(G)$ be a representation such that $T = \{u, v, w\}$ is represented as non-Helly triangle in it. Since $\rho(u) \cup \rho(v) \cup \rho(w)$ covers the whole circle it follows that $N[u] \cup N[v] \cup N[w] = V(G)$, which is the first condition of Definition 4.37. To see that the second condition holds we consider a vertex $x \in N_T(u)$ without loss of generality. Since x is not adjacent to v and w it holds that $\rho(x) \subseteq \mathbb{C} \setminus (\rho(v) \cup \rho(w))$ where $\mathbb{C}$ denotes the whole circle. Since $\mathbb{C} \setminus (\rho(v) \cup \rho(w)) \subset \rho(u)$ it follows that $\rho(x) \subset \rho(u)$. Due to the fact that ρ is a normalized representation this implies that x is contained by u. To see that the third condition of Δ_G holds let u', w' be vertices such that (u, w, w', u') is an induced 4-cycle of G. Since T is represented as non-Helly triangle in ρ it must hold that $\{u', v, w'\}$ is an interval triangle in ρ with $\rho(v) \subset \rho(u') \cup \rho(w')$ and therefore $N[v] \subseteq N[u'] \cup N[w']$.

For the second claim let G be a Δ-uniform CA graph. From the previous claim we know that $\mathcal{T}_{\mathrm{NHT}}(G) \subseteq \Delta_G$. Since every $\mathtt{ov}$-triangle must be in $\mathcal{T}_{\mathrm{NHT}}(G) \cup \mathcal{T}_{\mathrm{IT}}(G)$ it follows that $\Delta_G \subseteq \mathcal{T}_{\mathrm{NHT}}(G) \cup \mathcal{T}_{\mathrm{IT}}(G)$. The definition of Δ-uniform requires $\Delta_G \cap \mathcal{T}_{\mathrm{IT}}(G) = \emptyset$ and thus $\Delta_G \subseteq \mathcal{T}_{\mathrm{NHT}}(G)$. □

Fact 4.40. *Δ-uniform CA graphs are a strict subset of uniform CA graphs.*

Proof. Assume there exists a Δ-uniform CA graph G which is not uniform. This means there exists an $\mathtt{ov}$-triangle $T \in \mathcal{T}_{\mathrm{NHT}}(G) \cap \mathcal{T}_{\mathrm{IT}}(G)$. Due to the previous lemma it holds that $\mathcal{T}_{\mathrm{NHT}}(G) \subseteq \Delta_G$. This implies that $T \in \Delta_G \cap \mathcal{T}_{\mathrm{IT}}(G)$ which contradicts that G is Δ-uniform. Therefore every Δ-uniform CA graph is uniform.

An example of a uniform CA graph that is not Δ-uniform is the graph X_4 in Figure 4.7. In the third paragraph after Theorem 4.33 we argued that X_4 is a uniform CA graph because it has a unique normalized representation. Assume that the black arcs of X_4 are labeled with u, u', v, v', w, w' in the same way that the representation of $\overline{3K_2}$ is labeled in Figure 4.7. To see that X_4 is not Δ-uniform it suffices to check that the $\mathtt{ov}$-triangle $\{u, v, w\}$ is in Δ_{X_4} and represented as interval triangle. □

Corollary 4.41. *The globally invariant non-Helly triangle representability problem for Δ-uniform CA graphs can be solved in logspace.*

Proof. Given a CA graph G and an ov-triangle T output yes iff $T \in \Delta_G$. This is correct because in the case of a Δ-uniform CA graph G it holds that $\Delta_G = \mathcal{T}_{\mathrm{NHT}}(G)$ (Lemma 4.39). Clearly, Δ_G is computable in logspace and an invariant. □

Lemma 4.42. *Let G be a CA graph that is not Δ-uniform. Then there exists an induced 4-cycle $C = (u, w, w', u')$ such that $N[u] \cup N[w] = N[u'] \cup N[w'] = V(G)$ and a vertex v that overlaps with every vertex in C.*

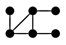
Proof. The argument is essentially the same as the one made for the "⇒"-direction in the proof of Theorem 4.33. The difference is that instead of the stronger assumption that $T \in \mathcal{T}_{\mathrm{NHT}}(G)$ we only require that $T \in \Delta_G$.

Since G is not Δ-uniform there exists an ov-triangle $T = \{u, v, w\}$ of G such that $T \in \Delta_G$ and there is a representation $\rho \in \mathcal{N}(G)$ such that T is represented as interval triangle in ρ. Furthermore, let us assume w.l.o.g. that $\rho(v) \subset \rho(u) \cup \rho(w)$. Since $T \in \Delta_G$ it holds that $N[u] \cup N[v] \cup N[w] = V(G)$. Due to the interval representation of T in ρ it follows that $N[u] \cup N[w] = V(G)$. Since u and w do not form a circle cover it must hold that there exists a vertex $u' \in N[u] \setminus N[w]$ such that $N[u'] \setminus N[u]$ is non-empty. If u' is disjoint from v it follows that u' must be contained by u from the second condition in Definition 4.37 of Δ_G. This cannot be the case and therefore $u' \in N_T(u, v)$. For u' to have a neighbor which is not adjacent to u it must hold that $\rho(u') \not\subseteq \rho(u)$. Therefore u' overlaps with u and v. Let $w' \in N[u'] \setminus N[u]$. If $w' \in N_T(w)$ then w' would be contained by w due to the second condition of Δ_G. Again, this cannot be the case and therefore $w' \in N_T(v, w)$. From the representation ρ it follows that w must overlap with u', v and w. Then $C = (u, w, w', u')$ is an induced 4-cycle of G such that v overlaps with every vertex of C. It remains to show that $N[u'] \cup N[w'] = V(G)$. Due to the third condition of Δ_G it holds that $N[v] \subseteq N[u'] \cup N[w']$. Additionally, it holds that $\rho(u) \setminus \rho(v) \subset \rho(u')$ and $\rho(w) \setminus \rho(v) \subset \rho(w')$. As a consequence $N[u'] \cup N[w'] = V(G)$. □

Corollary 4.43. *Canonical representations for CA graphs without induced 4-cycle can be computed in logspace.*

Proof. By Lemma 4.42 the class of CA graphs without induced 4-cycle is a subset of Δ-uniform CA graphs and due to Corollary 4.41 and Theorem 4.29 a canonical representation for such graphs can be computed in logspace. □

Corollary 4.44. *Helly CA graphs are a strict subset of Δ-uniform CA graphs.*

Proof. Assume G is a Helly CA graph which is not Δ-uniform. Then due to Lemma 4.42 there exists an induced 4-cycle C and a vertex v not in C which overlaps with every vertex in C. In any normalized representation of G it must hold that v forms a non-Helly triangle with two vertices from C. This contradicts that G is Helly. The graph  is a Δ-uniform CA graph which is not Helly. □

Theorem 4.45. *The canonical representation problem for CA graphs is logspace-reducible to the canonical representation problem for vertex-colored restricted CA matrices.*

Proof. For brevity let $\mathcal{Z}$ denote the set of all CA graphs which are not Δ-uniform. Since the globally invariant non-Helly triangle representability problem for Δ-uniform CA graphs can be solved in logspace (see Corollary 4.41) it follows from a modified version of Lemma 4.31 that the canonical representation problem for CA graphs is logspace-reducible to the canonical representation problem for vertex-colored $\mathcal{Z}$. To see this replace 'uniform' with 'Δ-uniform' and 'non-uniform' with 'non-Δ-uniform' in the statement (and proof) of Lemma 4.31.

For a CA graph G let us say a subset of vertices X of G is an R-flip set if $\lambda_G^{(X)}$ is a restricted CA matrix. To find canonical representations for $\mathcal{Z}$ we construct an invariant vertex set selector f such that $f(G)$ contains at least one R-flip set for every $G \in \mathcal{Z}$. Then to obtain a canonical representation for $G \in \mathcal{Z}$ let $\hat{X}$ denote the R-flip set in $f(G)$ such that $\text{canon}(\lambda_G^{(\hat{X})}, c_{\hat{X}})$ is lexicographically minimal with c_X being the coloring which assigns every vertex $v \in X$ the color red and the other vertices are blue. Let ρ be a canonical normalized representation for $(\lambda_G^{(\hat{X})}, c_{\hat{X}})$. Then $\rho^{(\hat{X})}$ is a canonical representation for G. Notice, that $\rho^{(\hat{X})}$ can be computed in logspace by computing canonical representations for vertex-colored restricted CA matrices. The correctness of this approach follows from the same argument made in the proof of Theorem 4.16 in the flip trick section. The analogy is straightforward. The R-flip sets in this context correspond to flip sets and the invariant vertex set selector f takes the place of the invariant flip set function. Given a CA graph G and $X \subseteq V(G)$ it can be easily checked in logspace whether $\lambda_G^{(X)}$ is a restricted CA matrix.

For a CA graph G let $C(G)$ denote the set of all ordered induced 4-cycles in G. Now, we claim that the following logspace-computable function f is an invariant vertex set selector with the desired property:

$$f(G) = \bigcup_{C \in C(G)} \{\, \{x \in V(G) \setminus C \mid \exists y \in C : x \,\texttt{cs}\, y\} \,\}$$

It is not difficult to check that f is invariant. It remains to argue why $f(G)$ contains at least one R-flip set for every $G \in \mathcal{Z}$. Let $G \in \mathcal{Z}$ and $C = (u, w, w', u')$ is an induced 4-cycle in G such that $N[u] \cup N[w] = N[u'] \cup N[w'] = V(G)$. The existence of such an induced 4-cycle is guaranteed by Lemma 4.42. Observe that if there exists a $u_1 \in N_C(u, w, u')$ with $u_1 \,\texttt{cs}\, u$ then $C_1 = (u_1, w, w', u')$ also satisfies the previous condition $N[u_1] \cup N[w] = V(G)$. Therefore we can assume that there exists no $z \in C$ and $z_1 \in N_C(N[z] \cap C)$ such that $z_1 \,\texttt{cs}\, z$. From $N[u] \cup N[w] = N[u'] \cup N[w'] = V(G)$ it immediately follows that $N_C(u, w)$, $N_C(u', w')$ and $N_C(x)$ are empty for every $x \in C$.

We prove that $\lambda^{(X)}$ is a restricted CA matrix with witness cycle C where λ is the neighborhood matrix of G and $X = \{x \in V(G) \setminus C \mid \exists y \in C : x \,\texttt{cs}\, y\}$. Note that $X \in f(G)$ via C. To reference the neighborhoods of G (which are the same as the ones of λ) or $\lambda^{(X)}$ we write N^G and $N^{\lambda^{(X)}}$ to distinguish between them. First, we show that $N_C^{\lambda^{(X)}}(u, w) = \emptyset$. Assume the opposite, i.e. there exists $x \in N_C^{\lambda^{(X)}}(u, w)$. If x was not flipped, i.e. $x \notin X$, then it also holds that $x \in N_C^G(u, w)$, which contradicts that $N_C^G(u, w)$ is empty. If x was flipped, i.e. $x \in X$, then it must be the case that x contains u' and w' in λ. This means $N_G[u'] \cup N_G[w'] \subseteq N_G[x]$ which implies that x is a universal vertex in G since $N_G[u'] \cup N_G[w'] = V(G)$, contradiction. For the same reason it holds that $N_C^{\lambda^{(X)}}(u', w')$ and $N_C^{\lambda^{(X)}}(z)$ are empty for all $z \in C$. It remains to show that for all $x \in N_C^{\lambda^{(X)}}(C)$ it holds that x overlaps with all vertices of C in $\lambda^{(X)}$. Notice that $\lambda_{x,z}^{(X)} \in \{\texttt{ov}, \texttt{cs}, \texttt{cc}\}$ for every $z \in C$. Otherwise x would not be in $N_C(C)$. We consider the following two cases: in the first one we assume that x contains one vertex of C in $\lambda^{(X)}$ and in the second one we assume that x forms a circle cover with one vertex of C in $\lambda^{(X)}$. We prove that neither of these cases can occur and therefore x must overlap with all vertices of C in $\lambda^{(X)}$. For the first case assume that w.l.o.g. x contains u in $\lambda^{(X)}$ and intersects with the other vertices of C in $\lambda^{(X)}$. If $x \in X$ then it was flipped. It follows that x was disjoint from u in λ and therefore $x \in N_C^G(w, w', u')$. Since $x \in X$ it also must hold that x contains at least one of the vertices w, w', u' in G. It follows that x contains w' since it cannot contain the other two in λ. However, this contradicts our choice of C which says that there exists no $w'_1 \in N_C^G(w, w', u')$ such that w'_1 contains w' in λ. If $x \notin X$ then it must hold that x already contained u in λ. But then x should be in X, contradiction. For the second case assume x forms a circle cover with u in $\lambda^{(X)}$. If x forms a circle cover with u then this implies that x contains w' in $\lambda^{(X)}$ and therefore this reduces to the first case. We conclude that both conditions of Definition 4.35 are satisfied and hence $\lambda^{(X)}$ is a restricted CA matrix. □

4.7 Flip Sets for Restricted CA Matrices

In this section we consider how to find an invariant flip set function for restricted CA matrices. From the previous section we know that this suffices to solve the canonical representation problem for CA graphs. In the first subsection we show that there is a subset of vertices in restricted CA matrices which makes it difficult to find flip sets and which cannot be avoided. In fact, this subset is the only obstacle. We prove that it suffices to only consider this subset of vertices in order to find flip sets in a restricted CA matrix. In the second subsection we analyze the structure of this particular subset and formulate two conjectures which, if true, show how an invariant flip set function for restricted CA matrices can be computed.

4.7.1 Partial Flip Sets

We consider the following refined notion of u-flip set called (u, w)-flip set which is defined in terms of two overlapping vertices u and w. In a restricted CA matrix one can always find two overlapping vertices due to the existence of a witness cycle.

Definition 4.46. *Given a CA matrix λ and two vertices u, w of λ with u $\mathtt{ov}$ w. We say the w-endpoint of u in a representation $\rho \in \mathcal{N}(\lambda)$ is the endpoint of $\rho(u)$ which is contained in $\rho(w)$. A flip set X of λ is a (u, w)-flip set if there exists a representation $\rho \in \mathcal{N}(\lambda)$ such that $x \in X$ iff $\rho(x)$ contains the w-endpoint of u in ρ for all $x \in V(\lambda)$.*

We say that x (instead of $\rho(x)$) contains the w-endpoint of u in ρ as it is clear from the context. Observe that (u, w)-flip sets can be equivalently defined as u-flip sets which contain w.

Now, our goal for a given a restricted CA matrix λ with witness cycle $C = (u, w, w', u')$ is to compute invariant (u, w)-flip sets (to be exact, we will compute (u_1, w)-flip sets where u_1 is a vertex similar to u). As we shall see for many vertices of λ it is clear whether they belong to a (u, w)-flip set or not. The only set of vertices that poses an obstacle is $N_C(C)$. Therefore we are interested in computing the following kind of subsets of $N_C(C)$.

Definition 4.47 (C-partial flip set)**.** *Let λ be a restricted CA matrix with witness cycle $C = (u, w, w', u')$. We call a subset Z of $N_C(C)$ a C-partial flip set if there exists a subset Y of $V(\lambda) \setminus N_C(C)$ such that $Y \cup Z$ is a (u, w)-flip set.*

Recall that a vertex $x \in N_C(C)$ overlaps with every vertex from C. In a normalized representation of λ there are four choices how x can be placed with respect to C. The following definition and Figure 4.8 describe these four choices.

Definition 4.48 (Type)**.** *Let λ be a restricted CA matrix with witness cycle $C = (u, w, w', u')$. Given $x \in N_C(C)$ and $\rho \in \mathcal{N}(\lambda)$ we define $\mathrm{type}_C(x, \rho)$ as $\{y, z\}$ with $y, z \in C$ such that $\rho(x) \subset \rho(y) \cup \rho(z)$. Additionally, we define $\mathrm{type}_C(x)$ as $\bigcup_{\rho \in \mathcal{N}(\lambda)} \{\mathrm{type}_C(x, \rho)\}$. If C is clear from the context we omit the subscript.*

Lemma 4.49. *Given a restricted CA matrix λ with witness cycle $C = (u, w, w', u')$. Let Z be a subset of $N_C(C)$. Then the following statements are equivalent:*

1. *Z is a C-partial flip set*
2. *There exists a representation $\rho \in \mathcal{N}(\lambda)$ such that $z \in Z$ iff $\rho(z)$ contains the w-endpoint of u in ρ for all $z \in N_C(C)$*
3. *There exists a representation $\rho \in \mathcal{N}(\lambda)$ such that $z \in Z$ iff $w \in \mathrm{type}(z, \rho)$ for all $z \in N_C(C)$*

Proof. Observe that for every $x \in N_C(C)$ and $\rho \in \mathcal{N}(\lambda)$ it holds that x contains the w-endpoint of u iff $w \in \mathrm{type}(x, \rho)$. From this it follows that the second and third statement are equivalent. It is also easy to see that the first implies the second statement. To see that the second statement implies the first consider a representation $\rho \in \mathcal{N}(\lambda)$ such that $z \in Z$ iff $\rho(z)$ contains the w-endpoint of u in ρ for all $z \in N_C(C)$. Let Y contain all vertices $x \in V(\lambda) \setminus N_C(C)$ such that $\rho(x)$ contains the w-endpoint of u in ρ. Clearly, $Y \cup Z$ is a (u, w)-flip set and therefore Z is a C-partial flip set. □

Definition 4.50. *Let f be a vertex set selector for restricted CA matrices. We call f a partial flip set function for restricted CA matrices if for every restricted CA matrix λ there exists a witness cycle C of λ such that $f(\lambda)$ contains a C-partial flip set. We call f an invariant partial flip set function if it is invariant for restricted CA matrices.*

Theorem 4.51. *The (canonical) representation problem for CA graphs is logspace-reducible to the problem of computing an (invariant) partial flip set function for restricted CA matrices.*

Proof. Due to Theorem 4.45 and Theorem 4.16 it suffices to compute an invariant flip set function for restricted CA matrices in logspace. Let f be an invariant partial flip set function for restricted CA matrices. Then we claim that the following vertex set selector is an invariant flip set function for restricted CA matrices. Let $C_4(\lambda)$ denote the set of all ordered induced 4-cycles of λ.

$$F(\lambda) = \bigcup_{\substack{x,y \in V(\lambda),\\ C=(u,w,w',u') \in C_4(\lambda),\\ Z \in f(\lambda)}} \Big\{ Z \cup \{x\} \cup \big(\langle N[x] \cap N[y] \rangle \setminus \langle N_C(C) \cup U \rangle \big) \Big\}$$

where U is shorthand for $N_C(C \setminus \{w'\}) \cup \{u\}$.

Clearly, if f can be computed in logspace then this also holds for F. The fact that F is invariant follows from f and the different neighborhoods used in F being invariant and the set of induced 4-cycles $C_4(\lambda)$ being a vertex set invariant as well.

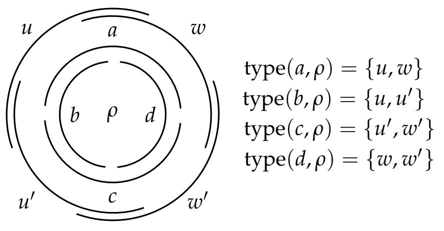

u
a
w
b
ρ
d
u'
c
w'
$\mathrm{type}(a,\rho) = \{u, w\}$
$\mathrm{type}(b,\rho) = \{u, u'\}$
$\mathrm{type}(c,\rho) = \{u', w'\}$
$\mathrm{type}(d,\rho) = \{w, w'\}$

FIGURE 4.8: Types of $N_C(C)$

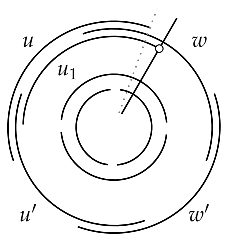


FIGURE 4.9: (u_1, w)-flip set in RCA matrix

It remains to prove that $F(\lambda)$ contains at least one flip set for every restricted CA matrix λ. Let $Z \in f(\lambda)$ be a C-partial flip set of λ with witness cycle $C = (u, w, w', u')$. We assume w.l.o.g. that there exists no $u_0 \in U$ such that $u_0 \texttt{ cs } u$ (otherwise we could consider $C_0 = (u_0, w, w', u')$ as witness cycle instead and Z is a C_0-partial flip set). Additionally, let $\rho \in \mathcal{N}(\lambda)$ be a representation such that $x \in Z$ iff $w \in \text{type}(x, \rho)$ for all $x \in N_C(C)$. The existence of such a representation follows from Lemma 4.49. Let $u_1 \in U$ denote the vertex such that $\rho(u_2) \cap \rho(w) \subseteq \rho(u_1) \cap \rho(w)$ for all $u_2 \in U$. Informally, u_1 is the 'rightmost' vertex of U in ρ. We show that the set

$$X = Z \cup \{u_1\} \cup \left(\langle N[u_1] \cap N[w] \rangle \setminus \langle N_C(C) \cup U \rangle\right)$$

with $X \in F(\lambda)$ is a (u_1, w)-flip set of λ. This follows from the following two claims:

1. For all $x \in N_C(C)$ it holds that $\rho(x)$ contains the w-endpoint of u_1 iff $w \in \text{type}(x, \rho)$
2. For all $x \in V(\lambda) \setminus N_C(C)$ with $x \neq u_1$ it holds that $\rho(x)$ contains the w-endpoint of u_1 iff $x \in (N[u_1] \cap N[w]) \setminus U)$

Consider the first claim. If $u_1 = u$ then this follows from Lemma 4.49. Therefore let us assume that $u_1 \neq u$. Consider a vertex $x \in N_C(C)$. Note that the w-endpoint of u_1 in ρ is contained in $A = \rho(w) \setminus (\rho(u) \cup \rho(w'))$. If $w \in \text{type}(x, \rho)$ then $A \subseteq \rho(x)$ and therefore $\rho(x)$ contains the w-endpoint of u_1 in ρ. If $w \notin \text{type}(x, \rho)$ then $\rho(x) \cap A = \emptyset$ and therefore $\rho(x)$ does not contain the w-endpoint of u_1 in ρ.

For the second claim let us first define the following sets which are similar to U:

- $W = N_C(C \setminus \{u'\}) \cup \{w\}$,
- $U' = N_C(C \setminus \{w\}) \cup \{u'\}$,
- $W' = N_C(C \setminus \{u\}) \cup \{w'\}$,
- $UU' = N_C(u, u')$,
- $WW' = N_C(w, w')$

It holds that these sets together with U and $N_C(C)$ partition $V(\lambda)$. Then the set $(N[u_1] \cap N[w]) \setminus U$ in the second claim is identical to $(W \cup W' \cup WW') \cap N[u_1]$ when restricted to vertices from $V(\lambda) \setminus N_C(C)$. We have chosen C such that either $u_1 = u$ or $u_1 \texttt{ ov } u$ must hold and therefore the other endpoint of u_1 (not its w-endpoint) is contained in $\rho(u) \cap \rho(u')$. If $x \in W \cup W' \cup WW'$ and adjacent to u_1 then it must overlap with u_1. Since $\rho(x) \cap (\rho(u) \cap \rho(u')) = \emptyset$ it holds that $\rho(x)$ cannot contain the other endpoint of u_1 in ρ. Therefore $\rho(x)$ contains the w-endpoint of u_1. If $x \notin W \cup W' \cup WW'$ then $x \in U \cup U' \cup UU'$. If $x \in U' \cup UU'$ then x is disjoint from w and therefore $\rho(x)$ does not contain the w-endpoint of u_1. If $x \in U$ and $x \neq u_1$ then $\rho(x) \cap \rho(w) \subset \rho(u_1) \cap \rho(w)$ since we have chosen u_1 to satisfy this. Therefore x does not contain the w-endpoint of u_1 in this case as well. □

4.7.2 Structure of Partial Flip Sets

Consider a restricted CA matrix λ with witness cycle C and let $\mathcal{Z}$ be the set of all C-partial flip sets. What can be said about the structure of $\mathcal{Z}$? A simple observation, for example, is that if two vertices $x, y \in N_C(C)$ are disjoint or form a circle cover then $\text{type}(x, \rho) \cup \text{type}(y, \rho) = C$ for all $\rho \in \mathbb{N}(\lambda)$. This implies that $x \in Z$ iff $y \notin Z$ for all

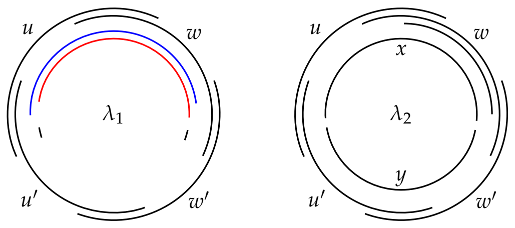


FIGURE 4.10: Mutual dependence relation

$Z \in \mathcal{Z}$. Analogously, if x contains or is contained by y then $x \in Z$ iff $y \in Z$ for all $Z \in \mathcal{Z}$. This means there are subsets of vertices X of $N_C(C)$ for which there exists a unique partition into two parts X^0 and X^1 such that $Z \in \mathcal{Z}$ implies $X^i \subseteq Z$ and $X^j \cap Z = \emptyset$ for some $(i,j) \in \{(0,1),(1,0)\}$. Informally, one has to make only one binary decision for the set X when trying to construct a partial flip set. The following definition describes these sets in terms of two equivalence relations $\sim_{\text{sibl}}$ and $\sim_{\text{md}}$.

Definition 4.52. *Let λ be a restricted CA matrix with witness cycle C. For two vertices $x, y \in N_C(C)$ we say they are siblings, in symbols $x \sim_{sibl} y$, if for all C-partial flip sets Z it holds that $x \in Z$ iff $y \in Z$. Similarly, we say x and y are anti-siblings if for all C-partial flip sets Z it holds that $x \in Z$ iff $y \notin Z$. We say x and y are mutually dependent, in symbols $x \sim_{md} y$, if x and y are siblings or anti-siblings.*

The observation of the previous paragraph can be strengthened as follows. For an intersection matrix μ let $G^{\text{ov}}(\mu)$ denote the graph which has the same vertex set as μ and two vertices are adjacent iff they overlap in μ. Let λ be a restricted CA matrix with witness cycle C. Let λ' be the submatrix of λ which is induced by $N_C(C)$. If two vertices occur in the same connected component of the edge-complement of $G^{\text{ov}}(\lambda')$ then they are mutually dependent. The contra-position of this statement tells us that $x \not\sim_{\text{md}} y$ implies $x \text{ ov } y$ for all $x, y \in N_C(C)$. The other direction of this implication does not hold, see λ_1 in Figure 4.10 for a counter-example. The red and blue arc are disjoint in the edge-complement of $G^{\text{ov}}(\lambda_1')$ but it holds that they are siblings.

Lemma 4.53. *Let λ be a restricted CA matrix with witness cycle C. Every equivalence class $X \subseteq N_C(C)$ of the mutual dependence relation is partitioned by at most two distinct equivalence classes of the sibling relation.*

Proof. Consider a $y \in X$ and the corresponding equivalence class $Y = \{x \in N_C(C) \mid x \sim_{\text{sibl}} y\}$ of y w.r.t. the sibling relation. Since $x \sim_{\text{sibl}} y$ implies $x \sim_{\text{md}} y$ it follows that $Y \subseteq X$. If $Y = X$ then the statement holds. Otherwise, let $z \in X \setminus Y$ and $Z = \{x \in N_C(C) \mid x \sim_{\text{sibl}} z\}$. It holds that $Z \subseteq X$ and clearly Y and Z are disjoint because they are distinct equivalence classes. We claim that $X = Y \cup Z$. Assume this is not the case. Then there exists a $w \in X \setminus (Y \cup Z)$ such that $w \not\sim_{\text{sibl}} y$ and $w \not\sim_{\text{sibl}} z$. Since $w \sim_{\text{md}} y$ this means for all C-partial flip sets V that $y \in V$ iff $w \notin V$. Since $y \not\sim_{\text{sibl}} z$ holds this also means that for all C-partial flip sets V that $y \in V$ iff $z \notin V$. From that it follows that $w \in V$ iff $z \in V$ for all C-partial flip sets V which contradicts that w and z are not siblings. $\square$

Definition 4.54. *Let λ be a restricted CA matrix with witness cycle C. We define the two subsets Γ^0_C and Γ^1_C of $N_C(C)$ as follows:*

- *$x \in \Gamma^0_C$ iff for all C-partial flip sets Z it holds that $x \notin Z$*
- *$x \in \Gamma^1_C$ iff for all C-partial flip sets Z it holds that $x \in Z$*

If C is clear from the context we omit the subscript.

Observe that Γ^0 and Γ^1 are equivalence classes of the sibling relation (or empty) and $\Gamma^0 \cup \Gamma^1$ is an equivalence class of the mutual dependence relation (or the empty set). From Lemma 4.49 it follows that these two sets can be alternatively characterized as:

$$\Gamma^1 = \{x \in N_C(C) \mid \text{type}(x) \subseteq \{\{u,w\},\{w,w'\}\}$$
$$\Gamma^0 = \{x \in N_C(C) \mid \text{type}(x) \subseteq \{\{\{u',w'\},\{u,u'\}\}$$

Consider λ_2 in Figure 4.10. The vertex x is in Γ^1 because x must be present in every C-partial flip set (x contains the w-endpoint of u in every representation). Since y is disjoint from x it follows that $y \in \Gamma^0$.

We conjecture that the sibling and mutual dependence relation contain enough information to characterize the set of C-partial flip sets.

Conjecture 4.55. *Let λ be a restricted CA matrix with witness cycle C. Let $X_1, \ldots, X_k$ be the equivalence classes of the mutual dependence relation but $\Gamma^0 \cup \Gamma^1$. Additionally, let $X_i^0 \neq X_i^1$ be equivalence classes of the sibling relation or empty such that $X_i = X_i^0 \cup X_i^1$ for all $i \in [k]$. For all $X \subseteq N_C(C)$ it holds that*

$$X \text{ is a } C\text{-partial flip set} \Leftrightarrow \exists w \in \{0,1\}^k : \Gamma^1 \cup \bigcup_{i=1}^{k} X_i^{w_i}$$

It is clear that the "$\Rightarrow$"-direction of this statement holds. Observe that to disprove this conjecture it would already suffice to find a restricted CA matrix λ with witness cycle C such that its number of C-partial flip sets is not a power of two. If this conjecture does not hold this implies that there are some other kind of dependencies among the equivalence classes of the mutual dependence relation which further restrict the set of C-partial flip sets.

Fact 4.56. *If Conjecture 4.55 holds then the representation problem for CA graphs is logspace-reducible to computing the sibling and mutual dependence relation.*

Proof. By Theorem 4.51 it suffices to compute a partial flip set function for restricted CA matrices in order to solve the representation problem for CA graphs. Assume that Conjecture 4.55 holds. Then a C-partial flip set can be computed by guessing which equivalence class of the sibling relation is Γ^0_C and which is Γ^1_C. Let $X_1, \ldots, X_k$ be the equivalence classes of the mutual dependence relation such that $X_i \neq \Gamma^0_C \cup \Gamma^1_C$ for all $i \in [k]$. Let $X_i^0 \neq X_i^1$ be chosen as in the statement of Conjecture 4.55. Then $\Gamma^1_C \cup X_1^1 \cup \cdots \cup X_k^1$ is a C-partial flip set which can be computed using the relations $\sim_{\text{sibl}}$ and $\sim_{\text{md}}$. □

Conjecture 4.57. *Let λ be a restricted CA matrix with witness cycle C. Let X be an equivalence class of the mutual dependence relation and let $X_0 \neq X_1$ be equivalence classes of $\sim_{sibl}$ or empty such that $X = X_0 \cup X_1$. Let λ_X denote the induced submatrix of λ on the vertex set $\big(V(\lambda) \setminus N_C(C)\big) \cup X$. It holds that X_0 and X_1 are in the same orbit w.r.t. λ iff X_0 and X_1 are in the same orbit w.r.t. λ_X.*

Fact 4.58. *If Conjecture 4.55 and 4.57 hold then the canonical representation problem for CA graphs is logspace-reducible to computing the sibling and mutual dependence relation.*

Proof. Let λ be a restricted CA matrix with witness cycle C. We argue how a single invariant C-partial flip set for λ can be computed under the assumption that the two previous conjectures hold. We proceed as in the proof of Fact 4.56. However, instead of $\Gamma^1_C \cup X^1_1 \cup \cdots \cup X^1_k$ (which is not guaranteed to be an invariant choice) we choose the C-partial flip set as follows. For each $i \in [k]$ we compute a canonical labeling τ_i of λ_{X_i} (λ_{X_i} is defined as in the statement of Conjecture 4.57). We say X^0_i is smaller than X^1_i if the lexicographically smallest vertex in $\tau_i(X^0_i)$ is smaller than that in $\tau_i(X^1_i)$. For each $i \in [k]$ let $w_i \neq \overline{w_i} \in \{0,1\}$ be such that $X^{w_i}_i$ is smaller than $X^{\overline{w_i}}_i$. Then $\Gamma^1_C \cup X^{w_1}_1 \cup \cdots \cup X^{w_k}_k$ is an invariant C-partial flip set. □

4.8 Summary and Outlook

Complexity classes such as GP can be understood as graph parameters and therefore can be used to parameterize algorithmic problems (Corollary 4.6). However, already GFO(=) seems to be a prohibitively complex parameterization when trying to find positive algorithmic results. The or-pointer number seems to be the most suitable candidate for algorithmic considerations since it is a simple generalization of uniformly sparse graph classes for which positive algorithmic results exist. In the non-parameterized context a fragment of GFO(<) has led us to consider the isomorphism problem for CA graphs.

We showed that the canonical representation problem for uniform CA graphs is logspace-equivalent to the non-Helly triangle representability problem and to the representation problem (Theorem 4.29). Since representations for CA graphs can be computed in polynomial-time [McC03] it follows that canonical representations for uniform CA graphs can be computed in polynomial-time. We then gave a partial characterization of uniform CA graphs in terms of what conditions an ov-triangle must satisfy in order to be representable as non-Helly triangle, see Definition 4.37. We called the resulting subset of uniform CA graphs Δ-uniform. An interesting problem is to extend the definition of Δ_G such that Δ-uniform CA graphs coincide with uniform CA graphs. If the newly defined set Δ_G remains logspace-computable and invariant this would imply that the canonical representation problem for CA graphs is logspace-reducible to the one for non-uniform CA graphs.

It is not known whether CA graphs can be recognized in logspace. If a (not necessarily canonical) flip set can be computed in logspace then a CA representation for CA graphs can be computed in logspace as well.

Finally, we reduced the canonical representation problem for CA graphs to that for restricted CA matrices (Theorem 4.51). These matrices capture some of the essential features of non-uniform CA graphs. We showed that finding flip sets for RCA matrices boils down to considering how vertices from $N_C(C)$ can be represented. The most relevant open problem is whether Conjecture 4.55 and 4.57 hold and, if so, how to compute the sibling and mutual dependence relations. Alternatively, one can also try to solve the isomorphism problem for non-uniform CA graphs using a different method than the flip trick. In that case it seems reasonable to further explore the structure of this class.

Bibliography

[AG07] Noga Alon and Shai Gutner. “Linear Time Algorithms for Finding a Dominating Set of Fixed Size in Degenerated Graphs”. In: *Proceedings of the 13th Annual International Conference on Computing and Combinatorics*. COCOON’07. Banff, Canada: Springer-Verlag, 2007, pp. 394–405. ISBN: 3-540-73544-5, 978-3-540-73544-1. URL: http://dl.acm.org/citation.cfm?id=2394650.2394689.

[Atm+15] A. Atminas, A. Collins, V. Lozin, and V. Zamaraev. “Implicit representations and factorial properties of graphs”. In: *Discrete Mathematics* 338.2 (2015), pp. 164 –179. ISSN: 0012-365X. DOI: https://doi.org/10.1016/j.disc.2014.09.008. URL: http://www.sciencedirect.com/science/article/pii/S0012365X14003690.

[Can88] John Canny. “Some Algebraic and Geometric Computations in PSPACE”. In: *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing*. STOC ’88. Chicago, Illinois, USA: ACM, 1988, pp. 460–467. ISBN: 0-89791-264-0. DOI: 10.1145/62212.62257. URL: http://doi.acm.org/10.1145/62212.62257.

[CGS17] Yixin Cao, Luciano N. Grippo, and Martín D. Safe. “Forbidden Induced Subgraphs of Normal Helly Circular-Arc Graphs: Characterization and Detection”. In: *Discrete Applied Mathematics* 216, Part 1 (2017). Special Graph Classes and Algorithms — in Honor of Professor Andreas Brandstädt on the Occasion of His 65th Birthday, pp. 67 –83. ISSN: 0166-218X. DOI: http://dx.doi.org/10.1016/j.dam.2015.08.023. URL: http://www.sciencedirect.com/science/article/pii/S0166218X15004308.

[Cha16a] Maurice Chandoo. “Deciding Circular-Arc Graph Isomorphism in Parameterized Logspace”. In: *33rd Symposium on Theoretical Aspects of Computer Science (STACS 2016)*. Vol. 47. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2016, 26:1–26:13. ISBN: 978-3-95977-001-9. DOI: http://dx.doi.org/10.4230/LIPIcs.STACS.2016.26. URL: http://drops.dagstuhl.de/opus/volltexte/2016/5727.

[Cha16b] Maurice Chandoo. “On the Implicit Graph Conjecture”. In: *41st International Symposium on Mathematical Foundations of Computer Science, MFCS 2016, August 22-26, 2016 - Kraków, Poland*. 2016, 23:1–23:13. DOI: 10.4230/LIPIcs.MFCS.2016.23.

[Cha17a] Maurice Chandoo. “A Complexity Theory For Labeling Schemes”. In: (2017).

[Cha17b] Maurice Chandoo. “Canonical Representations for Circular-Arc Graphs Using Flip Sets”. In: *Algorithmica* (2017).

[Cur+13] Andrew Curtis, Min Lin, Ross McConnell, Yahav Nussbaum, Francisco Soulignac, Jeremy Spinrad, and Jayme Szwarcfiter. “Isomorphism of Graph Classes Related to the Circular-Ones Property”. In: *Discrete Mathematics and Theoretical Computer Science* 15.1 (2013). ISSN: 1365-8050. URL: http://www.dmtcs.org/dmtcs-ojs/index.php/dmtcs/article/view/2298.

[Dab+11] Konrad Dabrowski, Vadim Lozin, Haiko Müller, and Dieter Rautenbach. "Parameterized Algorithms for the Independent Set Problem in Some Hereditary Graph Classes". In: *Combinatorial Algorithms: 21st International Workshop, IWOCA 2010, London, UK, July 26-28, 2010, Revised Selected Papers*. Ed. by Costas S. Iliopoulos and William F. Smyth. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 1–9. ISBN: 978-3-642-19222-7. DOI: 10.1007/978-3-642-19222-7_1. URL: https://doi.org/10.1007/978-3-642-19222-7_1.

[ELS10] David Eppstein, Maarten Löffler, and Darren Strash. "Listing All Maximal Cliques in Sparse Graphs in Near-Optimal Time". In: *Algorithms and Computation: 21st International Symposium, ISAAC 2010, Jeju Island, Korea, December 15-17, 2010, Proceedings, Part I*. Ed. by Otfried Cheong, Kyung-Yong Chwa, and Kunsoo Park. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 403–414. ISBN: 978-3-642-17517-6. DOI: 10.1007/978-3-642-17517-6_36. URL: https://doi.org/10.1007/978-3-642-17517-6_36.

[Esc98] Elaine Marie Eschen. "Circular-Arc Graph Recognition and Related Problems". UMI Order No. GAX98-03921. PhD thesis. Nashville, TN, USA: Vanderbilt University, 1998.

[Fel+09a] Michael R. Fellows, Danny Hermelin, Frances Rosamond, and Stéphane Vialette. "On the Parameterized Complexity of Multiple-interval Graph Problems". In: *Theor. Comput. Sci.* 410.1 (Jan. 2009), pp. 53–61. ISSN: 0304-3975. DOI: 10.1016/j.tcs.2008.09.065. URL: http://dx.doi.org/10.1016/j.tcs.2008.09.065.

[Fel+09b] Michael R. Fellows, Danny Hermelin, Frances Rosamond, and Stéphane Vialette. "On the parameterized complexity of multiple-interval graph problems". In: *Theoretical Computer Science* 410.1 (2009), pp. 53 –61. ISSN: 0304-3975. DOI: http://dx.doi.org/10.1016/j.tcs.2008.09.065. URL: http://www.sciencedirect.com/science/article/pii/S0304397508007329.

[FG06] J. Flum and M. Grohe. *Parameterized Complexity Theory (Texts in Theoretical Computer Science. An EATCS Series)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006. ISBN: 3540299521.

[Fid+98] Charles M. Fiduccia, Edward R. Scheinerman, Ann Trenk, and Jennifer S. Zito. "Dot product representations of graphs". In: *Discrete Mathematics* 181.1 (1998), pp. 113 –138. ISSN: 0012-365X. DOI: http://dx.doi.org/10.1016/S0012-365X(97)00049-6. URL: http://www.sciencedirect.com/science/article/pii/S0012365X97000496.

[Fom+10] Fedor V. Fomin, Petr A. Golovach, Daniel Lokshtanov, and Saket Saurabh. "Intractability of Clique-Width Parameterizations". In: *SIAM Journal on Computing* 39.5 (2010), pp. 1941–1956. DOI: 10.1137/080742270. eprint: https://doi.org/10.1137/080742270. URL: https://doi.org/10.1137/080742270.

[Gav74] F. Gavril. "Algorithms on Circular-Arc Graphs". In: *Networks* 4.4 (1974), pp. 357–369. ISSN: 1097-0037. DOI: 10.1002/net.3230040407. URL: http://dx.doi.org/10.1002/net.3230040407.

[GJT76] M. R. Garey, D. S. Johnson, and R. Endre Tarjan. "The Planar Hamiltonian Circuit Problem is NP-Complete". In: *SIAM Journal on Computing* 5.4 (1976), pp. 704–714. DOI: 10.1137/0205049. eprint: https://doi.org/10.1137/0205049. URL: https://doi.org/10.1137/0205049.

[Gol04] Martin Charles Golumbic. *Algorithmic Graph Theory and Perfect Graphs (Annals of Discrete Mathematics, Vol 57)*. Amsterdam, The Netherlands, The Netherlands: North-Holland Publishing Co., 2004. ISBN: 0444515305.

[Gro17] Martin Grohe. *Descriptive Complexity, Canonisation, and Definable Graph Structure Theory*. Lecture Notes in Logic. Cambridge University Press, 2017. DOI: 10.1017/9781139028868.

[GS15] Martin Grohe and Pascal Schweitzer. “Isomorphism Testing for Graphs of Bounded Rank Width”. In: *IEEE 56th Annual Symposium on Foundations of Computer Science, FOCS 2015, Berkeley, CA, USA, 17-20 October, 2015*. 2015, pp. 1010–1029. DOI: 10.1109/FOCS.2015.66. URL: https://doi.org/10.1109/FOCS.2015.66.

[Hes01] William Hesse. “Division Is In Uniform TC0”. In: *Automata, Languages and Programming: 28th International Colloquium, ICALP 2001 Crete, Greece, July 8–12, 2001 Proceedings*. Ed. by Fernando Orejas, Paul G. Spirakis, and Jan van Leeuwen. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 104–114. ISBN: 978-3-540-48224-6. DOI: 10.1007/3-540-48224-5_9. URL: https://doi.org/10.1007/3-540-48224-5_9.

[Hsu95] Wen-Lian Hsu. “$O(M \cdot N)$ Algorithms for the Recognition and Isomorphism Problems on Circular-Arc Graphs”. In: *SIAM J. Comput.* 24.3 (June 1995), pp. 411–439. ISSN: 0097-5397. DOI: 10.1137/S0097539793260726. URL: http://dx.doi.org/10.1137/S0097539793260726.

[Joe+11] Benson L. Joeris, Min Chih Lin, Ross M. McConnell, Jeremy P. Spinrad, and Jayme L. Szwarcfiter. “Linear-Time Recognition of Helly Circular-Arc Models and Graphs”. In: *Algorithmica* 59.2 (2011), pp. 215–239. ISSN: 1432-0541. DOI: 10.1007/s00453-009-9304-5. URL: http://dx.doi.org/10.1007/s00453-009-9304-5.

[KKV13] Johannes Köbler, Sebastian Kuhnert, and Oleg Verbitsky. “Helly Circular-Arc Graph Isomorphism is in Logspace”. English. In: *Mathematical Foundations of Computer Science 2013*. Vol. 8087. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2013, pp. 631–642. ISBN: 978-3-642-40312-5. DOI: 10.1007/978-3-642-40313-2_56. URL: http://dx.doi.org/10.1007/978-3-642-40313-2_56.

[KKV16] Johannes Köbler, Sebastian Kuhnert, and Oleg Verbitsky. “Solving the canonical representation and Star System Problems for proper circular-arc graphs in logspace”. In: *Journal of Discrete Algorithms* 38-41.Supplement C (2016), pp. 38 –49. ISSN: 1570-8667. DOI: https://doi.org/10.1016/j.jda.2016.03.001. URL: http://www.sciencedirect.com/science/article/pii/S1570866716300016.

[KM12] Ross J. Kang and Tobias Müller. “Sphere and Dot Product Representations of Graphs”. In: *Discrete & Computational Geometry* 47.3 (2012), pp. 548–568. ISSN: 1432-0444. DOI: 10.1007/s00454-012-9394-8. URL: https://doi.org/10.1007/s00454-012-9394-8.

[KM94] J. Kratochvil and J. Matousek. “Intersection Graphs of Segments”. In: *Journal of Combinatorial Theory, Series B* 62.2 (1994), pp. 289 –315. ISSN: 0095-8956. DOI: https://doi.org/10.1006/jctb.1994.1071. URL: http://www.sciencedirect.com/science/article/pii/S0095895684710719.

[KNR92] Sampath Kannan, Moni Naor, and Steven Rudich. “Implicit Representation of Graphs”. In: *SIAM Journal on Discrete Mathematics* 5.4 (Nov. 1992), pp. 596–603. ISSN: 0895-4801 (print), 1095-7146 (electronic).

[Kö+11] Johannes Köbler, Sebastian Kuhnert, Bastian Laubner, and Oleg Verbitsky. “Interval graphs: Canonical Representations in Logspace”. In: *SIAM Journal on Computing* 40.5 (2011), pp. 1292–1315. ISSN: 1501-1526. DOI: 10.1137/10080395X.

[LB79] George S. Lueker and Kellogg S. Booth. “A Linear Time Algorithm for Deciding Interval Graph Isomorphism”. In: *J. ACM* 26.2 (Apr. 1979), pp. 183–195. ISSN: 0004-5411. DOI: 10.1145/322123.322125. URL: http://doi.acm.org/10.1145/322123.322125.

[Lin92] Steven Lindell. “A Logspace Algorithm for Tree Canonization (Extended Abstract)”. In: *Proceedings of the Twenty-fourth Annual ACM Symposium on Theory of Computing*. STOC ’92. Victoria, British Columbia, Canada: ACM, 1992, pp. 400–404. ISBN: 0-89791-511-9. DOI: 10.1145/129712.129750. URL: http://doi.acm.org/10.1145/129712.129750.

[LMZ12] Vadim V. Lozin, Colin Mayhill, and Victor Zamaraev. “Locally bounded coverings and factorial properties of graphs”. In: *European Journal of Combinatorics* 33.4 (2012), pp. 534 –543. ISSN: 0195-6698. DOI: https://doi.org/10.1016/j.ejc.2011.10.006. URL: http://www.sciencedirect.com/science/article/pii/S0195669811002058.

[Lok+17] Daniel Lokshtanov, Marcin Pilipczuk, Michał Pilipczuk, and Saket Saurabh. “Fixed-Parameter Tractable Canonization and Isomorphism Test for Graphs of Bounded Treewidth”. In: *SIAM Journal on Computing* 46.1 (2017), pp. 161–189. DOI: 10.1137/140999980. eprint: https://doi.org/10.1137/140999980. URL: https://doi.org/10.1137/140999980.

[LS09] Min Chih Lin and Jayme L. Szwarcfiter. “Characterizations and Recognition of Circular-Arc Graphs and Subclasses: A Survey”. In: *Discrete Mathematics* 309.18 (2009). Combinatorics 2006, A Meeting in Celebration of Pavol Hell’s 60th Birthday (May 1–5, 2006), pp. 5618 –5635. ISSN: 0012-365X. DOI: http://dx.doi.org/10.1016/j.disc.2008.04.003. URL: http://www.sciencedirect.com/science/article/pii/S0012365X08002161.

[Luk82] Eugene M. Luks. “Isomorphism of graphs of bounded valence can be tested in polynomial time”. In: *Journal of Computer and System Sciences* 25.1 (1982), pp. 42 –65. ISSN: 0022-0000. DOI: https://doi.org/10.1016/0022-0000(82)90009-5. URL: http://www.sciencedirect.com/science/article/pii/0022000082900095.

[McC03] Ross M. McConnell. “Linear-time Recognition of Circular-Arc Graphs”. In: *Algorithmica* 37.2 (2003), pp. 93–147. ISSN: 0178-4617. DOI: 10.1007/s00453-003-1032-7. URL: http://dx.doi.org/10.1007/s00453-003-1032-7.

[MM13] Colin McDiarmid and Tobias Müller. “Integer realizations of disk and segment graphs”. In: *Journal of Combinatorial Theory, Series B* 103.1 (2013), pp. 114 –143. ISSN: 0095-8956. DOI: http://dx.doi.org/10.1016/j.jctb.2012.09.004. URL: http://www.sciencedirect.com/science/article/pii/S0095895612000718.

[Mul88] John Harold Muller. “Local Structure in Graph Classes”. Order No: GAX88-11342. PhD thesis. Atlanta, GA, USA, 1988.

[Pon91] Ilya Ponomarenko. “The isomorphism problem for classes of graphs closed under contraction”. In: 55 (June 1991), pp. 1621–1643.

[Rot16] Noy Rotbart. “New Ideas on Labeling Schemes”. PhD thesis. University of Copenhagen, 2016.

[Smo91] Craig Smoryński. *Logical Number Theory I: An Introduction*. Logical Number Theory I: An Introduction Bd. 1. Springer, 1991. ISBN: 9780387522364.

[Spi03] Jeremy P. Spinrad. *Efficient Graph Representations.: The Fields Institute for Research in Mathematical Sciences.* Fields Institute monographs. American Mathematical Soc., 2003. ISBN: 9780821871775.

[Sta67] Franklin W. Stahl. “Circular Genetic Maps”. In: *Journal of Cellular Physiology* 70.S1 (1967), pp. 1–12. ISSN: 1097-4652. DOI: 10.1002/jcp.1040700403. URL: http://dx.doi.org/10.1002/jcp.1040700403.

[SZ94] E.R Scheinerman and J Zito. “On the Size of Hereditary Classes of Graphs”. In: *Journal of Combinatorial Theory, Series B* 61.1 (1994), pp. 16 –39. ISSN: 0095-8956. DOI: http://dx.doi.org/10.1006/jctb.1994.1027. URL: http://www.sciencedirect.com/science/article/pii/S0095895684710276.

[Tuc70] Alan Tucker. “Characterizing Circular-Arc Graphs”. In: *Bull. Amer. Math. Soc.* 76.6 (Nov. 1970), pp. 1257–1260. URL: http://projecteuclid.org/euclid.bams/1183532398.

[Vol99] Heribert Vollmer. *Introduction to Circuit Complexity: A Uniform Approach*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1999. ISBN: 3540643109.

[WG86] Michael S. Waterman and Jerrold R. Griggs. "Interval Graphs and Maps of DNA". In: *Bulletin of Mathematical Biology* 48.2 (1986), pp. 189 –195. ISSN: 0092-8240. DOI: http://dx.doi.org/10.1016/S0092-8240(86)80006-4. URL: http://www.sciencedirect.com/science/article/pii/S0092824086800064.

[Wu83] Tsong-Ho Wu. "An $O(n^3)$ Isomorphism Test for Circular-Arc Graphs". PhD thesis. New York, NY, USA: SUNY Stony Brook, 1983.