Characterising parameterized graph classes:

Certifying algorithms for fixed-parameter tractable problems

Samuel Stuart Wilson

Submitted in accordance with the requirements
for the degree of Doctor of Philosophy

The University of Leeds
School of Computing

May 2015

The candidate confirms that the work submitted is his/her own, except where work which has formed part of a jointly authored publication has been included. The contribution of the candidate and the other authors to this work has been explicitly indicated below. The candidate confirms that appropriate credit has been given within the thesis where reference has been made to the work of others.

Some parts of the work presented in Chapters 5, 6 and 7 have been published in the following articles:

Müller H; Wilson S An FPT certifying algorithm for the vertex-deletion problem in: Lecroq T; Mouchard L (editors) Combinatorial Algorithms, 24th International Workshop, IWOCA 2013, pp.468-472. Springer. 2013.

Müller H; Wilson S Characterising subclasses of perfect graphs with respect to partial orders related to edge contraction in: Cornelissen K; Hoeksma R; Hurink J; Manthey B (editors) Proceedings of the 12th Cologne-Twente Workshop on Graphs and Combinatorial Optimization (CTW 2013), vol. WP 13-01, pp.183-186. Centre for Telematics and Information Technology. 2013.

The above publications are primarily the work of the second author. The first author is the primary supervisor of the candidate and therefore discussions on techniques, correctness and presentation were had. The contribution of the first author is that of a primary supervisor.

Acknowledgements

I would like to thank many people for supporting me throughout the duration of my PhD. I hope that people know that their support was vital for the completion of my thesis and I request that the omission of acknowledgement of any person who feels they have supported my studies does not diminish my gratitude for their support.

Firstly, I would like to thank my supervisory team; Haiko Müller and Kristina Vušković for their support and encouragement throughout my time at the University of Leeds. In particular I would like to thank Haiko for the countless hours he has spent expanding my enjoyment of mathematics and the patient support he has provided ensuring that I meet his expectations. I would like to thank Kristina for ensuring I understand the role of rigour and clarity in my research and also for her encouragement for me to teach during my studies. I am exceedingly honoured and proud to say that I have been supervised by them.

I would like to thank the members of the School of Computing for creating a happy and productive working environment. I have had many stimulating discussions on various topics which have given me insight into new areas of study. I would like to extend my gratitude to the Algorithms and Complexity research group for revealing a multitude of interesting theoretical problems to me. I would like to acknowledge the EPSRC and the School of Computing for funding my research through a doctoral training grant, without which pursing a PhD would not have been possible.

Lastly I would like to thank my friends and family who have supported me throughout my PhD; my family for their moral support and my friends for listening to me speak, often at length, about my work. I would like to recognise my brother, Ed, for his continued support throughout by academic studies. I would like to express my complete appreciation of the members of my research lab for their general support and their willingness to engage in random, but fruitful, discussion about anything. In particularly, Elaine and Sarah who have been invaluable, managing to keep my work/life balance, mostly, to a reasonable level. Without Elaine my thesis would certainly not have reached this point. Finally I would like to thank the examiners for their effort.

Abstract

In this thesis we consider the problem of characterising parameterized graph classes. The parameterized graph classes we consider are the graph classes where there exists a small number of modifications that yields a well studied graph class. This type of graph class appears in many fields such as in computational biology, data sciences and communication networks. We prove, under some weak assumptions, that many of the classes can be characterised by a finite minimal forbidden set.

We also provide a formalisation of properties of partial orders and demonstrate that many of the results in the literature, such as the well-quasi ordering theorems of Ding, of Damaschke, and of Robertson and Seymour can also be applied to other partial orders. We prove that it is possible to form a lattice structure from the set of all partial orders on finite graphs and introduce a set of tools for inferring properties of those partial orders.

The results presented in this thesis have a number of consequences. As a result of the finite characterisation of the parameterized graph classes, we develop a generic algorithm for enumerating the minimal forbidden set for each class where the results may be applied. The enumeration enables the development of structural theorems concerning the parameterized graph classes which leads to the development of efficient algorithms. The results presented in this thesis also have applications in the field of certifying algorithms. We provide motivation for the development of fixed-parameter certifying algorithms and provide the first fixed-parameter certifying algorithm. We apply the results to the vertex deletion problem, showing a general construction for a fixed-parameter certifying algorithm to recognise the parameterized graph classes we consider.

The results of this thesis also provide characterisations for a set of subclasses of perfect graphs with respect to partial orders that include edge contraction.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

The search for the existence of infinite antichains in combinatorial objects has gained attention for many years, especially in the field of graph theory where the combinatorial objects under consideration are finite graphs. A significant reason for the interests into infinite antichains is that there are favourable algorithmic consequences. One such algorithmic application is when considering the recognition problem. Consider the partially ordered set $(\mathcal{G}, \leqslant)$ of graphs which is well-founded and contains no infinite antichains, then observe that every class $\mathcal{C}$ that is closed with respect to $\leqslant$ can be characterised by a finite set; a kuratowski-esque theorem. Alternatively, for the class $\mathcal{C}$ there is a set $\{H_0, \ldots, H_k\}$ such that every element G belongs to $\mathcal{C}$ if and only if it is free from the elements of $\{H_0, \ldots, H_k\}$, *i.e.*, for all $H \in \{H_0, \ldots, H_k\}$ we have $H \nleqslant G$. The algorithmic implication of this observation is that the recognition problem for every class $\mathcal{C}$ can be decided—assuming the testing of $\leqslant$ is decidable.

A prime example of this idea in practice is the set of all finite graphs and the minor relation (Graph Minor Theorem) [139]. The efforts of Robertson and Seymour have shown the necessary properties for the set of all finite graphs to be well-quasi ordered with respect to the minor relation and consequently any minor closed class is characterised by a finite set of minimal graphs that are not in the class. Coupling this well-quasi ordering result with a later result of Robertson and Seymour, that for each fixed H deciding if $H \leqslant G$ is computable in polynomial time, yields in a set of algorithms for deciding the membership problem for any minor closed class.

The minor relation is a strong relation and many graph classes which are of interest in a practical setting are not minor closed. In this situation it is interesting to consider different partial orders. However, weaker partial orders often do not have the properties that exclude infinite antichains and therefore the class membership algorithm outlined above is not applicable. In this case the task is left to find the minimal graphs not in the class on a class by class basis (each class has to be considered individually).

A parameterized graph class is a class of graphs that has a parameter associated with it which constraints some property of the graphs in the class; for instance, to constrain the order of the graph or the edge density. Parameterized graph classes appear often in practical applications. The graphs that represent the problem domain share common structural properties which can be used to develop efficient algorithms. It is often the case however that datasets collected from practical applications have errors included which leads to special kinds of parameterized graph classes. These graph classes are closely related to the graph modification problems.

One example of where graph classes are applied to a problem domain is that of DNA sequencing. In [12] the author provides strong evidence that the structure of bacterial genes is linear much like the structure of genes in a chromosome. This suggest there exists a linear model for gene sequences. As a result of the linear structure, the class of interval graphs seems like a natural representation for sequences of DNA [171]. During the DNA sampling process strands of DNA are divided by a chemical process that separates the strands into shorter subsequences. The bases of each subsequence are then read, resulting in a set of intervals that then have to be rearranged in order to obtain the original sequence. The subsequences can be modelled as intervals on a line and the set of intervals form an interval graph. During this process the sequences of DNA can become damaged which can cause problems when trying to reassemble the original sequence. The damage to the subsequences manifests itself in the form of additional vertices in the interval graph representation or vertices with infeasible adjacency configurations. In order to reconstruct the DNA sequence the problem is to find the minimum number of vertices that when removed yields a viable DNA sequence. This problem is known as the vertex deletion problem. The interval deletion problem, which is the vertex deletion problem relating to reconstructing DNA sequences, has been solved by Cao and Marx [21] and has been shown to be fixed-parameter tractable.

For some partial orders the parameterized graph classes we consider are, in general, not closed with respect to the partial order, however, there may be specific parameterized graph classes for which they are closed. We highlight where this is the case and collate a set of results in this area (See Chapter 4).

For the graph classes associated with the graph modification problems we prove that the minimal forbidden set is finite under certain conditions. This avoids the need for a class by class analysis and allows the results to be applied more widely. Finite minimal forbidden set characterisations are desirable as they often shed light on structural properties of the graphs belonging to the class that can be exploited to develop efficient algorithms. For the class recognition problem a finite minimal forbidden set immediately yields a polynomial time algorithm assuming that the problem of deciding if one graph is contained within another with respect to the partial order under consideration can be solved in polynomial time.

As many of the partial orders we consider do not have polynomial time algorithms for the general containment problem (testing if $G \leqslant H$ for any $G, H \in \mathcal{G}$) and as we aim for a theorem in the most general setting we are often left with the prospect of fixed-parameter tractable algorithms, where the algorithm parameter is a function of the parameter of the graph class.

A different aspect of the class membership problem is that of providing a certificate of membership. A certificate is an additional piece of information from which membership can easily be ascertained. We introduce and provide foundations for the study of certifying algorithms for fixed-parameter tractable problems. In addition we give the first certifying algorithms for a fixed-parameter tractable problem (Chapter 6).

The results of Robertson and Seymour provide proof of the existence of a finite forbidden set for any minor closed class, however a method for generating the forbidden set is in general not known. We provide a procedure for the generation of the finite minimal forbidden sets for the parameterized graph classes we consider here.

In Chapter 2 and 3 (respectively) we introduce the notation used, including where it potentially differs from the notation found in the field, and we present the current state of research in the areas that are related to the results presented here.

In Chapter 4 we present a tool for investigating properties of partial orders, building a rich algebraic structure. The presented tool is used to present a set of results which excludes certain avenues of enquiry. We also introduce a set of properties which will be used in later chapters. We present a contribution in Chapter 5, providing a technique for establishing an upper bound on the order of a minimal forbidden graph for a parameterized graph class closed with respect to a partial order. This contribution is presented in a general setting making it applicable to any partial order satisfying a set of outlined conditions. We contribute a set of applications of the results in Chapter 5 in Chapter 6, specifically we provide an algorithm for enumerating the minimal forbidden set for a graph class closed with respect to a partial order which satisfies the conditions outlined in Chapter 5. In addition to this we introduce the theory of the amalgamation of the fields of fixed-parameter algorithms and certifying algorithms and motivate why they go hand in hand. We present a general certifying fixed-parameter algorithm construction for the recognition problem of the parameterized graph classes we consider in Chapter 5.

In Chapter 7 we present a set of results for characterising well studied graph classes with respect to partial orders that include edge contraction. The results of this section highlight some of the problems that are encountered when considering characterising graph classes with respect to partial order that include edge contraction. We introduce a previously undefined partial order which overcomes some of these problems.

In Chapter 8 we contribute some partial results for the topological minor relation and demonstrate why the technique developed in Chapter 5 does not work when considering the topological minor relation. We provide a number of results where we can establish an upper bound by showing that the bound does not differ from that for other considered partial orders. Lastly we present a more general case where the graph class has a single forbidden graph with respect to the topological minor relation.

Finally in Chapter 9 we summarise the contribution to the area of research and provide a set of possible avenues of research which seem interesting and fruitful.

Chapter 2

Notation

2.1 General

A *graph* is defined as a pair, consisting of a vertex set and an edge set, $G = (V, E)$ where $E \subseteq \{\{u, v\} \mid u, v \in V\}$. We write uv to mean $\{u, v\}$ therefore $uv = vu$. We adopt the notation from [162]. For a graph $G = (V, E)$ we define $V(G) = V$ and $E(G) = E$ additionally we use the notation $V_G = V(G)$ and $E_G = E(G)$. For a graph $G = (V, E)$ let $|V_G| = n$ and $|E_G| = m$. Where an edge has an associated direction we write uv to mean the edge directed from u to v, if a graph has multiple edges between a pair of vertices the edges are called parallel and the graph is not simple. Unless otherwise stated, when we refer to a graph we mean a finite simple undirected graph.

We make no distinction between isomorphic graphs, a pair of graphs G, H are *isomorphic* if there exists a bijective function $f : V_G \to V_H$ such that for all $\{u, v\} \in E_G \Leftrightarrow \{f(u), f(v)\} \in E_H$. Graph isomorphism is denoted by $\leqslant_{\mathrm{GI}}$. Graph isomorphism is an equivalence relation partitioning the set of all graphs into *equivalence classes*. The equivalence classes are referred to as isomorphism classes.

The *open neighbourhood* $N_G(v)$ of a vertex $v \in V(G)$ is the set $\{u \mid uv \in E\}$, i.e., the vertices which are adjacent to v. The *closed neighbourhood* of v is $N_G[v] = \{v\} \cup N_G(v)$. Both concepts generalise to sets. Where $S \subseteq V$ we have $N_G(S) = \bigcup_{v \in S} N_G(v) \setminus S$ and $N_G[S] = \bigcup_{v \in S} N_G[v]$ for open and closed neighbourhoods respectively. The *degree* of a vertex $v \in V(G)$ denoted $d_G(v) = |N_G(v)|$. A vertex of degree one is called a *pendent* vertex. A vertex is called isolated if it has degree 0.

A subgraph of a graph G is a graph G' whose vertex set is a subset of $V(G)$ and whose edge set is a subset of the edges restricted to the vertices of $V(G')$, i.e., $E(G') \subseteq \{uv \mid uv \in E(G) \wedge u, v \in V(G')\}$. Conversely G is a supergraph of G'. Let $G[S]$ denote the subgraph of G induced by S where $G[S] = (S, \{uv \mid uv \in E(G) \wedge u, v \in S\})$.

A *hypergraph* is a generalisation of the concept of a graph, a hypergraph $\mathcal{H}$ is defined as a pair (V, E) such that V is the set of vertices of $\mathcal{H}$ and the hyperedge set $E \subseteq \mathcal{P}(V) \setminus \{\varnothing\}$, where $\mathcal{P}(V)$ denotes the power set of V. A hypergraph $\mathcal{H}$ is an r-uniform hypergraph if all hyperedges are of cardinality r, that is for all $e \in E(\mathcal{H})$ we have $|e| = r$. It follows from this definition that a graph is a 2-uniform hypergraph.

2.1.1 Graph operations

The following operations are elementary graph operations which are required in subsequent sections:

Complement The *complement* of a graph G, denoted by $\overline{G}$ is $(V(G), \{uv \mid u, v \in V(G) \wedge uv \notin E(G) \wedge u \neq v\})$.

Substitution is the operation of replacing a vertex with a subgraph. Given two graphs G and H where $V(G) \cap V(H) = \varnothing$ and a vertex $v \in V(G)$. $G[H/v] = (V(G) \cup V(H) \setminus \{v\}, E(G - v) \cup \{uv \mid u \in N_G(v) \wedge w \in V(H)\} \cup E(H))$.

Disjoint union is the operation of combining two or more disjoint graphs. Given two graphs, G and H, the disjoint union $G \uplus H = (V(G) \cup V(H), E(E) \cup E(H))$ given that $V(G) \cap V(H) = \varnothing$. If $V(G) \cap V(H) \neq \varnothing$ then it is required that $V(G)$ is relabelled to meet the criteria that $V(G) \cap V(H) = \varnothing$. When G is isomorphic to H, we write $2G$ or $2H$ to have the meaning of $G \uplus H$. This extends to kG where $k \geqslant 1$ is an integer, $kG = \biguplus_{i=1}^{k} G$.

Union is the summation of two graphs, given G and H, $G \cup H = (V(G) \cup V(H), E(G) \cup E(H))$. Extending to a set of graphs $\mathfrak{Q} = \{Q_0, Q_1, \ldots, Q_k\}$, $\bigcup \mathfrak{Q} = Q_0 \cup Q_1 \cup \cdots \cup Q_k$.

Intersection is the operation of taking the intersection of two graphs, given G and H, $G \cap H = (V(G) \cap V(H), E(G) \cap E(H))$. Extending to a set of graphs $\mathfrak{Q} = \{Q_0, Q_1, \ldots, Q_k\}$, $\bigcap \mathfrak{Q} = Q_0 \cap Q_1 \cap \cdots \cap Q_k$.

Join is the binary operation of combining two graphs. Given the graphs G and H, $G \bowtie H = (V(G) \cup V(H), E(G) \cup E(H) \cup \{uv \mid u \in V(G), v \in V(H)\}$, or $G \bowtie H = \overline{\overline{G} \uplus \overline{H}}$.

Vertex operations

Vertex deletion is the operation of deleting a vertex from a graph. Given a graph G and a vertex $v \in V(G)$, $G - v = (V \setminus \{v\}, E \setminus \{uv \mid uv \in E(G)\})$. For a non-empty set $S \subseteq V(G)$, $G - S = \bigcap_{v \in S}(G - v)$. For $S = \varnothing$, $G - S = G$

Vertex addition is the operation of adding a vertex to a graph. Given a graph G and a vertex $v \notin V(G)$, $G \uplus v = (V(G) \cup \{v\}, E(G))$. For a set $S = \{s \mid s \notin V\}$, $G \uplus S = \bigcup_{v \in S} G \uplus v$.

Local complement is the operation of complementing the subgraph induced by the open neighbourhood of a vertex v. $G * v = (V(G), (E(\overline{F}) \cup E(G)) \setminus E(F))$ where $F = G[N_G(v)]$.

Local-complement-deletion is the operation of performing a local complement then deleting the vertex. Let G be a graph and let $v \in V(G)$, then $G \bullet v = (G * v) - v$.

Vertex dissolution Let G be a graph and $v \in V(G)$ such that $\deg(v) = 2$ and let $vu, vw \in E(G)$ then the operation of vertex dissolution is defined as deleting the vertex v and introducing the edge uw. This operation may introduce parallel edges; however, if the graphs being considered are simple then parallel edges are removed.

Inverse subdivision Let G be a graph and $v \in V(G)$ such that $\deg(v) = 2$ and let $vu, vw \in E(G)$ and $uw \notin E(G)$ then the operation of inverse subdivision is defined as deleting the vertex v and introducing the edge uw.

Vertex absorption Let G be a graph and $u, v \in V(G)$ such that $N_G[u] \subseteq N_G[v]$ then $G \between u = G - u$ which is equivalent to contracting the edge uv in a simple undirected graph.

Edge operations

Edge deletion is the operation of deleting an edge from a graph. Given a graph G such that $V(G) \cap E(G) = \varnothing$ and an edge e such that $e \in E(G)$. We define $G \backslash e = (V(G), E(G) \backslash \{e\})$. This operation extends to sets, given a non-empty set $S \subseteq E(G)$, $G \setminus S = \bigcap_{e \in S}(G \setminus e)$. For $s = \varnothing$, $G \setminus S = G$.

Edge addition is the operation of adding an edge to a graph. Given a graph G and an edge $e = uv$ such that $e \notin E(G)$ and $u, v \in V(G)$, $G + e = (V(G), E(G) \cup \{e\})$. This operation extends to sets, given a set $S = \{uv \mid u, v \in V(G) \wedge uv \notin E(G)\}$, $G + S = (V(G), E(G) \cup S)$.

Pivoting Let G be a graph and $uv \in E(G)$. Pivoting is denoted $G \times uv$ and is defined as $G \times uv = G * u * v * u$. Sometimes referred to as edge local complement.

Edge contraction Let G be a graph and $e = uv \in E(G)$. The graph G/e is the result of contracting the edge e in G and is defined as $((V(G) \cup \{w_{uv}\}) \setminus \{u, v\}, (E(G) \setminus (e \cup \{ab \mid a \in N_G(u) \cup N_G(v)\})) \cup \{\{w_{uv}, c\} \mid c \in \{N_G(u) \cup N_G(v)\}\})$ where $w_{uv} \notin V(G)$. Note that the operation of edge contraction is commutative [166]. If contracting an edge introduces parallel edges then the graph is reduced to a simple graph.

Subdivision is the operation of dividing an edge into two edges and introducing a new vertex. Let G be a graph and $e = uv \in E(G)$. $G \mathbin{-\!\bullet\!-} e = (V(G) \cup \{w_{uv}\}, \{uw_{uv}, vw_{uv}\} \cup (E(G) \backslash \{e\}))$ where $w_{uv} \notin V(G)$.

Edge lifting is the operation of deleting two adjacent edges with a common endpoint and adding an edge connecting the two endpoints. Given a graph G and the edges $uv, vw \in E(G)$ assuming $u \neq w$. $G \smile \{uv, vw\} = (V(G), (\{uw\} \cup E(G)) \setminus \{uv, vw\})$.

2.2 Relations

A *preorder* (or *quasi order*) on a set X is a subset of the Cartesian product $X \times X$ with the reflexive and transitive properties, that is

- reflexive $\forall u \in X \mid u \leqslant u$, and
- transitive $\forall u, v, w \in X \mid (u \leqslant v) \ \wedge \ (v \leqslant w) \implies u \leqslant w$.

The set X is referred to as the *ground set*. Two elements $x, y \in X$ are *incomparable* if $x \nleqslant y$ and $y \nleqslant x$, this is denoted as $x||y$. If two elements are not incomparable then they are *comparable*. An element x of a preordered set X is *minimal* if for all $y \in X$, $y \leqslant x$ implies $x \leqslant y$. An element $z \in X$ is *maximal* if for all $y \in X$, $z \leqslant y$ implies $y \leqslant z$. A preordered set can have multiple maximal and minimal elements or none. A *chain* is a set of elements of a preordered set such that any two elements are comparable. As a consequence of transitivity, given a chain $C = \{c_0, c_1, \ldots\}$ for any i, j we have $c_i \leqslant c_j$ where $i \leq j$. An *antichain* is a subset of the elements of a preordered set such that every pair of elements is incomparable.

A chain $C = \{c_0, c_1, \ldots\}$ is a *tight chain* in a preorder $\leqslant$ with set X if for all $0 \leqslant i$ and for all $y \in X$, $c_i \leqslant y \leqslant c_{i+1}$ implies $y \leqslant c_i$ or $c_{i+1} \leqslant y$.

An irreflexive order $<$ is an order such that $H \nless H$ for all H. A preorder is a *total order* if for all $x, y \in X$ we have $x \leqslant y$, $y \leqslant x$ or both. Given two preorders $\leqslant_1$ and $\leqslant_2$ on the set X, we call $\leqslant_2$ an *extension* of $\leqslant_1$ if $\leqslant_1 \subseteq \leqslant_2$, we also call $\leqslant_1$ a *restriction* of $\leqslant_2$.

A preorder that is also antisymmetric is called a *partial order*, the antisymmetric property is:

- $\forall u, v \in X \mid (u \leqslant v) \ \wedge \ (v \leqslant u) \implies u = v$.

Each partial order has a corresponding *strict* partial order, denoted by $<$. A strict partial order is a binary relation that is

- irreflexive $\forall u \in X \ u \nless u$,
- transitive $\forall u, v, w \in X \mid (u < v) \ \wedge \ (v < w) \implies u < w$ and
- asymmetric $\forall a, b \in X \mid a < b \implies b \nless a$.

The strict partial order of any partial order $\leqslant$ is the reflexive reduction of that partial order, that is, the strict partial order $<$ of $\leqslant$ is the strict partial order whose reflexive closure is equal to $\leqslant$. The partial order $\leqslant$ corresponding to a strict partial order $<$ is the reflexive closure of $<$. The dual of a partial order $\leqslant$, denoted $\geqslant$ is the partial order with elements $\{(y, x) \mid (x, y) \in \leqslant\}$. We adopt the definition of reflexive reduction, reflexive closure, transitive reduction and transitive closure from [153].

A binary relation $(\leqslant)$ on a set X is *well-founded* if and only if every non-empty subset $S \subseteq X$ contains a minimal element, that is:

$$\forall S \subseteq X \mid S \neq \varnothing \ \exists s \in S \ \forall x \in S \iff (x, s) \notin \leqslant$$

Alternatively a binary relation on a set X is *well-founded* if and only if X does not contain an infinite sequence $x_0, x_1, \dots$ such that for all $i \geq 0$ we have $x_{i+1} \leqslant x_i$.

Definition 1. A preorder (partial order) is well-founded if and only if the corresponding strict preorder (partial order) contains no infinite descending chains.

2.2.1 Well-quasi ordering

A *well-quasi ordering* is a reflexive and transitive binary relation, *i.e.*, a *quasi ordering*, on a set X such that for every infinite sequence $x_0, x_1, \dots \in X$ there are indices $i < j$ such that $x_i \leqslant x_j$. Any such pair (x_i, x_j) is called a *good pair.* A sequence is called a *good sequence* if the sequence has a good pair, otherwise the sequence is bad .

Corollary 2. Let X be a well-quasi ordered set then every infinite sequence is a good sequence.

Lemma 3. *[38, Proposition 12.1.1] A quasi ordering $<$ on X is a well-quasi ordering if and only if X contains neither an infinite antichain nor an infinite strictly decreasing sequence $x_0 > x_1 > \dots$*

Lemma 4. *Let $\leqslant$ be a quasi order on the set X, the following definitions are equivalent.*

- *$\leqslant$ is a well-quasi ordering.*
- *if $x_0, x_1, \dots \in X$ then there exists $i < j$ such that $x_i \leqslant x_j$.*
- *if $x_0, x_1, \dots \in X$ then there exists an infinite subsequence in $x_{f(0)}, x_{f(1)}, \dots$ such that for all $i < j \mid x_{f(i)} \leqslant x_{f(j)}$.*

If a preorder is well-founded then for it to also be a well-quasi ordering it is sufficient that there exists no infinite antichains. This is because well-foundedness prevents the existence of infinite descending chains.

An *ideal* in a partially ordered set $(X, \leqslant)$ is a subset of X that is closed with respect to $\leqslant$, *i.e.*, $(\mathcal{I}, \leqslant)$ is an ideal of $(X, \leqslant)$ if and only if $\mathcal{I} \subseteq X$ and if $x \in \mathcal{I}$ and $y \leqslant x$ implies $y \in \mathcal{I}$. In addition for every $x, y \in \mathcal{I}$ there exists a $z \in \mathcal{I}$ such that $x \leqslant z$ and $y \leqslant z$. For any ideal $(\mathcal{I}, \leqslant)$ of a well-quasi ordering $(X, \leqslant)$ the ideal can be expressed by excluding a finite set of elements in X. The set of excluded elements can be expressed as follows;

$$\mathcal{F} = \{x \in (X \setminus \mathcal{I}) \mid \forall y \in (X \setminus \mathcal{I}) \quad y \not< x\}.$$

As a consequence of X being well-quasi ordered by $\leqslant$ it is necessary that the set of minimal elements of $X \setminus \mathcal{I}$ is finite. The set of minimal elements of $X \setminus \mathcal{I}$ constitutes an antichain in $X \setminus \mathcal{I}$. As any antichain in $X \setminus \mathcal{I}$ is an antichain in X and all antichains in X are finite then this implies that $\mathcal{F}$ is finite.

A *filter* F in a partially ordered set $(X, \leqslant)$ is a subset of X if F is an ideal in $(X, \geqslant)$ where $\geqslant$ is the converse of $\leqslant$.

2.2.2 Partial orders on graphs

The binary relations on graphs that we consider are reflexive and transitive, that is they are preorders. As we make no distinction between isomorphism classes the relations are also anti-symmetric therefore the orders we consider are also partial orders. We will refer to the orders as partial orders. A partial order on the set of all graphs $\mathcal{G}$ defines a containment relation. A graph G is said to be contained in a graph H with respect to some partial order if $G \leqslant H$. We introduce a set of well studied partial orders on graphs (Summary provided in Table 2.1 on page 11):

Partial subgraph is defined by vertex deletion and edge deletion and is denoted by $\leqslant_{\mathrm{s}}$. Given G and H, $G \leqslant_{\mathrm{s}} H$ if $V(G) \subseteq V(H)$, $E(G) \subseteq E(H)$ and for all $uv \in E(G)$, $u, v \in V(G)$.

Induced subgraph is defined by vertex deletion and is denoted by $\leqslant_{\mathrm{i}}$. Given two graphs G and H, $G \leqslant_{\mathrm{i}} H$ if there is a set of vertices $U \subseteq V(H)$ such that the deletion of the vertices in U from H yields a graph isomorphic to G. The graph G is the subgraph of H induced by $V(H) \setminus U$.

Minor is defined by edge deletion, edge contraction and vertex deletion. The minor relation is denoted by $\leqslant_{\mathrm{m}}$. A graph G is a minor of H if G can be obtained from H by a sequence of edge deletions, edge contractions and the deletion of isolated vertices.

Induced Minor is defined by edge contraction and vertex deletion. The induced minor relation is denoted by $\leqslant_{\mathrm{e}}$. A graph G is an induced minor of H if G can be obtained from H by a sequence of edge contractions and vertex deletions.

Topological Minor is defined by edge deletion, vertex deletion and vertex dissolution. A graph H is a topological minor of a graph G if and only if a subdivision of H is isomorphic to a subgraph of G. The topological minor relation is denoted by $\leqslant_{\mathrm{t}}$.

Induced Topological Minor is defined by vertex deletion and vertex dissolution. The Induced topological minor relation is denoted by $\leqslant_{\mathrm{it}}$. A graph H is an induced topological minor of a graph G if and only if a subdivision of H is isomorphic to an induced subgraph of G.

Contraction Minor is defined by edge contraction. The contraction minor relation is denoted $\leqslant_{\mathrm{c}}$. A graph G is a contraction minor of H if G can be obtained from H by a sequence of edge contractions.

Partial contraction minor is defined by edge contraction and edge deletion.

Homeomorphic Minor is defined by inverse subdivision. A graph H is a homeomorphic minor of a graph G if a subdivision of H is isomorphic to G and is denoted by $\leqslant_{\mathrm{h}}$.

Partial homeomorphic image is defined by inverse subdivision and edge deletion.

Partial Order	Vertex deletion	Edge deletion	Edge contraction	Vertex dissolution	Pivoting	Edge lifting	Local complement
Partial subgraph	✓	✓					
Induced subgraph	✓						
Minor	✓	✓	✓				
Induced minor	✓		✓				
Topological minor	✓	✓		✓			
Induced topological minor	✓			✓			
Contraction minor			✓				
Vertex minor	✓						✓
Pivot minor	✓				✓		
Immersion minor	✓					✓	
Lift minor	✓	✓	✓			✓	
Lift contraction			✓			✓	
Graph isomorphism							

Table 2.1: Partial orders defined by graph operations

Vertex Minor is defined by local complement and vertex deletion. The vertex minor relation is denoted by $\leqslant_{\mathrm{v}}$ [127].

Pivot Minor is defined by vertex deletion and pivoting. The pivot minor relation is denoted by $\leqslant_{\mathrm{p}}$.

Immersion Minor is defined by edge lifting, vertex deletion and edge deletion. The immersion minor relation is denoted by $\leqslant_{\mathrm{I}}$.

Lift minor is defined by vertex deletion, edge deletion, edge contraction and edge lifting. The lift minor relation is denoted by $\leqslant_{\mathrm{lift}}$.

Lift contraction is defined by edge contraction and edge lifting. The lift contraction relation is denoted by $\leqslant_{\mathrm{lc}}$.

Graph isomorphism is denoted $G \leqslant_{\mathrm{GI}} H$ or $G \simeq H$. Two graphs are isomorphic if there exists a bijective function between the vertex sets that is adjacency preserving.

2.3 Graph classes

The graphs that satisfy a property define a binary partition of the set $\mathcal{G}$ of all graphs. The partition defines a graph class and its complement. Let $\mathcal{C}$ be the class of graphs satisfying a property and let $\overline{\mathcal{C}} = \mathcal{G} \setminus \mathcal{C}$. It is equivalent to say that a graph has a property and a graph belongs to a graph class. The property defines the class of graphs which have the property and the complement of the class, *i.e.*, those graphs that do not satisfy the property.

For any graph class $\mathcal{C}$ which is closed with respect to some partial order the class can be defined by forbidding a (possibly infinite) set of graphs. Trivially for any class $\mathcal{C}$ closed with respect to a partial order the forbidden set is defined as $\mathcal{G} \setminus \mathcal{C}$ (Theorem 5). Given a set $X \subseteq \mathcal{G}$ then $\text{minimal}(X) = \{x \mid x \in X \ \wedge \ \forall y \in X \ \wedge \ y \neq x \ \Rightarrow \ y \nleqslant x\}$; that is, minimal$(X)$ is the set of minimal elements of X with respect to some partial order. The *minimal* forbidden set for a class $\mathcal{C}$ is the set of graphs minimal$(\mathcal{G} \setminus \mathcal{C})$. The minimal forbidden set is denoted $\mathcal{F}$, where it is potentially ambiguous as to which partial order this is relating the minimal forbidden set is subscripted with the initial of the partial order, *e.g.* $\mathcal{F}_{\text{c}}$ denotes the minimal forbidden set with respect to the contraction minor partial order. For a class $\mathcal{C}$ with forbidden set $\mathcal{F}$ the class $\mathcal{C}$ can be described as being $\mathcal{F}$-free. The set $\mathcal{F}$ forms an antichain with respect to the partial order under consideration. For the class $\mathcal{C}$, if $\mathcal{C}$ is $\mathcal{F}$-free then $\mathcal{F} = \text{Forb}(\mathcal{C})$. The minimal forbidden set is often referred to as the obstruction set.

Theorem 5. *Any graph class $\mathcal{C}$ closed with respect to a partial order $\leqslant$ can be characterised by a forbidden set.*

Proof. Let $\mathcal{C}$ be a class closed with respect to $\leqslant$ and let $\overline{\mathcal{C}}$ denote the complement of $\mathcal{C}$. Since $\mathcal{C}$ is closed with respect to $\leqslant$ then for all $G \in \overline{\mathcal{C}}$ and $G \leqslant H$ implies that $H \in \overline{\mathcal{C}}$ leads to the conclusion that $G \in \mathcal{C}$ if and only if G does not contain a member of $\overline{\mathcal{C}}$ with respect to $\leqslant$. The set $\overline{\mathcal{C}}$ is a forbidden set of $\mathcal{C}$. □

If the partial order is well-founded then the minimal elements of the forbidden set uniquely characterises the graph class. The set obtained in Theorem 5 is not necessarily minimal with respect to the partial order. The minimal forbidden set is obtained by taking the minimal elements of the complement of $\mathcal{C}$. If the partial order is not well-founded then there may not be minimal elements of the complement of $\mathcal{C}$ and therefore the minimal forbidden set is the empty set. Given a set of graphs then that set of graphs defines a class of graphs with respect to any well-founded partial order. The class is the class of graphs that excludes that set of graphs with respect to $\leqslant$. If the set is minimal then the set is unique. Note that the antisymmetric property is required in order for the minimal set to be unique.

Theorem 6. *The minimal forbidden set for a class $\mathcal{C}$ closed with respect to a well-founded partial order $\leqslant$ is unique.*

Proof. The minimal forbidden set for a class $\mathcal{C}$ is defined as the minimal elements of the complement of $\mathcal{C}$. The definition of minimal elements in $\mathcal{G} \setminus \mathcal{C}$ is well defined as $\leqslant$ is well-founded. □

Corollary 7. For any graph class $\mathcal{C}$ closed with respect to a well-founded partial order $\leqslant$ if $G \notin \mathcal{C}$ then there exists a graph $H \in \text{Forb}(\mathcal{C})$ such that $H \leqslant G$.

A class is said to be *interesting* if there exists an infinite set of graphs belonging to the class and an infinite set of graphs not in the class. A class is *non-trivial* if there exists at least one graph which belongs to the class and the class is a proper subset of the set of all graphs.

A graph class is *monotone* if it is closed with respect to the partial subgraph relation, *i.e.*, the membership of the class is preserved under deleting vertices and edges. The class of planar graphs is monotone whereas the class of complete graphs is not, as the deletion of an edge from a complete graph does not yield a complete graph. A class is *hereditary* if the class is closed with respect to the induced subgraph relation, *i.e.*, the membership of the class is preserved under deleting vertices. It is easy to see that every monotone class is a hereditary class but not vice versa.

We introduce a set of graph classes and their notation. The classes relate to standard graph class definitions and well studied graph classes relating to perfect graphs. See also [17].

Trees

A *tree* is a connected graph with no cycles. Alternatively a tree is a connected graph on n vertices with $n-1$ edges.

Forests

A *forest* is the disjoint union of a set of trees.

Complete graphs

A *complete* graph on n vertices, denoted by K_n, is the graph G where $V(G) = \{v_i \mid 0 \leqslant i < n\}$ and $E(G) = \{\{u, v\} | v \neq u, \forall u, v \in V(G)\}$. A complete subgraph of a graph is called a *clique* (see Figure 2.1).

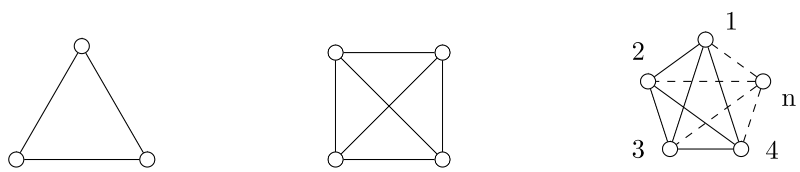


Figure 2.1: Complete graphs: K_3, K_4 and K_n

Bipartite graphs

A *bipartite* graph is a graph where the vertices can be partitioned into sets such that there is no edge between vertices in the same set. Alternatively the class of bipartite graphs forbids cycles of odd length with respect to the induced subgraph relation [7, Theorem 2.1.3].

Complete bipartite graphs

A *complete bipartite* graph is a bipartite graph with the vertex set partition X, Y and

$\forall x \in X\, \forall y \in Y\ \{x, y\} \in E$. Complete bipartite graphs are denoted $K_{a,b}$ where $a = |X|$ and $b = |Y|$.

Path graphs

A *path* graph on n vertices, denoted by P_n, is the graph G where $V(G) = \{v_i \mid 0 \leqslant i < n\}$ and $E(G) = \{\{v_i, v_{i+1}\} \mid 0 \leqslant i < n-1\}$ (see Figure 2.2).

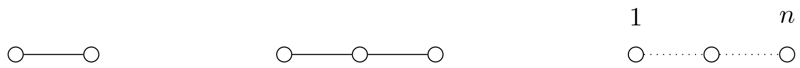


Figure 2.2: Path graphs: P_2, P_3 and P_n

Cycle graphs

A *cycle* graph on n vertices, denoted by C_n, is the graph $P_n + \{v_{n-1}, v_0\}$ (see Figure 2.3).

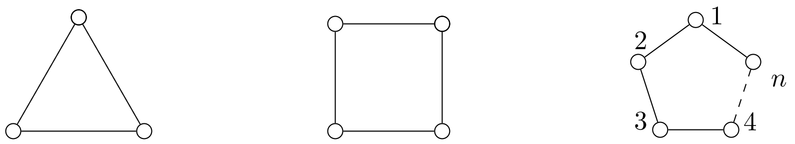


Figure 2.3: Cycle graphs: C_3, C_4 and C_n

Star graphs

A *star* graph on n vertices, denoted by $K_{1,n}$ is a complete bipartite graph where the vertex set partition consists of a single vertex and a set of n vertices (see Figure 2.4).

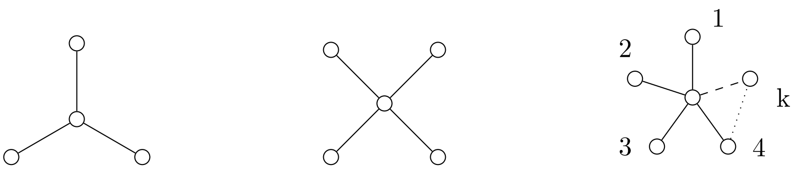


Figure 2.4: Star graphs: $K_{1,3}, K_{1,4}$ and $K_{1,k}$

Grid graphs An $n \times m$-grid graph is the graph on $\{1, \ldots, n\} \times \{1, \ldots, m\}$ with edge set $\{(i, j)(i', j') \mid |i - i'| + |j - j'| = 1\}$ (see Figure 2.5).

Perfect graphs

A *perfect* graph is a graph where the minimum number of colours required to colour the vertices, so that no two adjacent vertices are assigned the same colour, of every induced subgraph is equal to its size of the largest clique [72].

Wheel graphs

A *wheel* graph is a cycle with the addition of a vertex adjacent to all vertices of the cycle.

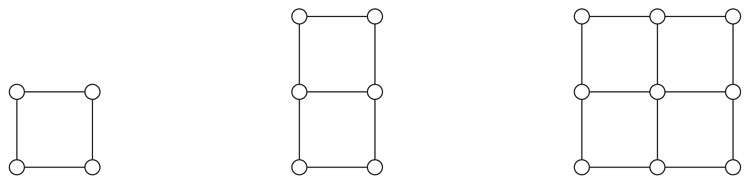


Figure 2.5: 2×2-grid, 2×3-grid and 3×3-grid respectively.

The wheel graph is notated W_n which consists of a cycle of length n with a dominating vertex. Alternatively $W_n = K_1 \bowtie C_n$ (see Figure 2.6).

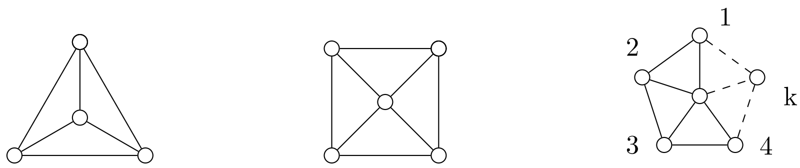


Figure 2.6: Wheel graphs: W_3, W_4 and W_k

Planar graphs

A *planar* graph is a graph that can be embedded on a plane without the edges intersecting. The property of planarity is closed with respect to many well studied partial orders, most notably the topological minor and minor relation leading to the characterisation of planar graphs as being $\{K_5, K_{3,3}\}$-free (see Figure 2.7) with respect to the minor and topological minor relations [106, 159].

Figure 2.7: Minimal forbidden planar graphs: $K_5, K_{3,3}$

Chordal graphs

A graph is a *chordal graph* if every cycle of length greater than 3 has an edge incident to two non consecutive vertices on the cycle. The class was first described by Hajnal and Surányi in [78] and has since had many different characterisations including; a graph is chordal if and only if it is $\{C_n \mid n \geq 4\}$-free with respect to induced subgraphs and a graph is chordal if and only if it has a perfect elimination ordering [61]. A *perfect elimination ordering* is an ordering of the vertices of a graph $(v_0, \ldots, v_n)$ such that for each vertex $v_i \in \{v_0, \ldots, v_n\}$ the neighbours of v_i that occur after v_i in the ordering form a clique (see Figure 2.8).

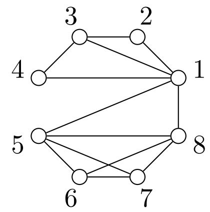


Figure 2.8: An example of a perfect elimination ordering. The elimination ordering of a graph may not be unique, the vertex ordering $(2, 3, 4, 1, 5, 6, 7, 8)$ is a perfect elimination ordering of the graph above.

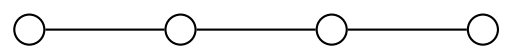


Figure 2.9: Minimal forbidden cograph: P_4

Co-chordal graphs

A graph is a *co-chordal graph* if it is the complement of a chordal graph. Co-chordal graphs have a forbidden set characterisation with respect to the induced subgraph relation. A graph is a co-chordal graph if it is $\{\overline{C_n} \mid n \geq 4\}$-free$_\mathrm{i}$.

Complement-reducible graphs

A *complement-reducible graph* (*cograph*) is a graph which can be constructed from the following basic operations:

- K_1 is a cograph.
- The disjoint union of two cographs is a cograph.
- The complement of a cograph is a cograph.

With respect to the induced subgraph relation cographs are $\{P_4\}$-free [27] (see Figure 2.9). Every cograph can be represented by a *cotree* [27]. A cotree is a tree where the internal nodes are denoted as either join or union nodes, the leaves of a cotree represent the vertices of the cograph. Nodes denoted as union nodes indicate the disjoint union of the children of that node and join nodes indicate the join ($\bowtie$) of all vertices in the children of that node.

Interval graphs

A graph is an *interval graph* if it can be represented by a set of line segments on the real line where each line segment represents a vertex and two vertices are adjacent if their corresponding line segments intersect. Every interval graph can be represented as an *interval model* where the vertices correspond to intervals. The set of intervals constitutes an interval model of the graph (see Figure 2.10). The class of interval graphs can be characterised as the set of graphs that are both chordal and co-comparability graphs [36]. With respect to the induced subgraph relation interval graphs are $\{C_{n+4}, T_2, X_{31}, XF_2^{n+1}, XF_3^n\}$-free where $n \geq 0$ (see [66]) (see Figure 2.10).

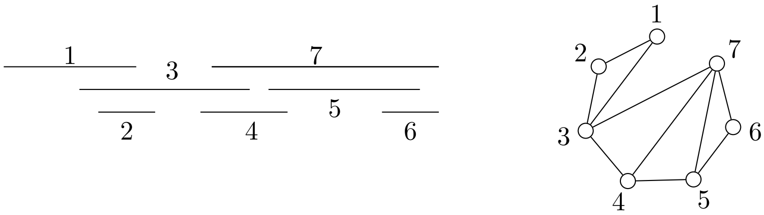


Figure 2.10: An example of an interval graph (left) and a corresponding intersection model (right).

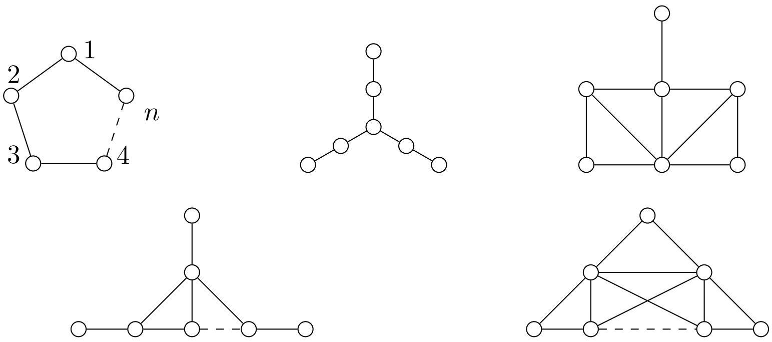


Figure 2.11: Minimal forbidden induced subgraphs for the class of interval graphs. C_n, T_2, XF_2^{n+1}, XF_3^n for $n \geq 0$.

Interval graphs are closed with respect to the induced minor relation and hence permit a characterisation, moreover, interval graphs have a finite forbidden set with respect to induced minors. The minimal forbidden induced minors are shown in Figure 2.12 adapted from [43].

Comparability graphs

A graph $G = (V, E)$ is a *comparability graph* if there is a strict partial order $(V(G), <)$ such that $uv \in E(G)$ if and only if $u < v$ or $v < u$. Comparability graphs are also known as transitively orientable graphs. The class has a forbidden set characterisation which is

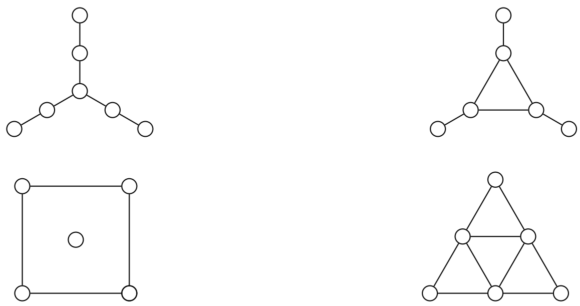


Figure 2.12: Minimal forbidden interval graphs with respect to induced minors

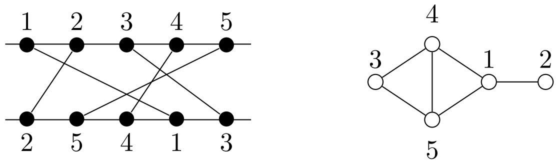


Figure 2.13: An example of a permutation graph (right) and a corresponding permutation model (left), $(1, 2, 3, 4, 5)$ $(2, 5, 4, 1, 3)$.

described by Gallai in [63] with respect to the induced subgraph relation.

Co-comparability graphs
A graph $G = (V, E)$ is a *co-comparability graph* if there is a strict partial order $(V(G), <)$ such that $uv \in E(G)$ if and only if $u||v$. The class has a forbidden set characterisation [63] with respect to the induced subgraph relation.

Permutation graphs A graph is a *permutation graph* if the graph models the inversions in a permutation, *i.e.*, the vertices represent elements of the ground set of the permutation and two vertices are adjacent if and only if the permutation reverses the natural ordering of the two corresponding elements. It is know that a graph is a permutation graph if and only if both the graph and its complement are comparability graphs [129]. Permutation graphs are closed with respect to induced subgraphs and therefore admit a forbidden set characterisation [63]. If a graph is a permutation graph then there exists a permutation model, which consists of two linear vertex orderings $(v_1, \ldots, v_n)$ and $(\pi(v_1), \ldots, \pi(v_n))$ such that two vertices v_i, v_j are adjacent if and only if v_i is before v_j in exactly one of the orderings (see Figure 2.13).

Trivially perfect graphs
A graph is *trivially perfect* if it is a cograph and an interval graph, that is the class is the intersection of the class of cographs and the class of interval graphs. A trivially perfect graph is easily observed to be a perfect graph as for every induced subgraph the size of the largest independent set is equal to the number of maximal cliques. With respect to the induced subgraph relation trivially perfect graphs are $\{P_4, C_4\}$-free (see Figure 2.14) [71].

Figure 2.14: Minimal forbidden induced subgraphs for trivially perfect graphs; P_4, C_4.

Co-trivially perfect graphs

A graph is *co-trivially perfect* if it is the complement of a trivially perfect graph. With respect to the induced subgraph relation co-trivially perfect graphs are $\{2K_2, P_4\}$-free (see Figure 2.15).

Figure 2.15: Minimal forbidden induced subgraphs for co-trivially perfect graphs; $2K_2, C_4$.

Threshold graphs

A graph is a *threshold graph* if and only if it can be constructed from the following basic graph operations:

- The graph $(\varnothing, \varnothing)$ is a threshold graph.
- The addition of an isolated vertex to a threshold graph is a threshold graph.
- The addition of a vertex adjacent to all other vertices in a threshold graph is a threshold graph.

An alternative definition for threshold graphs, and the origin of the class name, is a graph is a threshold graph if and only if there exists a real number s and a function $w : V \to \mathbb{R}$ such that if $\forall uv \in E$ then $w(u)+w(v) \geq s$. With respect to the induced subgraph relation, threshold graphs are $\{2K_2, C_4, P_4\}$-free (see Figure 2.16). The class is also closed under the operation of graph complement.

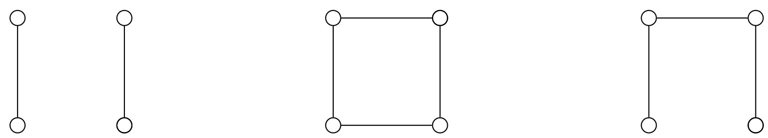


Figure 2.16: Minimal forbidden induced subgraphs for threshold graphs; $2K_2, C_4, P_4$.

Split graphs

A graph is a *split graph* if the graph permits a partition of the vertex set into two parts; one which induces a complete graph and one which induces an edgeless graph. Split graphs are the intersection of chordal graphs and co-chordal graphs, as a result of this definition split graph can be characterised as $\{2K_2, C_4, C_5\}$-free with respect to the induced subgraph relation (see Figure 2.17) [58]. The class is also closed under the operation of graph complement.

Knotless graphs

A graph is *knotless* if it can be embedded into a three dimensional space where every cycle

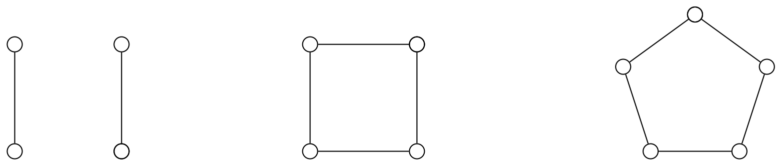


Figure 2.17: Minimal forbidden induced subgraphs for split graphs; $2K_2, C_4, C_5$.

is unknotted. The class of knotless graphs is closed with respect to the minor relation and therefore has a finite minimal forbidden set. A characterisation of knotless graphs with respect to the minor relation is unknown [2].

2.3.1 Parameterized graph classes

The concept of parameterized graph classes allows a graph class to be defined generally where for each value of the parameter the class of graphs is potentially an infinite set of graphs. Parameterized graph classes have interesting applications in complexity theory because for some problems the complexity class changes for different values of the parameter, the interesting problem in these cases is to establish the point at which the complexity classes changes. We introduce a set of parameterized graph classes and their notations.

k-connected graphs

A graph G is connected if there exists a path between every pair of vertices in $V(G)$. A graph is k-connected if there is a set of vertices $U \subset V(G)$ such that $|U| = k$ and $G \setminus U$ is either a disconnected graph or has one vertex. An equivalent definition for k-connected graphs is that for any pair of vertices there exist k vertex disjoint paths, this definition is Menger's theorem [121].

Graphs of bounded treewidth

Treewidth is defined in Section 2.4 on page 21. A class of graph $\mathcal{C}$ has bounded treewidth if for all $G \in \mathcal{C}$ we have $\text{tw}(G) \leq k$ for some value k. The class of graphs of bounded treewidth is the set $\{G \mid \text{tw}(G) \leq k\}$ where k is the parameter.

k-apex graphs

The class of *apex graphs* is related to the class of planar graphs. A graph is an apex graph if there is a vertex so that the removal of the vertex results in a planar graph. This generalises to the class of *k-apex graphs* which is the class of graphs where there exists a set of k vertices so that the removal of these k vertices result in a planar graph, an alternative notation for the class k-apex is $\{K_5, K_{3,3}\}\text{-free}_{\leqslant_m}{+}k\mathsf{v}$ [17].

$\mathcal{C}{+}k\mathsf{v}$

For a class $\mathcal{C}$, the class $\mathcal{C}{+}k\mathsf{v}$ is defined inductively as;

$$\mathcal{C}{+}k\mathsf{v} = \{G \mid \exists u \in V(G)\,(G - u) \in \mathcal{C}{+}(k-1)\mathsf{v} \vee G \in \mathcal{C}{+}(k-1)\mathsf{v}\}$$

for $k \geq 1$. For $k = 0$, $G \in \mathcal{C}{+}k\mathsf{v}$ if and only if $G \in \mathcal{C}$. Alternatively the class can be defined as;

$$\mathcal{C}{+}k\mathsf{v} = \{G \mid \exists U \subseteq V(G) \wedge |U| \leq k \wedge (G - U) \in \mathcal{C}\}.$$

$\mathcal{C}{+}k\mathsf{e}$

For a class $\mathcal{C}$ the class $\mathcal{C}{+}k\mathsf{e}$ is defined inductively as;

$$\mathcal{C}{+}k\mathsf{e} = \{G \mid \exists u \in E(G)\,(G \setminus u) \in \mathcal{C}{+}(k-1)\mathsf{e} \;\vee\; \mathcal{C}{+}(k-1)\mathsf{e}\}$$

for $k \geq 1$. For $k = 0$, $G \in \mathcal{C}{+}k\mathsf{e}$ if and only if $G \in \mathcal{C}$. Alternatively the class can be defined as;

$$\mathcal{C}{+}k\mathsf{e} = \{G \mid \exists U \subseteq E(G) \wedge |U| \leq k \wedge (G \setminus U) \in \mathcal{C}\}.$$

$\mathcal{C}{-}k\mathsf{e}$

For a class $\mathcal{C}$ the class $\mathcal{C}{-}k\mathsf{e}$ is defined inductively as;

$$\mathcal{C}{-}k\mathsf{e} = \{G \mid \exists u \in E(\overline{G})\,(G - u) \in \mathcal{C}{-}(k-1)\mathsf{e} \;\vee\; G \in \mathcal{C}{-}(k-1)\mathsf{e}\}$$

for $k \geq 1$. For $k = 0$, $G \in \mathcal{C}{-}k\mathsf{e}$ if and only if $G \in \mathcal{C}$. Alternatively the class can be defined as;

$$\mathcal{C}{-}k\mathsf{e} = \{G \mid \exists U \subseteq E(\overline{G}) \wedge |U| \leq k \wedge (G + U) \in \mathcal{C}\}.$$

2.4 Width parameters

Treewidth

Treewidth is a measure for comparing the structure of some arbitrary finite graph with the structure of a tree, allowing tree properties to be used with respect to general graphs. A *tree-decomposition* of a graph $G = (V, E)$ is a pair (X, T) where $X = \{X_0, X_1, \ldots, X_n\}$ and $X_i \subseteq V$ for all $0 \leq i \leq n$ and T is a tree on the vertex set X such that the following conditions are satisfied:

- $\bigcup_{i=0}^{n} X_i = V$
- $\forall uv \in E(G)$ there exists an $X_i \in X$ such that $u, v \in X_i$
- $\forall X_i, X_j \in V(T)$ if $u \in X_i$ and $u \in X_j$ then all vertices of the tree that lie on the unique path between X_i and X_j also contain u.

The sets X_i for all $0 \leq i \leq n$ are referred to as *parts* or *bags*. The width of the tree-decomposition is $\max(|X_i| - 1 \mid i \in I)$. The *treewidth* of a graph G, denoted $\mathrm{tw}(G)$, is the minimum width over all tree decompositions of G, $\min(\max(|X_i| - 1 \mid i \in I))$ [133, 79].

Pathwidth

Pathwidth is a measure for comparing the structure of some arbitrary finite graph with the structure of a path graph. Pathwidth relies on the definition of a path-decomposition. A *path-decomposition* of a graph $G = (V, E)$ is a pair (X, P) where $X = \{X_0, X_1, \ldots, X_n\}$ is a sequence and $X_i \subseteq V$ and P is a path on the vertex set X such that the following conditions are satisfied:

- $\bigcup_{i=0}^{n} X_i = V$
- $\forall uv \in E(G)$ there exists an $X_i \in X$ such that $u, v \in X_i$
- $\forall X_i, X_j, X_k \in V(P)$ where $i \leq j \leq k$, $X_i \cap X_k \subseteq X_j$

The width of the path-decomposition is $\max(|X_i| - 1 \mid i \in I)$. The *pathwidth* of a graph G, denoted $\mathrm{pw}(G)$, is the minimum over all path-decompositions of G, $\min(\max(|X_i| - 1 \mid i \in I))$ [132].

Clique-width

Clique-width, defined in [31], of a graph is the minimum number of labels needed to construct a graph using the following operations:

1. Creation of a new vertex with label i,
2. Disjoint union of two labelled graphs,
3. Addition of edges between all vertices of label i to all vertices of label j, and
4. Relabelling all vertices with label i to label j.

Branchwidth

The definition of *branchwidth* requires the definition of a branch-decomposition. A *branch-decomposition* can be represented by an unrooted binary tree T and a bijective function between the leaves of T and the edges of the graph G. For every edge $e \in T$, the components of $T \setminus e$ induces a bipartition of the set of leaves of T The width of an edge e is the number of vertices of G that are adjacent to an edge in $E(G_1)$ and $E(G_2)$ where G_1, G_2 are components of $T \setminus e$. The width of the branch-decomposition is $\max(\text{width of } e \mid \forall e \in T)$. The *branchwidth* of a graph G, denoted $\mathrm{bw}(G)$, is the minimum width over all branch-decompositions of G [146].

Rankwidth

The definition of *rankwidth* requires the definition of a rank-decomposition. A *rank-decomposition* of a graph G is a pair (T, L) where T is a tree with maximum degree 3 and L is a bijection between $V(G)$ and the leaves of T. For any edge $uv \in E(T)$, $T \setminus uv$ produces two connected components C_u, C_v such that $u \in C_u$ and $v \in C_v$. Let V_1, V_2 be two disjoint subsets of $V(G)$, let N_{V_1,V_2} be a $|V_1| \times |V_2|$ matrix whose rows are labelled by V_1 and columns are labelled by V_2 where the entry relating the pair (v_1, v_2) is 1 if and only if $v_1v_2 \in E(G)$. The *cutrank* of the bipartition V_1, V_2 denoted $\rho_G(V_1, V_2) = \text{rank}(N_{V_1,V_2})$.

The width of a rank-decomposition is defined as $\max_{uv\in E(T)} \rho_G(C_u, C_v)$. The rank-width is the minimum over all rank-decompositions;

$$\min_{(T,L)} \max_{uv\in E(T)} \rho_G(C_u, C_v).$$

2.4.1 Connections between width parameters

Some of the width parameters provide upper and lower bounds for other width parameters. These relations are useful as they prevent bounds having to be proved for each width parameter on each class. The definition of pathwidth is similar to that of treewidth adding the additional restriction that the underlying tree must be a path therefore treewidth is bounded from above by pathwidth. The clique-width parameter is monotone with respect to the induced subgraph relation. It has been shown that clique-width is bounded from above by a function of treewidth, namely $\text{cw}(G) \leq 3 \cdot 2^{\text{tw(G)-1}}$ [29]. Clique-width has also been shown to be bounded from above and below by a function of rankwidth, $\text{rw}(G) \leq \text{cw}(G) \leq 2^{\text{rw}(G)+1} - 1$ and furthermore there exists a polynomial time algorithm that produces a clique-width expression of at most $2^{k+1} - 1$ given a rank-decomposition of rankwidth k [127]. It has been shown by Oum that $\text{rw}(G) \leq \text{bw}(G)$ [126] and this implies that $\text{rw}(G) \leq \text{tw}(G) + 1$ assuming that $\text{bw}(G) \neq 0$ because $\text{bw}(G) \leq \text{tw}(G) + 1$. There is a strong relation between branchwidth and treewidth, Robertson and Seymour show that they are related by a constant factor of 3/2 [136].

2.5 Fixed-parameter tractability

Fixed-parameter tractability (FPT) is a developing field within computational complexity and aims to provide a method of classifying real world problems which are NP-hard. In traditional computational complexity theory the running time of an algorithm is measured as a function of the size of the input, this yields results that classify some problems as intractable. For some problems there exist natural parameters such that these problem can be solved in $f(k) \cdot n^{O(1)}$, on inputs of length n, that is the algorithm runs in polynomial time with respect to the size of the input and there exists an arbitrary computable function with k as a parameter with the condition that the degree of the polynomial is independent of k.

As with traditional complexity theory, FPT problems are modelled as strings over a finite alphabet Σ. Each instance is modelled as a pair representing the problem and the parameter, it is then a case of recognising the language of yes instances.

Definition 8. A parameterized problem $L \subseteq \Sigma^* \times \Sigma^*$ is FPT if there is an algorithm that correctly decides, for input $(x, y) \in \Sigma^* \times \Sigma^*$, whether $(x, y) \in L$ in time $f(k) \cdot n^{\alpha}$, where $n = |x|$, $k = |y|$ and α is independent of k, and f is an arbitrary function [43].

An algorithm that computes the output for an FPT problem in $f(k) \cdot n^{O(1)}$ is called a fixed-parameter tractable algorithm.

For FPT there is an internal hierarchy of classes called the $\mathcal{W}$-hierarchy, it is thought that each class represents a distinct class of problems such that each class is contained within its successors, *i.e.*, $\mathcal{W}[i] \subseteq \mathcal{W}[i+1]$. Each class has the concept of completeness, for $\mathcal{W}[1]$ the first problem to be shown to be $\mathcal{W}[1]$-complete was the independent set problem [64, GT20] and for $\mathcal{W}[2]$ the dominating set problem [64, GT2] was shown to be $\mathcal{W}[2]$-complete. The results of the previous three problems are presented in [43]. The class $\mathbb{XP}$ defines the upper bound of the $\mathcal{W}$-hierarchy such that the entire $\mathcal{W}$-hierarchy is a subset of it, $\mathbb{XP}$ represents the class of problems where the best algorithm runs in $O(n^{f(k)})$. Problems in $\mathbb{XP}$ are generally called intractable problems.

2.6 Certifying algorithms

A certifying algorithm is an algorithm which justifies its output by providing a "proof" that the output is correct. This provides a level of confidence in the implementation of the given algorithm. A certifying algorithm produces a certificate or witness with each output. We follow the approach from [116].

Formally, a certifying algorithm takes as input an element $x \in X$ and produces $y \in Y$. It is required that the input satisfies some precondition $\varphi(x)$ such that $\varphi : X \rightarrow \{\mathrm{T}, \mathrm{F}\}$ and the pair x, y where $x \in X$ and $y \in Y$ is supposed to satisfy a postcondition $\psi(x, y)$ where $\psi : X \times Y \rightarrow \{\mathrm{T}, \mathrm{F}\}$. We say that an input x satisfies the precondition if $\varphi(x) = \mathrm{T}$ and that $\varphi(x)$ is unsatisfied otherwise. For technical reasons it is favourable to introduce a new symbol to the output set Y to indicate a violated precondition. Let the set $Y^{\perp}$ be the set of all outputs including the symbol $\perp$ to indicate a precondition violation.

Let $\mathbb{W}$ be the witness or certificate predicate such that $\mathbb{W} : X \times Y^{\perp} \times W \rightarrow \{\mathrm{T}, \mathrm{F}\}$ with preconditions/postcondition pair (φ, ψ) where W is the set of witnesses. We distinguish between three types of certifying algorithms.

Strongly certifying algorithms

A *strongly certifying algorithm* provides an output and a witness on every input $x \in X$. The algorithm produces evidence to the user that the output of the algorithm is correct, because the input/output pair satisfy the postcondition, or the input did not meet the precondition, *i.e.*, the input was illegal. A strongly certifying algorithm also indicates as to which of the two options holds.

Strong witness property: Let $(x, y, w) \in X \times Y^{\perp} \times W$ satisfy the certificate predicate then:

$$\forall (x, y, w) \qquad \begin{cases} (y = \perp \wedge \mathbb{W}(x, y, w)) \implies \neg\varphi(x) \\ (y \in Y \wedge \mathbb{W}(x, y, w)) \implies \psi(x, y) \end{cases}$$

That is, a strongly certifying algorithm terminates on all inputs $x \in X$ and provides a proof that the witness predicate is correct.

Certifying algorithms

A lesser variant of strongly certifying algorithms is that of *ordinary certifying algorithms*, the algorithm will prove that either the precondition was violated or the postcondition was satisfied but will not provide an indication of which of the two cases hold. Formally;

$$\forall (x, y, w) \qquad \begin{cases} (y = \perp \wedge \mathbb{W}(x, ywz)) \implies \neg\varphi(x) \\ (y \in Y^{\perp} \wedge \mathbb{W}(x, y, w)) \implies \neg\varphi(x) \vee \psi(x, y) \end{cases}$$

That is, an ordinary certifying algorithm terminates on all inputs $x \in X$ and provides proof that the witness predicate is correct.

Weakly certifying algorithms

A lesser variant of ordinary certifying algorithms is that of a *weakly certifying algorithm*. A weakly certifying algorithm is an algorithm that for any $x \in X$ such that $\varphi(x)$ is satisfied the algorithm terminates and returns a certificate that satisfies the certificate predicate. For any $x \in X$ that does not satisfy the precondition the algorithm may not terminate. Note that if the precondition is trivial, *i.e.*, any string $x \in X$, then the three types of certifying algorithm are indistinguishable.

Theorem 9. *Let (φ, ψ) be a precondition/postcondition pair. The combination of a certifying algorithm for φ and a weakly certifying algorithm for (φ, ψ) can be formulated to form a strongly certifying algorithm for (φ, ψ).*

Proof. [116] Let P be a certifying algorithm for precondition φ and let Q be a weakly certifying algorithm for postcondition ψ. If P returns F then the algorithm returns F and the certificate produced by P otherwise the output and certificate of ψ is returned. The algorithm clearly terminates on all inputs as P will terminate because it is an ordinary certifying algorithm and Q will terminate as φ is met. In both case P and Q return an witness that justifies its output. □

A certifying algorithm is *efficient* if the algorithm and the associated checker have asymptotic running time at most that of the best known algorithm. We call the algorithm that produces the certificate the prover and the algorithm that authenticates the tuple (x, y, w) the checker.

Chapter 3

Related Work

3.1 Overview

The efforts to characterise graph classes stretch back to the very foundations of graph theory itself where Euler quested to prove necessary and sufficient conditions for a graph to contain an Euler tour. Since then characterisation theorems are scattered liberally through the literature for many different graph classes. The benefits of characterisation theorems are that they often expose structural properties that can be utilised to develop efficient algorithms. This idea can be seen in the study of subclasses of perfect graphs where characterisations are used to expose structural properties which are then used to develop efficient algorithms for the maximum independent set, maximum clique and colouring problems. Subclasses of perfect graphs are often characterised by forbidding a set of minimal induced subgraphs, if this set is finite this yields a polynomial time algorithm for recognising the class. Often this approach does not produce the most efficient recognition algorithm but its generality is passed by no other method. This general approach can be abstracted to any partial order which has resulted in a number of efficient algorithms for recognising many graph classes which have practical importance. For this approach to work two components are required: (1) a polynomial time algorithm for the containment problem for the partial order, and (2) a finite minimal obstruction set.

The requirement of a finite minimal obstruction set, and its potential benefits for illuminating the structural properties of graph classes that have practical applications, motivates the research presented here. The area of research spans a number of different fields in mathematics and theoretical computer science. The literature for characterising graph classes by forbidding a set of graphs is rich including many different partial orders which have varying motivations. We survey the area of partial orders defined on graphs including their containment complexity and their applications in characterising graph classes.

The parameterized graph classes we consider arise naturally in the area of graph modification problems. In this area many problems are fixed-parameter tractable. We therefore survey the literature for results relating to the graph modification problems relating to the graph classes that we consider. We also survey the literature for abstract approaches to solving fixed-parameter tractable problems.

3.2 Partial orders

Partial orders have an important role in the topic of graph theory. An example of this can be seen in famous areas of research such as graph colouring, the strong perfect graph theorem [22] and the graph minor theorem [139]. The partial orders defined in Chapter 2 provide a natural ordering of graphs when a specific property is considered. Consider the vertex colouring problem: that is the problem of assigning colours to the vertices of a graph such that no two adjacent vertices are assigned the same colour. The minimum number of colours required to colour a graph is called the *chromatic number*. It is easy to observe that if $\chi(G)$ denotes the chromatic number of G then $\chi(H) \leq \chi(G)$ for all induced subgraphs H of G [38]. This observation can often be used to prove decomposition theorems which allow for efficient algorithms for computing the chromatic number of a graph, if the graph belongs to a specific graph class, by using an induction argument. As many of the partial orders defined on graphs are well-founded then the technique of structural induction is valid and is a useful proof technique. It is essential to ensure that the relation claiming to be a partial order is indeed a partial order.

The interest in partial orders in the field of graph theory is beneficial to other fields of theoretical computer science. Many interesting problems for partial orders on graphs are equivalent to problems on graphs that expose the boundary of what is $\mathbb{NP}$-complete and polynomial time solvable. For instance many of the partial order containment problems resolve themselves into instances of the k-disjoint path or k-induced disjoint path problem which can be shown to be easy for specific graph classes but hard on slightly larger classes.

A graph H is contained in G with respect to a partial order $\leqslant$ if and only if $(H, G) \in \leqslant$. The algorithms that are developed using the structural properties that use partial orders often require the recognition of pairs of graphs that belong to a partial order. The computational complexity of deciding if a pair of graphs belong to a partial order is called the containment complexity of a partial order. The containment problem for the induced minor relation is denoted INDUCED MINOR and named similarly for other partial orders. Where one of the graphs is fixed and the problem is to decide if G contains a copy of the fixed graph then the problem is prefixed with "H-", *e.g.*, H-INDUCED MINOR problem is the induced minor problem where H is fixed. As many algorithms require the recognition of the elements of a partial order, the containment complexity is important in determining the overall complexity of an algorithm. Unfortunately, the containment complexity of a partial order is often not trivial to determine.

For the induced subgraph and partial subgraph relation the containment complexity is $\mathbb{NP}$-complete as it contains as a subproblem the problem of determining if a graph contains a complete graph, however, both the H-INDUCED SUBGRAPH and H-PARTIAL SUBGRAPH problems have polynomial time algorithms for every fixed H that run in $O\left(n^{|H|}\right)$ time. For some graphs H there exist more efficient algorithms; for P_4 there exists a linear time algorithm [28, 77], for K_3 there exists a subcubic algorithm [3] and for $K_{1,3}$ there exists an algorithm that runs in $O\left(m^{\alpha+1/2}\right)$ where $O\left(n^{\alpha}\right)$ is the time required for matrix multiplication [101].

For the MINOR problem, where both H and G are part of the input, the problem is known not to be solvable in polynomial time, unless $\mathbb{P} = \mathbb{NP}$, however, when H is fixed the problem is fixed-parameter tractable running in $O\left(f(|H|) \cdot n^3\right)$ time [137], later improved to $O\left(f(|H|) \cdot n^2\right)$ [97]. This bound has been improved further when the input domain is restricted. The problem is linear time solvable for graphs of bounded treewidth via Courcelle's theorem (See Section 3.5.4). The classification of the induced version of the minor relation is a little more varied. The INDUCED MINOR problem has been well studied. There are instances of H where the H-INDUCED MINOR problem is known to be $\mathbb{NP}$-complete. Fellows *et al.* show the existence of a specific graph H with 68 vertices for which the problem is $\mathbb{NP}$-complete [51]. Also it is shown in [115] that the problem is $\mathbb{NP}$-complete for trees of bounded degree. For the general problem with no restrictions on H or G the complexity of INDUCED MINOR is $\mathbb{NP}$-complete. When restrictions are placed on H and G the problem has been shown to be solvable in polynomial time. It has been shown by Fellows *et al.* that for every fixed graph H the H-INDUCED MINOR problem can be solved in linear time on planar graphs [51]. Three open problems are posed by Fellows *et al.* [51] and subsequently two have been answered. The first open question asks if there is a planar graph H for which the H-INDUCED MINOR problem is $\mathbb{NP}$-complete. This was partially answered by van't Hof *et al.* in [156] where the authors show that for any fixed planar graph H the H-INDUCED MINOR problem can be solved in polynomial time on any minor closed graph class providing the class is not the class of all graphs. The second open question of Fellows *et al.* [51] asks if the H-INDUCED MINOR problem can be solved in polynomial time for all fixed trees H, this is answered negatively by Fiala *et al.* [57]. Fiala *et al.* show the existence of a tree H for which the H-INDUCED MINOR problem is not polynomial time solvable (shown in Figure 3.1). They also go further, showing that for all fixed forest H which is not isomorphic to the exception shown in Figure 3.1 the H-INDUCED MINOR problem can be solved in polynomial time. It was latter shown by Belmonte *et al.* that when the input domain is restricted to chordal graphs the H-INDUCED MINOR problem is polynomial time solvable for any fixed graph H [10].

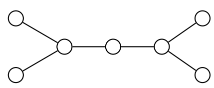


Figure 3.1: A graph H such that the H-INDUCED MINOR problem is not polynomial time solvablem, unless $\mathbb{P} = \mathbb{NP}$. The graph is the single exception of a forest on less that 8 vertices for which the H-INDUCED MINOR problem is not polynomial time solvable.

The topological minor and induced topological minor containment problems have also attracted some interest. It is known that both containment problems are $\mathbb{NP}$-complete. The topological minor containment problem has been shown to be polynomial [136] if H is fixed and fixed-parameter tractable if H is the parameter [75]. This settles the case for the topological minor containment problem. For the induced topological minor containment problem the containment problem is more problematic. There are instances of H for which the containment problem is solvable in polynomial time, such as K_3 [3] and where H is the family of all cycles. Lévêque *et al.* show a number of interesting examples where the graphs H have a similar structure but the containment complexity varies between being solvable in polynomial time and being $\mathbb{NP}$-complete [107]. These examples show that it is not a trivial task to determine if the H-INDUCED TOPOLOGICAL MINOR problem is solvable in polynomial time or not based on the structure of the graph. When the class of graphs is restricted to elements of $\{K_{1,3}\}$-free$_{\mathrm{i}}$ the problem is polynomial time solvable [56].

For the contraction minor relation the containment complexity results are well-classified. Brouwer *et al.* provide a description of the H-CONTRACTIBILITY problem for all graphs with at most 4 vertices [18]. The H-CONTRACTIBILITY problem can be solved in polynomial time for all graphs H with at most 4 vertices with the exception of C_4 and P_4, in which case the problems are shown to be $\mathbb{NP}$-complete [18]. A more general result is presented in [18] stating that for any graph H which is connected and triangle-free (other than a star) the H-CONTRACTIBILITY problem is $\mathbb{NP}$-complete. This work is extended by Levin *et al.* [108, 109] to a complete description of the H-CONTRACTIBILITY problem on all graphs with at most 5 vertices. The work of Levin *et al.* generalises the arguments used by Brouwer *et al.* [18]. They also provide polynomial time algorithms for two specific graphs, namely W_5 and $\overline{K_{1,3} \uplus K_1}$. Levin *et al.* also make the observation that if H is connected, contains a dominating vertex and the order of H is less than 5 then the H-CONTRACTIBILITY problem can be solved in polynomial time.

A summary of the complexities of the partial order containment problems, for those partial orders defined in Chapter 2, is provided in Table 3.1.

Partial order	All graphs	Bounded treewidth	Planar graphs	Chordal graphs	Split graphs
Partial subgraph	ℕℙ-c (ℙ)	ℕℙ-c (ℙ [30])	ℕℙ-c (ℙ [46])	ℕℙ-c (?)	ℕℙ-c (ℙ [69])
Induced subgraph	ℕℙ-c (ℙ)	ℕℙ-c (ℙ [30])	ℕℙ-c (ℙ [46])	? (?)	? (ℙ [69])
Minor	ℕℙ-c (ℙ [137])	ℕℙ-c (ℙ [137])	ℕℙ-c (ℙ [137])	ℕℙ-c (ℙ [137])	ℕℙ-c (ℙ [137, 69])
Induced minor	ℕℙ-c ([115])	ℕℙ-c (ℙ [30])	ℕℙ-c (ℙ [51])	ℕℙ-c (ℙ [9])	ℕℙ-c [69] (ℙ [11])
Topological minor	ℕℙ-c (ℙ [75])	ℕℙ-c [115] (ℙ [30, 75])	ℕℙ-c (ℙ [69])	ℕℙ-c (ℙ [75])	ℕℙ-c (ℙ [69])
Induced topological minor	ℕℙ-c (ℕℙ-c [10])	ℕℙ-c [115] (ℙ [30])	ℕℙ-c (ℙ [70])	ℕℙ-c (ℙ [10, 70])	(ℙ [69, 10])
Contraction minor	ℕℙ-c (ℕℙ-c [18])	ℕℙ-c (ℙ [30])	ℕℙ-c (ℙ [90])	ℕℙ-c (ℙ [9])	ℕℙ-c (ℙ [69])
Graph isomorphism	𝔾𝕀-c (ℙ)	ℙ [112] (ℙ [30])	ℙ [88] (ℙ [88])	𝔾𝕀-c [16]	𝔾𝕀-c [16] (ℙ) [69]

Table 3.1: Known results for the containment problem $H \leqslant G$ where G is part of the class indicated by the column. Bracketed results indicate the complexity when H is fixed. ℕℙ-c denotes ℕℙ-complete.

3.2.1 Kruskal's tree theorem

Kruskal's tree theorem was one of the first well-quasi ordering theorems on the set of finite graphs and laid the foundations for a number of significant results including the graph minor theorem. Kruskal's tree theorem is a generalisation of Higman's lemma [85] and states that the set of finite trees is well-quasi ordered with respect to the topological minor relation, the result was first published in [105]. Kruskal's tree theorem is an affirmative proof for a conjecture by Vázsonyi. The conjecture stated;

> "There is no infinite set $\{t_1, t_2, \ldots\}$ of finite trees such that t_i is not homeomorphically embeddable in t_j for all $i \neq j$."

This statement is equivalent to the statement in Theorem 10.

Theorem 10. *[105] The finite trees are well-quasi ordered with respect to the topological minor relation.*

The theorem was originally proved in [105], alternative proofs include one presented by Diestel [38] (outlined below) and one presented by Lovász [113].

In the proof of Theorem 10 a more restrictive partial order is used which implies the theorem. Consider two rooted directed trees T, T' with roots r, r' respectively with the edges oriented away from the roots. The pair of trees (T, T') are in the partial order if there is an isomorphism φ from a subdivision of T to a subtree of T' that preserves the tree ordering of the vertices. That is if $x, y \in V(T)$ and x is a predecessor of y then $\varphi(x)$ is a predecessor of $\varphi(y)$. If $T \leqslant T'$ then clearly $\overline{T} \leqslant_t \overline{T'}$ where $\overline{T}$ represents the undirected unrooted tree represented by T.

Assume the Theorem 10 is not true, then there exists an infinite antichain. Let us construct the antichain inductively. For a given $n \in \mathbb{N}$ assume inductively that the sequence $T_0, T_1, \ldots, T_{n-1}$ is the start of some bad sequence (defined on page 9). Choose T_n such that $|T_n|$ is as small as possible such that some bad sequence starts with $T_0, T_1, \ldots, T_{n-1}, T_n$. Let the root of each rooted tree in this sequence be denoted r_n. The sequence $(T_n)_{n\in\mathbb{N}}$ is a bad sequence from its construction. For each n let A_n denote the set of directed rooted trees obtained from T_n by removing the root r_n and selecting, in each subtree, the vertex adjacent to r_n as the new root of the subtree. The tree order on these subtrees is that induced by the tree order of the supertree. Let $A = \bigcup_{n\in\mathbb{N}} A_n$ it is shown that A is well-quasi ordered.

Let $(T^k)_{k\in\mathbb{N}} \subseteq A$, for each $k \in \mathbb{N}$ choose an n such that $T^k \in A_n$, let $f(k) = n$. Select a k such that $f(k)$ is minimum. The sequence $T_0, \ldots, T_{f(k)-1}, T^k, T^{k+1}, \ldots$ is a good sequence, by the minimal choice of $T_{f(k)}$ and that T^k is a subtree of $T_{f(k)}$. As $T_0, \ldots, T_{f(k)-1}, T^k, T^{k+1}, \ldots$ is a good sequence then there exists a good pair (by definition), let (T, T') be such a good pair. Since $(T_n)_{n\in\mathbb{N}}$ is bad, $T \notin (T_n)_{0\leq n<f(k)-1}$ therefore the good pair must be of the form (T^u, T^v) for some $k \leq u < v$. As (T^u, T^v) is a good pair in $(T^k)_{k\in\mathbb{N}} \subseteq A$ and $(T^k)_{k\in\mathbb{N}}$ was chosen without condition then A is well-quasi ordered.

As A is well-quasi ordered by $\leqslant$ then the extension of $\leqslant$ to finite subsets of A follows by Higman's lemma [85] consequently the sequence $(A_n)_{n\in\mathbb{N}}$ is well-quasi ordered. Let (A_i, A_j) be a good pair in $(A_n)_{n\in\mathbb{N}}$ and let $f : A_i \to A_j$ be injective with $T \leqslant f(T)$ for all $T \in A_i$. By extending the union of the embedding of T into $f(T)$ to a map φ from $V(T_i)$ to $V(T_j)$ by letting $\varphi(r_i) = r_j$ a mapping is obtained that preserves the tree ordering of the vertices. The edge $r_i r \in E(T_i)$ maps easily onto the paths $r_j T_j \varphi(r)$. Hence (T_i, T_j) is a good pair in the original bad sequence forming a contradiction.

A generalisation of Kruskal's tree theorem transfers the result to the set of arbitrary graphs, first by showing that graphs of bounded treewidth are well-quasi ordered by the minor relation then extending this to all finite graphs.

3.2.2 Graph minor theorem

The celebrated work of Robertson and Seymour's graph minor theorem was published in a series of papers over a 25-year period. The main results of the series of papers is a proof of Wagner's conjecture, which Wagner denies conjecturing [38]. Wagner's conjecture states that for any infinite set of graphs, there exists two graphs in the set such that one graph is a minor of the other. This is equivalent to stating that in any infinite set of graphs the set contains a good pair. As a consequence of Wagner's conjecture and the proof of Robertson and Seymour given in [139], the set of all finite graphs is well-quasi ordered with respect to the minor relation. The implication can be confirmed by observing that the minor relation is well-founded and therefore contains no infinite strictly descending chains and Wagner's conjecture implies there are no infinite antichains.

Outline of the Graph Minor Theorem

An overview of the graph minor theorem is provided next, many of the details are omitted for brevity however the outline of the proof is provided. Tree decompositions have an important role in the graph minor theorem allowing structural properties to be extracted. The results are structural theorems regarding graph classes that exclude certain graphs with respect to the minor relation.

Kruskal's tree theorem (see Section 3.2.1) proves that the set of trees is well-quasi ordered with respect to the topological minor relation. An interesting question is to enquire which, if any, of the properties of trees can be transferred to general graphs. This is where treewidth plays an important role. Treewidth is a measure of how 'tree like' a graph is. The smaller the parts of a tree decomposition the more the graph resembles a tree. Tree decompositions permit certain tree properties to be generalised and allow the properties to be applied to a more general class of graphs, specifically Kruskal's tree theorem may be extended to those graphs that resembles trees. However, this is only possible if the graphs under consideration have bounded treewidth. Effectively this ensures that the parts of the tree decomposition have an insignificant size allowing Kruskal's tree theorem to be generalised resulting in the following result.

Theorem 11. *[135] Let $k \geq 0$ be an integer. The class of graphs with treewidth $\leq k$ are well-quasi ordered with respect to the minor relation.*

As a result of Theorem 11 we know that graphs of bounded treewidth are well-quasi ordered. However, the theorem makes no remark for the graphs of unbounded treewidth. For a class of graphs to have unbounded treewidth, there must be some structural property which prevents this: indeed this is the case. There are a number of obstructions for small treewidth, complete graphs being one of them, as each complete subgraph must be entirely contained in a part of the tree decomposition. Complete subgraphs are not the only obstruction to small treewidth; the class of grid graphs also have unbounded treewidth but do not contain arbitrarily large complete graphs with respect to the minor relation. The obstructions to small treewidth are numerous; however, there is a structural theorem that states a necessary and sufficient condition as an obstruction to small treewidth.

A *bramble* is a set of mutually touching connected subgraphs in a graph. Two subgraphs are said to *touch* if they have a vertex or edge in common in the graph. The *order* of a bramble is the least number of vertices that cover the elements of a bramble. The classical example of a bramble is the set of crosses of a $k \times k$-grid:

$$C_{u,v} = \{(u, l) \mid l \in \{1, \ldots k\}\} \cup \{(l, v) \mid l \in \{1, \ldots k\}.\}$$

That is, the crosses comprise the vertices of the uth column and the vth row of a $k \times k$-grid. A result of Robertson and Seymour proves that every graph of large treewidth contains a bramble of large order.

Theorem 12. *[145] Let $k \geq 0$ be an integer. A graph has treewidth greater than or equal to k if and only if the graph contains a bramble of order greater than k.*

The obstructions for the classes of graphs of treewidth less than k ($k \leq 4$) are known [6]. For $k < 3$ the obstruction set is K_{k+2} with respect to the minor relation. The set of minimal obstructions for $k = 3$ are shown in Figure 3.2. For $k = 4$ the obstruction set is considerably larger. As k increases, the size of the obstruction set grows quickly [131]. Interestingly in [134] it is shown that the class of graphs of bounded treewidth must forbid a planar graph.

Theorem 13. *[134] Given a graph H the graphs in $\{H\}$-free$_{\mathrm{m}}$ have bounded treewidth if and only if H is planar.*

The one direction of this proof is easily observed: every class of graphs not forbidding a planar graph contains all grid graphs and therefore must have unbounded treewidth as grids are an obstruction to small treewidth. The reverse direction is more complex but it suffices to prove it in the special case of when the graph is a grid. This is because every planar graph

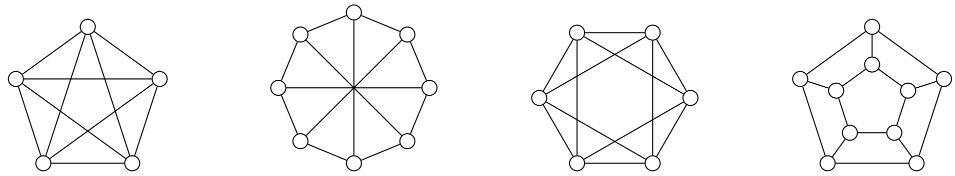


Figure 3.2: Minimal forbidden graphs for treewidth 3 with respect to the minor relation.

is contained in a grid with respect to the minor relation. It is a theorem of Robertson and Seymour that provides the reverse direction.

Theorem 14. *[134] For every integer r there is an integer k such that all graphs with treewidth at least k contains an $r \times r$ grid with respect to the minor relation.*

As a result of Theorem 13 the graphs that are members of a class that forbids a planar graph with respect to the minor relation have bounded treewidth and as a consequence of Theorem 11 are well-quasi ordered. In the context of graph classes this translates to: any graph class that forbids a planar graph with respect to the minor relation is well-quasi ordered. If all the graphs we forbid are not planar then the class does not have bounded treewidth, due to the biconditional of Theorem 13, and the generalisation of Kruskal's tree theorem does not work. Instead, if we forbid non-planar graphs, the result is a more subtle structural constraint. In general, it is sufficient to show the case where the forbidden graph is a complete graph of order greater than 4. For each n there is a finite set of surfaces $\mathcal{S}$ such that each graph in $\{K_n\}$-free$_{\mathrm{m}}$ has a tree decomposition into parts that are nearly embeddable into a surface $s \in \mathcal{S}$ which K_n is not. The finiteness of $\mathcal{S}$ is guaranteed by the following result of Robertson and Seymour.

The *torsos* of a tree decomposition (X, T) of a graph G are the graphs H_i where $i \in V(T)$ obtained from $G[X_i]$ by adding all the edges xy such that $x, y \in X_i \cap X_{i'}$ for some neighbour i' of i in T.

Theorem 15. *[138] For every $n \geq 5$ there exists a $k \in \mathbb{N}$ such that every graph not containing a K_n with respect to the minor relation has a tree decomposition whose torsos are k-nearly embeddable in a surface in which K_n is not embeddable.*

The proof of the graph minor theorem is outlined in Figure 3.3. The graph minor theorem states;

Theorem 16. *The set of finite graphs is well-quasi ordered with respect to the minor relation.*

Therefore, as a consequence of Corollary 2, any infinite sequence of graphs must contain a good pair. Let $G_0, G_1, \ldots$ be an infinite sequence of finite graphs then there are indices i and j such that $i < j$ and $G_i \leqslant_{\mathrm{m}} G_j$. Assume that this is not the case and $G_0, G_1, \ldots$ is an infinite antichain then observe that $G_1, G_2, \ldots \in \{G_0\}$-free$_{\mathrm{m}}$ otherwise $G_0 \leqslant_{\mathrm{m}} G_i$ for some integer $i \geq 1$. If G_0 is planar then from Theorem 13 the set $\{G_0\}$-free$_{\mathrm{m}}$ has bounded treewidth and therefore is well-quasi ordered by Theorem 11. As $G_1, G_2, \ldots \subseteq \{G_0\}$-free$_{\mathrm{m}}$

then $G_1, G_2, \ldots$ cannot be an infinite antichain as it would be an infinite antichain in a well-quasi ordered set. However all antichains in a well-quasi ordered set are finite therefore there must exist a good pair in $G_1, G_2, \ldots$. This contradicts the assumption that $G_1, G_2, \ldots$ is an infinite antichain. If G_0 is not planar then the generalisation of Kruskal's algorithm cannot be applied. The proof continues by considering a general case: if $G_0 = K_n$ where $n = |G_0|$. This is reasonable as $G_0 \leqslant_{\mathrm{m}} K_n$ and therefore $\{G_0\}$-free$_{\mathrm{m}} \subseteq \{K_n\}$-free$_{\mathrm{m}}$. It is shown that $\{K_n\}$-free$_{\mathrm{m}}$ for each integer n is well-quasi ordered and consequently $\{G_0\}$-free$_{\mathrm{m}}$ is well-quasi ordered because $\{G_0\}$-free$_{\mathrm{m}} \subseteq \{K_n\}$-free$_{\mathrm{m}}$.

We may assume that $G_1, G_2, \ldots \in \{K_n\}$-free$_{\mathrm{m}}$. The graphs in $\{K_n\}$-free$_{\mathrm{m}}$ have the structural property that there exists a finite set of surfaces $\mathcal{S}$ such that the graphs in $\{K_n\}$-free$_{\mathrm{m}}$ have a tree decomposition into parts that are nearly embeddable into a surface $s \in \mathcal{S}$. By a generalisation of Theorem 11, if the set of all parts is well-quasi ordered then the graphs that decompose into those parts are well-quasi ordered. To prove this, the proof considers a single surface $s \in \mathcal{S}$ and shows that the set of parts nearly embeddable in that surface are well-quasi ordered and therefore contain no infinite antichain. As $\mathcal{S}$ is finite this extends then to the set of all parts being well-quasi ordered.

The proof that the set of all parts nearly embeddable in a surface $s \in \mathcal{S}$ is well-quasi ordered uses an induction argument on the genus of the surface. Using a similar argument as before, the set of parts nearly embeddable in the surface s form an infinite sequence. We assume that it is an antichain and therefore $H_1, H_2, \ldots \in \{H_0\}$-free$_{\mathrm{m}}$. If the surface is homomorphically equivalent to the sphere then it is the case that H_0 is planar and are therefore well-quasi ordered (Theorem 13), this forms the base of the induction. The induction step reduces the genus of the surface by performing 'surgery'. A circle is found in the surface that does not bound a disc in more than a bounded number of vertices (X_i) for each H_i where $i \geq 1$. By cutting along this circle and mending the structure to be a surface again, either one or two new surfaces are obtained with reduced genus. If only one surface S_i is obtained then $H_i \setminus X_i$ is nearly embeddable in S_i as X_i is bounded in size. If this occurs for infinitely many H_i's then infinitely many of the surfaces S_i are homeomorphically equivalent and the induction hypothesis provides a good pair. If the 'surgery' obtains two surfaces S'_i and S''_i for infinitely many H_i's then the graph H_i is separated into two subgraphs H'_i and H''_i by the separator X_i, which are nearly embeddable into S'_i and S''_i respectively. Infinitely many of the S'_i and S''_i must be homeomorphically equivalent and by the induction hypothesis the graphs embeddable into the surfaces S'_i and S''_i are well-quasi ordered. By extension the subgraphs H'_i and H''_i are well-quasi ordered. Reconstructing the embedding of H_i into S_i that considers the layout of X_i in the subgraphs H'_i and H''_i indices i, j are obtained that such that $H'_i \leqslant_{\mathrm{m}} H'_j$, $H''_i \leqslant_{\mathrm{m}} H''_j$ and $H_i \leqslant_{\mathrm{m}} H_j$, demonstrating the existence of a good pair.

Applications and limitations of the graph minor theorem

The graph minor theorem is a remarkable result which is partially distracted from by a fixed-parameter algorithm for checking if one graph is a minor of another. Robertson and Seymour [137] provide a parameterized algorithm for testing if a fixed graph H is a minor of another graph in $O\left(f(|H|) \cdot n^3\right)$ time. With these two results it makes it theoretically possible to recognise any graph class which is closed with respect to the minor relation. Given a graph class $\mathcal{C}$ closed with respect to the minor relation then the set $\mathcal{G} \setminus \mathcal{C}$ is the set containing all non-members of $\mathcal{C}$. The minimal non-members of $\mathcal{C}$ form an antichain in $\mathcal{G} \setminus \mathcal{C}$. As $\mathcal{G} \setminus \mathcal{C}$ is a subset of $\mathcal{G}$ and $\mathcal{G}$ is well-quasi ordered then the antichain must be finite. Therefore any minor closed class has a finite minimal forbidden set. As every class $\mathcal{C}$ closed with respect to the minor relation has a finite forbidden set then each element of the minimal forbidden set has bounded order. When this is coupled with the minor checking algorithm an algorithm for recognising the class $\mathcal{C}$ is obtained. The algorithm checks if a graph contains any graph in the forbidden set. The overall run time for recognising any minor closed class is $O\left(n^3\right)$.

The results of Robertson and Seymour are monumental and in no way should their efforts towards the fields of mathematics and computer science be diminished, but the practical applications of the graph minor theorem are restricted by a number of technicalities. The first being that the graph minor theorem is a non-constructive proof (an existence proof). The minimal forbidden set is guaranteed to be finite however an algorithm for enumerating it is not given nor is any guide to its size. Moreover there is a number of later results regarding computing the minimal forbidden sets of minor closed graph classes which implies it is not an easy task. In [55] it is observed that there is no algorithm that can compute the minimal forbidden set for a minor closed class given a Turing machine that can recognise the class. A similar result is provided by Courcelle *et al.* in [33] where it is shown that there is no algorithm that, given a minor closed property expressed as a sentence in monadic second order logic, can compute the minimal forbidden set. Despite the obstacles outlined in [33, 55] on computing the forbidden set for a minor closed graph class there have been techniques developed to overcome the non-constructiveness of the graph minor theorem. The work of Fellows and Langston in [54] provides a general method based on an extension of the Myhill-Nerode theorem to graph languages. In [1] an alternative technique is proposed along the same lines but uses definability in monadic second order logic. By applying the results in [1], it is possible to construct an algorithm that computes the minimal forbidden set for the class of bounded treewidth graphs, the class of bounded branchwidth graphs and the class of graphs with a fixed genus. In addition to the bounded parameter graph classes Adler *et al.* [1] provide a method for computing the forbidden set for the union of two minor closed graph classes. Their result uses an observation of Fellows and Langston [54] that states the forbidden set for the union of two minor closed graph classes could be computed if it were possible to bound the treewidth of the union of the two graph classes. The results in [1] also extends to computing the minimal forbidden set for the graph class $Planar{+}k\mathsf{v}$, referred to as apex graphs in the context of [1].

The second restricting technicality of the graph minor theorem is the construction of the H-MINOR checking algorithm. Although the proof of the algorithm is constructive, that is the paper outlining the algorithm does provide an explicit process for constructing the algorithm [137], the algorithm is far from practical even in the most trivial of cases. Robertson and Seymour in [137] provide a fixed-parameter algorithm for the H-MINOR containment problem, where the running time is $O\left(f(|H|) \cdot n^3\right)$. As a result of the fixed-parameter algorithm for each fixed graph H the H-MINOR containment problem is polynomial time solvable. The practical difficulty arises when considering the constant factor hidden by the big-Oh notation. The constant depends super-exponentially on the order of H [140, 142]. In [89] Johnson shows how vastly impractical the cubic time algorithm is. Johnson states that in computing part of the constant there are three steps which cause most of the "damage". The "damage" is caused by a repeated application of a tower of twos generator $t(k)$ defined as follows $t(1) = 2$, $t(k) = 2^{t(k-1)}$. The resulting application of this generator yields a constant which is dependent on the order of H and is bounded from below by:

$$2^{\uparrow 2^{2^{2^{2^{\uparrow 2^{|H|}}}}}}$$

where $2^{\uparrow n} = \left.2^{2^{\cdot^{\cdot^{\cdot^{2}}}}}\right\}n$. Clearly for even small graphs this constant factor is massively impractical. Johnson comments in [89];

> "...for any instance $G = (V, E)$ that one could fit into the known universe, one would easily prefer $|V|^{70}$ to even constant time, if that constant had to be one of Robertson and Seymour's."

There have been some developments for H-MINOR checking, the complexity has been reduced from $O\left(n^3\right)$ to $O\left(n^2\right)$ in [97]. However this result only reduced the asymptotic complexity. The constant factor hidden by the big-Oh is not reduced from that of the Robertson and Seymour result [137].

The importance of the graph minor theorem is not only in its contribution towards mathematics or computer science but also in providing motivation for the development of algorithms especially in the field of parameterized complexity. The graph minor theorem proves the existence of algorithms to recognise minor closed graph classes but provides no concrete method of constructing such an algorithm. However, the existence of an algorithm is a good incentive to strive towards it. This has lead to much work on developing algorithms and new techniques for constructing the minimal forbidden sets.

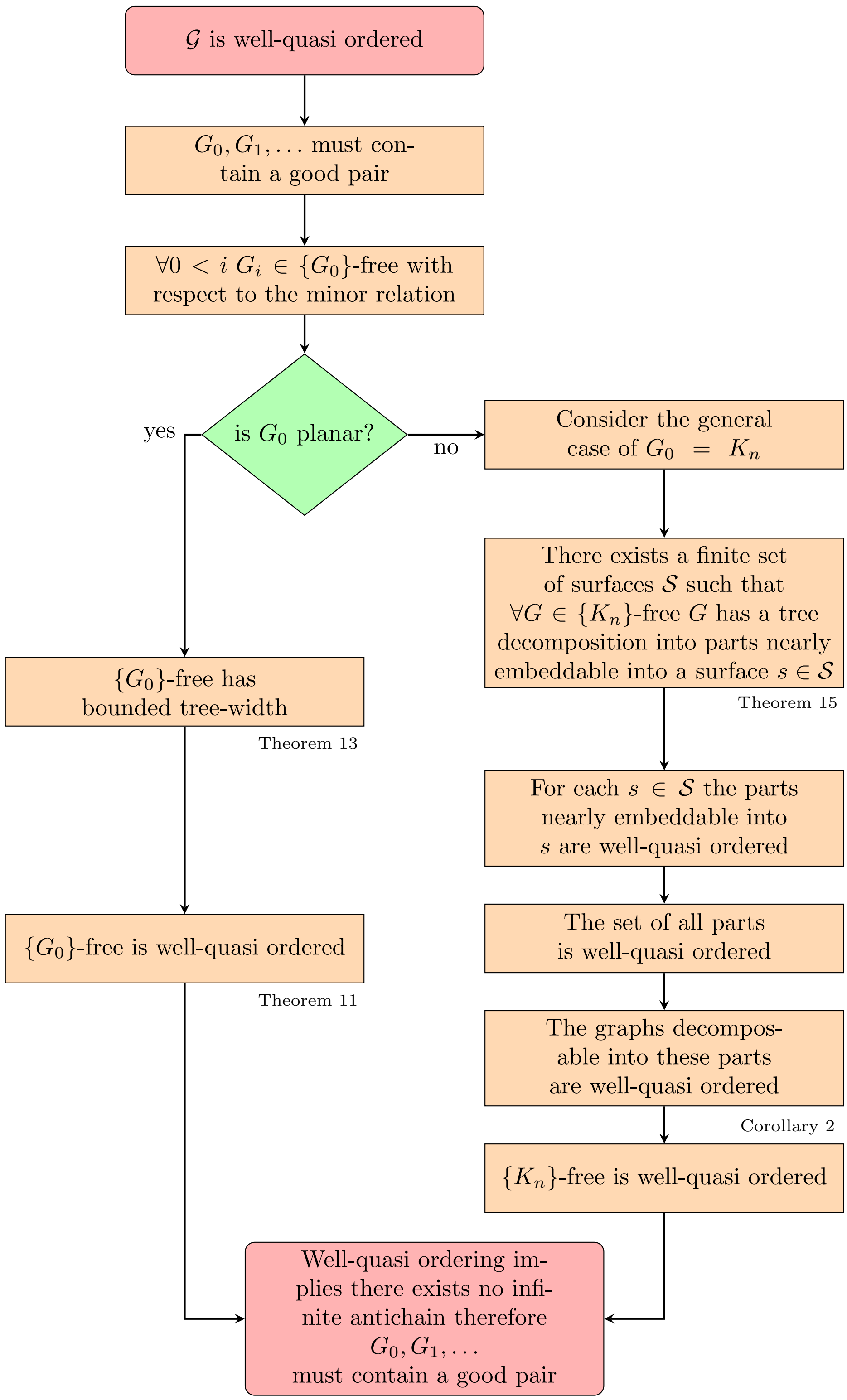


Figure 3.3: Outline of the graph minor theorem

3.2.3 Well-quasi ordering

The seminal work of Robertson and Seymour proves, amongst other things, that the class of all graphs is well-quasi ordered with respect to the minor relation which implies that every graph class that is closed with respect to the minor relation has a finite forbidden set. This on its own is a monumental result but coupled with a cubic time algorithm for testing if $H \leqslant_{\mathrm{m}} G$ results in a polynomial time recognition algorithm for every minor closed class.

There are similar results for other partial orders. The immersion minor relation is a well-quasi ordering on the set of all graphs [141]. The consequence of this is the same as that for the minor relation: there exists a polynomial time recognition algorithm for each immersion minor closed graph class. Although it may seem that a graph class being well-quasi ordered implies that there exists a polynomial time recognition algorithm it is not necessarily accurate. There exist well-quasi ordered graph classes that have infinite forbidden sets. Consider the class of linear forests, that is, the class of graphs where each connected component induces a path [52]. The minimal forbidden set with respect to the induced subgraph relation is infinite therefore the naive approach of testing if the graph contains a forbidden graph does not lead to a polynomial time recognition algorithm. For this class there is, however, a trivial recognition algorithm which avoids the knowledge of forbidden set characterisation.

There exist partial orders for which the class of all graphs is not well-quasi ordered but when the set of graphs is restricted the partial order is a well-quasi ordering. Consider the induced subgraph relation and the set of all graphs $(\mathcal{G}, \leqslant_{\mathrm{i}})$. It is easy to observe that $(\mathcal{G}, \leqslant_{\mathrm{i}})$ is not a well-quasi ordering. The set of all cycles C_n where $n \geq 3$ or the set of 'H'-graphs, shown in Figure 3.4, are antichains in $\mathcal{G}$ with respect to $\leqslant_{\mathrm{i}}$. Table 3.2 shows some antichains in the set of all graphs with respect to some well studied partial orders.

Figure 3.4: Antichains in the set of all graphs with respect to $\leqslant_{\mathrm{i}}$. The cycle C_n and H_n.

Lemma 17. $(\mathcal{G}, \leqslant_{\mathrm{I}})$ *is a well-quasi ordering.*

Proof. Observe that every $G \leqslant_{\mathrm{m}} H$ implies $G \leqslant_{\mathrm{I}} H$ and that $\leqslant_{\mathrm{I}}$ is well-founded. For any infinite sequence $G_0, G_1, \ldots \in \mathcal{G}$ there exists an i, j such that $i < j$ where $G_i \leqslant_{\mathrm{m}} G_j$ as every minor is also an immersion minor this implies that $G_i \leqslant_{\mathrm{I}} G_j$, proving that $\mathcal{G}$ is well-quasi ordered by $\leqslant_{\mathrm{I}}$. □

For the induced subgraph relation when the class of graphs is restricted to the class of cographs the class is well-quasi ordered [34]. In [34] the author provides the following theorem about well-quasi ordered hereditary graph classes.

Theorem 18. *[34, Proposition 1] For any hereditary graph classes $\mathcal{H}$ and $\mathcal{G}$ where $\mathcal{G}$ is well-quasi ordered and $\mathcal{H} \subseteq \mathcal{G}$ then $\mathcal{H}$ is well-quasi ordered. Further if* $\text{Forb}(\mathcal{G})$ *is finite then* $\text{Forb}(\mathcal{H})$ *is finite.*

The additional comment to this theorem regarding the finiteness of $\text{Forb}(\mathcal{H})$ depending on the finiteness of $\text{Forb}(\mathcal{G})$ is as a result of the class of all graphs not being well-quasi ordered with respect to the induced subgraph relation. There are well-quasi ordered sets of graphs from the set of all graphs which have an infinite forbidden set. Consider the class of graphs which are the disjoint union of path graphs. The class is well-quasi ordered but the forbidden set is $\{C_n \mid n \geq 3\} \cup \{K_{1,3}\}$. This can be stated more generally.

Theorem 19. *For any classes $\mathcal{H}$ and $\mathcal{G}$ where $\mathcal{G}$ is well-quasi ordered with respect to $\leqslant$ and $\mathcal{H} \subseteq \mathcal{G}$ then $\mathcal{H}$ is well-quasi ordered with respect to $\leqslant$.*

It has been shown that the class $\{K_3, P_5\}$-free$_\text{i}$ is well-quasi ordered by the induced subgraph relation [34]. By Theorem 18 any subclass of $\{K_3, P_5\}$-free$_\text{i}$ is also well-quasi ordered with respect to the induced subgraph relation. The cycles form an antichain with respect to the induced subgraph relation. Therefore if a class is well-quasi ordered and has a finite minimal forbidden set the class must exclude an induced path of some length. This condition is however not sufficient as the complements of cycles also form an antichain. Therefore it is also required to exclude the complement of an induced path. An open question posed in [34] was answered affirmatively in [102] where it is proven that the class of bipartite $\{P_6\}$-free$_\text{i}$ graphs is well-quasi ordered with respect to induced subgraph relation and the class of bipartite $\{P_7\}$-free$_\text{i}$ is not well-quasi ordered.

There are similar results to those presented in [34] for the partial subgraph relation published in [39]. Ding provides two useful tools in [39]: firstly it is proved that a class $\mathcal{C}$ is well-quasi ordered with respect to induced subgraph relation if and only if $\mathcal{C}$ is well-quasi ordered with respect to the partial subgraph relation. Secondly, Ding proved that for the partial subgraph relation there exist only two infinite antichains, that is, $\{C_n \mid n \geq 3\}$ and $\{H_n \mid n \geq 3\}$ form antichains with respect to partial subgraph relation in the set of all graphs. Figure 3.4 show the general construction for H_n. Therefore it follows that a class is well-quasi ordered with respect to partial subgraph relation if and only if it contains only a finite number of graphs in $\{C_n \mid n \geq 3\}$ and $\{H_n \mid n \geq 3\}$. The class of graphs of bounded vertex cover number is well-quasi ordered with respect to the induced subgraph relation [49].

The results of Kruskal, mentioned in Section 3.2.1, show that the class of connected acyclic graphs is well-quasi ordered with respect to the topological minor relation. Further, the class of graphs with bounded feedback vertex set number is well-quasi ordered by the topological minor relation [49]. The class of graphs of bounded circumference is well-quasi ordered with respect to the induced minor relation [49].

In [49] a general theorem is presented regarding the class $\mathcal{C}{+}k\mathsf{v}$. It is shown that for any class $\mathcal{C}$ that is well-quasi ordered the class $\mathcal{C}{+}k\mathsf{v}$ is well-quasi ordered if the partial order under consideration is either the partial subgraph, induced subgraph or topological minor relation. The context of the results in [49] is that of subclasses of graphs of bounded treewidth. Because the context is graphs of bounded treewidth, the partial order containment complexity is linear time for the partial orders they consider, taking this into account and that the class has a finite minimal forbidden set results in linear time recognition algorithms. The algorithm they develop takes as input a graph from a restricted domain and outputs whether the input graph belongs to some specific subclass of the input domain. The drawback to this is that the algorithm is a "promise" algorithm: if the input to the algorithm is a member of the well-quasi ordered set then the result follows; however, if the input to the algorithm is not a member of the well-quasi ordered set then the algorithm may not terminate or may produce the wrong output. The interesting result from this paper is the biconditional relation between $\mathcal{C}+k\mathsf{v}$ and $\mathcal{C}$ being well-quasi ordered.

Theorem 20. *Given a class $\mathcal{C}$ closed with respect to the partial subgraph or induced subgraph relation and for all $k \geq 0$ the class $\mathcal{C}+k\mathsf{v}$ is well-quasi ordered if and only if $\mathcal{C}$ is well-quasi ordered.*

Proof. The forwards implication is presented in [49] and the backwards implication follows from Theorem 18. □

Having noted this relation it is noteworthy to highlight that this does not imply that $\mathcal{C}+k\mathsf{v}$ has a finite minimal forbidden set if $\mathcal{C}$ is well-quasi ordered. The combination of the result in Theorem 20 and the result of Theorem 46 culminates in the following theorem.

Theorem 21. *If $\mathcal{C}$ is well-quasi ordered with respect to the induced subgraph relation then $\mathcal{C}+k\mathsf{e}$ is well-quasi ordered with respect to the induced subgraph relation for all $k \geq 0$.*

Proof. If $\mathcal{C}$ is well-quasi ordered with respect to the induced subgraph relation then $\mathcal{C}+k\mathsf{v}$ is well-quasi ordered by Theorem 20 and Theorem 46 implies that $\mathcal{C}+k\mathsf{e} \subseteq \mathcal{C}+k\mathsf{v}$ then by applying Theorem 18, $\mathcal{C}+k\mathsf{e}$ is also well-quasi ordered. □

It has been shown in [125] that the class of graphs of bounded rankwidth is well-quasi ordered with respect to the pivot minor relation and that for each $k \geq 0$ the class of graphs of rankwidth at most k is characterised by a finite set of minimal forbidden pivot minors. As every pivot minor is a vertex minor and the class of graphs of bounded rankwidth are closed with respect to vertex minors then they may be characterised by a finite set of minimal forbidden vertex minors.

3.3 Graph classes

The study of graph classes is a natural topic of interest in theoretical computer science. The computational complexity for many problems such as vertex colouring, maximum independent

Partial order	Antichain
Induced subgraph	$\{C_n \mid n \geq 3\}$, $\{\overline{C_{2n+1}} \mid n \geq 2\}$
Partial subgraph	$\{C_n \mid n \geq 3\}$
Minor	well-quasi ordered
Topological minor	Long double paths [40] ,$\{S_n \mid n \geq 3\}$
Contraction minor	$\{2K_2 \bowtie kK_1 \mid k \geq 1\}$, $\{kK_1 \mid k \geq 1\}$
$\leqslant_{\text{ftm}}$	$\{\overline{C_n} \mid n \geq 6\}$
Immersion minor	well-quasi ordered [141]
Induced topological minor	Long double paths [40] ,$\{S_n \mid n \geq 3\}$
Lift minor	well-quasi ordered (Lemma 17)
Lift contraction	$\{kK_1 \mid k \geq 1\}$

Table 3.2: Antichains in the set of all graphs with respect to a partial order.

set and maximum clique is $\mathbb{NP}$-complete when the input to the algorithm is an element from the set of all finite graphs. However, for some graph classes these problems are solvable in polynomial time when the input is restricted. By restricting the domain of the algorithm it is possible to provide efficient algorithms for hard problems that have practical applications. Graph classes form the restriction of the domain for which efficient algorithms are developed. The set of graphs where an algorithm correctly computes the answer forms a graph class and it is possible to extract and distil elegant structural theorems about them. Many of the structural theorems rely on the absence of certain substructures which makes obtaining minimal forbidden set characterisations fruitful to the development of graph theory and to the design of efficient algorithms. A classical example of this is for the subclasses of perfect graphs, where the classes are defined because of a specific property they have that allows for efficient vertex colouring algorithms to be applied. From the definition of perfect graphs we get that the class is closed with respect to the induced subgraph relation and from the strong perfect graph theorem a forbidden set characterisation is given [22]. Although a purely combinatorial algorithm is unknown for the vertex colouring problem on perfect graphs, for some subclasses of perfect graphs combinatorial algorithms are well known. For chordal graphs a linear time algorithm is known for the vertex colouring problem using a lexicographical breadth first search approach to find a perfect elimination ordering [72]. The class has a forbidden set characterisation with respect to the induced subgraph relation and has also been characterised with respect to the contraction minor relation (see Chapter 7). By restricting the domain of an algorithm it is often possible to show that some problem is solvable in polynomial time. The next question that often leads from this type of result is 'Is there a superclass where the problem is also solvable in polynomial time?'. This type of question allows the area of graph theory to expose the boundaries of computational complexity classes.

For different properties it is necessary to consider different partial orders such that the property under consideration is closed with respect to the partial order. That is, if G has a property then for all $H \leqslant G$ then H has the property. The choice of partial order has implications for the development of algorithms and for using the characterisations in structural theorems. In order to achieve the most concise characterisation it is beneficial to consider the most "powerful" partial order for which the property is still closed with respect to. However, the literature focuses on only a finite number of partial orders which encourages characterising graph classes with respect to well studied partial orders. For example, many of the subclasses of perfect graphs are closed with respect to the induced subgraph relation but have an infinite minimal forbidden set. Some of the same classes are closed with respect to the induced topological minor relation and admit a finite minimal forbidden set characterisation. This seems to be an advantage; however, the containment complexity for the induced topological minor relation remains an open question for the general case. Because partial orders are amenable to decomposition theorems, many partial orders have been defined on the basis that they preserve some property. This is evident for the rankwidth property and the vertex minor and pivot minor relations. Table 3.3 summarises closure of some well studied graph classes with respect to partial orders that appear in the literature.

The computation and description of minimal forbidden sets for some property with respect to some partial order is a task which is interesting not only because it exposes some interesting computability questions but because the minimal forbidden sets have practical applications. Many of the well-quasi ordering results in the field of graph theory are non-constructive, proving that there exist finite minimal forbidden sets but not providing a method of computing such a set. The computation of the minimal forbidden set is nowhere more important than for the classes of bounded width parameters. This is motivated by the need to recognise the classes of bounded width parameters because they permit efficient polynomial time algorithms for many real world problems. From the graph minor theorem, it is known that the minimal forbidden set for the class of bounded treewidth graphs is finite but it does not provide a construction. Similarly, it is known that the class of graphs of bounded rankwidth can be characterised by a finite minimal forbidden set with respect to the vertex minor relation [124]. The approach used is to establish that the set of all graphs of rankwidth at most k is well-quasi ordered with respect to the vertex minor relation and that the set of minimal forbidden graphs for graphs of rankwidth at most k belong to the class of graphs of rankwidth at most $k + 1$. As graphs of rankwidth at most $k + 1$ are well-quasi ordered then the set of minimal forbidden graphs must be finite as it is an antichain in a well-quasi ordered class. From these results it is possible to construct polynomial time algorithms to recognise the classes of graphs of bounded rankwidth [32]. The same approach can be used to show that the minimal forbidden set for the class of graphs of bounded treewidth is finite with respect to the minor relation. However, these algorithms require that the minimal forbidden set is known in order to construct the algorithm which neither of the constructions provide.

Adler *et al.* provide the construction of an algorithm that will compute the minimal for-

	Partial orders							
	Partial Subgraph	Induced Subgraph	Topological minor	Induced topological minor	Minor	Induced Minor	Contraction Minor	Pivot minor
Perfect graphs		✓(✓)						
Chordal graphs		✓(✓)		✓		✓	✓	
Interval graphs		✓(✓)		✓		✓	✓	
Comparability graphs		✓(✓)						
Permutation graphs		✓(✓)						
Split graphs		✓(✓)		✓(✓)		✓(✓)	✓(✓)	
Bipartite graphs	✓	✓(✓)						
Cographs		✓(✓)		✓(✓)		✓(✓)	✓(✓)	
Trivially perfect graphs		✓(✓)		✓(✓)		✓(✓)	✓(✓)	
Threshold graphs		✓(✓)		✓(✓)		✓(✓)	✓(✓)	
Forests	✓	✓	✓	✓	✓	✓	✓	
Planar graphs	✓	✓	✓	✓	✓	✓	✓	
Bounded treewidth	✓	✓	✓	✓	✓	✓	✓	
Bounded pathwidth	✓	✓	✓	✓	✓	✓	✓	
Bounded rankwidth		✓						✓

Table 3.3: Summary of which well studied graph classes are closed with respect to partial orders defined in Chapter 2. Bracketed ticks, *i.e.*, (✓), indicate that the class of graphs that contains the complement of each graph in the class is closed with respect to the partial order.

bidden set for the classes of graphs of bounded treewidth, bounded branchwidth and bounded genus [1]. The results of Adler *et al.* rely heavily on the decidability of monadic second order logic on trees. In [124] an upper bound on the order of a minimal forbidden graph for the class of graphs of bounded rankwidth is provided. This is extended in [91] where they prove an upper bound for the class of bounded linear rankwidth. With an efficient algorithm to recognise the classes of bounded width parameters it is possible to apply the metatheorems discusses in Section 3.5.4.

A technique similar to the approach used by [125] is employed by [52]. They use the technique to show that some subclasses of bounded treewidth graphs are well-quasi ordered with respect to the induced subgraph, topological minor and induced minor relations. They introduce two tools which provide a means of recognising the subclasses they consider in linear time. The technique can be applied to the parameterized classes $\mathcal{C}+k\mathsf{v}$ and $\mathcal{C}+k\mathsf{e}$ when characterised with respect to the partial subgraph or induced subgraph relations provided that the class $\mathcal{C}$ has bounded vertex cover number and is well-quasi ordered [52].

Due to the definition of a graph class, determining the intersection of two graph classes is trivial. Assuming both graph classes are closed with respect to the same partial order then the minimal forbidden set can be described concisely as the minimal elements of the union of the minimal forbidden sets for the two classes. This approach leads to a number of interesting new graph classes to prove results on. A survey of the known results for graph classes derived from the intersection of graph classes can be found on the Information System on Graph classes and their Inclusions (ISCGI) [17]. The problem of characterising the union of two graph classes is harder: even assuming that both classes are closed with respect to the same partial order, the minimal forbidden set for the union does not follow easily from the characterisations of the two classes. For some partial orders it has been shown that the minimal obstruction set can be computed for the union of two classes assuming that the minimal forbidden sets for the two classes are given as input to the algorithm, this is true for the minor and immersion minor relations [1, 65]. Due to the fact that both the minor and immersion minor relations are well-quasi ordering on the set of all graphs, the minimal forbidden set for the union of two classes is finite. However, for other partial orders it is not clear whether the union of two closed classes has a finite minimal forbidden set even if the two classes have finite obstruction sets. This is particularly the case for the induced subgraph and partial subgraph relations.

3.4 Graph modification problems

A concept of parameterized graph classes arises commonly in the context of graph modification problems. Graph modification problems concern adding or deleting a set of edges or vertices from/to a graph to satisfy some property. Graph modification problems arise in many natural settings, the idea of modelling data in the form of a graph is an intuitive approach to problem solving but often the data have errors introduced by poor data collection, noise in the data, loss of information as a result of compression algorithms or poor quality recording equipment. These kinds of problems in data are often dealt with via two means, either treating them as a graph modification problem or using fixed-parameter algorithm where the error is known to be bounded. In the case of fixed-parameter algorithms this bound on the error is used as the parameter.

A graph modification problem can be formulated from erroneous or noisy data where much of the data is assumed to be correct and it is desired to find the smallest set of modifications such that the data have some specific property. The property is often required to validate the correctness of the data and to be able to achieve good performance in subsequent data processing. For example, if it is required to compute a colouring of the underlying graph class then the graph class should have an efficient colouring algorithm, *i.e.*, bipartite or perfect graphs *etc.* This type of problem trivially translates into a graph modification problem of looking for the smallest set of modifications.

We formally introduce a set of graph modification problems then relate them to parameterized graph classes.

$\mathcal{C}$-Vertex Deletion Given a graph $G = (V, E)$, find a set $U \subseteq V$ such that the removal of the vertices in U yields a graph belonging to $\mathcal{C}$, *i.e.*, $(G - U) \in \mathcal{C}$.

$\mathcal{C}$-Deletion Given a graph $G = (V, E)$, find a set $U \subseteq E$ such that the removal of the edges in U yields a graph belonging to $\mathcal{C}$, *i.e.*, $(G - U) \in \mathcal{C}$.

$\mathcal{C}$-Editing Given a graph $G = (V, E)$, find a set $U \subseteq E$ and a set $U' \in (V \times V) \setminus E$ such that $(G - U + U') \in \mathcal{C}$.

$\mathcal{C}$-Completion Given a graph $G = (V, E)$, find a set of edges $U \subseteq (V \times V)$ such that the addition of the edges in U to G yields a graph belonging to the class $\mathcal{C}$ and $E \cap U = \varnothing$.

The problems as they are posed above often have trivial solutions. For example for any class $\mathcal{C}$ the $\mathcal{C}$-Editing problem can be solved by removing all of the edges and adding edges to construct a graph in $\mathcal{C}$. The problem only becomes interesting and useful when restrictions are placed on the number of modifications or the problem asks to find the minimum number of modifications.

k-$\mathcal{C}$-Vertex Deletion Given a graph $G = (V, E)$, find a set $U \subseteq V$ where $|U| \leq k$ such that the removal of the vertices in U yields a graph belonging to $\mathcal{C}$.

k-$\mathcal{C}$-Deletion Given a graph $G = (V, E)$, find a set $U \subseteq E$ where $|U| \leq k$ such that the removal of the edges in U yields a graph belonging to $\mathcal{C}$.

k-$\mathcal{C}$-Completion Given a graph $G = (V, E)$, find a set of edges $U \subseteq (V \times V)$ where $|U| \leq k$ such that the addition of the edges in U to G yields a graph belonging to the class $\mathcal{C}$ and $E \cap U = \varnothing$.

$\mathcal{C}$	$\mathcal{C}+k\mathsf{v}$	$\mathcal{C}+k\mathsf{e}$
Perfect	$\mathbb{NP}$-complete [168]	$\mathbb{NP}$-complete [122]
Interval	$\mathbb{NP}$-complete [168]	$\mathbb{NP}$-complete [67]
Chordal	$\mathbb{NP}$-complete [168]	$\mathbb{NP}$-complete [122]
Split	$\mathbb{NP}$-complete [168]	$\mathbb{NP}$-complete [122]
Cluster	$\mathbb{NP}$-complete [168]	$\mathbb{NP}$-complete [45]
Chain	$\mathbb{NP}$-complete [168]	$\mathbb{NP}$-complete [122]
Tree	$\mathbb{NP}$-complete [168]	$\mathbb{P}$

Table 3.4: A summary of the complexity for the k-vertex deletion and k-edge deletion problem.

The above problems all have a related decision problem, which instead of trying to find a set of vertices or edges merely asks if such a set exists. Where k is not fixed (as above) but a minimum value of k is sought, the following problem formulations arise.

Minimum-$\mathcal{C}$-Vertex Deletion Given a graph $G = (V, E)$, find a set $U \subseteq V$ where $\neg\exists U' \subseteq V \quad |U'| < |U| \wedge (G - U') \in \mathcal{C}$ such that the removal of the vertices in U yields a graph belonging to $\mathcal{C}$.

Minimum-$\mathcal{C}$-Deletion Given a graph $G = (V, E)$, find a set $U \subseteq E$ where $\neg\exists U' \subseteq E \quad |U'| < |U| \wedge (G - U') \in \mathcal{C}$ such that the removal of the edges in U yields a graph belonging to $\mathcal{C}$.

Minimum-$\mathcal{C}$-Editing Given a graph $G = (V, E)$, find a set $U \subseteq E$ and a set $U' \in (V \times V) \setminus E$ such that $(G - U + U') \in \mathcal{C}$ and there does not exist a smaller set of edges and non-edges for which the property holds.

Minimum-$\mathcal{C}$-Completion Given a graph $G = (V, E)$, find a set of edges $U \subseteq (V \times V)$ such that the addition of the edges in U to G yields a graph belonging to the class $\mathcal{C}$ and $E \cap U = \varnothing$ and $\neg\exists U' \subseteq (V \times V) \quad |U'| < |U| \wedge (G + U') \in \mathcal{C}$.

The decision problem version of the k-$\mathcal{C}$-Vertex Deletion, k-$\mathcal{C}$-Deletion and k-$\mathcal{C}$-Completion defines a partition of the set of all graphs into two parts, those graph for which the decision problem answer is yes and for those that the decision problem answer no. Observe that if $\mathcal{C}$ is a property preserved by removing vertices then the class $\mathcal{C}+k\mathsf{v}$ is the class containing the graph for which the k-$\mathcal{C}$-Vertex Deletion decision problem answers yes. Analogously for the class $\mathcal{C}+k\mathsf{e}$ which contains the graphs for which the k-$\mathcal{C}$-Deletion decision problem answers yes, given that the property $\mathcal{C}$ is preserved by deleting edges. From these observations it is clear that there is a strong link between graph modification problems and parameterized graph classes. As there is a strong link between graph modification problems and graph class recognition we outline the results from the literature in Table 3.4.

At the outset of investigations into graph modification problems, efforts were focused on the vertex deletion problem considering properties that are closed with respect to the induced subgraph relation. A result by Yannakakis and Lewis in [168, 110] proves a very general problem with wide reaching implications.

Theorem 22. *For any non-trivial interesting properties closed with respect to the induced subgraph relation, finding a maximum subgraph with the property is* $\mathbb{NP}$*-hard.*

This result provides the complexity of the minimum-$\mathcal{C}$-vertex-deletion problem for any property closed with respect to the induced subgraph relation. This result was extended later in [169] to consider the effects of connectivity. The results in [168, 110] apply only for properties that are closed with respect to the induced subgraph relation. There are many properties that are not closed with respect to the induced subgraph relation but each connected subgraph has the property. For example the class of trees is not closed with respect to the induced subgraph relation however every connected subgraph of a tree is a tree. The result in [169] implies that it is $\mathbb{NP}$-hard to find the minimum number of vertices to remove from a graph to obtain a cycle-free graph. This result is consistent with the result of Karp [94], where the feedback vertex set problem is shown to be $\mathbb{NP}$-complete.

Despite the unfavourable complexity results for many interesting graph modification problems, one approach to handle this is to restrict the input to a specific graph class. The structure of the specific graph class may allow the problem to be solved in polynomial time. An alternative method to develop "good" algorithms for graph modification problems is that of parameterized algorithms. The best outcome of these efforts is to show that a graph modification problem is fixed-parameter tractable when parameterized by the maximum number of modifications.

An example of the advantages of restricting the input graph can be seen in a result of Peng *et al.* [128]. The result in [128] shows that there is a polynomial time algorithm for finding a maximum interval graph if the input is restricted to the class of distance-hereditary graphs. The minimum interval vertex deletion problem has been shown to be fixed-parameter tractable in [21] where the input is the class of all graphs. The restriction often seems arbitrary, the minimum interval bipartite deletion problem is $\mathbb{NP}$-complete even when the input graph is restricted to graphs of bounded degree [23] however if the input graph is restricted to being a tree then the problem becomes polynomial time solvable [160]. The approach of restricting the input to an algorithm is not that helpful when considering graph class recognition algorithms unless the restriction is a superclass of the property the algorithm recognises and there is an efficient algorithm to test membership of the restricted graph class. Even in the case when both of these conditions are met it does not ensure that the product is a practical algorithm.

The alternative approach of developing parameterized algorithms for graph modification problems has been fruitful. Many of the graph modification problems have been shown to be fixed-parameter tractable which has also lead to the development of parameterized algorithms for many different problems on parameterized graph classes. The work of Cai shows that it is possible to recognise the class $\mathcal{C}+k\mathsf{v}$ for every hereditary graph class $\mathcal{C}$ where $\mathcal{C}$ has a finite minimal forbidden set [19]. For other hereditary graph classes which do not have a finite characterisation, the task of recognising the class $\mathcal{C}+k\mathsf{v}$ and $\mathcal{C}+k\mathsf{e}$ is considered on a class by

$\mathcal{C}$	$\mathcal{C}+k\mathsf{v}$	$\mathcal{C}+k\mathsf{e}$	$\mathcal{C}-k\mathsf{e}$
Interval	FPT [20]	FPT [20]	FPT [20]
Proper Interval	FPT [155, 157]	FPT [157]	?
Chordal	FPT [114]	FPT [114]	FPT [19]
Strongly Chordal	FPT [93]	?	?
$\{W_n \mid n > 4\}$-free$_\mathrm{i}$	W[2] [111]	W[2] [111]	?
$\mathcal{F}$-free$_\mathrm{i}$*	FPT [19]	FPT [19]	FPT [19]
$\mathcal{F}$-free$_\mathrm{m}$*	FPT [137]	FPT [137]	-

Table 3.5: A summary of the complexity for recognition of $\mathcal{C}+k\mathsf{v}$, $\mathcal{C}+k\mathsf{e}$ and $\mathcal{C}-k\mathsf{e}$. Asterisks denote that the set $\mathcal{F}$ must be finite.

class basis. For interval graphs the recognition problem for both $+k\mathsf{v}$, $+k\mathsf{e}$ and $-k\mathsf{e}$ has been shown to be fixed-parameter tractable [21, 20, 158]. The problem of recognising the classes $\mathcal{C}+k\mathsf{v}$ and $\mathcal{C}+k\mathsf{e}$ where $\mathcal{C}$ is the class of chordal graphs has been shown to be fixed-parameter tractable [114]. The recognition problem for the class $\mathcal{C}-k\mathsf{e}$ where $\mathcal{C}$ is the class of chordal graphs is fixed-parameter tractable [19]. A summary of the results is provided in Table 3.5.

The minimum-$\mathcal{C}$-completion problem, sometimes referred to as the minimum fill-in problem, has been well studied. For the class of chordal graphs it has been shown to be $\mathbb{NP}$-complete [170]. The minimum completion problems for the class of interval graphs, proper interval graphs and trivially perfect graphs are $\mathbb{NP}$-complete. The results for interval graphs were also discovered in [95].

For the class of planar graphs it has been shown that the minimum vertex deletion and minimum edge deletion problem are $\mathbb{NP}$-hard [168, 161]. However, for fixed valued of k the k-vertex deletion problem and k-edge deletion problem have been shown to be fixed-parameter tractable. This results comes as a consequence that the parameterized graph classes associated with the graph modifications are minor closed and therefore have a finite minimal forbidden set. The finite minimal forbidden set can then be used to recognise the class although the computation of the minimal forbidden set is not trivial.

An alternative generalisation of graph modification problems are sandwich problems. A sandwich problem is; given two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ such that $V_1 \subseteq V_2$ and $E_1 \subseteq E_2$ is there a graph H such that $G_1 \leqslant H \leqslant G_2$ where H has a specific property. For the case that V_1 is equal to V_2 the problem is equivalent to either

- finding a $\mathcal{C}$-completion of G_1 using only the edges of $E_2 \setminus E_1$, or
- finding a $\mathcal{C}$-deletion of G_2 only removing the edges in $E_2 \setminus E_1$.

The sandwich problems are well motivated by practical applications in the fields of computational biology, scheduling and linear algebra. A significant effort was made by Golumbic *et al.* to establish the complexity of a number of sandwich problems [73]. For the classes of comparability graphs, permutation graphs, chordal graphs, interval graphs, and circular arc graphs the problem is ℕℙ-complete, however, for the classes of split graphs, threshold graphs and cographs the problem is solvable in polynomial time [73]. The result for threshold graphs improves on the complexity from a result by Hammer *et al.* [80]. Golumbic *et al.* leave three graph classes that resist classification, namely the class of chordal bipartite graphs, strongly chordal graphs and perfect graphs. The problem for the first two classes has since been resolved, being proved ℕℙ-complete by de Figueiredo *et al.* [35].

3.5 Fixed-parameter algorithms

Fixed-parameter tractability is a field of study that aims to provide practical solutions to hard real world problems. The field of fixed-parameter tractability takes a different approach to classifying problem complexity compared to traditional complexity theory. Instead of characterising 'good' and 'bad' algorithms purely on whether there exists a polynomial function of the input size that bounds the running time of the algorithm, fixed-parameter tractability enhances the framework under which algorithms are analysed by enriching it with parameters. The hope is that for interesting problems which are known to be ℕℙ-hard there exist natural parameters such that the runtime of the algorithm can be bounded by the product of a polynomial function of the input size and some computable function of the parameter. Of course, when the parameter is fixed the resulting algorithm runs in polynomial time for all input. In practice, it is hoped that all interesting instances of the problem have a small parameter value which results in a practically useful algorithm.

The function parametrized by the parameter is often sizeable, containing large combinatorial terms. It has been noted in [43] that fixed-parameter tractability can be likened to making a deal with the devil: a compromise for a polynomial time bound with respect to the input size can result in the parameter function being arbitrarily large. Fixed-parameter tractable problems can therefore be viewed as partitioning the original problem instance into two parts, one where the problem can computed in polynomial time and the other where some brute force approach is used. From this viewpoint fixed-parameter tractability exposes useful structural properties of the problems that can be useful for other problems.

Although the field of fixed-parameter tractability is young several abstract algorithm design techniques have emerged, as is much the same for traditional algorithm design. The main techniques are:

- Bounded search,
- Kernelization, and
- Iterative compression.

For each technique a brief description is provided along with its application to the classic fixed-parameter tractable problem of vertex cover. All of the techniques have been applied to the recognition of parameterized graph classes.

3.5.1 Bounded search tree

The bounded search tree techniques are among the first fixed-parameter tractability techniques to be utilised in the field, being used as early as 1996 [19]. The technique involves constructing a search tree that spans the search space and using a polynomial time algorithm on each node of the tree. The search tree should be bounded in size by a function of the parameter. The search tree is often exponential in the size of the parameter. The worse case complexity analysis of bounded search tree algorithms occurs when all branches of the search tree are explored. If we denote $f(k)$ as the size of the search tree then it is easily observed that we obtain an algorithm that runs in $f(k) \cdot n^{O(1)}$ time if a polynomial time algorithm gets executed at each leaf of the search tree.

A classic example of the bounded search tree technique is when it is applied to the vertex cover problem. The problem asks if there is a vertex cover of size less than or equal to k in G where k is the parameter. The search tree is constructed as follows: create a root of the tree, labelled $\varnothing$. By selecting an arbitrary edge $uv \in E(G)$, clearly if there is a vertex cover of size at most k then either u or v is a member of the vertex cover. Create children of the root corresponding to the two possibilities, *i.e.*, u is in the vertex cover or v is in the vertex cover. Label the node with the vertex union the label of its parent. Recursively construct the tree labelling the nodes of the search tree with the vertex set of its parent union a vertex x or y from an edge $xy \in E(G)$ such that x and y are not members of the label of the parent node. Observe that the constructed tree is a binary tree. The height of the tree is at most k therefore there are at most 2^k leaf nodes. If at any point in the construction it is not possible to select a disjoint edge then a vertex cover has been found. As the height of the tree is bounded the size of the vertex cover must be at most k. If all leaf labels do not cover the edges of G then the graph does not contain a vertex cover of size at most k.

The bounded search tree technique is employed in an algorithm for the graph modification problem of hereditary properties presented in [19]. The algorithm requires the hereditary properties to have a finite characterisation. The algorithm constructs the search tree recursively in a depth first approach. At each branch the algorithm makes a modification to the input graph and continues to recursively test if the graph is a member of the graph class. The result of Cai in [19] does not cover all hereditary properties: there are many hereditary graph which do not have a finite characterisation. For example consider the class of chordal graphs. This class forbids all cycles of length greater than or equal to four with respect to the induced subgraph relation. Because of the infinite minimal forbidden set, the bounded search tree method cannot be applied; however, the problem has been shown to be fixed-parameter tractable via kernelization, the technique is also presented in [19]. Other such hereditary classes where the technique of Cai cannot be applied is the class of interval graphs. The class does not have a finite minimal forbidden set characterisation with respect to the induced subgraph relation; however the vertex deletion problem for interval graphs has been shown to be fixed-parameter tractable [21]. The interval completion problem has been shown to be fixed-parameter tractable by Villanger *et al.* in [158] using the bounded search tree technique. For the class of proper interval graphs, Bevern *et al.* have shown that the vertex deletion problem is fixed-parameter tractable [155], this was later improved on by van 't Hof & Villanger [157].

Another application of the bounded search tree technique is that in Chapter 6 where an algorithm is presented that enumerates a finite minimal forbidden set (subject to some condition) and a certifying algorithm is presented for an FPT problem.

3.5.2 Kernelization

The kernelization technique uses the concept of reductions to reduce an instance of the problem to an instance of bounded size. Let I denote an instance of a problem and let k be the parameter then the kernelization technique uses a set of reduction rules that transforms an instance (I, k) into an instance (I', k') such that:

- $k' \leq k$,
- $|I'| \leq f(k)$, and
- (I, k) is a yes instance if and only if (I', k') is.

The transformation should be computable in polynomial time. The output for the transformed instance is then computed and the result is then transformed into an output for the original instance. The computation of the output for the transformed instance may take exponential (or even greater) time. Because the size of the transformed instance is bounded only by the parameter, the computation can be done in constant time for each fixed parameter value. The transformed instance (I', k') is called the *problem kernel* or just *kernel*.

There is a strong link between fixed-parameter tractability and kernelization, it has been proved that every fixed-parameter tractable problem admits a kernelization [44, Proposition 4.7.1], however the proof does not guarantee the optimality of the kernel size. The result is interesting from a theoretical viewpoint but algorithmically its significance is little. The kernelization technique has been successfully applied to many graph modification problems. The interval vertex deletion problem has been shown to be fixed-parameter tractable using a set of reduction rules which yield a kernelization result. The results was proven by Cao *et al.* [21]. Their technique first attempts to destroy all 'small' forbidden graphs using two reduction rules. They progress by studying the structure of the reduced graphs, obtaining a cycle cover and then destroying all asteroidal triples. The resulting algorithm runs in $10^k \cdot n^{O(1)}$ time. This problem is significantly improved upon by Cao [20] where an algorithm is presented that runs in linear time with respect to the size of the input graph. Another graph modification problem that has been shown to be fixed-parameter tractable is that of recognising the class of graphs of bounded feedback vertex set, that is deciding if a graph can be made acyclic by removing a bounded number of vertices. The problem was shown to be fixed-parameter tractable using a kernelization technique which immediately yields an fixed-parameter tractable algorithm. The size of the kernel was originally shown to be $5k^2 + k$ in [150] and was later improved to $4k^2$ in [151]. A similar problem is the hitting set problem, this relates to the problem of finding a cover for a set of sets of some specific size. For the 3-Hitting set problem the problem has been shown to admit a quadratic kernel, specifically $5k^2 + k$ [99]. A generalisation to the d-hitting set problem is also presented in [99], where they show that the kernel size is $k^d \cdot d! \cdot d^2$. In [99] the bound is obtained from a careful generalisation of the 3-Hitting set. This bound can also be obtained from the sunflower lemma by Erdös & Rado [47]; however in [99] the author remarks that this bound is reasonable only for small values of d.

3.5.3 Iterative compression

Iterative compression is a technique where the problem can be posed as a minimisation problem. The technique starts at a base case where the problem is assumed to be trivial and then iteratively extends the partial solution to a solution closer to the solution of the problem instance, the procedure terminates when the partial solution is the solution to the problem instance. At each iteration either the algorithm returns a negative answer indicating that there is no output satisfying the output criteria or the algorithm returns a partial output for the extended problem. The latter option for each iteration is referred to as the *compression* stage.

There are a number of parameterized graph classes that can be recognised using an iterative compression approach. The class of bounded vertex cover graphs can be recognised using the iterative compression technique [44].

3.5.4 Meta theorems

Courcelle's Theorem

Monadic second order logic is a fragment of second order logic which allows quantification over unary relations and elements of the domain only. In the context of graph theory this allows for quantification over sets of vertices and sets of edges. Monadic second order logic is sufficient to express many common graph problems including k-vertex cover, k-colourability and k-dominating set. Courcelle's theorem concerns the relationship between graph properties expressible in this logic and graphs of bounded treewidth.

Theorem 23 ([30])**.** *Given a graph G of treewidth at most k and a graph property P expressed in monadic second order logic then there exists an algorithm that runs in $f(k) \cdot n$ time that decides correctly if G has property P.*

The work of Courcelle utilises the decidability of fragments of monadic second order logic on tree structures. The fragment of monadic second order logic Courcelle considers allows for quantification over sets of vertices and sets of edges, the fragment includes a binary relation that asks if a vertex is incident with an edge. The technique uses a dynamic programming approach to compute the property expressed on each subtree of the tree decomposition. When the property is computed for the root of the tree, the algorithm returns deciding the property for the input graph. Some of the technicalities of Courcelle's theorem make it very difficult to apply in practice. This has been partially overcome by Kloks. By defining *nice tree decomposition* as a special kind of tree decompositio,n Kloks improves the accessibility of the theorem to a wider audience [100].

This meta theorem has found many applications in graph theory. The crossings number of a graph is the minimum number of edge crossing of a graph embedded in a plane. This problem was shown to be solvable in cubic time via the graph minor theorem. By using Courcelle's theorem, Grohe was able to prove that the problem is fixed-parameter tractable [74] running in $f(k) \cdot n^2$. Grohe's approach applies Courcelle's theorem directly to the input graph if the input graph has small treewidth otherwise the graph must contain a large grid minor which can be 'ignored' as it does not alter the crossing number. The algorithm simplifies the input graph until the graph has small treewidth, then applying Courcelle's theorem directly. This approach can be considered as using the kernelization technique. This result was later improved to a linear time algorithm by Kawarabayashi and Reed [98].

Other examples of the application of Courcelle's theorem include those for proving that the minimal forbidden elements of some minor closed classes are computable. Adler *et al.* show that, given two minor closed classes $\mathcal{C}_1$ and $\mathcal{C}_2$, it is possible to compute the obstruction set for the class $\mathcal{C}_1 \cup \mathcal{C}_2$. It is known that the class $\mathcal{C}_1 \cup \mathcal{C}_2$ is closed with respect to the minor relation and as a result of the graph minor theorem the class has a finite minimal forbidden set. However, it has been shown previously in [33, 53] that computing the set is hard. The proof commences by establishing that all graphs in the minimal forbidden set either have bounded treewidth, contain a large clique as a minor or contain a large substructure that contradicts the minimality. Where the minimal forbidden graph contains a large clique then the clique contains a minimal forbidden element from one of the two base classes and is therefore not minimal. In the remaining case the graphs have bounded treewidth and Courcelle's theorem can be applied as $\mathcal{C}_1 \cup \mathcal{C}_2$ can be expressed in monadic second order logic. A similar approach is used by Giannopoulou *et al.* to show that it is possible to compute the minimal forbidden set for the union of two immersion minor closed graph classes [65].

For other width parameters there are similar decomposition theorems which allow dynamic programming techniques to be applied to the decomposition. For pathwidth it has been shown that using a dynamic programming technique over a path decomposition can yield efficient algorithms for hard problems. This has been exploited by many people including Arnborg [5] and Andreica [4]. In the latter of the two listed applications the author introduced the concept of a nice path decomposition which play a similar role to nice tree decompositions in implementing Courcelle's theorem in practice. Branchwidth has also been used as a framework for dynamic programming [42, 24]. For clique-width there is a similar theorem to Courcelle's theorem for graphs of bounded treewidth. If a property is expressible in a fragment of monadic second order logic and the input graph has clique-width at most k then there exists a linear time algorithm to decide if the input graph has that property. This fragment of monadic second order logic allows the property to be expressed in terms of sets of vertices and has a single binary relation for determining if two vertices are adjacent.

Well-quasi of fixed-parameter tractable problems

A recent result of Fellows & Jansen states that fixed-parameter tractable problems are characterised by useful obstruction sets [50]. Their work extends the knowledge that fixed-parameter tractable problems have kernels by showing that the kernels can be quasi ordered by some appropriate quasi ordering. It is then possible to define a class of kernels which are yes/no instances of the problem, consequently it is possible to characterise the no instances with respect to a quasi ordering, resulting in a set of obstructions. The cardinality of the obstruction set may not be finite and it may not be immediate how the forbidden kernels relate to obstructions in the original problem.

3.6 Certifying algorithms

The importance of the correctness of algorithms is paramount in software engineering, especially as ever more complicated algorithms are developed and are implemented in critical systems. This is a motivating factor in developing the field of *certifying algorithms*. As algorithms get more complicated, the user is increasingly relying on the reputation and the correctness of the implementer. It is therefore desirable for an algorithm to provide some justification that the output is valid with respect to the input. By providing such justification the user need only validate that the tuple of (input, output, justification) is correct. Checking the tuple is often easier than proving that the algorithm has been implemented correctly. The name given to an algorithm that justifies its output is a *certifying algorithm* and was first used in [103]. Previous to the adoption of this term, a collection of terms were used to refer to similar concepts, including proof-carrying code [123] and interactive proof systems [68]. The concept of Robust algorithms bears a strong resemblance [130]. The formal definition of a certifying algorithm was given in Section 2.6.

The difference between conventional algorithms and certifying algorithms can be seen in Figure 3.5 and Figure 3.6. Certifying algorithms have an additional step where the output of the algorithm is validated by a second algorithm, called the *checker*. The part of a certifying algorithm that produces the output to the problem and the certificate is called the *prover*. The advantage of certifying algorithms is evident, the user can be assured that the output is correct without having to understand the details of the implementation of the prover. Despite much effort on program verification [86] it is still often beyond the state of the art to be able to verify the implementation of complex algorithms. Implementation verification involves the verification of the software and hardware stack making it impractical to maintain a reasonable number of verified platforms. With certifying algorithms, it is not necessary to verify the implementation of the prover or the hardware stack the algorithm runs on. It is merely enough to trust the hardware and the implementation of the checker and why the certificate justifies the output of the algorithm. This suggests a requirement for the checker to be conceptually simpler than the prover. There are few areas of study other than graph theory where certifying algorithms have be applied so widely. A possible reason for this is the potential of graph theory to model many real world problems. That is not to say that certifying algorithms have not been successfully applied to other areas of study. Certifying algorithms have been developed for string matching algorithms and numerical algorithms [116].

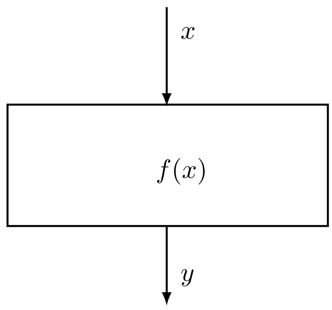


Figure 3.5: Expected behaviour of a conventional program. Input x is provided to some algorithm which outputs an answer y. The end user must trust the implementation of f.

Approaches for developing certifying algorithms

An approach used in the development of certifying algorithms is that of reductions. As with many problems in theoretical computer science it is possible to reduce one problem to another. This reduces the effort required in developing new algorithms and aids in ensuring the best asymptotic complexity for the problem; as once a problem has been reduced to another problem any advances in the latter automatically propagate to the former. This technique involves transforming a problem P into an instance of a new problem P' such that there is a certifying algorithm for problem P'. Using this technique a certificate is obtained for the input P' which may be sufficient to the end user in certifying the output for input P, or another transformation may be applied to transform the output for input P' back into the context of the original problem.

An application of the reduction technique can be seen in the following example. Consider the problem of maximum cardinality matching in bipartite graphs, it is well known that this problem can be transformed into the maximum network flow problem [26, p. 732]. The maximum network flow problem is known to have a certifying algorithm as a consequence of the max-flow min-cut theorem [59].

Another approach used to develop certifying algorithm is that of composition. Certifying algorithm composition is when a certifying algorithm is used as a subprocedure to another certifying algorithm and the certificate of the subprocedure is used as part of the computation

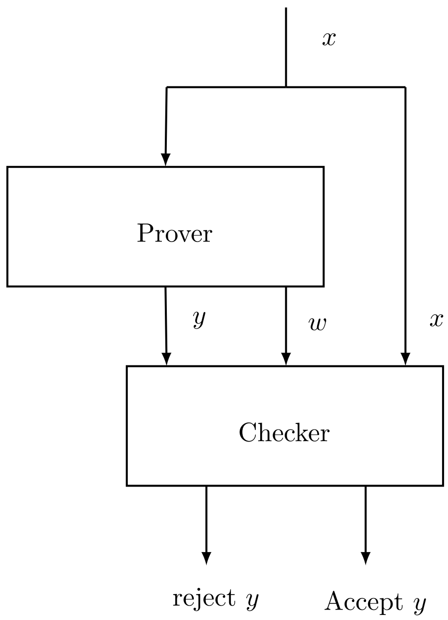


Figure 3.6: Behaviour of a certifying program. An immutable input x is provided to the prover and the checker. The checker also receives the output y from the prover and a certificate w which justifies the correctness of y. Note that x must be immutable otherwise the prover may make modifications which could cause the checker to verify the correctness of (x', y, w) where x' is a modified version of x.

of the final certificate. This type of design is common when the certifying algorithm is applied to a restricted domain and a certifying algorithm is known for the domain membership. This type of design can be seen in Algorithm 13 in Appendix B where a call to Algorithm 12 is made in order to establish if the input is a split graph. As the class of threshold graphs is a subclass of the class of split graphs then if `Certifying-Split` returns false then `Certifying-Threshold` must also return false, the certificate returned by `Certifying-Split` makes the computation of the final certificate trivial.

When developing certifying algorithms selecting an appropriate certificate which is checkable is frequently the challenging part. For many types of problems there are characterisation theorems which often provide insight into candidate certificates for a problems. This can be seen in the software library LEDA [119] where a Kuratowski subgraph is provided as proof that the graph does not have a planar embedding. In addition to this the work of Heggerness & Kratsch [83] uses forbidden induced subgraphs as certificates for non-members of other graph classes.

In the following section we provide a set of certifying graph class recognition algorithms which are significant in the field either because they introduce a new approach to certifying algorithms or that the techniques used applies to a large set of problems.

Certifying algorithm for bipartite graph recognition

The example that is often provided for certifying algorithms is that of recognising bipartite graphs. Recall a bipartite graph is a graph $G = (V, E)$ such that V can be partitioned into two sets X, Y such that $\forall uv \in E(G)\ u \in X \wedge v \in Y$ or vice versa. From this definition it is easy to see that if a graph is a bipartite graph then it has a 2-colouring and if a graph is not bipartite then there exists an odd length cycle [7, Theorem 2.1.3]. These two characterisations serve as the certificate for the certifying algorithm. If the graph is bipartite the algorithm provides an affirmative answer and a 2-colouring or in the event that the graph is not bipartite it returns a negative answer and the edges that induce an odd length cycle. Algorithm 1 provides an outline of the algorithm.

Algorithm 1: Certifying bipartite graph recognition algorithm

```
Input: A connected graph G = (V, E)
Output: {True, False} and either a 2-colouring or a set of vertices that induce an odd
        length cycle respectively.
Q := ∅ // initial an empty queue.
u ∈ V(G)
X := {u}
Y := ∅
Q.enqueue(u)
while Q is not empty do
    u := Q.dequeue()
    for v ∈ N_G(u) do
        if v ∉ (X ∪ Y) then
            if u ∈ X then
                Y := Y ∪ {v}
            else
                X := X ∪ {v}
            end
            Q.enqueue(v)
        end
    end
end
for uv ∈ E(G) do
    if u ∈ X ∧ v ∈ X or u ∈ Y ∧ v ∈ Y then
        Let P be a path from u to v in G \ uv
        return (False, P ∪ {uv})
    end
end
return (True, (X, Y))
```

The certificates outlined above for membership and non-membership are sublinear and weak respectively. To check the membership certificate the algorithm must check that each of the two parts in the partition is independent, this take $O(n + m)$ time. For the non-membership certificate the checker must check that the set of vertices induces a cycle and that the cycle is of odd length, this can be achieved in $O(n)$. The disparity between validating the certificate for a decision problem is evident in Table 3.6 where it is an exception that the complexity of the checker is equal for both outcomes of the prover.

Certifying algorithms for recognising subclasses of perfect graphs

Many subclasses of perfect graphs have linear time certifying recognition algorithms, this is partly due to the efforts that have gone into providing characterisations of these graph classes. The certificates rely on these characterisations for both membership and non-membership certificates. For chordal graphs a linear time certifying algorithm was presented in [149], the result extends the work of Rose *et al.* in [143] where a lexicographical breadth first search procedure is used to find a perfect elimination ordering in a perfect graphs. The perfect elimination ordering is provided to the checker as the membership certificate and a reduced input graph as a certificate of non-membership, either certificate can be verified in linear time. Other candidate non-membership certificates include a minimal separator that is not a clique (see [41]) or an induced chordless cycle of length 4 or more.

An algorithm for certifying the recognition of cographs is provided in [28]. Cographs are $\{P_4\}$-free$_{\mathrm{i}}$ (See Section 2.3) with respect to the induced subgraph relation. In the case of membership the certifying algorithm provides a restricted cotree which can be checked in linear time and in the case of non-membership provides an induced P_4. This work is extended in [83] where they apply a similar approach for certifying trivially perfect graphs which are the intersection of the class of cographs and chordal graphs.

Linear time certifying recognition algorithms are also know for the class of interval and permutation graphs [103], proper interval graphs [84], proper circular-arc graphs and unit circular-arc graphs [92].

The certifying recognition algorithms for the class of split graphs and threshold graphs are well-known and are a direct result of [72]. The authors of [83] reject the use of the degree sequence as the certificate, not because it does not justify the correctness of the algorithm but because the checker would recompute the work of the certifying algorithm. As the checker is recomputing the certificate and the end user accepts the implementation of the checker then he may as well use the checker to test for membership. Instead, for the class of split graphs the authors provide an alternative membership certificate in the form of a vertex ordering which can be checked in linear time and in the case of non-membership an induced subgraph isomorphic to a graph in $\{2K_2, C_4, C_5\}$. The authors also present a representation of an induced subgraph that can be checked in constant time for each fixed size forbidden graph. We provide details of two algorithms from [83] as they are used to illustrate a point in Chapter 6. A summary of certifying algorithms for graph class recognition problems can be found in Table 3.6.

Certifying algorithm for recognising Split graphs

Algorithm 12 in Appendix B runs in time $O(n+m)$, the procedures to check if a vertex ordering is a perfect elimination ordering and to find the size of a largest clique can be found in [72]. The checking algorithm runs in time $O(n+m)$ in the case that the algorithm returns an affirmative output and $O(1)$ otherwise.

Certifying algorithm for recognising Threshold graphs

Algorithm 13 in Appendix B runs in time $O(n+m)$. The checking algorithm runs in time $O(n+m)$ in the case that the algorithm return an affirmative output and $O(1)$ otherwise. Note that the class of threshold graphs is a subclass of split graphs. The certifying algorithm for threshold graphs uses Algorithm 12 (in Appendix B) as a subprocedure.

Graph class	Complexity	Membership certificate		Non-membership certificate	
Chordal graphs [149]	$O(n+m)$	Perfect elimination ordering	$O(n+m)$	Chordless cycle	$O(n)$
Split graphs [83, 72]	$O(n+m)$	(K, I) where K is a clique, I is an independent set	$O(n+m)$	$\{2K_2, C_4, C_5\}$	$O(1)$
Threshold graphs [83, 72]	$O(n+m)$	Nested neighbour ordering of a maximum size independent set	$O(n+m)$	$\{2K_2, C_4, P_4\}$	$O(1)$
Cographs [28]	$O(n+m)$	Cotree	$O(n)$	$\{P_4\}$	$O(1)$
Chain graphs [83]	$O(n+m)$	Nested neighbourhood ordering of a bipartition	$O(n+m)$	$\{2K_2, K_3, C_5\}$	$O(1)$
Co-chain graphs [83]	$O(n+m)$	Nested neighbourhood ordering of a bipartition in the complement of the input	$O(n+m)$	$\{3K_1, C_4, C_5\}$	$O(1)$
Trivially perfect graphs [83]	$O(n+m)$	Universal-in-a-component vertex ordering	$O(n+m)$	$\{P_4, C_4\}$	$O(1)$
Interval graphs [103]	$O(n+m)$	Interval model	$O(n)$	Chordless cycle or Asteroidal triple	$O(n)$
Proper Interval graphs [120, 84]	$O(n+m)$	Proper interval model	$O(n+m)$	Forbidden substructure	$O(n)$
Permutation graphs [103]	$O(n+m)$	Permutation model	$O(n)$	Forbidden substructure	$O(n)$
Unit circular-arc graphs [92]	$O(n+m)$	Circular-arc model	$O(n+m)$	Forbidden substructure	$O(n)$

Table 3.6: Summary of Certifying graph class recognition algorithms

Chapter 4

Properties of Partial Orders

Characterising graph classes with respect to some partial order has been a fruitful line of inquiry for many years, leading to results such as the graph minor theorem and those results listed on the Information System on Graph Classes and their Inclusions [36], to highlight a few. The consequence of such characterisations is often a better understanding of the graph class being considered which provides insight for developing specialised algorithms. This can be seen in the results of Kratsch *et al.* [103] where the characterisations of interval and permutation graphs obtained using the induced subgraph partial order play an important role in the certifying recognition algorithms for these classes.

To date, partial orders have played an important role in characterising graph classes but the relationship between the partial orders has gone undocumented which leads to the replication of results. Another unexplored avenue is that of specific properties of partial orders that make the partial order favourable to work with. Examples of this are the results of Kruskal [105] and Robertson and Seymour [134] showing that specific graph classes are well-quasi ordered with respect to some partial orders. These properties are not arbitrary to some partial orders, rather the property is inherited (in a sense which will be explained in Section 4.1) which raises a number of interesting questions to investigate. Here we provide a formal framework for reasoning about properties of partial orders, collect a set of results which have previously been known and we show how they fit into the formalisation. We then restrict our attention to a set of well studied partial orders and use the formalisation to expose a number of interesting questions relating to parameterized graph classes, which will then be answered in following chapters. We exclude a number of partial orders on the account that they are uninteresting for our line of investigation.

4.1 Lattice of partial orders

Consider the set of all finite graphs and consider a binary relation R on this set. If R is reflexive and transitive then R is a preorder (quasi order). Let us restrict the ground set to the set of

Figure 4.1: P_3 and K_3 (respectively). $P_3 \leqslant_{\mathrm{v}} K_3$ and $K_3 \leqslant_{\mathrm{v}} P_3$ therefore $K_3 \equiv_{\mathrm{v}} P_3$.

equivalence classes defined as follows; if G, H are finite graphs and $G \,\mathrm{R}\, H$ and $H \,\mathrm{R}\, G$ then G and H belong to the same equivalence class. This restriction of R to the (representatives of) equivalence classes of all finite graphs defines a partial order on $\mathcal{G}$. Let R be the aforementioned restriction then $\mathrm{R} \subseteq \mathcal{G} \times \mathcal{G}$ such that for all $x, y, z \in \mathcal{G}$:

- $(x, x) \in \mathrm{R}$ (reflexivity)
- $(x, y), (y, z) \in \mathrm{R}$ then $(x, z) \in \mathrm{R}$ (transitivity)
- $(x, y), (y, x) \in \mathrm{R}$ then $x = y$ (antisymmetry)

Let $(x, y) \in \mathrm{R}$ be denoted by $x \leqslant y$. When there is ambiguity as to which partial order this may refer, the order will be subscripted to ensure distinguishability. As a partial order is a subset of $\mathcal{G} \times \mathcal{G}$ then the set of all partial orders on $\mathcal{G} \times \mathcal{G}$ can be ordered by subset inclusion. This ordering of partial orders is itself a partial order. Let $\mathcal{P}$ denote the set of all partial orders on the set $\mathcal{G} \times \mathcal{G}$ then $(\mathcal{P}, \subseteq)$ is a partially ordered set.

Theorem 24. *$(\mathcal{P}, \subseteq)$ is a partially ordered set.*

Proof. The subset inclusion relation on a set of subsets of a set is a partial ordering [144, Example 1.1.2.2]. □

Observe that the partial order $(\mathcal{P}, \subseteq)$ has a unique maximal element called the maximum element and is denoted by $\top$, the maximum element in $\mathcal{P}$ is the partial order $\mathcal{G} \times \mathcal{G}$, which is also a total order. The partially ordered set $(\mathcal{P}, \subseteq)$ also has a unique minimal element called the minimum element and is denoted by $\bot$, the minimum element is the partial order $\{(x, x) \mid x \in \mathcal{G}\}$, this is the smallest reflexive partial order in $\mathcal{G} \times \mathcal{G}$. From the definitions of $\top$ and $\bot$ we have;

$$\forall x \in \mathcal{P} \quad \bot \subseteq x \subseteq \top$$

Note that for each partial order the equivalence classes may differ. For the induced subgraph relation the equivalence classes are the classes of pairwise isomorphic graphs, however for the vertex minor partial order, P_3 and K_3 belong to the same equivalence class (Figure 4.1). This effect is most profound for $\top$ where all graphs belong to the same equivalence class.

Let us define a binary operation on the partially order set $(\mathcal{P}, \subseteq)$. The operation is the greatest lower bound of two elements of $\mathcal{P}$. An element $z \in \mathcal{P}$ is the greatest lower bound of $x, y \in \mathcal{P}$ if

– $z \leqslant x$ and $z \leqslant y$, and

– for all $w \in \mathcal{P}$ if $w \leqslant x$ and $w \leqslant y$ then $w \leqslant z$.

Definition 25. The *meet* operation, denoted $\curlywedge$, on the partially ordered set $(\mathcal{P}, \subseteq)$ is defined as $x \curlywedge y = x \cap y$ for all $x, y \in \mathcal{P}$.

Lemma 26. *The meet operation as defined in Definition 25 is associative, commutative and idempotent.*

Proof. From definition 25 the meet operation is commutative, associative and idempotent by virtue of the intersection operation having these properties. □

Let us define a binary operation on the partially ordered set $(\mathcal{P}, \subseteq)$. The operation is the least upper bound of two elements of $\mathcal{P}$. The element $z \in \mathcal{P}$ is the join of $x, y \in \mathcal{P}$ if

– $x \leqslant z$ and $y \leqslant z$, and

– for all $w \in \mathcal{P}$ if $x \leqslant w$ and $y \leqslant w$ then $z \leqslant w$.

Definition 27. The *join* operation, denoted $\curlyvee$, on the partially ordered set $(\mathcal{P}, \subseteq)$ is defined as the transitive closure of the union of any two elements of $\mathcal{P}$.

The correctness that the operation defined in Definition 27 satisfies the necessary properties for it to be a join operation do not follow directly from the definition. The *transitive closure* of a relation R is the minimal transitive relation containing R. The union of two partial order may not be transitive. Therefore in order to ensure that the relation is a member of $\mathcal{P}$ it is required that the transitive closure is used. Let us prove that the join operation as defined in Definition 27 is associative, commutative and idempotent.

Lemma 28. *The join operation as defined in definition 27 is associative, commutative and idempotent.*

Proof. For a relation R, we denote the transitive closure of R as R^+. We first prove that the join operation is idempotent, let $x \in \mathcal{P}$ then from Definition 27 $x \curlyvee x = (x \cup x)^+$ clearly $(x \cup x)^+ = x^+$. As $x \in \mathcal{P}$ then x is a partial order, from the definition of a partial order x is transitive and x is the smallest transitive relation containing x therefore $x \curlyvee x = x$. Next we show that the join relation is commutative, let $x, y \in \mathcal{P}$ then from the definition $x \curlyvee y = (x \cup y)^+$ and $y \curlyvee x = (y \cup x)^+$ as union is commutative it follows that join is also commutative. Finally we show that the join relation is associative by proving that $((x \cup y)^+ \cup z)^+ = (x \cup y \cup z)^+$ and $(x \cup (y \cup z)^+)^+ = (x \cup y \cup z)^+$. Observe that if $A \subseteq B$ then $A^+ \subseteq B^+$. Let us prove $((x \cup y)^+ \cup z)^+ = (x \cup y \cup z)^+$ by showing both directions of the subset inclusion. For the reverse direction observe that $(x \cup y \cup z) \subseteq (x \cup y)^+ \cup z$ and therefore $(x \cup y \cup z)^+ \subseteq ((x \cup y)^+ \cup z)^+$. For the forwards direction observe that from the definition of transitive closure $(x \cup y)^+$ is the smallest transitive relation containing $(x \cup y)$ and as $(x \cup y \cup z)^+$ contains $(x \cup y)$ then

$(x \cup y)^+ \subseteq (x \cup y \cup z)^+$. Also $z \subseteq (x \cup y \cup z)^+$ therefore $(x \cup y)^+ \cup z \subseteq (x \cup y \cup z)^+$ consequently $((x \cup y)^+ \cup z)^+ \subseteq ((x \cup y \cup z)^+)^+$. To conclude that $((x \cup y)^+ \cup z)^+ \subseteq ((x \cup y \cup z)^+$ it is sufficient to observe that $((x \cup y \cup z)^+)^+ = (x \cup y \cup z)^+$. The same argument can be applied to show that $(x \cup (y \cup z)^+)^+ = (x \cup y \cup z)^+$. We have demonstrated that $((x \cup y)^+ \cup z)^+ = (x \cup y \cup z)^+$ and $(x \cup (y \cup z)^+)^+ = (x \cup y \cup z)^+$ therefore $((x \cup y)^+ \cup z)^+ = (x \cup (y \cup z)^+)^+$ proving that the join operation is associative. $\square$

We next show that the meet and join operations as defined above satisfy the absorption law. The absorption law states;

$$x \curlyvee (x \curlywedge y) = x \curlywedge (x \curlyvee y) = x$$

Lemma 29. *The meet and join operations satisfy the absorption law.*

Proof. It is required to prove

$$x \cap (x \cup y)^+ = (x \cup (x \cap y))^+ = x$$

Observe that $x \subseteq (x \cup y)^+$ therefore $x \cap (x \cup y)^+ = x$. To show $(x \cup (x \cap y))^+ = x$ observe that $x \cup (x \cap y) = x$, as x is a partial order x is a transitive relation and therefore $x^+ = x$. Therefore meet and join operations satisfy the absorption law. $\square$

A partially ordered set equipped with two commutative, associative and idempotent binary operations connected by the absorption law forms a *lattice*. A lattice with a maximum and minimum element is called a bounded lattice.

Theorem 30. $\mathcal{L} = ((\mathcal{P}, \subseteq), \curlywedge, \curlyvee, \top, \bot)$ *is a bounded lattice.*

Proof. From Lemmas 28 and 26 we have that the binary operations $\curlyvee$ and $\curlywedge$ are commutative, associative and idempotent. From Lemma 29 we have that the operations satisfy the absorption law. It remains to show that $\mathcal{L}$ is closed with respect to $\curlywedge$ and $\curlyvee$. Observe that the intersection of two partial orders is reflexive, transitive and antisymmetric and is therefore a partial order. From this it follows that for all $x, y \in \mathcal{P}$ we have $x \curlywedge y \in \mathcal{P}$ as $\mathcal{P}$ contains all partial orders on $\mathcal{G}$, this partial order $x \curlywedge y$ is unique as $x \cap y$ is unique, therefore meet $\mathcal{L}$ is closed with respect to the meet operation. Observe that the union of two partial orders is a reflexive antisymmetric relation, by taking the transitive closure we obtain a relation that is also transitive and is therefore a partial order and consequently must be a member of $\mathcal{P}$. As $\top$ and $\bot$ are the maximum and minimum elements of $(\mathcal{P}, \subseteq)$ then $\mathcal{L}$ is a bounded lattice. $\square$

We introduce notation for the ideal and filter of an element $x \in \mathcal{P}$. Let $\text{ideal}(x)$ and $\text{filter}(x)$ denote the ideal and filter of an element $x \in \mathcal{P}$ respectively, that is, $\text{ideal}(x) = \{y \mid y \leqslant x \wedge y \in \mathcal{P}\}$ and similarly for $\text{filter}(x)$. Let $p : \mathcal{P} \to \{\mathrm{T}, \mathrm{F}\}$ be a property of a partial order, we say that the filter inherits a property if $x \in \mathcal{P}$;

$$p(x) \implies \forall y \in \text{filter}(x) \quad p(y)$$

analogously we say the ideal inherits a property if $x \in \mathcal{P}$

$$p(x) \implies \forall y \in \text{ideal}(x) \quad p(y).$$

For example the property of being well-founded is inherited by the ideal and the property of being a member of some filter is inherited by the filter, that is for some $x, y \in \mathcal{P}$ where $x \subseteq y$ we have $\text{filter}(y) \subseteq \text{filter}(x)$. We present a number of properties of partial orders that will be required in later chapters.

Theorem 31. *Let $x \in \mathcal{P}$ and let x be a well-founded partial order then for all $y \in \text{ideal}(x)$ y is well-founded.*

Proof. Let $x \in \mathcal{P}$ such that x is well-founded and let $y \in \text{ideal}(x)$. Suppose y is not well-founded then there exists an infinite descending chain in y. As $y \subseteq x$ then x also contains this infinite descending chain contradicting the choice of x. □

The meet of two partial orders, as defined in Definition 25, is the intersection of two partial orders. A consequence of Theorem 31 is that the meet of two well-founded partial orders is also well-founded. This can be seen by observing the intersection of two partial orders is in the ideal of both partial orders, therefore, as both partial orders are well-founded then the intersection must be well-founded. The join of two well-founded partial orders is not well-founded this can be seen in Example 32.

Example 32. Let $(G_i)_{i=0}^{\infty}$ be a sequence of graphs in $\mathcal{G}$. Consider the partial orders $\leqslant_1$ and $\leqslant_2$ defined on the set $\mathcal{G}$ where

$$\leqslant_1 = \{(G_i, G_i), (G_{2i+1}, G_{2i}) \mid i \in \mathbb{Z}^+\}$$
$$\leqslant_2 = \{(G_i, G_i), (G_{2i}, G_{2i-1}) \mid i \in \mathbb{Z}^+\}.$$

Observe that both $\leqslant_1$ and $\leqslant_2$ are well-founded, however there exists an infinite descending chain in $(\leqslant_1 \cup \leqslant_2)^+$. The sequence $G_0, G_1, G_2, \ldots$ is an infinite descending chain with respect to $(\leqslant_1 \cup \leqslant_2)^+$.

Theorem 33. *Let $x \in \mathcal{P}$ and let x be without an infinite antichain then for all $y \in \text{filter}(x)$ then y is without an infinite antichain.*

Proof. Let $x \in \mathcal{P}$ and let x be without an infinite antichain. Let $y \in \text{filter}(x)$ and assume that y has an infinite antichain. Therefore there must exist an infinite antichain A in $\mathcal{G}$ with respect to y. As $x \subseteq y$ then A is also an infinite antichain in x contradicting the assertion that x is without infinite antichains. □

Theorem 33 proves that the property of a partial order being without an infinite antichain is inherited by the filter and Theorem 31 proves that the well-founded property is inherited by the ideal. If a partial order is both well-founded and without infinite antichains then the partial order is a well-quasi ordering. The graph minor theorem shows that the class of all finite graphs is well-quasi ordered with respect to the minor relation. Using the lattice defined above this result can be extended, this extension is trivial and is highlighted only to demonstrate that the lattice can be used as a tool. The consequence of the above theorems is that the class of all graphs $\mathcal{G}$ is well-quasi ordered by a number of partial orders that are defined in Chapter 2. These include the immersion minor and lift minor partial orders. The property of a partial order being a well-quasi ordering is inherited by the filter provided that the partial order is well-founded. All of the partial orders defined in Chapter 2 are well-founded.

The property of a graph class being closed with respect to a partial order is inherited by the ideal of $\leqslant$.

Theorem 34. *Let $\mathcal{C}$ be a graph class closed with respect to $\leqslant$ where $\leqslant \in \mathcal{P}$ then for all $y \in \mathrm{ideal}(\leqslant)$ we have that $\mathcal{C}$ is closed with respect to y.*

Proof. Let $\mathcal{C}$ be a graph class closed with respect to $\leqslant$ where $\leqslant \in \mathcal{P}$ and let $y \in \mathrm{ideal}(\leqslant)$ such that $\mathcal{C}$ is not closed with respect to y. There must exist a pair $(G, H) \in y$ such that $H \in \mathcal{C}$ and $G \notin \mathcal{C}$. As $y \subseteq \leqslant$ then $(G, H) \in \leqslant$ therefore contradicting the assertion that $\mathcal{C}$ is closed with respect to $\leqslant$. □

The above theorem proves that graph class closure with respect to a partial order is inherited by the ideal in the lattice structure. It is noteworthy that the above theorem states nothing regarding the cardinality of the forbidden set. Theorem 5 on page 12 states that if a graph class is closed with respect to a partial order then the class can be characterised by a forbidden set. Therefore if $\mathcal{C}$ is closed with respect to $\leqslant$ then $\mathcal{C}$ can be characterised by a forbidden set with respect to $\leqslant$ and $\mathcal{C}$ can also be characterised by a forbidden set with respect to any partial order $y \in \mathcal{P}$ such that $y \in \mathrm{ideal}(\leqslant)$. The cardinality of the forbidden set is however not inherited by the ideal, in fact quite the opposite. For example, consider a graph class $\mathcal{C}$ closed with respect the minor relation then by the graph minor theorem [139] the class $\mathcal{C}$ has a finite minimal forbidden set with respect to the minor relation, by Theorem 34 the class $\mathcal{C}$ is also closed with respect to the induced subgraph relation but $\mathcal{C}$ might have an infinite forbidden set with respect to the induced subgraph relation. A concrete example of this is of the class of graphs without cycles; with respect to the minor relation the class is $\{K_3\}$-free$_\mathrm{m}$ but with respect to the induced subgraph relation the class is $\{C_k \mid k \leq 3\}$-free$_\mathrm{i}$, this demonstrates the potential difference in cardinality. It is possible to infer some results about the cardinality of the minimal forbidden set under certain conditions. Using the lattice structure the following theorems become evident.

Theorem 35. *For a graph class $\mathcal{C}$ closed with respect to $\leqslant_1$ and $\leqslant_2 \in \mathrm{ideal}(\leqslant_1)$ then for all $H \in \mathrm{Forb}(\mathcal{C})_2$ there exists a graph $H' \in \mathrm{Forb}(\mathcal{C})_1$ such that $H' \leqslant_1 H$.*

Proof. Let $\mathcal{C}$ be a graph class closed with respect to $\leqslant_1$ and let $\leqslant_2 \in \mathrm{ideal}(\leqslant_1)$. Suppose there exists a graph $H \in \mathrm{Forb}(\mathcal{C})_2$ such that for all $H' \in \mathrm{Forb}(\mathcal{C})_1$ we have $H' \nleqslant_1 H$. The class $\mathcal{C}$

is closed with respect to $\leqslant_1$ and $H \notin \mathcal{C}$ therefore there must exist a graph $H' \in \text{Forb}(\mathcal{C})_1$ such that $H' \leqslant_1 H$, contradicting the supposition. □

Corollary 36. For a graph class $\mathcal{C}$ closed with respect to $\leqslant_1$ and $\leqslant_2 \in \text{ideal}(\leqslant_1)$ and $\text{Forb}(\mathcal{C})_2$ is finite then $\text{Forb}(\mathcal{C})_1 = \text{minimal}(\text{Forb}(\mathcal{C})_2)_1$.

It is a natural extension of graph classes to consider if the union and intersection of two graph classes is closed with respect to some partial order when individually the two classes are closed with respect to the same partial order. This has been considered for the minor relation where the focus was to establish a means of computing the minimal forbidden set [1]. The union of two graph classes which are closed with respect to $\leqslant$ is also closed with respect to $\leqslant$ for all $\leqslant \in \mathcal{P}$. The cardinality of the forbidden set may not be finite even if the two classes are finite, unless there is some underlying property that precludes infinite forbidden sets.

Theorem 37. *For all graph classes $\mathcal{C}$ and $\mathcal{D}$ closed with respect to $\leqslant$ the class $\mathcal{C} \cup \mathcal{D}$ is closed with respect to $\leqslant$.*

Proof. Observe that $\mathcal{C}$ and $\mathcal{D}$ are subsets of $\mathcal{C} \cup \mathcal{D}$. Assume that $\mathcal{C} \cup \mathcal{D}$ is not closed with respect to $\leqslant$ then there must exist a pair G, H such that $G \leqslant H$ and $G \notin \mathcal{C} \cup \mathcal{D}$ and $H \in \mathcal{C} \cup \mathcal{D}$. If $G, H \in \mathcal{C}$ or $G, H \in \mathcal{D}$ then this contradicts the statement that $\mathcal{C}$ and $\mathcal{D}$ are closed with respect to $\leqslant$. Therefore G and H are members of different classes. Without loss of generality assume $G \in \mathcal{C}$ and $H \in \mathcal{D}$, as $\mathcal{D}$ is closed with respect to $\leqslant$ then for all $H' \leqslant H$ we have $H' \in \mathcal{D}$. Observe that $G \leqslant H$ which implies that $G \in \mathcal{D}$ and therefore $G \in \mathcal{C} \cup \mathcal{D}$ a contradiction that $G \notin \mathcal{C} \cup \mathcal{D}$. □

For the intersection of two graph classes closed with respect to $\leqslant$ the resulting class is closed with respect to $\leqslant$ and the minimal forbidden set can be expressed generally for all well-founded partial orders.

Theorem 38. *For all graph classes $\mathcal{C}$ and $\mathcal{D}$ closed with respect to $\leqslant$ the class $\mathcal{C} \cap \mathcal{D}$ is closed with respect to $\leqslant$.*

Proof. Assume that $\mathcal{C} \cap \mathcal{D}$ is not closed with respect to $\leqslant$ then there exists a pair of graphs G, H such that $G \leqslant H$ where $G \notin \mathcal{C} \cap \mathcal{D}$ and $H \in \mathcal{C} \cap \mathcal{D}$. As $H \in \mathcal{C}$ and $H \in \mathcal{D}$ and both classes $\mathcal{C}$ and $\mathcal{D}$ are closed with respect to $\leqslant$ implies that $G \in \mathcal{C}$ and $G \in \mathcal{D}$. This implies that $G \in \mathcal{C} \cap \mathcal{D}$ contradicting that $G \notin \mathcal{C} \cap \mathcal{D}$. □

The fact that the minimal forbidden set can be expressed generally is partially related to the definition of a graph class. Consider a graph class $\mathcal{C}$ closed with respect to a well-founded partial order $\leqslant$ and $\text{Forb}(\mathcal{C}) = \{H_0, \ldots, H_n\}$ then;

$$\forall 0 \leq i \leq n \quad \mathcal{C} \subseteq \{H_i\}\text{-free}.$$

The class $\mathcal{C}$ contains those graphs that are free from all of the minimal forbidden graphs and hence;

$$\mathcal{C} = \bigcap_{0 \leq i \leq n} \{H_i\}\text{-free.} \tag{4.1}$$

The intersection of two classes $\mathcal{C}$ and $\mathcal{D}$ is the restriction of the class $\mathcal{C}$ to the members of $\mathcal{D}$ and therefore the forbidden set for $\mathcal{C} \cap \mathcal{D}$ will contain elements from Forb($\mathcal{C}$) and Forb($\mathcal{D}$), however the elements of Forb($\mathcal{C}$) $\cup$ Forb($\mathcal{D}$) may not be minimal with respect to $\leqslant$.

Theorem 39. *For all graph classes $\mathcal{C}$ and $\mathcal{D}$ closed with respect to $\leqslant$ the class $\mathcal{C} \cap \mathcal{D}$ is characterised by the forbidden set* minimal(Forb($\mathcal{C}$) $\cup$ Forb($\mathcal{D}$))

Proof. Let $\mathcal{C}$ and $\mathcal{D}$ be graph classes closed with respect to $\leqslant$ and let $\mathcal{E} = \mathcal{C} \cap \mathcal{D}$. From Theorem 38, $\mathcal{E}$ is closed with respect to $\leqslant$ and can therefore be characterised with respect to $\leqslant$, it remains to show that Forb($\mathcal{E}$) = minimal(Forb($\mathcal{C}$) $\cup$ Forb($\mathcal{D}$)). Observe that for all $G \in \mathcal{E}$ the graph G does not contain a graph in Forb($\mathcal{C}$) or Forb($\mathcal{D}$) with respect to $\leqslant$. Therefore $\mathcal{E}$ = (Forb($\mathcal{C}$) $\cup$ Forb($\mathcal{D}$))-free however some elements of Forb($\mathcal{C}$) $\cup$ Forb($\mathcal{D}$) may be comparable. By taking the minimal elements of Forb($\mathcal{C}$) $\cup$ Forb($\mathcal{D}$) the minimal forbidden set is obtained. In conclusion that $\mathcal{E}$ = (minimal(Forb($\mathcal{C}$) $\cup$ Forb($\mathcal{D}$)))-free. □

Corollary 40. For all graph classes $\mathcal{C}$ and $\mathcal{D}$ characterised by a finite minimal forbidden set, the class $\mathcal{C} \cap \mathcal{D}$ is characterised by a finite forbidden set.

It is noteworthy that when considering the minimal forbidden set the partial order under consideration must be well-founded otherwise the existence of minimal elements is not certain.

Partial orders defined by graph operations

The lattice as defined in Section 4.1 is an infinite bounded lattice, the cardinality of the lattice is infinite however there is a maximum and minimum element. All of the partial orders defined in Chapter 2 are members of $\mathcal{L}$ and have been shown to be interesting for specific reasons. It remains that the lattice $\mathcal{L}$ contains a number of uninteresting partial orders that have not yet appeared in the literature. The partial orders that have appeared in the literature have demonstrated some practical usefulness. Due to their practical usefulness the partial orders often have intuitive description, past that of just an abstract relation satisfying the reflexive, transitive and antisymmetric properties. The descriptions often stems form a finite set of a transformations which may be applied to the graphs; the partial order can then be defined as the composition of a finite set of these transformations. Here we call the transformations *graph operations*. Considering the partial orders defined in Chapter 2, these partial orders are defined by; vertex deletion, edge deletion, inverse subdivision, edge contraction, local complement and pivoting.

Consider a sublattice of $\mathcal{L}$ of those partial orders defined in Chapter 2 as shown in Figure 4.2. The sublattice is constructed by taking the set of operations $\{-, \backslash, /, \multimap\}$ and forming a lattice type structure from the set $2^{\{-,\backslash,/,-\!\bullet\!-\}}$ ordered by subset inclusion (see Page 6). The elements of this structure define partial orders that allow for the given operations. As the elements are partial orders then the elements of $2^{\{-,\backslash,/,-\!\bullet\!-\}} \subseteq \mathcal{L}$, some elements are removed such as $\{/, -\!\bullet\!-\}$ as this is equivalent to the partial order defined by the set $\{/\}$. Figure 4.2 shows the Hasse diagram of the described structure.

From the definitions of the partial orders on graphs defined in Chapter 2 the following facts are easily inferred:

(i) All topological minors are also minors, *i.e.*, $G \leqslant_{\mathrm{t}} H \implies G \leqslant_{\mathrm{m}} H$.

(ii) All pivot minors are also vertex minors, *i.e.*, $G \leqslant_{\mathrm{p}} H \implies G \leqslant_{\mathrm{v}} H$.

(iii) All minors are also lift minors, *i.e.*, $G \leqslant_{\mathrm{m}} H \implies G \leqslant_{\mathrm{lift}} H$.

(iv) All immersion minors are lift minors, *i.e.*, $G \leqslant_{\mathrm{l}} H \implies G \leqslant_{\mathrm{lift}} H$.

(v) All lift contractions are lift minors, *i.e.*, $G \leqslant_{\mathrm{lc}} H \implies G \leqslant_{\mathrm{lift}} H$.

Properties

Certain properties of partial orders have useful consequences, like the consequences of being a well-quasi ordering. In order to prevent a case by case exploration of all partial orders it is favourable to identify individual properties that partial orders can possess that imply a result. The partial orders in Figure 4.2 have additional properties which will be used in later chapters.

Dual well-founded

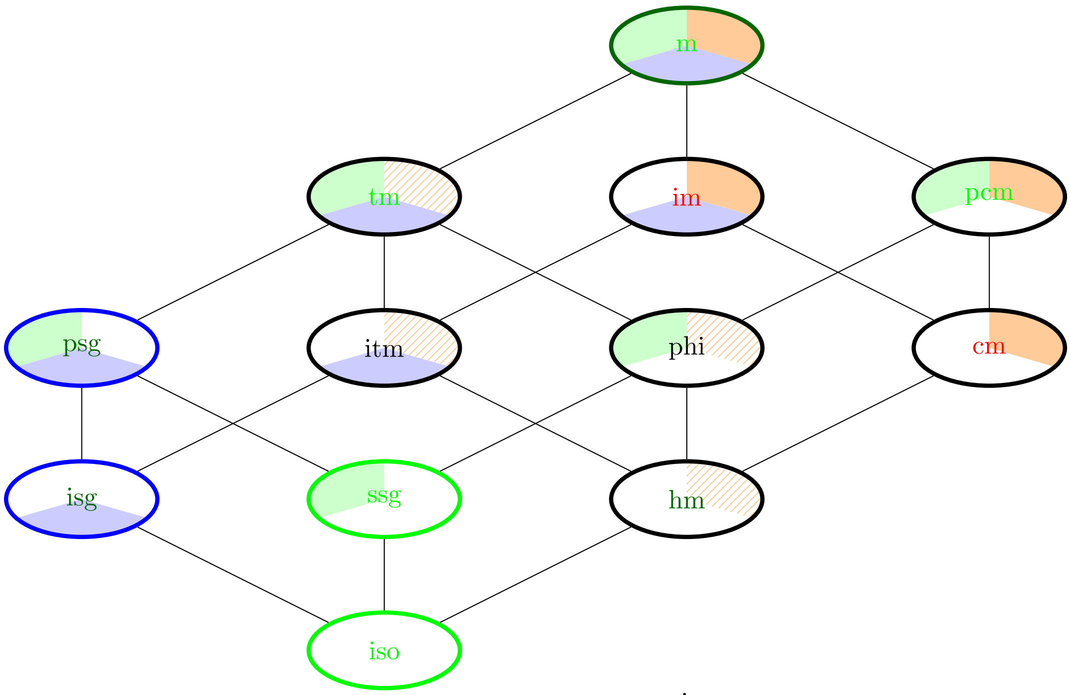
The dual well-founded property states that the ground set is partitioned into finite classes. This property is possessed by the graph isomorphism and spanning subgraph partial orders.

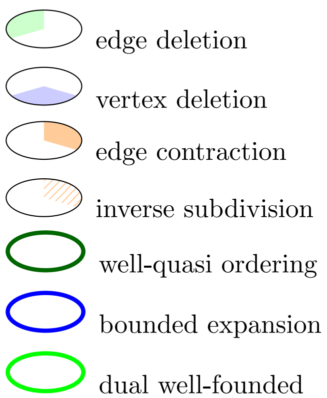
Definition 41. A partial order $\leqslant$ is *dual well-founded* if and only if $\leqslant$ and its dual $\geqslant$ contains no strictly descending chains.

Order descending partial orders

The *order descending* property states that the order of the graph does not increase as a chain in the partial order is descended. All partial orders that are defined in Chapter 2 possess this property. This property implies that the partial order is well-founded, however, well-foundedness does not imply the order descending property.

Definition 42. A partial order $\leqslant$ is *order descending* if $G \leqslant H$ implies $|G| \leq |H|$ for all graphs $G, H \in \mathcal{G}$.





m	minor
tm	topological minor
im	induced minor
pcm	partial contraction minor
psg	partial subgraph
itm	induced topological minor
phi	partial homeomorphic image
cm	contraction minor
isg	induced subgraph
ssg	spanning subgraph
hm	homeomorphic minor
iso	isomorphism

Figure 4.2: Hasse diagram of a lattice of partial orders

Hasse diagram of a lattice of partial orders. The problem $\{G \mid H \leqslant G\}$ parameterized by H is fixed-parameter tractable, polynomial for every H or $\mathbb{NP}$-complete for some fixed H.

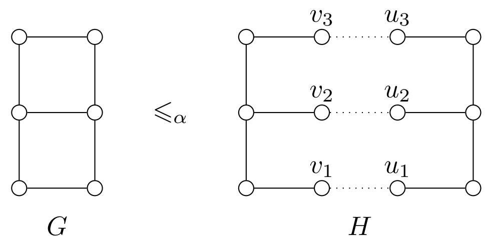


Figure 4.3: Figure illustrating the $\leqslant_\alpha$ relation does not have the bounded expansion property where $\alpha \in \{\leqslant_{\mathrm{t}}, \leqslant_{\mathrm{it}}, \leqslant_{\mathrm{c}}, \leqslant_{\mathrm{e}}, \leqslant_{\mathrm{m}}\}$. G is not $\leqslant_\alpha$ of any proper subgraph of H implying the only minimal set of vertices U of H such that $G \leqslant_\alpha H[U]$ is $V(H)$. This shows that the size of U cannot be bounded by the order of G.

Bounded expansion partial orders

The *bounded expansion* property states that the image of an embedding of a graph should be bounded in size. This property is possessed by spanning subgraph, partial subgraph, induced subgraph, pivot minor, vertex minor partial orders.

Definition 43. A partial order $\leqslant$ has the *bounded expansion* property if for all $G \leqslant H$ and for any $U \subseteq V(H)$ where U is minimal with the property that $G \leqslant H[U]$ then $|U| \leq f(G)$ for some function $f : \mathcal{G} \to \mathbb{Z}^+$ and the partial order has the order descending property.

The set U, in Definition 43, is called the preimage of G in H. The topological minor, induced topological minor, contraction minor, induced minor and minor relations do not have the bounded expansion property. As shown in Figure 4.3 the size of a minimal set of vertices of H that are in the relationship with G is unbounded, *i.e.*, the v_iu_i-paths may be of any length where $i = 1, 2, 3$. For the aforementioned partial orders the function, as described in Definition 43, is the function $f(G) = |G|$. All of the partial orders in Figure 4.2 are ordering descending partial orders and are therefore all also well-founded; of those partial orders graph isomorphism and spanning subgraphs are dual well-founded. The partial subgraph and induced subgraph partial orders both have the bounded expansion property. If all of the partial orders in Chapter 2 are considered then the pivot minor and vertex minor partial orders also have the bounded expansion property.

Complexity

The complexity of determining if two graphs are in a specific relation is a theoretically interesting question as it is essential if the relation is to have algorithmic applications. The complexity of the containment problem, that is determining if $G \leqslant H$, is difficult to classify generally. There is no simple property of a partial order that implies the containment problem will be polynomial, $\mathbb{NP}$-complete or fixed-parameter tractable. The results in Table 3.1 on page 31 highlights the apparent lack of structure in the complexity of the containment problem. Table 3.1 does not provide a complexity result for all partial orders considered in Figure 4.2, those partial orders are partial homeomorphic image, homeomorphic minor and spanning subgraph. The containment complexity for the latter two partial orders is fixed-parameter tractable, both containment algorithms use a similar concept. The underlying concept can be extended more generally to apply to many partial orders.

The homeomorphic minor partial order is defined by inverse subdivision. The partial order is well-founded and has the order descending property. Note that for each graph $G \in \mathcal{G}$ there is a unique minimal element with respect to $\leqslant_{\text{hm}}$. This unique element is called the *core* and is denoted by core(G). If $G \leqslant H$ then core$(G) \simeq$ core(H).

Figure 4.4: core(G) and G respectively

This observation forms the basis of an algorithm for deciding $G \leqslant_{\text{hm}} H$. It has been shown that determining if two graphs are isomorphic is fixed-parameter tractable when parameterized by a number of graph parameters including feedback vertex set [104], eigenvalue multiplicity [8], and treewidth [14]. The general idea behind the algorithm is to reduce the graphs to their cores, if the cores are not isomorphic then $G \nleqslant_{\text{hm}} H$ otherwise we check each bijection to see if it can be extended to an embedding of G into H. The interpretation of "extended" is dependent on the partial order being considered. For homeomorphic minors the interpretation of extends is if there is a mapping that maps paths between vertices $u, v \in V(G)$ to paths of equal to or greater length between the vertices $f(u), f(v) \in V(H)$ (See Figure 4.5). The outline of this algorithm is provided in Algorithm 2.

The complexity of Algorithm 2 for each fixed graph G is polynomial. The algorithm iterates over all possible bijections of $V(\text{core}(G)) \to V(\text{core}(H))$, as the size of $V(\text{core}(G))$ is bounded by the size of $V(G)$ the number of bijections in finite. For each bijection the algorithm gets the set of paths between each pair of vertices in core(G) in G and the set of paths between the corresponding vertices of core(H) in H, disjoint from $V(\text{core}(G))$ and $V(\text{core}(H))$ respectively. This can be achieved using depth-first search. If $G \leqslant_{\text{hm}} H$ then there exists a bijective function between these sets of paths such that for each path in H there is a path in G of at least the same length. The overall run time is of the order $f(|G|) \cdot n^2$ where f is dominated by the number of bijections between core(G) and core(H).

A similar approach is also possible for the spanning subgraph containment problem. For the approach to work there must exist an embedding of the core into the original graph. The embedding is in the form of an injective function between $V(\text{core}(G))$ and $V(G)$ that is structure

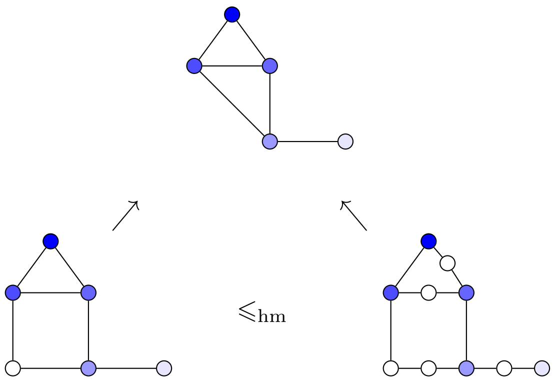


Figure 4.5: An example of two homeomorphic graphs (bottom) with their unique core (top). Shading identifies a valid embedding of the core into each of the graphs.

preserving. The "structure preserving" constraint depends on the partial order; for homeomorphic minors the structure is the existence of paths, for spanning subgraphs the structure being preserved is adjacency. The contraction minor and induced subgraph relations have the property that if $G \leqslant H$ then $\mathrm{core}(G) \simeq \mathrm{core}(H)$, however the partial orders destroy all structure making it impossible to reconstruct an embedding in fixed-parameter tractable time. The partial homeomorphic image partial order does not have a unique minimal element for each graph therefore checking if the cores are isomorphic does not work.

The lattice provides a tool for determining the containment complexity for some partial orders. Given two partial orders $\leqslant_1$ and $\leqslant_2$ where the containment complexities are known it is possible to determine an upper bound on the containment complexity for the partial order $\leqslant_1 \curlywedge \leqslant_2$ and to provide an algorithm for the containment problem, *i.e.*, the proof is constructive. As the meet of two partial orders is defined as the intersection then it follows that the conjunction of the two containment algorithms is the algorithm for the meet (see Algorithm 3).

The computational complexity of Algorithm 3 is the "maximum" of the containment complexities for $\leqslant_1$ and $\leqslant_2$, *e.g.* if the containment complexity for $\leqslant_1$ and $\leqslant_2$ are polynomial then the resulting algorithm will be polynomial, likewise if both containment complexities are fixed-parameter tractable.

For the join of two partial orders the lattice does not help in determining the containment complexity. It would be reasonable to consider that as the meet is the conjunction of the two containment algorithms that the join might be the disjunction of the two containment algorithms however this is not the case. This idea fails to provide the correct answer for the elements of the union of the partial orders introduced by taking the transitive closure.

Algorithm 2: Algorithm for homeomorphic minor containment problem

Input: A graph $G = (V, E)$
Input: A graph $H = (U, F)$
Output: {True, False} returning True if $G \leqslant_{\mathrm{hm}} H$ and False otherwise

$G' = \mathrm{core}(G)$
$H' = \mathrm{core}(H)$
// F is a bijective function
$F = \{f : V(G') \to V(H') \mid \forall uv \in E(G') \iff f(u)f(v) \in E(H')\}$
for $f \in F$ **do**
  for $uv \in E(G')$ **do**
    S = the set of uv-paths in G disjoint from $V(G')$
    S' = the set of $f(u)f(v)$-paths in H disjoint from $V(H')$
    if $\neg\exists B : S \to S'$ *such that* B *is a bijection and* $\forall s \in S\ |s| \leq |B(s)|$ **then**
      break // next $f \in F$
    end
  end
  return True
end
return False

Algorithm 3: Generic algorithm for testing $G \leqslant H$ where $\leqslant = \leqslant_1 \wedge \leqslant_2$.

Input: A graph $G = (V, E)$
Input: A graph $H = (U, F)$
Output: True if $G \leqslant H$ and False otherwise
Data: A function `Containment(`$\leqslant$,G,H`)` to test $G \leqslant H$

return `Containment(`$\leqslant_1$,G,H`)` $\wedge$ `Containment(`$\leqslant_2$,G,H`)`

4.2 Partial orders & parameterized graph classes

The parameterized classes defined in Chapter 2 have many practical applications which make them of interest for study. As demonstrated in previous chapters characterising graph classes with respect to some partial order has provided a number of insightful structural properties which have been exploited to develop efficient algorithms. This encourages the question as to whether the parameterized graph classes can also be characterised with respect to the same partial order that is used to characterise the base class. For certain parameterized graph classes such as graphs of bounded treewidth the algorithmic implications are evident. The class of graphs of bounded treewidth is closed with respect to the minor relation and therefore has a finite obstruction set and can be recognised in cubic time [137]. Coupling this with the results of Courcelle [30] then any property that is treewidth bounding and expressible in monadic second order logic can be recognised in cubic time. Some parameterized graph classes lend themselves easily to being characterised by a partial order because the parameter is closely related to a graph property that the partial order preserves. Examples of this can be seen for the minor relation and treewidth and the vertex minor relation and rankwidth.

The parameterized graph classes that are considered here are those related to the graph modification problems, *i.e.*, $\mathcal{C}+k\mathsf{v}$, $\mathcal{C}+k\mathsf{e}$, $\mathcal{C}-k\mathsf{v}$ and $\mathcal{C}-k\mathsf{e}$. A set of relationships between the classes is established below. Assuming the classes are closed with respect to the partial order under consideration then for the classes $\mathcal{C}+k\mathsf{v}$, $\mathcal{C}+k\mathsf{e}$ and $\mathcal{C}-k\mathsf{e}$ for each distinct $k > 0$, $\mathcal{C}+(k-1)\mathsf{v} \subseteq \mathcal{C}+k\mathsf{v}$, $\mathcal{C}+(k-1)\mathsf{e} \subseteq \mathcal{C}+k\mathsf{e}$ and $\mathcal{C}-(k-1)\mathsf{e} \subseteq \mathcal{C}-k\mathsf{e}$.

Theorem 44. $\mathcal{C}+(k-1)\mathsf{v} \subseteq \mathcal{C}+k\mathsf{v}$

Proof. This is immediate from the definition. □

Theorem 45. $\mathcal{C}+(k-1)\mathsf{e} \subseteq \mathcal{C}+k\mathsf{e}$

Proof. This is immediate from the definition. □

Theorem 46. $\mathcal{C}+k\mathsf{e} \subseteq \mathcal{C}+k\mathsf{v}$ *where* $\mathcal{C}$ *is closed with respect to the induced subgraph relation.*

Proof. Let $G \in \mathcal{C}+k\mathsf{e}$ and $U \subseteq E(G)$ such that $(G - U) \in \mathcal{C}$ and $|U| \leq k$. We can always find a set of vertices U' of at most size $|U|$ such that $(G - U') \in \mathcal{C}$. The set U' is obtained by selecting an end point of each of the k edges that should be removed in order to obtain a graph in $\mathcal{C}+k\mathsf{e}$. □

A graph class can be characterised with respect to a partial order if the class is closed with respect to the partial order and the partial order is well-founded. Without the well-founded property there is no guarantee that there will be a minimal non-member of the class. The characterisation with respect to a partial order yields a forbidden set which may be finite or infinite. The characterisation provides an insight into the structural properties of the graphs belonging to the graph class. For example, consider the class of chordal graphs which forbid all chordless cycles of length four or more. This characterisation has led to a number of efficient algorithms being developed. The elements of the minimal forbidden set often have a vital role in the proof of an algorithms correctness where each of the minimal forbidden graphs are handled separately. It is therefore of interest to establish the minimal forbidden set with respect to a partial order. For some partial orders there is the possibility that the minimal forbidden set is infinite. The graphs in an infinite minimal forbidden set can often be grouped together into families that share common structural features; again consider chordal graphs, the minimal forbidden set is infinite but all elements have a common structural feature that they are simple cycles. For those partial orders that are well-quasi orderings then all minimal forbidden set are finite, but if the partial order is not a well-quasi ordering then the minimal forbidden set may be infinite. It is of theoretical interest to establish the minimal forbidden set and whether this set is finite. To establish if a graph class is closed with respect to a partial order and to determine the cardinality of the minimal forbidden has up to date been considered on a class by class basis, each class being considered individually. For the parameterized graph classes we consider it interesting to look for properties of the partial orders that imply that a parameterized graph class is closed and has a finite forbidden set. An advantage to obtaining such a result is that it removes the requirement for each class to be considered separately and allows results to be applied more generally. For example, if a graph class is closed with respect to some partial order, has a finite minimal forbidden set and the containment problem for that partial order has an efficient algorithm then the combination of these results yields a generic graph class recognition algorithm (see Algorithm 4).

Algorithm 4: Generic algorithm for class recognition given that the class is closed with respect to $\leqslant$ and has a finite forbidden set,.

Input: A graph $G = (V, E)$
Output: $\{\mathsf{True}, \mathsf{False}\}$ returning True if $G \in \mathcal{C}$ and False otherwise
Data: The minimal forbidden set for $\mathcal{C}$ denoted $\{H_0, \dots H_n\}$

for $H \in \{H_0, \dots H_n\}$ **do**
    if $H \leqslant G$ **then**
        return False
    end
end
return True

Algorithm 4 is a generic graph class recognition algorithm. The algorithm requires that the graph class is closed with respect to the partial order being considered, the minimal forbidden set is finite and an algorithm for testing if $(G, H) \in \leqslant$. The first requirement is taken as a promise, without the graph class being closed with respect to the partial order then there exists no minimal forbidden set. The problem of generating the minimal forbidden set is a challenge. There is not always an efficient algorithm for this task. Consider the minor relation, from the graph minor theorem we know that all minimal forbidden sets are finite, however it has been shown in [55] that it is often an undecidable problem to compute the obstruction set. The last requirement that there is an algorithm to test $(G, H) \in \leqslant$ is the component of the algorithm which dictates the runtime of the resulting algorithm.

Cases when the three requirements for the above algorithm are not met are still of theoretical interest. To determine if a class is closed with respect to a partial order and if the minimal forbidden set is finite can still lead to a better understanding of the graph class.

$\mathcal{C}+k\mathsf{v}$

Characterising the parameterized graph class $\mathcal{C}+k\mathsf{v}$ is dependent on the partial order as to whether it is possible or useful. There are a number of factors that may have an effect when characterising the class $\mathcal{C}+k\mathsf{v}$. These factors include whether the class $\mathcal{C}$ has a finite minimal forbidden set, whether $\mathcal{C}+k\mathsf{v}$ is closed with respect to $\leqslant$ and for applications such as in Algorithm 4 whether Forb($\mathcal{C}+k\mathsf{v}$) is finite. For some partial orders these factors follow easily from known results, for others the results have not been shown in a general setting owing to the previous class by class investigations.

For the induced subgraph relation if Forb($\mathcal{C}$+kv) is to be finite it is necessary that Forb($\mathcal{C}$) is finite. To demonstrate, let $\mathcal{C}$ be the class of chordal graphs, the minimal forbidden graphs for the class of chordal graphs is $\{C_n \mid n \geq 4\}$. The class $\mathcal{C}$+kv has an infinite minimal forbidden set, this can be seen by observing that the elements of $\{(k+1)C_n \mid n \geq 4\}$ are minimal and forbidden for the class and therefore are members of Forb($\mathcal{C}$+kv). This behaviour is not specific to the induced subgraph relation and applies more generally, *e.g.*, to the partial subgraph relation.

Although the class of Chordal+kv has an infinite forbidden set the class has been successfully characterised in [19] where a recognition algorithm is presented. For the class Interval+kv a characterisation was presented in [21] despite interval graphs having an infinite minimal forbidden set. In both cases the result of the characterisation is a fixed-parameter recognition algorithm and for the latter an algorithm for the minimum-Interval-completion problem. In both of these cases no explicit minimal forbidden set is provided, rather structural properties of the graph classes are used.

The assumption that the parameterized graph class, $\mathcal{C}$+kv, is closed with respect to a partial order on the account that the base class is closed with respect to that partial order is incorrect. There is no straightforward implication that can be formed regarding the closure of a parameterized graph class. This can be seen in Example 47 where a counterexample is provided demonstrating that the parameterized graph class $\mathcal{C}$+kv, in general, is not closed with respect to the contraction minor relation, even if the class $\mathcal{C}$ is closed with respect to this relation.

Example 47. Let $\mathcal{C} = \{iK_1 \mid 0 < i < c\}$-free$_{\mathrm{c}}$, the class contains all those graphs with at least c connected components. Observe that $K_{1,c} \in \mathcal{C}$+1v and $K_{1,c-1} \notin \mathcal{C}$+1v and that $K_{1,c-1} \leqslant_{\mathrm{c}} K_{1,c}$. For $c = 5$ Figure 4.6 illustrates the example.

Figure 4.6: $K_{1,4}$ and $K_{1,5}$ respectively. An example of $\mathcal{C}$+1v not being closed with respect to $\leqslant_{\mathrm{c}}$.

There exist partial orders that given a graph class $\mathcal{C}$ where Forb($\mathcal{C}$) is finite then the class $\mathcal{C}$+kv does not have a finite minimal forbidden set. This can be seen in Example 48 where a counterexample is presented demonstrating that for the induced topological minor relation there exist graph classes which have a finite minimal forbidden set but the parameterized class $\mathcal{C}$+kv where $k = 1$ has an infinite minimal forbidden set.

Example 48. Let $\mathcal{C} = K_3$-free$_{\mathrm{it}}$, the class $\mathcal{C}$+1v is closed with respect to $\leqslant_{\mathrm{it}}$ and the forbidden set is infinite. Observe that $\{W_n \mid n \geq 4\} \subset$ Forb($\mathcal{C}$+1v) (see Figure 4.7).

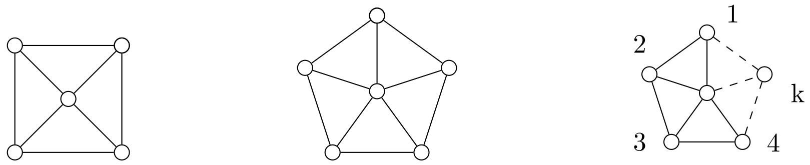


Figure 4.7: W_4, W_5 and W_k. An example of Forb($\mathcal{C}$+1v) not being finite where Forb($\mathcal{C}$) is finite with respect to $\leqslant_{\text{it}}$.

For some partial orders the classes $\mathcal{C}$+1v and $\mathcal{C}$+2v are not distinct despite $\mathcal{C}$ being closed with respect to the considered partial order. An example of this can be seen in Example 49.

Example 49. Let $\mathcal{C}$ be the class of $\{K_3\}$-free$_{\text{hm}}$. The class $\mathcal{C}$+1v is the set of all graphs. Let $G \in \mathcal{G}$ we show that $G \in \mathcal{C}$+1v. If G has no vertex of degree 2 then $G \in \mathcal{C}$ and therefore is a member of $\mathcal{C}$+1v. If G has a vertex $u \in V(G)$ of degree 2 and v is a neighbour of u then $G - v \in \mathcal{C}$ implying $G \in \mathcal{C}$+1v. As $\mathcal{G} = \mathcal{C}$+1v and $\mathcal{C}$+1v $\subseteq \mathcal{C}$+2v then $\mathcal{G} = \mathcal{C}$+2v.

Positive results

Despite the obstacles to characterising the class $\mathcal{C}+k$v outlined in the previous sections there are properties of the partial order that imply the parameterized graph class is closed with respect to the partial order and has a finite minimal forbidden set.

For all of the partial orders included in Figure 4.2 the cardinality of the set Forb($\mathcal{C}$) affects the cardinality of the set Forb($\mathcal{C}+k$v). If $\mathcal{C}$ has an infinite minimal forbidden set then $\mathcal{C}+k$v has an infinite minimal forbidden set for all $k \geq 0$ assuming that for each k the class $\mathcal{C}+k$v is distinct from $\mathcal{C}+(k+1)$v and $\mathcal{G}$. Therefore, if we wish to obtain finite minimal forbidden set characterisations then we should restrict our attention to those graph classes where the minimal forbidden set for $\mathcal{C}$ is finite. For the minor relation this is all minor closed classes. For any other partial order included in Figure 4.2 these are the classes that forbid any finite set of graphs, such as; threshold graphs, cographs, split graphs, triangle-free graphs and bull-free graphs [17].

It is essential that the class $\mathcal{C}+k$v is closed with respect to $\leqslant$ if we are to obtain a characterisation by forbidding a set of graphs. For a number of well-studied partial orders if $\mathcal{C}$ is closed with respect to $\leqslant$ then $\mathcal{C}+k$v is also closed. As has been shown in Example 47 it is not generally the case that $\mathcal{C}+k$v is closed with respect to $\leqslant$ if $\mathcal{C}$ is closed with respect to $\leqslant$. For a general result we require the property of $\leqslant$ stating that $\mathcal{C}$ is closed with respect to $\leqslant$ implies $\mathcal{C}+k$v is closed with respect to $\leqslant$. As the definition of the parameterized class $\mathcal{C}+k$v includes the removal of vertices then it would be reasonable to restrict $\leqslant$ to the set of partial order that include vertex deletion as an operation. Theorem 50 proves that for any class $\mathcal{C}$ closed with respect to $\leqslant_{\text{i}}$ then the class $\mathcal{C}+k$v is also closed with respect to $\leqslant_{\text{i}}$.

Theorem 50. *Let $\mathcal{C}$ be a graph class closed with respect to the induced subgraph relation then the class $\mathcal{C}+k$v is closed with respect to the induced subgraph relation.*

Proof. Let $\mathcal{C}$ be a graph class closed with respect to the induced subgraph relation. Suppose that the statement is not true, that is $\mathcal{C}$ is closed with respect to the induced subgraph relation and $\mathcal{C}+k\mathsf{v}$ is not. Let $\mathcal{C}$ be a graph class closed with respect to $\leqslant_{\mathrm{i}}$ and the class $\mathcal{C}+k\mathsf{v}$ is not closed with respect to the induced subgraph relation then there must exist a pair of graphs H and G such that $H \leqslant_{\mathrm{i}} G$ where $H \notin \mathcal{C}+k\mathsf{v}$ and $G \in \mathcal{C}+k\mathsf{v}$. As $H \leqslant_{\mathrm{i}} G$ then there exists an injective adjacency preserving function $\varphi : V(H) \to V(G)$. From the definition of $\mathcal{C}+k\mathsf{v}$ there must exist a set of vertices U where $|U| \leq k$ such that $G - U \in \mathcal{C}$. Observe that $H - U' \leqslant_{\mathrm{i}} G - U$ where $U' = \{\varphi^{-1}(u) \mid u \in U \cap V(H)\}$ however as $H \notin \mathcal{C}+k\mathsf{v}$ then for any subset $T \subseteq V(H)$ where $|T| < k+1$ then $H - T \notin \mathcal{C}$. Recall that $|U| \leq k$ therefore $H - U' \notin \mathcal{C}$ which implies there exists a graph $H' \in \mathrm{Forb}(\mathcal{C})$ such that $H' \leqslant_{\mathrm{i}} H - U'$. From the transitivity of $\leqslant_{\mathrm{i}}$ we obtain that $H' \leqslant_{\mathrm{i}} G - U$. Recall that $G - U \in \mathcal{C}$ and $H' \notin \mathcal{C}$. A contradiction that $\mathcal{C}$ is closed with respect to the induced subgraph relation, a diagrammatic representation is provided in Figure 4.8. □

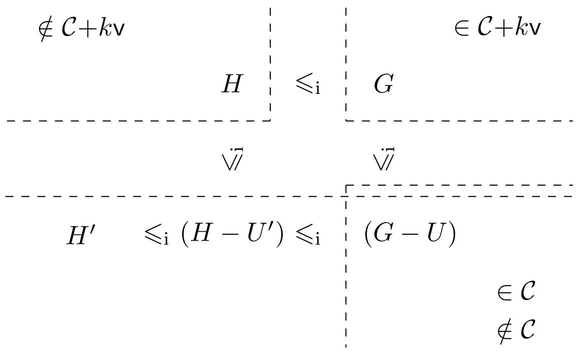


Figure 4.8: Diagrammatic representation of Theorem 50

This result can be extended to a wider set of partial orders including the partial subgraph relation, topological minor relation and minor relation.

Theorem 51. *Let $\mathcal{C}$ be a class closed with respect to the minor relation then $\mathcal{C}+k\mathsf{v}$ is closed with respect to the minor relation.*

Proof. Let $\mathcal{C}$ be a class closed with respect to the minor relation. Let $G \in \mathcal{C}+1\mathsf{v}$ and let $u \in V(G)$ such that $G - u \in \mathcal{C}$. Let $H \leqslant_{\mathrm{m}} G$ such that G covers H. There are three cases to consider; H is obtained from G by deleting a vertex, H is obtained from G by deleting an edge or H is obtained from G by contracting an edge. Consider the case where H is obtained by a vertex deletion and let v be that vertex. Assume $u \neq v$ otherwise it follows easily that $H \in \mathcal{C}$ and therefore is in $\mathcal{C}+1\mathsf{v}$. Observe that $((G - u) - v) = ((G - v) - u)$ and $H = G - v$. As $((G - u) - v) \in \mathcal{C}$ from the assertion that $\mathcal{C}$ is closed and $H - u = ((G - v) - u)$ then clearly $H - u \in \mathcal{C}$ and therefore $H \in \mathcal{C}+1\mathsf{v}$. Consider the case where H is obtained by an edge deletion and let e be that edge. Observe that $((G \setminus e) - u) = ((G - u) \setminus e)$ and $H = G \setminus e$.

As $G-u \in \mathcal{C}$ then $((G-u) \setminus e) \in \mathcal{C}$ from the assertion that $\mathcal{C}$ is closed therefore $(H-u) \in \mathcal{C}$ and $(H-u) \in \mathcal{C}{+}1\mathsf{v}$. Consider the case where H is obtained by an edge contraction and let $e = ab$ be that edge. If $a \neq u$ and $b \neq u$ then observe that $((G/e)-u) = ((G-u)/e)$ and that $((G-u)/e) \in \mathcal{C}$ because $((G-u)/e)$ is a minor of $G-u$. Therefore $((G/e)-u) \in \mathcal{C}$ which implies that $((G/e)-u) \in \mathcal{C}{+}1\mathsf{v}$. Without loss of generality assume that $a = u$, observe that $((G/e) - \{ab\}) = ((G-a)-b)$. As $((G-a)-b)$ is a minor of $(G-u)$ then $((G-a)-b) \in \mathcal{C}$ and therefore $((G-a)-b) \in \mathcal{C}{+}1\mathsf{v}$, consequently $((G/e) - \{ab\}) \in \mathcal{C}$ and $(G/e) \in \mathcal{C}{+}1\mathsf{v}$. A simple induction argument on k implies the theorem. □

Theorem 52. *Let $\mathcal{C}$ be a class closed with respect to the topological minor relation then $\mathcal{C}{+}k\mathsf{v}$ is closed with respect to the topological minor relation.*

Proof. Let $\mathcal{C}$ be a class closed with respect to the topological minor relation. Let $G \in \mathcal{C}{+}1\mathsf{v}$ and $u \in V(G)$ such that $G-u \in \mathcal{C}$. Let $H \leqslant_{\mathrm{t}} G$ such that G covers H. There are three cases to consider; H is obtained from G by deleting a vertex, H is obtained from G by deleting an edge or H is obtained from G by vertex dissolution. Consider the case where H is obtained from G by deleting a vertex and let v be that vertex. Observe that $((G-v)-u) = ((G-u)-v)$ and that $((G-u)-v) \in \mathcal{C}$ therefore $((G-v)-u) \in \mathcal{C}$ and $(G-v) \in \mathcal{C}{+}1\mathsf{v}$. Consider the case where H is obtained from G from deleting an edge and let e be that edge. Observe that $((G \setminus e)-u) = ((G-u) \setminus e)$ and $((G-u) \setminus e) \in \mathcal{C}$ therefore $((G \setminus e)-u) \in \mathcal{C}$ and consequently $(G \setminus e) \in \mathcal{C}{+}1\mathsf{v}$. Lastly consider the case where H is obtained from G by vertex dissolution and let v be the vertex that is dissolved with neighbours v_1, v_2. If $u \notin \{v, v_1, v_2\}$ then observe that $((G \mathbin{-\!\bullet\!-} v)-u) = ((G-u) \mathbin{-\!\bullet\!-} v)$ and that $((G-u) \mathbin{-\!\bullet\!-} v) \in \mathcal{C}$ therefore $((G \mathbin{-\!\bullet\!-} v)-u) \in \mathcal{C}$ and $(G \mathbin{-\!\bullet\!-} v) \in \mathcal{C}{+}1\mathsf{v}$. If $u \in \{v_1, v_2\}$ then observe that $((G \mathbin{-\!\bullet\!-} v)-u) = ((G-u)-v)$ implying that $((G \mathbin{-\!\bullet\!-} v)-u) \in \mathcal{C}$ and $(G \mathbin{-\!\bullet\!-} v) \in \mathcal{C}{+}1\mathsf{v}$. If $u = v$ then observe that $((G \mathbin{-\!\bullet\!-} v)-v_1) = ((G-u)-v_1)$ implying that $((G \mathbin{-\!\bullet\!-} v)-v_1) \in \mathcal{C}$ and $(G \mathbin{-\!\bullet\!-} v) \in \mathcal{C}{+}1\mathsf{v}$. □

In general the parameterized graph class $\mathcal{C}{+}k\mathsf{v}$ is closed with respect to a partial order if the partial order can emulate vertex deletion and the modifications to the graph can be reordered. A partial order $\leqslant$ emulates vertex deletion if for all $G \in \mathcal{G}$ and for all $u \in V(G)$ then $G-u \leqslant G$.

For all of the partial orders included in Figure 4.2, if $\mathcal{C}$ is not the class of all graphs then the class $\mathcal{C}{+}1\mathsf{v}$ is distinct from $\mathcal{C}$. If for some k the graph class $\mathcal{C}{+}k\mathsf{v}$ is the same graph class as $\mathcal{C}{+}(k+1)\mathsf{v}$ then it is interesting to find the value of k and if $\mathcal{C}{+}k\mathsf{v}$ can be characterised by a finite minimal forbidden set.

The minimal forbidden set for the class $\mathcal{C}{+}k\mathsf{v}$ is finite if the partial order under consideration has the bounded expansion property and $\mathrm{Forb}(\mathcal{C})$ is finite. The proof of this result is provided in Chapter 5. As it is required that $\mathcal{C}{+}k\mathsf{v}$ is closed with respect to the partial order under consideration then Theorems 50, 51 and 52 may be applied. For other partial order it is required to show that $\mathcal{C}{+}k\mathsf{v}$ is closed with respect to the partial order under consideration.

$\mathcal{C}+k\mathsf{e}$

As with the parameterized graph class $\mathcal{C}+k\mathsf{v}$, if the class $\mathcal{C}+k\mathsf{e}$ is closed with respect to the partial order under consideration and Forb($\mathcal{C}$) is infinite then the minimal forbidden set for the class $\mathcal{C}+k\mathsf{e}$ is also infinite for all of the partial orders considered in Figure 4.2. This can be shown using the same construction as for the class $\mathcal{C}+k\mathsf{v}$. To demonstrate, let $\mathcal{C}$ that forbids the set $\{C_n \mid n \geq 4\}$. The class $\mathcal{C}+k\mathsf{e}$ has an infinite minimal forbidden set, this can be seen by observing that the elements of $\{(k+1)C_n \mid n \geq 4\}$ are minimal and forbidden for the class and therefore are members of Forb($\mathcal{C}+k\mathsf{e}$). Therefore for the partial orders we consider we should only consider graph classes where the minimal forbidden set for the base class is finite.

It is not generally the case that the class $\mathcal{C}+k\mathsf{e}$ is closed with respect to a partial order on the account that the base class is closed with respect to that partial order. This can be seen in Example 53 where a counterexample is provided to demonstrate that the parameterized class $\mathcal{C}+k\mathsf{e}$, in general, is not closed with respect to the minor relation, however, the base class is closed.

Example 53. Let $\mathcal{C} = \{3K_2 \bowtie K_1\}$-free$_{\mathrm{m}}$, the class $\mathcal{C}+1\mathsf{e}$ is not closed with respect to $\leqslant_{\mathrm{m}}$. Observe that $G \in \mathcal{C}+1\mathsf{e}$ and $G' \notin \mathcal{C}+1\mathsf{e}$ and that $G' \leqslant_{\mathrm{m}} G$ (G, G' are shown in Figure 4.9).

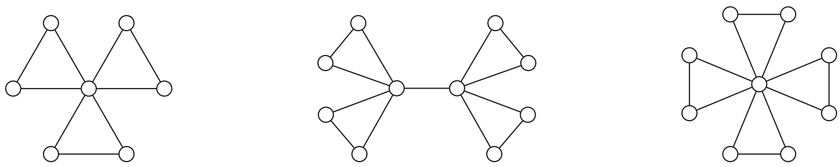


Figure 4.9: $3K_2 \bowtie K_1, G$ and G' respectively. An example of $\mathcal{C}+1\mathsf{e}$ not being closed with respect to $\leqslant_{\mathrm{m}}$.

Unlike for the class $\mathcal{C}+k\mathsf{v}$ where we are able to obtain a partial characterisation of partial orders for which we can expect the class $\mathcal{C}+k\mathsf{v}$ to be closed if $\mathcal{C}$ is closed; for the class $\mathcal{C}+k\mathsf{e}$ this is not possible. For $\mathcal{C}+k\mathsf{v}$ the partial order should be able to emulate vertex deletion and the modifications to the graph should be able to be applied in any order and result in isomorphic graphs. One might expect a similar condition for the partial orders that can emulate edge deletion but this is not the case. Example 53 demonstrates this. The minor relation can emulate edge deletion and the operations can be applied in any order yet the class $\mathcal{C}+k\mathsf{e}$ is not generally closed.

4.3 Summary

The problem of characterising parameterized graph classes has numerous practical applications. For the parameterized graph classes $\mathcal{C}+k\mathsf{v}$, $\mathcal{C}+k\mathsf{e}$ and $\mathcal{C}-k\mathsf{e}$, trying to characterise the class is not a straightforward problem and many obstacles obscure the route to a general technique. There is no trivial property of a partial order that implies that any of the parameterized graph classes we consider will be closed given that the base class is closed with respect to the partial order. The best that can be achieved in this avenue is that for specific well-studied partial orders we can show that in general the parameterized graph classes are closed, or exhibit a counterexample that precludes a general theorem. However, just because the parameterized graph class is not generally closed with respect to a partial order, does not exclude the possibility of special cases. The applicability of these special cases to practical problems is likely to be limited, and it would be little more than a theoretical exercise to attempt to provide a characterisation of those special cases where closure can be ascertained.

For the generic graph class recognition algorithm proposed in Algorithm 4 to be correct the complexity for the containment problem for the partial order should be efficient and the class must have a finitely many minimal forbidden graphs with respect to that partial order. The complexity for the containment problem is an important problem and for many of the partial orders defined in Chapter 2 the complexity of the containment problem is well-established. There appears to be no conclusions that can be drawn from the lattice regarding the complexity of the containment problem for any given partial order other than the observation that the complexity of $\leqslant_1 \wedge \leqslant_2$ is bounded by the higher complexity class of the containment complexities of $\leqslant_1$ and $\leqslant_2$. A summary of the containment complexity for the partial orders included in Figure 4.2 can be found in Table 3.1.

For those partial orders where it can be shown that $\mathcal{C}+k\mathsf{v}$ is closed and the partial order has the bounded expansion property and $\mathcal{C}$ can be characterised by a finite forbidden set then it is possible to prove that the minimal forbidden set for $\mathcal{C}+k\mathsf{v}$ is finite. Further it is possible to provide a bound on the maximum size of a graph in the minimal forbidden set and consequently provide a routine to generate the minimal forbidden set. A similar proof using the same techniques can be used to prove that any graph class $\mathcal{C}+k\mathsf{e}$, that is closed with respect to a partial order and $\mathcal{C}$ has a finite minimal forbidden set, has a finite minimal forbidden set assuming that the partial order under consideration has the bounded expansion property.

The topological minor relation does not have the bounded expansion property and thus the results presented in Chapter 5 cannot be directly applied. However, for some special cases such as those classes that are also minor closed, those classes that forbid a single topological minor for the base class and those classes that are identical to classes that can be characterised by finitely many minimal forbidden graphs with respect to a partial order with the bounded expansion property we are able to provide a proof that the parameterized graph class $\mathcal{C}+k\mathsf{v}$ is finite.

Chapter 5

Characterising *almost* graphs

The classes of graphs related to the graph modifications problems have recieved an increasing amount of attention with the developing interest in fixed-parameter algorithms. Generally the graph modification problems concern adding or removing edges and vertices from a graph until some property is satisfied. The graphs where there exists a small number of modifications that results in the graph belonging to a class are sometimes called *almost* graphs– G is almost a member of $\mathcal{C}$.

For hereditary graph classes the problem has been well-studied and a number of $\mathbb{NP}$-completeness results have been shown for the vertex deletion problem [64, GT21], edge deletion [64, GT28] and vertex and edge deletion problem [168] (see Chapter 3). This has led to the investigation of cases where the problem is polynomial time solvable. Recognition of graph classes is an interesting problem studied for almost as long as graph theory itself. The original problem in graph theory, that of determining if a graph has an Eulerian trail, can be reformulated into a graph class recognition problem. The recognition of parameterized graph classes, such as those defined by graph modification problems, has been shown to be $\mathbb{NP}$-complete and some of the problems have been shown to be fixed-parameter tractable. An example is that of finitely characterised hereditary graph classes [19]. In this chapter we provide a characterisation of the parameterized graph classes $\mathcal{C}+k\mathsf{v}$ and $\mathcal{C}+k\mathsf{e}$, demonstrating that if the class $\mathcal{C}$ is closed and has a finite minimal forbidden set with respect to a partial order there is a sufficient condition for the existence of a finite minimal forbidden set for the classes $\mathcal{C}+k\mathsf{v}$ and $\mathcal{C}+k\mathsf{e}$. This result considerably extends the work of Cai [19]. Cai shows that it is possible to recognise the class $\mathcal{C}+k\mathsf{v}$ by constructing an algorithm to do so (see Chapter 3). We contribute a combinatorial construction which proves that the classes can be characterised by a finite minimal forbidden set. This explicit construction surpasses the purely algorithmic approach of Cai [19]. Our results extend more widely than the partial order considered by Cai. We prove that there is a sufficient property of the partial order which implies the class can be characterised by a finite minimal forbidden set. This result can be seen to be interesting for a number of fields in computer science, including algorithmic graph theory and fixed-parameter tractability. The recent results of Fellows [50] show that all fixed-parameter tractable problems have "useful" obstruction sets, however in this context "useful" does not necessarily imply finite size.

From the viewpoint of algorithmic graph theory, a characterisation of the graph class in question aids in the development of algorithms. The minimal forbidden set is often used to prove the correctness of the algorithm. The results of this chapter may seem weak when viewed in the light of the graph minor theorem; however, as the partial orders we consider are not well-quasi orderings on the set of all graphs, the results are in a sense best possible solutions.

An application of these results can be seen in the field of certifying algorithms where the forbidden set can be used as a certificate for non-membership of a graph in a class. For certifying algorithms it is desirable to have the proof of correctness of a certificate to utilise a different insight to the proof of correctness of the algorithm. Without the general proof that there always exists a finite minimal forbidden set it would be left to a class by class search for suitable non-membership certificates. This application is exploited in Chapter 6 where an algorithm for enumerating the minimal forbidden set is given.

5.1 Constructing a bound for almost graphs

In the following section we provide a construction for bounding the order of a critical hypergraph. This bound is then applied in Section 5.1.2 to bound the order of the classes $\mathcal{C}+k\mathsf{v}$ and $\mathcal{C}+k\mathsf{e}$.

Critical hypergraphs

Let X be a finite set and let $\mathcal{P}(X)$ denote the power set of X. A *hypergraph* is a tuple (X, E) where X is a finite set and $E \subseteq \mathcal{P}(X)$. If $\mathcal{H} = (X, E)$ then $X = V(\mathcal{H})$ and $E = E(\mathcal{H})$ are the vertex set and the edge set of $\mathcal{H}$ respectively. A hypergraph $\mathcal{H}' = (X', E')$ is a partial hypergraph of $\mathcal{H} = (X, E)$ if $X = X'$ and $E' \subseteq E$. The degree of a vertex in a hypergraph is the number of edges the vertex belongs to

$$\deg(x) = |\{e \mid x \in e, e \in E\}| .$$

The rank of a hypergraph $\mathcal{H}$ is denoted $\mathrm{r}(\mathcal{H})$ and is the maximum size of an edge

$$\mathrm{r}(\mathcal{H}) = \max\{|e| \mid e \in E(\mathcal{H})\} .$$

We call a hypergraph r-uniform if all the edges are of size r, that is for all $e \in E(\mathcal{H})$, $|e| = r$. Notice that if we consider only the case for $r = 2$ then we are considering graphs as defined in Chapter 2. A *strongly stable set* in a hypergraph $\mathcal{H} = (X, E)$ is a set $S \subseteq X$ such that for all $e \in E$, $|S \cap e| \leq 1$. For our purposes we consider hypergraphs with additional constraints

(i) for all $v \in X$ there exists an $e \in E$ such that $v \in e$

(ii) for all $e \in E$, $|e| \geq 1$.

Simply, there are no isolated vertices and no empty edges. A *transversal* of a hypergraph $\mathcal{H} = (X, E)$ is a set $T \subseteq X$ such that every edge intersects with T. A minimal transversal, with respect to set inclusion, is a transversal T such that for all $T' \subset T$ there exists an edge $e \in E$ such that $e \cap T' = \varnothing$. A minimum transversal is a transversal of smallest size. We denote the size of a minimum transversal of a hypergraph $\mathcal{H}$ as

$$\tau(\mathcal{H}) = \min\{|T| \mid T \text{ is a transversal of } \mathcal{H}\}\,.$$

Notice that for the case of 2-uniform hypergraphs, the minimum transversal size is the same as the minimum vertex cover number. Each edge in the graph must be incident to a vertex in the cover.

Let $\mathcal{H} - e$ denote the removal of the edge $e \in E(\mathcal{H})$ from $\mathcal{H}$. A hypergraph $\mathcal{H}$ is *τ-critical* if the removal of any edge in the hypergraph reduces $\tau(\mathcal{H})$, that is, a hypergraph is *τ-critical* if

$$\tau(\mathcal{H} - e) < \tau(\mathcal{H})$$

for all $e \in E$. For example, if we consider 2-uniform hypergraphs then the graph kK_2 has a minimum transversal of size k. Any transversal must contain at least one vertex from each component and as the transversal is minimal it can contain at most one vertex from each component. Therefore the graph kK_2 is critical as the removal of any edge reduces the size of the minimum transversal by one. However, the graph C_{2k} where $k \geq 2$ which has a minimum transversal of size k is not critical as the removal of any edge yields a graph in which every minimum transversal is of size k.

The maximum number of pairwise disjoint edges in a hypergraph is denoted $\nu(\mathcal{H})$. A graph is *ν-edge-critical* if the contraction of any edge (Figure 5.1) increases the number of disjoint edges, that is

$$\nu(\mathcal{H}') > \nu(\mathcal{H})$$

whenever $\mathcal{H}' = (V(\mathcal{H}), E')$ with $E' = (E \setminus \{e\}) \cup e'$ where $e' \subsetneq e$ and $e \in E$. Observe that neither of the hypergraphs in Figure 5.1 are ν-edge-critical. Unless explicitly stated when we refer to critical hypergraphs we refer to transversal critical hypergraphs.

A set of pairs (A_i, B_i) for $1 \leq i \leq m$ forms an *intersecting set pair system* if and only if

(a) $A_i \cap B_j = \varnothing$ if and only if $i = j$ where $0 \leq i, j \leq m$.

Additionally an intersecting set pair system forms an (a, b)-system if

(b) $|A_i| = a$ and $|B_i| = b$ where $0 \leq i \leq m$.

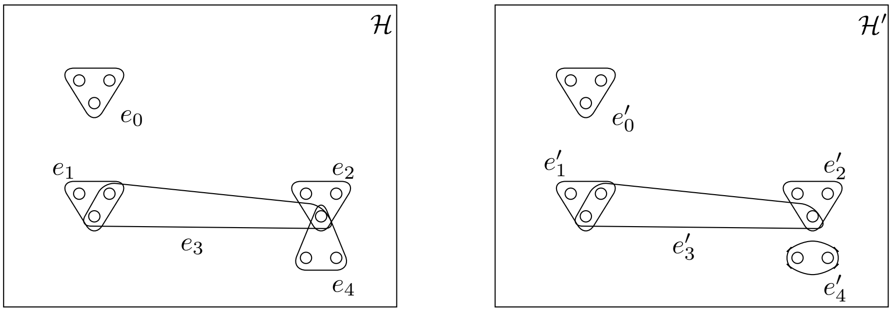


Figure 5.1: Example of hyperedge contraction. The edge e_4 is contracted to form the edge e'_4 in $\mathcal{H}'$. $\nu(\mathcal{H}) = 3$ and $\nu(\mathcal{H}') = 4$.

An important result of Bollobás in [15] is that for an (a, b)-system

$$m \leq \binom{a+b}{a}. \tag{5.1}$$

Recall from Chapter 2 that the number of edges in a graph is denoted by m, *i.e.*, $m = |E|$. The result of Bollobás in [15] has many formulations and a number of special cases have been considered in [96, 60, 62].

We will require the following result regarding a relationship between the order of a strongly stable set and the number of edges the strongly stable set *touches*. Let us define a quantity for the number of edges a set $S \subseteq V$ touches

$$\Gamma(S) = \{e \setminus \{s\} \mid s \in e \wedge e \in E \wedge s \in S\}.$$

Note that if the hypergraph is r-uniform then the set $\Gamma(S)$ contains $(r-1)$-element sets. For ease we denote $\Gamma(\{x\})$ as $\Gamma(x)$ and $\deg(x) = |\Gamma(x)|$.

Theorem 54 ([76])**.** *If $\mathcal{H} = (X, E)$ is a τ-critical hypergraph without isolated vertices and $S \subseteq X$ is a strongly stable set in $\mathcal{H}$ then $|S| \leq |\Gamma(S)|$.*

Proof. Suppose the statement is not true. Let $t = \tau$ and let S be a strongly stable set of minimal cardinality such that $|\Gamma(S)| < |S|$. There must exist an element $a \in S$ such that $Y \subseteq S \setminus \{a\}$ where

$$|\Gamma(Y) \setminus \Gamma(a)| < |Y| \tag{5.2}$$

consequently $\Gamma(Y) \cap \Gamma(a) \neq \varnothing$ otherwise $|S| \leq |\Gamma(S)|$ is true. Let $Y \subseteq S \setminus \{a\}$ such that Y is minimal for some $a \in S$ and Y satisfies Equation 5.2.

As a result of $\Gamma(Y) \cap \Gamma(a) \neq \varnothing$ there must exist an element $b \in Y$ and a set $W \in \Gamma(Y) \cap \Gamma(a)$ such that $W \in \Gamma(a)$ and $W \in \Gamma(b)$ where $a, b \in S$ and $a \neq b$. Let $f_a = W \cup \{a\}$ and $f_b = W \cup \{b\}$, clearly $f_a, f_b \in E$.

As Y is minimal there exists a bijection from $Y \setminus \{b\}$ to $\Gamma(Y \setminus \{b\}) \setminus \Gamma(a)$ (as a consequence of the Kőnig-Hall Theorem [13, Theorem 5]). Let n be a bijective function $n : Y \setminus \{b\} \to \Gamma(Y \setminus \{b\}) \setminus \Gamma(a)$ such that for all $z \in Y \setminus \{b\}$ we have $n(z) \in \Gamma(z)$ and set $n(b) = W$. Observe that $n(z)$ is defined for all $z \in Y$. For each $z \in Y$ let $p(z)$ be an element of $n(z)$.

From the assumption that $\mathcal{H}$ is τ-critical the hypergraph $\mathcal{H}' = (X, E \setminus \{f_a\})$ has a transversal T of size $t-1$. Consequently $e \cap T \neq \varnothing$ for all $e \in E \setminus \{f_a\}$ and $T \cap f_a = \varnothing$ otherwise T would be a transversal of $\mathcal{H}$. It follows that $b \in T$ and $a \notin T$ because $T \cap W = \varnothing$ otherwise $T \cap f_a \neq \varnothing$ and therefore T would be a transversal of $\mathcal{H}$. Note that for all $z \in \Gamma(a) \setminus \{W\}$ the intersection between T and z is not empty.

Let $T' = (T \setminus Y) \cup \{p(z) \mid z \in T \cap Y\}$, observe that $|T'| \leq |T|$ as we substitute at most one element for each element of Y. We now show that T' is a transversal of $\mathcal{H}$ having at most $t-1$ elements which contradicts the assumption that $\mathcal{H}$ is a τ-critical hypergraph. There are three types of edges to consider in order to prove that T' is a transversal of $\mathcal{H}$;

(i) edges that are disjoint from $Y \cup \{a\}$

(ii) the edge f_a (the edges containing a)

(iii) the edges that intersect with Y

Clearly T' intersects every edge that is disjoint from $Y \cup \{a\}$, as these edges are unaffected by substituted elements of the transversal. As $b \in Y$, b is substituted in T' by the element $p(b)$, *i.e.* $p(b) \in T'$. Recall from the definition of the function p that $p(b) \in n(b)$ and that $n(b) = W$ therefore $p(b) \in W$. We know that $f_a \setminus \{a\} \subseteq W$ therefore $p(b) \in f_a$ and consequently $T' \cap f_a \neq \varnothing$. As T intersects every set $z \in \Gamma(a) \setminus W$ and $T' \cap f_a \neq \varnothing$ then $T' \cap z \neq \varnothing$ for all $z \in \Gamma(a)$. It remains to show that the edges associated with Y are covered by T'. Recall that n is a bijection between Y and $\Gamma(Y) \setminus \Gamma(a)$ therefore $\Gamma(Y) \setminus \Gamma(a) = \{n(z) \mid z \in Y \setminus \{b\}\}$. If a set $n(z) \cap T = \varnothing$ then $z \in T$ and from the substitution $p(z) \in T'$ therefore $n(z) \cap T' \neq \varnothing$ for all $z \in Y$. From this we obtain that T' is a transversal of $\mathcal{H}$. From the construction of T' we have that $|T'| \leq (t-1)$ therefore $\mathcal{H}$ is not a τ-critical hypergraph. We reject that $|\Gamma(S)| < |S|$ and therefore accept that $|S| \leq |\Gamma(S)|$. □

Establishing a bound on the order and size of a hypergraph with certain propertes is an interesting problem from a combinatorial point of view. We intend to show that the problem we consider can be formulated into a problem of bounding the order of a hypergraph.

For an r-uniform t-critical hypergraph with minimum transversal number t, the order is bounded from above by a function of r and t [76]. Let $\nu_{\max}(r,t)$ denote the maximum order of a r-uniform τ-critical hypergraph with minimum transversal number t. The investigation into determining a value for $\nu_{\max}(r,t)$ has received a lot of attention with researchers establishing special cases for fixed values of r and t. The problem first appeared in [48] and for $r = 2$ the problem was solved, where $r = 3$ an implication of the correct order was provided in [148]. In [76] Gyárfás *et al.* provided a solution for the general case for all $r, t \geq 1$. The result was later extended in [154] by Tuza using a method which generalises to a number of similar problems. The result in [154] uses the concept of intersecting set pair systems. A general bound is established for an intersecting set pair system then a number of problems are shown to be representable using the intersecting set pair systems, including the maximum order of r-uniform τ-critical hypergraphs.

The main result of [76] is;

Theorem 55. *Let $\mathcal{H}$ be an r-uniform τ-critical hypergraph then $|V(\mathcal{H})|$ is bounded,*

$$|V(\mathcal{H})| \leq \nu_{\max}(r,t) \leq t^{r-1} + t\binom{t+r-2}{r-2} \tag{5.3}$$

where $r \geq 2$ and t is the minimum transversal number of $\mathcal{H}$.

We present a proof that follows the outline of the proof in [76, proof of Theorem 2].

Proof. Let $\mathcal{H} = (X, E)$ be an r-uniform τ-critical hypergraph with $\tau(\mathcal{H}) = t$. Let T be the t-uniform hypergraph formed by the t-element minimal transversals of $\mathcal{H}$. If T' is a partial hypergraph of T and $e \in E$ we define $\mathrm{m}(e, T')$ to be the minimum cardinality of all subsets of e that intersect every edge in T', that is;

$$\mathrm{m}(e, T') = \min\{|Y| \mid Y \subseteq e \wedge \forall f \in E(T')\,(Y \cap f \neq \varnothing \wedge)\}$$

Observe some properties of the quantity $\mathrm{m}(e, T')$, for all $e \in E$

(i) $\mathrm{m}(e, T') \leq r$

(ii) $\mathrm{m}(e, T) = r$

Property (i) is easily observed from the definition of $\mathrm{m}(e, T')$, $\mathrm{m}(e, T')$ is the smallest subset of a finite set satisfying a property therefore the size of the set cannot be larger than the original set, as e is an edge of an r-uniform hypergraph then $\mathrm{m}(e, T') \leq r$. Property (ii) is a consequence of the criticality of $\mathcal{H}$. Assume $\mathrm{m}(e, T) \neq r$. Then $\mathrm{m}(e, T) < r$, and there exists an element $x \in e$ such that for all $f \in E(T)$ we have $f \cap (e \setminus \{x\}) \neq \varnothing$. Let $f \in E(T)$. Clearly, f is also a transversal of $\mathcal{H} - e$, and since $\mathcal{H}$ is τ-critical, f is not a minimum transversal of $\mathcal{H} - e$. Therefore, there exists some $x' \in f$ such that $f \setminus \{x'\}$ is a transversal of $\mathcal{H} - e$. Since $|f \setminus \{x'\}| < t$, $f \setminus \{x'\}$ is not a transversal of $\mathcal{H}$, therefore, $f \setminus \{x'\} \cap e = \varnothing$. But now, $(f \setminus \{x'\}) \cup \{x\}$ is a minimum transversal of $\mathcal{H}$ with $f \cap e = \{x\}$ and consequently $f \cap (e \setminus \{x\}) = \varnothing$, contradicting the assumption on e.

There must exist an element $x \in e$ such that for all $f \in E(T)$ we have $x \notin f$ but as T contains all t-element transversals of $\mathcal{H}$ then for all $f \in E(T)$ $f \cap (e \setminus \{x\}) \neq \varnothing$ because f is a transversal of $\mathcal{H}$. Clearly for all $f \in E(T)$, f is a transversal of $\mathcal{H} \setminus e$. As $\mathcal{H}$ is τ-critical f cannot be a minimal transversal of $\mathcal{H} \setminus e$ therefore there must exist an element $y \in f$ such that $f \setminus y$ is a minimal transversal of $\mathcal{H} \setminus e$ but then $(f \setminus \{y\}) \cup \{x\}$ is a transversal of $\mathcal{H}$ and therefore $(f \setminus \{y\}) \cup \{x\} \in E(T)$ contradicting our assumption that there exists no transversal containing x as an element. Therefore $\mathrm{m}(e,T) = r$.

We next show that we can modify T until we obtain a hypergraph with a specific set of properties. We require the following claim.

Claim 56. For all $e \in E$ if $f \in E(T')$ then $\mathrm{m}(e,T') \leq \mathrm{m}(e,T' \setminus f) + 1$.

Proof. Let $Y \subseteq e$ be a set achieving the minimum in the definition of $\mathrm{m}(e,T' \setminus f)$, let $e \in e \cap f$, and let $Y' = Y \cup \{x\}$. Then, Y' is a subset of size at most $\mathrm{m}(e,T' \setminus f) + 1$ intersecting all edges of T'. ■

From Claim 56 we may therefore repeatedly remove edges from T until we obtain a hypergraph T^* satisfying the following properties

(i) for all $e \in E \quad r-1 \leq \mathrm{m}(e,T^*) \leq r$

(ii) for all $f \in E(T^*)$ there exists an edge $e \in E$ such that $\mathrm{m}(e,T^* \setminus f) = r-2$.

If Y is a subset of some edge in E then by $f(Y)$ let us denote an edge $f \in E(T^*)$ such that $f \cap Y = \varnothing$. For some $e \in E$ and $Y \subset e$ where $|Y| \leq r-2$ then property (i) ensures there exists an $f \in E(T^*)$ such that $f \cap Y = \varnothing$.

Let $f_0 \in E(T^*)$ be chosen arbitrarily and be fixed. We construct a hypergraph $\mathcal{H}^*$ from all the different $(r-1)$-element sets constructed such that the following constraints are satisfied. For each $e \in E$ select a $(r-1)$-element subset $\{x_1,\ldots,x_{r-1}\} \subset e$ such that

$$\begin{aligned} x_1 &\in e \cap f_0 \\ x_2 &\in e \cap f(\{x_1\}) \\ x_3 &\in e \cap f(\{x_1,x_2\}) \\ &\vdots \\ x_{r-1} &\in e \cap f(\{x_1,\ldots,x_{r-2}\})\,. \end{aligned}$$

The size of $\mathcal{H}^*$ can be seen to be bounded, $|E(\mathcal{H}^*)| \leq t^{r-1}$. For x_1 there are at most t choices as $|e \cap f_0| \leq t$. For each fixed $x_1,\ldots,x_i - 1$ where $2 \leq i \leq r-1$ there are at most t choices for x_i.

For every $f \in E(T^*)$ property (ii) ensures that there exists an $(r-2)$-element subset of some $e \in E$, denoted $X(f)$, such that $f \cap X(f') = \varnothing$ if and only if $f = f'$. That is $\{(f,X(f)) \mid f \in E(T^*)\}$ form an intersecting set pair system. Moreover f, $X(f)$ form an (a,b)-system where $a = r-2$ and $b = t$ implying that;

$$|E(T^*)| \leq \binom{t+r-2}{r-2}.$$

by the application of Equation 5.1. As T^* is a t-uniform hypergraph then;

$$|V(T^*)| \leq |E(T^*)|t \leq \binom{t+r-2}{r-2}t.$$

Let $S = X \setminus V(\mathcal{H}^*)$, observe that S is a strongly stable set of $\mathcal{H}$. The set S is strongly stable on the account of each edge in $E(\mathcal{H})$ being a superset of some edge in $E(\mathcal{H}^*)$ therefore the maximum intersection between the set S and an edge in $E(\mathcal{H})$ is the difference in the size of the edges in $E(\mathcal{H})$ and $E(\mathcal{H}^*)$, as $\mathcal{H}$ is r-uniform and $\mathcal{H}^*$ is $(r-1)$-uniform the difference is at most 1. Therefore the intersection is at most 1 and agrees with the definition of a strongly stable set. Also observe that $\Gamma(S) \subseteq E(\mathcal{H}^*)$. From the definition of Γ we get that $\Gamma(S)$ contains $(r-1)$-element subsets of edges in E that touch the set S. As $E(\mathcal{H}^*)$ also contains $(r-1)$-element subsets of edges in E it remains to show that no set in $\Gamma(S)$ contains an element not in $V(\mathcal{H}^*)$. Suppose some element $Y \in \Gamma(S)$ contains an element that is not in $V(\mathcal{H}^*)$ then there must be an edge $e \in E$ such that $|e \cap S| > 1$. If this is the case then by the pigeon hole principle at least one of the elements in $e \cap S$ must belong to some edge in $E(\mathcal{H}^*)$ and therefore the element is not a member of S.

We continue with some elementary rearrangements and substitutions.

$$\begin{aligned}
|X| &= |S| + |V(\mathcal{H}^*)| && \text{rearrangement from above}\\
&\leq |\Gamma(S)| + |V(\mathcal{H}^*)| && \text{substituting } |S| \text{ for } |\Gamma(S)| \text{ from Theorem 54}\\
&\leq |E(\mathcal{H}^*)| + |V(\mathcal{H}^*)| && \text{substituing } |\Gamma(S)| \text{ for } |E(\mathcal{H}^*)|\\
&\leq t^{r-1} + \binom{t+r-2}{r-2}t
\end{aligned}$$

As $t = \tau(\mathcal{H})$ the proof is concluded and;

$$|X| \leq \tau^{r-1} + \binom{\tau+r-2}{r-2}\tau.$$

□

5.1.1 Overview

An overview of the technique used to characterise $\mathcal{C}+k\mathsf{v}$ and $\mathcal{C}+k\mathsf{e}$ is given followed by the detailed result. In order to provide a characterisation of the parameterized classes $\mathcal{C}+k\mathsf{v}$ and $\mathcal{C}+k\mathsf{e}$ we demonstrate that there is a bound on the maximum order of a graph in the minimal forbidden set, therefore the size of the minimal forbidden set is finite as there are only a finite number of graphs with order less than or equal to a given bound. The bound is established by constructing a hypergraph and demonstrating that the constructed hypergraph satisfies a set of properties that allow the application of Theorem 55. This technique also provides an explicit upper bound on the maximum order of a minimal forbidden graph.

5.1.2 $\mathcal{C}+k\mathsf{v}$

Recall the definition of the class $\mathcal{C}+k\mathsf{v}$, the class $\mathcal{C}+k\mathsf{v}$ is the class of graphs where there exists a set of at most k vertices such that the removal of the vertices yields a graph belonging to the class $\mathcal{C}$. The definitions and results of the previous section can be used to show that the minimal forbidden set for the class $\mathcal{C}+k\mathsf{v}$ is finite where the class $\mathcal{C}$ is closed with respect to the induced subgraph relation and has a finite forbidden set. More generally, for any partial order which has the bounded expansion property (see Section 43 on page 73) if $\mathcal{C}$ has a finite minimal forbidden set then the set $\mathcal{C}+k\mathsf{v}$ has a finite minimal forbidden set if $\mathcal{C}+k\mathsf{v}$ is closed with respect to the partial order under consideration. Observe that to prove that the minimal forbidden set for $\mathcal{C}+k\mathsf{v}$ is finite it is sufficient to demonstrate that the maximum order of a graph in the minimal forbidden set is bounded. As the partial order under consideration has the bounded expansion property it is therefore order descending and as a consequence is well-founded, consequently there is a minimal forbidden set.

Let $\mathcal{C}$ be a class of graphs closed with respect to a partial order that has the bounded expansion property, has a finite minimal forbidden set and the class $\mathcal{C}+k\mathsf{v}$ is closed with respect to the partial order. Let $\mathrm{Forb}(\mathcal{C})$ denote the minimal forbidden set for $\mathcal{C}$. As $\mathrm{Forb}(\mathcal{C})$ is finite there exists a maximum order of a graph in the set, *i.e.*, there exists an integer such that $r \geq |H|$ for all $H \in \mathrm{Forb}(\mathcal{C})$. Let r denote the maximum order of a graph in $\mathrm{Forb}(\mathcal{C})$.

Let $\mathcal{C} = \{F_0, \ldots, F_n\}$-free with respect to the partial order under consideration. For a fixed graph $H \in \mathrm{Forb}(\mathcal{C}+k\mathsf{v})$ construct the hypergraph $\mathcal{H}$ as follows:

1. $V(\mathcal{H}) = V(H)$
2. $E(\mathcal{H}) = \{e \mid e \subseteq V(\mathcal{H}) \wedge \exists F \in \mathrm{Forb}(\mathcal{C})\ H\langle e\rangle \simeq F\}$

The notation $F'\langle e\rangle$ is defined as $F'\langle e\rangle = G'$ where $G' \leqslant F'[e]$. That is the hypergraph $\mathcal{H}$ is on the same vertex set as H and for each instance of a minimal forbidden graph there is a hyperedge containing those vertices that induce a copy with respect to the partial order under consideration. Note that every vertex belongs to some edge of $\mathcal{H}$ as a result of the minimality of H. All graphs $F \in \mathrm{Forb}(\mathcal{C})$ are of bounded order, *i.e.*, for all $G \in \mathrm{Forb}(\mathcal{C})$ we have $|G| \leq r$ for some integer r. The minimum transversal size of the constructed hypergraph is $k+1$.

Lemma 57. *The hypergraph $\mathcal{H}$ has minimum transversal size of $k+1$.*

Proof. Let $\mathcal{H}$ be the hypergraph constructed as above. Suppose $\tau(\mathcal{H}) < k+1$. Then there exists a set of vertices $T \subseteq V(H)$ such that $H - T \in \mathcal{C}$ and $|T| < k+1$. Contradicting that H is forbidden for the class $\mathcal{C}+k\mathsf{v}$. Therefore $\tau(\mathcal{H}) \geq k+1$. Suppose $\tau(\mathcal{H}) > k+1$. Then H is not minimal for the class $\mathcal{C}+k\mathsf{v}$. Consequently $\tau(\mathcal{H}) = k+1$. □

In order to apply Theorem 55 it is required that the hypergraph be τ-critical and r-uniform. The hypergraph $\mathcal{H}$ as constructed may not be τ-critical but there exists a τ-critical partial hypergraph of $\mathcal{H}$ that contains all vertices of H. A hypergraph $\mathcal{H}' = (Y, F)$ is a partial hypergraph of a hypergraph $\mathcal{H} = (X, E)$ if $X = Y$ and $F \subseteq E$.

Lemma 58. *If $\mathcal{H}$ is a hypergraph with $\tau(\mathcal{H}) = t$ then $\mathcal{H}$ contains a t-critical hypergraph as a partial hypergraph.*

Corollary 59. The hypergraph constructed as above contains a $(k+1)$-critical hypergraph as a partial hypergraph.

From Corollary 59 the hypergraph $\mathcal{H}$, as constructed above, contains a $(k+1)$-critical hypergraph, let $\mathcal{H}'$ be such a hypergraph. Therefore $\mathcal{H}'$ is a $(k+1)$-critical hypergraph. It remains to show that we can construct an r-uniform hypergraph while retaining the property of being $(k+1)$-critical. For each hyperedge $e \in E(\mathcal{H})$ where $|e| \leq r$ let $X(e)$ be a set of $r - |e|$ vertices disjoint from $V(\mathcal{H}) \cup \bigcup_{f \in E(\mathcal{H}) \setminus \{e\}} X(f)$. Construct the hypergraph $\mathcal{H}^*$ from $\mathcal{H}'$ as follows:

1. $V(\mathcal{H}^*) = V(\mathcal{H}') \cup \bigcup_{f \in E(\mathcal{H})} X(f)$
2. $E(\mathcal{H}^*) = \{e \cup X(e) \mid e \in E(\mathcal{H}')\}$

In order to prove that $\mathcal{H}^*$ is an r-uniform $(k+1)$-critical hypergraph. We require the following lemmas.

Lemma 60. *If T is a minimum transversal of $\mathcal{H}^*$ such that $T \cap \bigcup_{f \in E(\mathcal{H})} X(f) = \varnothing$ then T is a minimum transversal of $\mathcal{H}'$.*

Proof. Let T be a minimum transversal of $\mathcal{H}^*$ such that $T \cap \bigcup_{f \in E(\mathcal{H})} X(f) = \varnothing$. Clearly T is a transversal of $\mathcal{H}'$ from the construction of the edges of $\mathcal{H}^*$ and that $T \subseteq V(\mathcal{H}')$. Let T' be a minimum transversal of $\mathcal{H}'$. From the construction of $\mathcal{H}^*$ the transversal T' is also a minimum transversal of $\mathcal{H}^*$ therefore $|T| = |T'|$. As T is a transversal of $\mathcal{H}'$ and $|T| = |T'|$ then T must be a minimum transversal of $\mathcal{H}'$. □

Lemma 61. *Every minimum transversal of $\mathcal{H}'$ is a minimum transversal of $\mathcal{H}^*$.*

Proof. Let T be a minimum transversal of $\mathcal{H}'$. From the construction of $\mathcal{H}^*$ we have that there is a bijection $N : E(\mathcal{H}') \to E(\mathcal{H}^*)$ such that $N(e) = e'$ implies $e \subseteq e'$ therefore T is a transversal of $\mathcal{H}^*$. Suppose T is not a minimum transversal of $\mathcal{H}^*$ then there exists a transversal T' of $\mathcal{H}^*$ such that $|T'| < |T|$. From Lemma 60 we have that T' is a minimum transversal of $\mathcal{H}'$ therefore $|T'| = |T|$. Showing that T is a minimum transversal of $\mathcal{H}^*$. □

Lemma 62. *The hypergraph $\mathcal{H}^*$ is an r-uniform $(k+1)$-critical hypergraph.*

Proof. Let $\mathcal{H}^*$ be the hypergraph as constructed above. Clearly $\mathcal{H}^*$ is r-uniform as $|e \cup X(e)| = r$ for all $e \in E(\mathcal{H}')$. Let T be a minimum transversal of $\mathcal{H}^*$. If $T \cap \bigcup_{f \in E(\mathcal{H})} X(f) = \varnothing$ then T is also a minimum transversal of $\mathcal{H}'$ by Lemma 60. Implying that $|T| = k + 1$. Otherwise $T \cap \bigcup_{f \in E(\mathcal{H})} X(f) \neq \varnothing$. For each $x \in T \cap \bigcup_{f \in E(\mathcal{H})} X(f)$ we have that x belongs to exactly one edge of $\mathcal{H}^*$. Let e denote that edge. Let T' be constructed from T such that each $x \in T \cap \bigcup_{f \in E(\mathcal{H})} X(f)$ is replaced by an element $u \in e \cap V(\mathcal{H}')$. Observe that T' is a transversal of $\mathcal{H}^*$ which implies that $|T'| \geq |T|$. Also observe that each element of T that was replaced in T' was replaced by at most one element therefore $|T'| \leq |T|$. Thus $|T'| = |T|$, that is, T' is a minimum transversal of $\mathcal{H}^*$. As $T' \cap \bigcup_{f \in E(\mathcal{H})} X(f) = \varnothing$ we have that T' is also a minimum transversal of $\mathcal{H}'$, by Lemma 60, therefore $|T'| = k+1$. Therefore $\tau(\mathcal{H}^*) = k+1$.

Supose that $\mathcal{H}^*$ is not $(k+1)$-critical then there exists an edge $e \in E(\mathcal{H}^*)$ such that $\tau(\mathcal{H}^* - e) = (k+1)$. Let $e' = e \setminus X(e)$. From the construction of $\mathcal{H}^*$ the edge $e' \in E(\mathcal{H}')$. Let T be a minimum transversal of $\mathcal{H}^*$, clearly T is a minimum transversal of $\mathcal{H}^* - e$. If $T \cap \bigcup_{f \in E(\mathcal{H})} X(f) \neq \varnothing$ then we may construct a minimum transversal T' as above such that T' is a minimum transversal of $\mathcal{H}^*$ and $T' \cap \bigcup_{f \in E(\mathcal{H})} X(f) = \varnothing$. Let $T = T'$. From Lemma 60 if $T \cap \bigcup_{f \in E(\mathcal{H})} X(f) = \varnothing$ then T is a minimum transversal of $\mathcal{H}'$. As $\mathcal{H}'$ is $(k+1)$-critical then $\tau(\mathcal{H}' - e') < k+1$. Let T^* be a minimum transversal of $\mathcal{H}' - e'$. We claim that $\tau(\mathcal{H}^* - (e' \cup X(e))) < k+1$. The transversal T^* clearly intersects all edges of $\mathcal{H}^*$ except $(e' \cup X(e))$ and therefore T^* is a transversal of $\mathcal{H}^* - (e' \cup X(e))$ and has size less than $k+1$. Contradicting the assumption that $\mathcal{H}^*$ was not $(k+1)$-critical. □

From Lemma 62 we have that $\mathcal{H}^*$ is an r-uniform τ-critical hypergraph where $\tau = k+1$ and $r = \max\{|G| \mid G \in \mathrm{Forb}(\mathcal{C})\}$.

Lemma 63. *If $\mathcal{L} = (X, E)$ is an r-uniform t-critical hypergraph then $\mathcal{L}$ has bounded order.*

Proof. Let $\mathcal{L}$ be an r-uniform t-critical hypergraph. We bound $|X|$ for all values of $r \geq 1$ and $t = k+1$.

Case 1. $r = 1$

In this case $\mathcal{L}$ is a collection of isolated vertices, with each edge containing exactly one vertex. For a transversal to intersect with each edge its size equals to the size of the vertex set.

Case 2. $r \geq 2$

The maximum order of such a hypergraph is proved in Theorem 55 to be

$$\binom{t+r-2}{r-2} t + t^{r-1}$$

Therefore N is bounded in terms of s and t as follows:

$$N \leqslant \begin{cases} t & \text{if } s = 1 \\ \binom{t+s-2}{s-2} t + t^{s-1} & \text{if } s \geq 2 \end{cases}$$

□

Corollary 64. $\mathcal{H}^*$ has a number of vertices bounded from above by a function of r and k.

Lemma 65. $|H| \leq |\mathcal{H}^*|$.

Proof. The hypergraph $\mathcal{H}^*$ is constructed from $\mathcal{H}'$ by the addition of vertices therefore $|\mathcal{H}'| \leq |\mathcal{H}^*|$. The hypergraph $\mathcal{H}'$ is a partial hypergraph of $\mathcal{H}$ consequently $V(\mathcal{H}') = V(\mathcal{H})$. The hypergraph $\mathcal{H}$ is constructed from a graph $H \in \mathrm{Forb}(\mathcal{C}+k\mathsf{v})$ such that $V(H) = V(\mathcal{H})$ therefore $|V(\mathcal{H})| = |V(H)|$. From transitivity we obtain that $|H| \leq |\mathcal{H}^*|$. □

From Lemma 65 we obtain that if the order of $\mathcal{H}^*$ is bounded then so is the order of H. Recall that $H \in \mathrm{Forb}(\mathcal{C}+k\mathsf{v})$. As $\mathcal{H}^*$ is an r-uniform $(k+1)$-critical hypergraph from Lemma 63 we get that the order of $\mathcal{H}^*$ is bounded from above by a function of r and k. This leads to the following theorem.

Theorem 66. *For every class $\mathcal{C}$ which is characterised by a finite forbidden set with respect to a partial order ($\leqslant$) that has the bounded expansion property and every $k \geq 0$ the class $\mathcal{C}+k\mathsf{v}$ is closed with respect to $\leqslant$ then $\mathcal{C}+k\mathsf{v}$ has a finite minimal forbidden set.*

The consequence of Theorem 66 is clear, for any class of graphs closed with respect to a partial order which has the bounded expansion property and can be characterised by a finite set of minimal forbidden graphs then the parameterized class $\mathcal{C}+k\mathsf{v}$ can also be characterised by a finite minimal forbidden set if $\mathcal{C}+k\mathsf{v}$ is closed with respect to the partial order under consideration. This result is presented in the most general way in order to demonstrate the wide ranging applications in graph theory. The result far passes that of Fellows *et al.* in [52] by showing that the minimal forbidden set is finite independent of the well-quasi ordering condition. This has many theoretical and practical implications which can be exploited to develop algorithms for practical problems.

5.1.3 Worked example for the induced subgraph relation

We present a completed worked example for the result of characterising $\mathcal{C}+k\mathsf{v}$ with respect to a specific partial order. The induced subgraph relation is used for this example as it is a commonly used partial order and many studied graph classes admit a characterisation with respect to it. Firstly it is necessary to demonstrate that the induced subgraph relation has the required properties, recall that for a partial order to have the bounded expansion property it must satisfy the following conditions:

1. $\leqslant$ is well-founded
2. $\leqslant$ is order descending
3. $\forall G \leqslant H$ and $\forall U \subseteq V(H)$ where U is minimal with the property that $G \leqslant H[U]$ then $|U| \leq f(G)$ for some function $f : \mathcal{G} \to \mathbb{Z}^+$.

Lemma 67. *The induced subgraph relation has the bounded expansion property.*

Figure 5.2: Forbidden graphs for the class $\mathcal{C}$.

Proof. It is easily observed that the induced subgraph relation is order descending and is therefore also well-founded. It remains to show that for all $G \leqslant_{\mathrm{i}} H$ and for all $U \subseteq V(H)$ where U is minimal with the property that $G \leqslant_{\mathrm{i}} H[U]$ then $|U| \leq g(G)$ for some function g. Observe that G is an induced subgraph of H if and only if there exists an injective function between $V(G)$ and $V(H)$ that preserves adjacency. Let $U \subseteq V(H)$ such that U is minimal with the property that $G \leqslant_{\mathrm{i}} H[U]$ and $|U| > |V(G)|$. As there exists an injective function $g : V(G) \to U$ there must exist a vertex $u \in U$ such that u is not a member of the image of g. Therefore $G \leqslant_{\mathrm{i}} H[U \setminus \{u\}]$, contradicting that U is minimal. □

Corollary 68. $\forall G \leqslant_{\mathrm{i}} H$ and $\forall U \subseteq V(H)$ where U is minimal with the property that $G \leqslant H[U]$ then $|U| = |V(G)|$.

Example

Consider the class of $\{K_3, C_4\}$-free graphs. The class is not well-quasi ordered with respect to the induced subgraph relation, the class contains the set $\{C_n \mid n \geq 5\}$ which is an antichain with respect to the induced subgraph relation, and therefore the results of Fellows *et al.* [52] cannot be applied. Let $\mathcal{C}$ be the class of $\{K_3, C_4\}$-free graphs. Clearly $\mathrm{Forb}(\mathcal{C}) = \{K_3, C_4\}$ (see Figure 5.2). As the graphs in $\mathrm{Forb}(\mathcal{C})$ are not of a uniform order we construct the hypergraph $\mathcal{H}$ such that

1. $V(\mathcal{H}) = V(H)$
2. $E(\mathcal{H}) = \{e \mid e \subseteq X \wedge \exists F \in \mathrm{Forb}(\mathcal{C})\ H\langle e\rangle \simeq F\}$.

For a concrete example consider the graph $S_3 \notin \mathcal{C}{+}1\mathsf{v}$ (see Figure 5.3). The hypergraph constructed for the graph S_3 is

$$\begin{aligned} V(\mathcal{H}) =& \{a, b, c, d, e, f\} \\ E(\mathcal{H}) =& \{\{a, b, c\}, \{b, d, e\}, \{b, d, e\}, \{c, e, f\}.\} \end{aligned}$$

Notice that S_3 contains no induced cycles of length 4. It is easily observed that $S_3 \notin \mathcal{C}{+}1\mathsf{v}$. Observe that $\tau(\mathcal{H}) = 2$, e.g., $\{e, c\}$ is a transversal of size 2. Also observe that $\mathcal{H}$ is not τ-critical as $(V, E \setminus \{b, c, e\})$ has transversal number 2, i.e., $\tau(\mathcal{H}) = \tau((V, E \setminus \{b, c, e\}))$. However

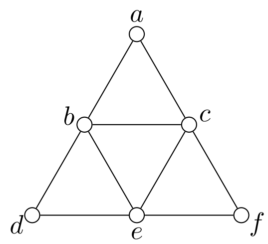


Figure 5.3: $S_3 \notin \mathcal{C}+1\mathsf{v}$

from Lemma 58 we can find a τ-critical partial hypergraph in $\mathcal{H}$. Observe that the following hypergraph $\mathcal{H}'$ is τ-critical

$$\begin{aligned} V =& \{a, b, c, d, e, f\} \\ E =& \{\{a, b, c\}, \{b, d, e, \}, \{c, e, f\}\} \end{aligned}$$

shown in Figure 5.4.

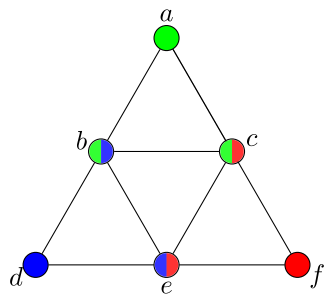


Figure 5.4: An illustration of $\mathcal{H}'$, the colours indicate the edges each vertex belongs to; green=$\{a, b, c\}$, blue=$\{b, d, e\}$ and red=$\{c, e, f\}$.

From the construction of $\mathcal{H}'$ we can apply Lemma 63 to obtain that the maximum order of a minimal forbidden induced subgraph for the class $\mathcal{C}+k\mathsf{v}$ is;

$$\binom{k+3}{2}(k+1) + (k+1)^3 \tag{5.4}$$

5.1.4 $\mathcal{C}+k\mathsf{e}$

Recall the definition of the class $\mathcal{C}+k\mathsf{e}$: the class $\mathcal{C}+k\mathsf{e}$ is the class of graphs where there exists a set of at most k edges such that the removal of the edges yields a graph belonging to the class $\mathcal{C}$. The definitions and results of the previous sections can be used to show that the minimal forbidden set for the class $\mathcal{C}+k\mathsf{e}$ is finite when the class $\mathcal{C}$ has a finite minimal forbidden set and the class $\mathcal{C}+k\mathsf{e}$ is closed with respect to the partial order under consideration. To prove that the minimal forbidden set for the class $\mathcal{C}+k\mathsf{e}$ is finite we formulate the problem as a hypergraph problem, similar to the formulation for $\mathcal{C}+k\mathsf{v}$. However, the construction of the hypergraph differs from that used in proving a bound for the graphs in Forb($\mathcal{C}+k\mathsf{v}$).

In constructing the hypegraph, in the process of bounding the maximum order of a graph in Forb($\mathcal{C}+k\mathsf{e}$), the vertices of the hypergraph are the edge of the graph in Forb($\mathcal{C}$) and the edges of the hypergraph consist of sets of edges that induces a graph in the set Forb($\mathcal{C}$). As Forb($\mathcal{C}$) is finite there is a maximum size of one of the graphs. Let r denote the maximum size of a member of Forb($\mathcal{C}$). The hypergraph constructed may not be $k+1$ critical but the graph contains a $(k+1)$-critical hypergraph as a partial hypergraph by Lemma 58.

As for the construction of the hypergraph for the case of $\mathcal{C}+k\mathsf{v}$ the edges of the hypergraph are then inflated to be of a uniform size. The result is an $(k+1)$-critical r-uniform hypergraph, whose maximum order can be bounded from above by Lemma 65.

If we denote the largest uniform τ-critical hypergraph by N then the maximum order of a minimal forbidden graph in Forb($\mathcal{D}+k\mathsf{e}$) is bounded to $2N$. The restrictions where $\mathcal{C}+k\mathsf{e}$ is finite are similar to those for the class $\mathcal{C}+k\mathsf{v}$. The partial order under consideration must have the bounded expansion property.

5.2 Summary

In this chapter we have proved that the minimal forbidden sets for the classes $\mathcal{C}+k\mathsf{v}$ and $\mathcal{C}+k\mathsf{e}$ are finite under some weak conditions. This is shown by demonstrating that the minimal forbidden graphs for those classes can be represented as τ-critical hypergraphs. We provide a construction of a τ-critical hypergraph for the classes $\mathcal{C}+k\mathsf{v}$ and $\mathcal{C}+k\mathsf{e}$ which allows us to apply a result to bound the maximum number of vertices in such a hypergraph. This bound translates into bounding the number of vertices in the minimal forbidden graphs. As this bound is a finite bound the technique results in a finite minimal forbidden set for the considered classes.

Although when the techniques of this chapter are viewed in parallel with the graph minor theorem they may seem weak, covering only a small number of cases, they provide a useful technique for proving that a parameterized graph class has a finite minimal forbidden set. In practice, many of the graph classes that we consider here are not closed with respect to the minor relation which prevents us using the machinery of the graph minor theorem to obtain results. The techniques developed in this chapter have a number of applications, some of which are explored in Chapter 6. One of these applications is the ability to enumerate the minimal forbidden set for a particular graph class. A potential impact of this application is that by observing the structure of the minimal forbidden graphs it may be possible to construct efficient algorithms for that class, which could have impact on areas of science such as computational biology, data cleaning and theoretical computer science where the classes $\mathcal{C}+k\mathsf{v}$ and $\mathcal{C}+k\mathsf{e}$ occur naturally.

Chapter 6

Applications

The results of the previous chapters have applications spanning numerous fields outside computer science. Any problem that can be formulated into recognising a parameterized graph class can benefit. Two interesting problems from a theoretical viewpoint are that of enumerating the minimal forbidden set and providing certifying algorithms for fixed-parameter tractable problems. In this chapter we outline two algorithms that answer both of these questions in a general setting and provide a concrete example to demonstrate the improvements that can be achieved when considering specific graph classes.

6.1 Enumerating the minimal forbidden sets

The results of the previous chapters have applications in enumerating the minimal forbidden set. The task of enumerating the minimal forbidden set is that of generating the set of minimal forbidden graphs for a specific class. This set is unique up to equivalence. This is an interesting application on many accounts. From a theoretical viewpoint the process of enumerating the minimal forbidden set highlights an interesting distinction between the partial orders we consider here and some of those that are considered in the literature, such as the minor relation (see Chapter 3). For the minor relation it has been shown that there exists a finite minimal forbidden set for any minor closed property; however, the task of computing the minimal forbidden set is hard. For some graph classes such as $\mathcal{C}+k\mathsf{v}$ it has been shown that the the minimal forbidden set is computable, however there is no explicit bound on the order of a minimal forbidden graph nor is there any explicit bound on the time complexity for computing such a set [1]. It is therefore interesting that for some partial orders we have a bound on the maximum order of a minimal forbidden graph and this provides a method for enumerating the minimal forbidden set, providing that the containment problem for the partial order under consideration is decidable. The second account for justifying the interest in the problem of enumerating the minimal forbidden set is that a better understanding of the graph class being characterised can be gained by observing structural properties of the minimal forbidden graphs.

We provide a generic algorithm for enumerating the minimal forbidden graphs for a graph class that is covered by the results of the previous chapters; *i.e.*, the class is closed with respect to a partial order with the bounded expansion property and $\mathcal{C}$ has a finite minimal forbidden set. This algorithm allows the enumeration of the minimal forbidden graphs for the graph classes $\mathcal{C}{+}k\mathsf{v}$ and $\mathcal{C}{+}k\mathsf{e}$ closed with respect to a partial order whose containment problem is decidable. For some partial orders and for some graph classes it is possible to avoid using the generic algorithm by applying class specific algorithms which yield better time complexity. There exist graph classes where it is possible to explicitly construct the minimal forbidden set. Here we present the generic algorithm for enumerating the minimal forbidden set for the class $\mathcal{C}{+}k\mathsf{v}$, however the techniques differ little for the enumeration of the minimal forbidden set for the class $\mathcal{C}{+}k\mathsf{e}$. A discussion is provided for the practical and technical considerations an implementer should be aware of.

For the generic algorithm for enumerating the minimal forbidden set a naive approach is used. The algorithm computes the set of graphs $\mathrm{Forb}(\mathcal{C}{+}k\mathsf{v}) \subseteq \mathcal{J}$ such that $\mathcal{J} \subseteq \mathcal{C}{+}(k+1)\mathsf{v}$ and the maximum order of a graph in $\mathcal{J}$ is bounded from above by n, the value of n is computed using the results of Chapter 5. Because of minimality, the minimal forbidden graphs for the class $\mathcal{C}{+}k\mathsf{v}$ will be members of $\mathcal{C}{+}(k+1)\mathsf{v}$. In order to recognise the class $\mathcal{C}{+}k\mathsf{v}$ it is essential that the class $\mathcal{C}$ can be recognised. For any class $\mathcal{C}$ closed with respect to a partial order $\leqslant$ where $\mathrm{Forb}(\mathcal{C})$ is finite and $\leqslant$ is decidable then the class $\mathcal{C}$ can be recognised. The algorithm should iterate over the minimal forbidden graphs checking if the input graph contains any of the minimal forbidden graphs; if the input graph is free from all minimal forbidden graphs then the algorithm should return an affirmative output otherwise the algorithm should return a negative output. Algorithm 5 defines a procedure named `recogniseClass` that implements the previously outlined algorithm. This algorithm is used as a sub-procedure in the subsequent algorithms.

Algorithm 5: Generic algorithm to recognise a graph class, closed with respect to a partial order, that has a finite minimal forbidden set.

Input: A graph $G = (V, E)$
Output: True if $G \in \mathcal{C}$, False otherwise.
Data: The minimal forbidden set $\mathcal{F}$ of $\mathcal{C}$ with respect to $\leqslant$.

```
procedure recogniseClass(G)
    for H ∈ F do
        if H ⩽ G then
            return False
        end
    end
    return True
end
```

Lemma 69. *Algorithm 5 recognises the class $\mathcal{C}$.*

Proof. The correctness of the algorithm follows trivially from Theorem 5. □

The decidability of class membership is determined by two factors: first the decidability of the containment problem for the partial order under consideration, and second the cardinality of the minimal forbidden set. As the cardinality of the minimal forbidden set is presumed to be finite and from the restriction that the partial order containment problem is decidable then we infer that the class membership problem is decidable. To be specific, the complexity of Algorithm 5 is dependent on the complexity of the containment problem. If the containment problem is polynomial (for each pattern graph) or fixed-parameter tractable (where the parameter is the order of pattern graph) then the algorithm is a polynomial time algorithm for each fixed class $\mathcal{C}$. The term *pattern* graph refers to the first element of a pair in a partially ordered set, *e.g.* if $G \leqslant H$ $((G, H) \in \leqslant)$ then G is referred to as the pattern graph.

To recognise the class $\mathcal{C}+k\mathsf{v}$ we generalise the algorithm proposed by Cai [19] for class membership of hereditary closed parameterized graph classes to recognise any parameterized graph class closed with respect to a partial order with the bounded expansion property. The following algorithm (Algorithm 6) finds the preimage of a minimal forbidden graph of the base class in the input graph if one exists. In the case where a preimage is found the algorithm returns a set of vertices else an empty set is returned. Algorithm 6 uses Algorithm 5 as a subprocedure, let $T(n)$ denote the time complexity of Algorithm 5. For each vertex in the input graph Algorithm 5 is called, therefore the overall time complexity of Algorithm 6 is $O(n \cdot T(n))$.

Lemma 70. *Algorithm 6 finds a bounded size preimage of a minimal forbidden graph for $\mathcal{C}$.*

Proof. We claim that the set F returned by `findMinimalForbidden`, if nonempty, is a preimage of a minimal forbidden graph for $\mathcal{C}$ in the input graph G. Let us consider the instance of time before the algorithm removed the last vertex v. Let G' be the graph before v is removed and $G'' = G' - v$. As v was removed from G' then it must be the case that $G'' \notin \mathcal{C}$. We show that G'' is of bounded size and is a preimage of a minimal forbidden graph for the class $\mathcal{C}$. Suppose that G'' is not a minimal preimage of a minimal forbidden graph for the class $\mathcal{C}$ then there must exist a vertex $u \in V(G'')$ such that $G'' - u \notin \mathcal{C}$. Let G''' be the graph when the vertex u was considered, $G''' - u \notin \mathcal{C}$ as $V(G') \subseteq V(G''')$ and $G' \notin \mathcal{C}$ therefore u should have been deleted contradicting the existence of $u \in V(G'')$. As F is a minimal preimage of a minimal forbidden graph and $\leqslant$ is a partial order that has the bounded expansion property, it follows that $|F|$ is bounded by some function of the maximum order of a graph in $\text{Forb}(\mathcal{C})$. □

Given that a set of vertices can be found that is a preimage of a minimal forbidden graph then the set of vertices can be used to recognise the class $\mathcal{C}+k\mathsf{v}$. Algorithm 7 uses the vertices of the preimage and recursively constructs a search tree in order to identify a set $U \subseteq V(G)$ of k vertices such that $G - U \in \mathcal{C}$. On line 4 a minimal forbidden graph is found in the input graph, the input graph is then modified by deleting each vertex in the minimal forbidden graph

Algorithm 6: Algorithm to find a minimal forbidden graph of $\mathcal{C}$ contained in G.

Input: A graph $G = (V, E)$

Output: $F \subseteq V(G)$ such that $G[F]$ is minimal with respect to the partial order under consideration and there exists a graph $H \in \text{Forb}(\mathcal{C})$ such that $H \leqslant G[F]$ or $\varnothing$ if $G \in \mathcal{C}$.

Data: A procedure `recogniseClass` that recognises the class $\mathcal{C}$.

```
procedure findMinimalForbidden(G)
    if recogniseClass(G) then
        return ∅
    end
    F = ∅
    V = V(G)
    while V ≠ ∅ do
        choose a vertex v ∈ V
        V = V \ {v}
        if recogniseClass(G − v) then
            F = F ∪ {v}
        else
            G = G − v
        end
    end
    return F
end
```

and recursively calling Algorithm 7 on a smaller instance of the same problem. Note that on all paths through the pseudocode the algorithm will return a value. If a minimal forbidden graph cannot be found then the algorithm returns True. The algorithm has two cases which distinguish between handling the base class of the parameterized graph class and the inductive step. The base class is handled by line 14. Between lines 8–11 the algorithm modifies the input graph and recursively calls the algorithm. If one of the possible modifications leads to $G - v \in \mathcal{C}{+}(k-1)\mathsf{v}$ then B obtains the value True and is returned on line 12.

The time complexity of Algorithm 7 is dependent on the time complexity of Algorithm 6 (which in turn depends on Algorithm 5). Let the running time of Algorithm 6 be denoted by $T'(n)$. At each level of the recursion there are a maximum number of c vertices to remove and the recursion has bounded depth, at most depth k. Each step of the recursion makes a call to Algorithm 6 therefore the overall time complexity is $O\left(c^k \cdot T'(n)\right)$.

Now that an algorithm has been defined to recognise the class $\mathcal{C}{+}k\mathsf{v}$ closed with respect to a partial order with the bounded expansion property where $\mathcal{C}$ has a finite minimal forbidden set then we can construct an algorithm to enumerate the minimal forbidden set for the class $\mathcal{C}{+}k\mathsf{v}$. From the definition of the minimal forbidden set for the class $\mathcal{C}{+}k\mathsf{v}$, the elements of this set will be members of the graph class $\mathcal{C}{+}(k+1)\mathsf{v}$. Algorithm 8 first constructs a set of graphs F from the set of all graphs up to a given order then computes the minimal elements.

Algorithm 7: Algorithm to test membership of $\mathcal{C}$+kv.

```
Input: A graph G = (V, E), an integer k ≥ 0, the remaining number of vertices to be
       removed.
Output: True if G ∈ C+kv, False otherwise.
Data: A procedure recogniseClass that recognises the class C.
procedure recogniseCkv(G,k)
    B := False
    if k > 0 then
        F := findMinimalForbidden(G)
        if F = ∅ then
            B := True
        else
            for v ∈ F do
                B := B ∨ recogniseCkv(G − v,k − 1)
            end
        end
        return B
    else
        return recogniseClass(G)
    end
end
```

The bound on the maximum order of a minimal forbidden graph was establish in Chapter 5.

Lemma 71. *Algorithm 8 is correct.*

Proof. The minimal forbidden set for the class $\mathcal{C}$+kv is a subset of $\mathcal{C}$+$(k+1)$v by definition. The bound on the maximum order of a minimal forbidden graph was established by Lemma 63 in Chapter 5. Therefore we may compute all graphs in $\mathcal{C}$+$(k+1)$v up to the maximum order and then compute the minimal elements of this set. □

Technical considerations

There are a number of improvements that can be made on the generic algorithm presented previously at the cost of compromising the generality. The algorithm for enumerating the minimal forbidden set is a fixed-parameter algorithm running in $f(k)$ and also depends on the graph class. This provides two avenues for improving the time complexity of the algorithm. It is noteworthy that for each graph class $\mathcal{C}$+kv, generating the minimal forbidden set takes $f(k) \cdot O(1)$ time. The first potential avenues for improving the complexity is to improve the function $f(k)$. Although this practically is likely to produce a notable improvement, as parts of the function contain exponential components, by improving the algorithm that recognises the base class a more noticeable improvement is achieved. To improve the complexity it is possible to exchange the generic class recognition algorithm, used in Algorithm 5, for a class specific one. Consider the class of split graphs, using the generic class recognition algorithm

Algorithm 8: Algorithm to enumerate the minimal forbidden set of the class $\mathcal{C}$+kv with respect to the partial order $\leqslant$.

Input: An integer $k \geq 1$
Output: A set F of minimal forbidden graphs for the class $\mathcal{C}$+kv.
Data: The set $\mathcal{G}_n$ of graphs of order less than or equal to n where n is an upper bound on the maximum order of a minimal forbidden graph.

```
procedure generateForbiddenSet()
    Compute n using Lemma 63 in Chapter 5
    F = ∅
    for G ∈ G_n do
        if ¬recogniseCkv(G,k) ∧ recogniseCkv(G,k + 1) then
            F = F ∪ {G}
        end
    end
    X = ∅
    for f ∈ F do
        m = True
        for f′ ∈ F \ {f} do
            if f′ ⩽ f then
                m = False
            end
        end
        if m then
            X = X ∪ f
        end
    end
    return X
end
```

Class	Class recognition algorithm Generic	Specific
Split graphs	$O\left(n^5\right)$	$O(n)$ [82]
Cographs	$O\left(n^4\right)$	$O(n+m)$ [28]
Threshold graphs	$O\left(n^4\right)$	$O(n+m)$ [81]
Triangle-free graphs ($\leqslant_\mathrm{i}$)	$O\left(n^3\right)$	$O\left(n^{2.3727}\right)$ [163]
Trivially perfect graphs	$O\left(n^4\right)$	$O(n+m)$ [167]
P_3-free graphs ($\leqslant_\mathrm{i}$)	$O\left(n^3\right)$	$O(n+m)$

Table 6.1: Summary of the complexity of some graph class recognition algorithms which can be used in place of the generic recognition algorithm used in Algorithm 7.

(Algorithm 5) results in an $O\left(n^5\right)$ time algorithm whereas the use of a specific split graph recognition algorithm results in an $O(n)$ algorithm. A brief summary of other graph classes that would potentially benefit from the use of a class specific recognition algorithm is provided in Table 6.1. In practice the set of graphs $\mathcal{G}_n$ should be the set of non-isomorphic graphs on at most n vertices. The set of non-isomorphic graphs can be generated using *nauty* [117].

In practice, improvements to the function $f(k)$ yield noticeable improvements to the running time. One of the components of the function is the number of graphs that have to be checked for minimality. By reducing the number of graphs to be checked, improvements on the running time are achieved. Algorithm 8 takes the set $\mathcal{G}_n$ of all graphs of order less than or equal to n. The size of this set is of the order $2^{\binom{n}{2}}$. Restricting this set can provide improvements in running time, however the restriction must not alter the correctness of the algorithm. Two restrictions which have been successful in practice are that of providing a lower bound on the order of a graph and also restricting the set of graphs to only connected graphs (the second restriction is only valid if the minimal forbidden graphs for the class $\mathcal{C}$ are connected). Both restrictions have their limitations; for the first method only crude lower bounds have been proven which limit its effectiveness. The restriction to connected graphs applies for partial subgraphs and induced subgraphs and stems from the observation that if the underlying τ-critical hypergraph is disconnected then each of the components of the hypergraph induces a minimal forbidden graph for a smaller value of k.

Lemma 72. *For a graph class $\mathcal{C}$ characterised with respect to a finite set of connected minimal forbidden graphs with respect to the partial subgraph (induced subgraph) relation if $G \in \mathrm{Forb}(\mathcal{C}{+}k\mathsf{v})$ then G is the disjoint union of $G_1, \dots, G_l$ such that for all $i = 1, \dots, l$ there exists a k_i such that $G_i \in \mathrm{Forb}(\mathcal{C}{+}k_i\mathsf{v})$ and $\Sigma_{i=1}^{l}(k_i + 1) = k + 1$.*

Proof. The theorem follows by a proof by induction on the number of components of G. Let $G \in \mathrm{Forb}(\mathcal{C}{+}k\mathsf{v})$ and let $G_1, \dots G_l$ be the connected components of G. The base of the induction is for $l = 1$; the graph G is connected and therefore $\Sigma_{i=1}^{1}(k_i + 1) = k + 1$. Let G' and G'' be graphs induced by a partition of the connected components of G into two nonempty parts and let l' and l'' denote the number of connected components in G' and G'' respectively. For some k'

and k'' we have $G' \in \text{Forb}(\mathcal{C}{+}k'\mathsf{v})$ and $G'' \in \text{Forb}(\mathcal{C}{+}k''\mathsf{v})$. As $G = G' \cup G''$ we have $l = l' + l''$ and $k = k' + k'' + 1$. By the induction hypothesis we have

$$\sum_{i=1}^{l'}(k_i+1) - 1 + \sum_{i=l'+1}^{l}(k_i+1) - 1 + 1 = k$$

which leads to

$$\sum_{i=1}^{l}(k_i+1) = k+1 \qquad \square$$

The consequence of Lemma 72 is that the disconnected minimal forbidden graphs can be explicitly computed for a class $\mathcal{C}{+}k\mathsf{v}$ given the set of minimal forbidden graphs for all classes $\mathcal{C}{+}l\mathsf{v}$ where $l < k$. The computation should compute all integer partitions of k and then for each partition compute the disjoint union of the graphs, one from each forbidden set indexed by the partition. Standard software engineering approaches could improve the running time of Algorithm 8. An approach which could be applied to Algorithm 8 is that of adapting the algorithm for parallel computation. As there is no computational dependency between elements of $\mathcal{G}_n$ the application of a programming framework, such as MapReduce [37], to distribute this task could easily be implemented.

Example

We provide concrete examples for the generation of the minimal forbidden set for two well-studied classes of graphs, namely $\{K_2\}$-free$_\mathrm{i}$ and $\{K_3\}$-free$_\mathrm{i}$. The classes are closed with respect to the induced subgraph relation and have finite minimal forbidden sets therefore there exists an algorithm to recognise the class of graphs. These two examples are provided as they illustrate where applying the techniques discussed in the technical considerations sections make significant practical improvements on the running time.

The class $\{K_3\}$-free$_\mathrm{i}$ is a well studied class. The time complexity of many of the interesting graph theoretical problems on the class has been established. The recognition problem is polynomial as a consequence of the class having a finite minimal forbidden set. The naive approach to recognising the class is to test each 3-tuple for a K_3 adjacency configuration; however, as discussed in the technical considerations section it is possible to improve on the naive approach by using a class specific recognition algorithm. By computing the adjacency matrix of the input graph we obtain a $(1,0)$-adjacency matrix with zero diagonal entries as the input graph is a simple undirected graph. Let A denote the adjacency matrix of the input graph. The matrix A^n has the property of counting the number of walks of length n between any two vertices of the graph (A^n denotes the matrix product of n copies of A). From this property we get that if the trace of A^3 is non-zero then the input graph contains a K_3 as an induced subgraph. Thus the problem of recognising K_3-free$_\mathrm{i}$ graphs is asymptotically bound to the complexity of matrix multiplication. The asymptotic complexity of matrix multiplication has been shown to be better than $O(n^3)$. Strassen improved on $O(n^3)$ to achieve $O\left(n^{\log_2 7}\right) \approx O\left(n^{2.8074}\right)$ [147]. This was later improved up on by many research most notably Coppersmith and Winograd [25]. Coppersmith and Winograd achieves $O\left(n^{2.375477}\right)$. Although the Coppersmith-Winograd algorithm improved the asymptotic complexity of matrix multiplication, the advantage is only observed when the matrices are substantially large, larger than what is feasible to be computed with modern computing. Since Coppersmith-Winograd Williams has improved on the complexity of matrix multiplication to $O\left(n^{2.3727}\right)$. Figure 6.1 shows the minimal forbidden set for the class of K_3-free$_\mathsf{i}$+1v computed by an implementation of Algorithm 8 using [165].

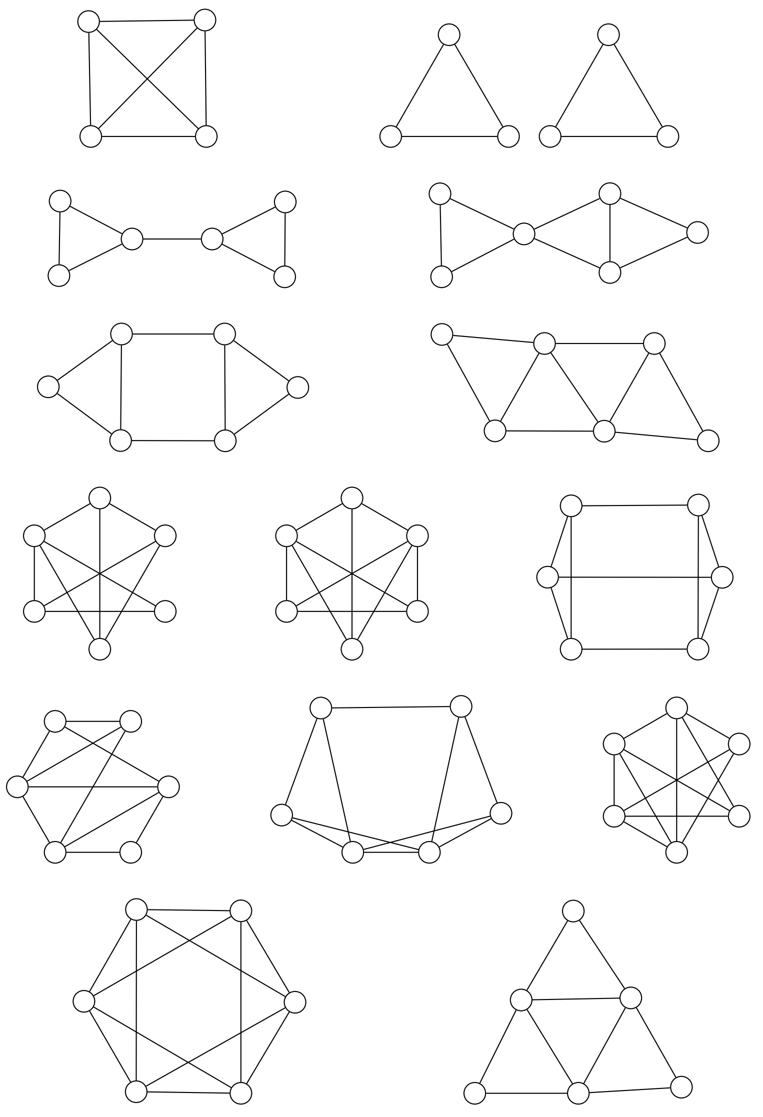


Figure 6.1: Minimal forbidden graphs for the class $\mathcal{C}$+1v, as computed by Algorithm 8.

The second example is for the class $\{K_2\}$-free$_\mathsf{i}$+$k\mathsf{v}$. The class is very restrictive and con-

k	$\in$ Forb($\{K_2\}$-free$_\mathsf{i}$+kv)
0	K_2
1	$2K_2$, K_3
2	$3K_2$, $K_3 \uplus K_2$,K_4
3	$4K_2$, $2K_3$, $2K_2 \uplus K_3$, $K_2 \uplus K_4$, K_5

Table 6.2: A subset of the minimal forbidden graphs for the class $\{K_2\}$-free$_\mathsf{i}$+kv constructed from the constraint in Equation 6.1.

sequently many of the interesting graph theoretical problems are solvable in linear time. The motivation for this example is that the minimal forbidden set can be explicitly computed rather than using the search method outlined in Algorithm 8. The class $\{K_2\}$-free$_\mathsf{i}$+kv contains those graphs where these exists a set of at most k vertices whose removal will result in a graph without edges. The class is equivalent to the graphs with vertex cover at most k. The upper bound for the maximum order of a minimal forbidden graph can be improved from the bound provided in Chapter 5. The maximum order of a minimal forbidden graph for the class $\{K_2\}$-free$_\mathsf{i}$+kv is $2(k+1)$. This upper bound is realised for each class by the graph $(k+1)K_2$. Further to the improved upper bound the graphs in the minimal forbidden set have a definable structure. For the class $\{K_2\}$-free$_\mathsf{i}$+kv each graph in the minimal forbidden set can be partitioned into a set of cliques $c_0, \ldots, c_l$ such that;

$$\sum_{i=0}^{l}(|c_i| - 1) = k + 1. \tag{6.1}$$

Table 6.2 shows the minimal forbidden graphs constructed from the constraint in Equation 6.1. The complete minimal forbidden set can be computed from the graphs in Table 6.2 by adding edges between non-adjacent vertices while still conserving minimality. For the class $\{K_2\}$-free$_\mathsf{s}$+kv Table 6.2 provides a complete listing of the minimal forbidden set for $k < 4$.

6.2 Certifying fixed-parameter algorithms

There are a number of motivating factors which support the development of certifying algorithms. The sentiment held by many researchers in the field is that;

> "a program should justify (prove) its answer in a way that is easily checked by the user of the program." [116, p. 20]

This sentiment leads to the hypothesis that certifying algorithms are superior to conventional algorithms and provide an added value to practical applications. Mehlhorn *et al.* outline a number of advantages of certifying algorithms in [116].

- Instance Correctness: If the checker accepts the tuple then correctness of the algorithm is guaranteed for that instance however this does not guarantee that for all inputs the algorithm is correct.

– Testing on all inputs: Traditionally testing is carried out on a number of known test cases (as is the case with test driven development), it is assumed that if the algorithm passes all test cases then the algorithm is correct however this is a misguided effort as testing in this way only demonstrates the existence of errors and not their absence. For certifying algorithms, the algorithm is tested on all inputs as the certificate and the checker is the proof the output is correct.

– Trust with minimal intellectual investment: The end user only needs to trust the checker and understand why the witness proves that the output is correct, it is not necessary to understand the technicalities of the algorithm.

– Remote computation: The algorithm can be executed on a remote server, maybe with specialist hardware or libraries, then the output can be checked locally.

– Black-box programs: The details of the algorithm may be restricted intellectual property, for a software provider it is not necessary to provide source code for the algorithm itself as the end user only has to accept the correctness of the checker and that the witness proves that the output is correct.

– Integrity insurance: The end user can ensure that the integrity of the computation is complete and that the tuple has not been corrupted, either intentionally or accidentally. The algorithm can be kept private and by sharing the code to the checker the user can see be assured that the output is valid.

– Knowledge advancement: The development of certifying algorithms has provided insight into many problems, the ingenuity required in developing appropriate certificates often leads to a deeper understanding of the problem structure.

The history of certifying algorithms is rich, stretching back to the 8th century. The work of Mehlhorn and Näher has brought certifying algorithms to the attention of the software development community with their work on the LEDA project [118, 119]. The efforts of Mehlhorn and Näher have seen many of the algorithms in the LEDA system made certifying. The motivation for the LEDA project to implement certifying algorithms came from the discovery that a planarity checker implemented in LEDA produced erroneous results on some inputs. In the early 1990's a linear time planarity checker, devised by Hopcroft and Tarjan [87], was added to the LEDA library and in 1993 the implementers were made aware of a counterexample to the correctness of the implementation. This led the implementer to take the approach that certifying algorithms are superior and that it is insufficient for an algorithm to return a single bit without providing a certificate. The certificate in this instance is an embedding in the affirmative case and a topological minor isomorphic to K_5 or $K_{3,3}$ (shown in Figure 6.2) in the case of the graph being non-planar. This is a valid certificate of non-planarity due to Kuratowski's theorem [152]. There also exists a linear time algorithm for finding a K_5 or $K_{3,3}$ [164]. As a result of these two results a strongly certifying algorithm could be implemented with only a constant factor increase in the running time.

Figure 6.2: Minimal forbidden planar graphs: $K_5, K_{3,3}$

The benefits of certifying algorithms rest on the existence of a suitable certificate that justifies the output of the algorithm and that the certificate is "simple" to check. The checking algorithm is an algorithm that checks the validity of the certificate. As described in Section 2.6 the checking algorithm should reject the output of the algorithm if it cannot validate that the certificate justifies the output given the input or accept the output otherwise. The formal definition of certifying algorithms provides no guide to developing the certificate nor restriction on the checking algorithm but clearly sensible restrictions should be imposed in order to reap the intended benefit from certifying algorithms. This leaves two questions which have no definitive answer but intuition provides some insights: (1) What constitutes a good certificate? (2) What restrictions can be placed on the checking algorithms in order to achieve the benefits of certifying algorithms?

Since the checker must validate that the certificate justifies the output of the algorithm then the certificate must justify the correctness of the output. A consideration for a certificate is the complexity of the checking algorithm to validate the certificate. A certificate is called a *sublinear* certificate if the time required to validate the certificate is less than linear time. In [103] the authors define the concept of a *strong certificate* and a *weak certificate* which is distinct from the concept of strongly and weakly certifying algorithms. A certificate is *strong* if there is a checking algorithm that correctly verifies the validity of the certificate in better time than the current fastest algorithm that solves the problem without the addition information provided by the certificate. A certificate is weak if the checking algorithm takes the same amount of time as the best known algorithm. This, in essence, seems to be an acceptable interpretation of attempting to limit the complexity of the checking algorithm. By limiting the running time of the algorithm it prevents the checking algorithm from computing the solution anew and therefore the checking algorithm must rely on a different insight.

Nothing should be drawn from the naming of strong and weak certificates regarding their usefulness. A weak certificate may have benefits over a strong certificate as the implementation of the checking algorithm for the weak certificate may be simpler or even verifiable or the certificate may be conceptually simple to understand. As the practical applications of certifying algorithms are motivated by a non-expert end user it is highly desirable that the concept that substantiates the correctness of the certificate is intellectually within reach. The worth of a certificate should not entirely be measured by its computational complexity to verify. For instance; consider a checking algorithm which for all inputs that occur in practice has a better running time that the asymptotically optimal checking algorithm. From a purely practical viewpoint one would settle for a checking algorithm that is good in practice.

A "good" certificate should have a checking algorithm that is conceptually simpler than the algorithm for the original problem. However, simplicity is subjective; if the checker was conceptually simpler than the proving algorithm,such that the checker could be formally verified or that the end user could feel confident of its correctness then it may be acceptable for the checking algorithm to have a asymptotically higher complexity than the proving algorithm. Assuming that it is essential for the checking algorithm to be conceptually simpler than the proving algorithm then it could be argued that the conceptual complexity between the two algorithms is relative and that for a very complex algorithm a more complicated checking algorithm would suffice.

The formal definition of certifying algorithms stipulates no restriction on the checking algorithm, it is intentionally defined in a vague manner to allow certifying algorithms to be applied generally. However, to achieve the intended benefits of certifying algorithms there are a number of sensible restriction that may be applied:

– The checker runs in linear time.

– The size of the certificate is bounded.

– There is an elementary proof that the checker is implement correctly.

– There is a simple logical system for which it is trivial to show that the witness predicate holds.

The last two items are of particular interest as they most accurately capture the intuition behind certifying algorithms, however, up to date the field has focused its efforts on providing checking algorithms that run in bounded time. Checking algorithms that run in bounded time do not guarantee the benefits of certifying algorithms are obtained. As the time bound for the checking algorithm becomes tighter it is often the case that the reasoning behind the certificate becomes more involved, which contradicts one of the motivating factors of certifying algorithms: that the end user should trust the validity of the certificate. It is often not sensible to apply hard and fast rules as to what constitutes a "good" certificate but a measured and balanced justification should be provided on a case by case basis.

There is often a disparity in the asymptotic time complexity for the checking algorithm to validate the certificate. For example, sublinear certificates are common (see Table 3.6 on page 62) in the case when the checking algorithm rejects the input, however to validate the accepting certificate requires more time.

Assuming one understands the motivation for certifying algorithms, it is easy to see that it is more imperative that fixed-parameter algorithms are designed to be made certifying. Although fixed-parameter algorithms are a compromise for providing a solutions to solving $\mathbb{NP}$-complete problems the running time of these algorithms can still be considerable. As the running time of the algorithm increases the effective cost of the computation increases which extends the need of the algorithm to output the correct result. For traditional algorithms where output validity is critical, consensus schemes are often used however this approach only reduces the probability of an incorrect output. In applications where the validity of the output is critical and the running time is considerable, certifying fixed-parameter algorithms provide an approach which is matched by few others. Instead of running several expensive algorithms to gain the level of confidence required a single algorithm can provide the "peace of mind" that is required.

Because the running time of fixed-parameter algorithms is often considerable a complete test suite that thoroughly tests the implementation of the algorithm could be wildly impractical to execute. With certifying fixed-parameter algorithms there is no need for a complete test suite as the checker validates the correctness of the algorithm on each instance. The importance of instance correctness is not diminished because the algorithm designer considers a fixed-parameter algorithm for a solution to a problem.

Algorithms for fixed-parameter tractable problems are often complex relying on concepts such as iterative compression, kernelization and bounded search spaces which may be alien to the implementer or end user. It is therefore essential that the end user can trust that the computation is correct despite being unable to understand each step in the computation. Certifying fixed-parameter algorithms offer a solution to this by minimising the intellectual investment required of the end user. The end user is only required to understand that the checker is implemented correctly and that the certificate validates the output.

Fixed-parameter algorithms may require large amounts of computation as the terms in the function $f(k)$ may be substantial. The computation of a fixed-parameter algorithm may be best suited to remote computation where excess computational resources, specialist hardware and expensive proprietary libraries are available. To ensure the integrity of the remote computation a certificate justifying the correctness of the output can be provided to the end user enabling the end user to have confidence that the integrity of the result is untarnished.

Of course the advancement in knowledge and understanding of fixed-parameter algorithms can only benefit from the amalgamation of the fields of fixed-parameter tractability and certifying algorithms. It is an open question as to if all algorithms have a certifying algorithm, likewise the question as to whether all fixed-parameter algorithms have a certifying algorithm is equally open. With the result of Fellows that all fixed-parameter tractable algorithms have "useful" obstruction sets the question is laid before the community as to whether the obstruction set in fixed-parameter tractable problems can be used as a certificate [50]. In the same way the field of fixed-parameter algorithms can benefit from the advancement introduced by the field of certifying algorithms. As certifying algorithms often require an alternative insight into the problem in order to develop a suitable certificate, the insight may prove useful to developing new reduction rules and kernelization techniques.

Therefore, it stands that fixed-parameter algorithms should be made certifying where possible with advantages gained by the end user, the implementer and the wider academic community in the form of a better insight into the problem.

Certifying the recognition of $\mathcal{C}+k\mathsf{v}$

Certifying the recognition of $\mathcal{C}+k\mathsf{v}$ extends the recognition of $\mathcal{C}+k\mathsf{v}$ in [19] and is the first known certifying fixed-parameter algorithm to be published. The motivation for striving for certifying algorithms for fixed-parameter tractable problems has be laid out in previous sections and thus justifies the contribution of the development of a certifying fixed-parameter algorithm. For the problem of recognising a parameterized graph class, certifying algorithms have additional benefits. Given an hereditary graph class $\mathcal{C}$ where the maximum independent set problem is solvable in polynomial time and a graph G belonging to the class $\mathcal{C}+k\mathsf{v}$ then the maximum independent set of G can be found in fixed-parameter tractable time. The parameter for the fixed-parameter tractable algorithm is the graph class parameter k.

Lemma 73. *Let $\mathcal{C}$ be an hereditary graph class such that the maximum independent set problem can be solved in polynomial time then the maximum independent set problem for the class $\mathcal{C}+k\mathsf{v}$ is fixed-parameter tractable.*

Proof. Let $G = (V, E)$ be a graph belonging to $\mathcal{C}+k\mathsf{v}$ and let $U \subseteq V$ such that $G - U \in \mathcal{C}$ and $|U| \leq k$. Let $S \subseteq V$ be a maximum independent set of G and let $S_U = S \cap U$ and $S_V = S \setminus U$. Observe that S_V is a maximum independent set of $G[V \setminus (U \cup N_G(S_U))]$. Suppose this is not true then there must exist a set S' such that S' is a maximum independent set of $G[V \setminus (U \cup N_G(S_U))]$ and $|S'| > |S_V|$ and $S' \cup S_U$ is a maximum independent set of G, contradicting that S was a maximum independent set because $|S_U \cup S_V| < |S_U \cup S'|$. Therefore a maximum independent set of G can be considered to be made up of two parts a set S_V which is an independent set of $G[V \setminus (U \cup N_G(S_U))]$ and a subset of U. As $G-U \in \mathcal{C}$ and $\mathcal{C}$ is hereditary then $G-(U \cup N_G(S_U)) \in \mathcal{C}$, consequently a maximum independent set of $G-(U \cup N_G(S_U))$ can be found in polynomial time. To compute S_U we may compute all subsets of U and check which set obtains a maximum independent set, this can be done in $2^k \cdot k^2$ (independent of the input size). The overall algorithm runs in time $2^k \cdot k^2 \cdot n^{O(1)}$, hence a fixed-parameter algorithm. □

The algorithm that can be implemented from Lemma 73 is given in Algorithm 9. The algorithm is only correct if the set of vertices U is a valid modifier that results in a graph belonging to the class $\mathcal{C}$. This is a prime example of where a certifying fixed-parameter algorithm has a practical application. It is noteworthy that the Lemma 73 outlines a general technique and to consider it as written is unjust. The same technique can be applied to the maximum clique and vertex cover problem. In addition the technique can also be applied to the classes $\mathcal{C}+k\mathsf{e}$ and $\mathcal{C}-k\mathsf{e}$ as both classes are subsets of $\mathcal{C}+k\mathsf{v}$ (see Theorem 46 on page 77).

In the general case of certifying the recognition of the class $\mathcal{C}+k\mathsf{v}$ it is necessary to assume either that there exists a certifying algorithm for the class $\mathcal{C}$ or that recognition of the base

Algorithm 9: Fixed-parameter algorithm for the maximum independent set problem for the class $\mathcal{C}+k\mathsf{v}$ where $\mathcal{C}$ is a graph class closed with respect to the induced subgraph relation and the maximum independent set problem can be solved efficiently for $\mathcal{C}$.

Input: A graph $G = (V, E)$ and a set U such that $G - U \in \mathcal{C}$.
Output: A maximum independent set of G.
Data: A procedure `misC`, returning a maximum independent set of a graph in $\mathcal{C}$.

procedure `MIS`(G, U)
    $S := \varnothing$
    for $S_U \subset U$ *and* S_U *is independent* **do**
        $S_V :=$ `misC`$(G - U - N_G(U))$
        if $|S_U \cup S_V| > |S|$ **then**
            $S := S_U \cup S_V$
        end
    end
    return S
end

class is trivial and does not require a certificate. Assuming there exists a certifying algorithm for the base class the certifying algorithm will return a set U of k vertices and a certificate that the input graph with the k vertices removed is a member of $\mathcal{C}$ when the input graph is in $\mathcal{C}+k\mathsf{v}$ or an embedding of a minimal forbidden graph in the input graph when the input graph is not a member of $\mathcal{C}+k\mathsf{v}$. The certificate for non-membership is an index of a minimal forbidden graph and an embedding of that graph in the input graph. The verifying algorithm verifies the certificate in $O(f(k))$ (*i.e.*, $O(1)$ for each k) when the input graph is not a member of $\mathcal{C}+k\mathsf{v}$ and verifies the certificate of membership in $O(T(n))$ where $T(n)$ is the complexity of the certifying membership of $\mathcal{C}$.

Prover

Every graph class $\mathcal{C}$ closed with respect to a partial order, $\leqslant$, with $|V(H)| \leq c$ for all $H \in \text{Forb}(\mathcal{C})$ and where $\leqslant$ can be checked in $T(n)$ time can be recognised in $O(T(n))$ time. For some graph classes and some partial orders there may exist more efficient recognition algorithms other than the algorithm outlined in Algorithm 5. Let $T(n)$ denote the optimal time complexity for recognising class $\mathcal{C}$.

The prover (Algorithm 10) uses a recursive approach, attempting to find a set of vertices U such that the removal of U from the input graph yields a graph belonging to the base class. Let l denote k minus the recursion depth. Assuming that $l > 0$ the algorithm finds a minimal forbidden graph for the base class and removes one of the vertices from the input graph and recursively tests if the modified graph is a member of $\mathcal{C}+l\mathsf{v}$. When $l = 0$ the algorithm calls the recognition algorithm for the base class returning either a set of vertices U indicating that the removal of U from the input graph is a member of $\mathcal{C}$ or a set containing the empty set marking that the removal of the vertices does not yield a graph belonging to $\mathcal{C}$. The algorithm will reach line 13 if the graph is not a member of $\mathcal{C}+k\mathsf{v}$ and in this case the algorithm will return a minimal forbidden graph induced by the vertices of Q.

Algorithm 10: Certifying algorithm, running in fixed-parameter time, for the recognition of the class $\mathcal{C}+k\mathsf{v}$ (an extension of the algorithm presented in [19]).

Input: A graph $G = (V, E)$, an integer k, the maximum number of vertices to be removed, and an integer $l \geq 0$, the remaining number of vertices to be removed, and a set U of already removed vertices

Output: A tuple (True, U) where U is a set of at most k vertices such that $G - U \in \mathcal{C}$; or (False, H) where $H \in \mathrm{Forb}(\mathcal{C}+k\mathsf{v})$ contained G if $G \notin \mathcal{C}+k\mathsf{v}$.

Data: A procedure $\texttt{findMinimalForbidden}_k$ that finds a minimal forbidden graph for the class $\mathcal{C}+k\mathsf{v}$.

```
1  Q := ∅
2  procedure certifyCkv(G, k, l, U)
3      if l > 0 then
4          F := findMinimalForbidden_0(G)
5          Q := Q ∪ F
6          for v ∈ F do
7              let (A, B) := certifyCkv((G − v), k, l − 1, (U ∪ {v}))
8              if B ≠ {∅} then
9                  return (A, B)
10             end
11         end
12         if l = k then
13             return (False, findMinimalForbidden_k(G⟨Q⟩))
14         end
15         return (False, {∅})
16     else
17         if recogniseClass(G) then
18             return (True, U)
19         else
20             return (False, {∅})
21         end
22     end
23 end
```

Lemma 74. *Algorithm 10 is correct and has running time $c^k \cdot n^{O(1)}$.*

Proof. The correctness of the algorithm is trivial to observe. In line 4 a minimal forbidden graph is found in G. The input graph is then modified by deleting each vertex in the minimal forbidden set and recursively calling Algorithm 10 on the smaller instance. The algorithm will terminate in one of two ways: (1) line 18 is reached and the set U is returned up the stack of recursive calls or (2) line 13 is reached because no set of modifications yields a graph belonging to $\mathcal{C}$. The running time of this algorithm is parameterized by the number of vertices to remove. At each level of the recursion in Algorithm 10 there are a maximum of c possible vertices to remove, where c is the maximum size of a preimage of a minimal forbidden graph. The depth of recursion is bounded by k. At each level of recursion a minimal forbidden graph must be found that takes $T(n)$ therefore the overall complexity is $c^k \cdot n^{O(1)}$. □

We now outline an extension of Algorithm 10 to return an embedding of a minimal forbidden graph or a set of k vertices. Observe that Algorithm 10 will return a set of U vertices such that $G - U \in \mathcal{C}$ what remains is to create an embedding of a minimal forbidden graph into the input graph in the event that $G \notin \mathcal{C}{+}k\mathsf{v}$. The minimal forbidden graphs for $\mathcal{C}{+}k\mathsf{v}$ can be found in $O(f(k))$ time by Algorithm 8. Let H be the minimal forbidden graph found in G. The prover sequentially generates the forbidden set for the class $\mathcal{C}{+}k\mathsf{v}$ in the same predetermined order as the verifier will. For each forbidden graph the algorithm should check if the forbidden graph is isomorphic to H. If the graphs are isomorphic then the algorithm should output the index of the minimal forbidden graph, *i.e.*, the number of graphs generated prior to finding an isomorphic graph, and the bijection associated with the isomorphism. As the generation of the minimal forbidden set and the bijection generation is independent of the size of the input then the outlined algorithm has the same running time as Algorithm 10, *i.e.*, $f(k) \cdot n^{O(1)}$. The correctness of the outlined algorithm follows directly from the proof of the upper bound on the maximum order of an element of the minimal forbidden set for $\mathcal{C}{+}k\mathsf{v}$.

Assuming a certifying algorithm is known for the base class $\mathcal{C}$ it is possible to extend Algorithm 10 to return a set of k vertices and justification that the removal of the vertices yields a graph in the base class (see Algorithm 11).

Algorithm 11: Certifying algorithm, running in fixed-parameter tractable time, for the recognition of the class $\mathcal{C}{+}k\mathsf{v}$ (an extension of Algorithm 10). The algorithm provides a certificate of membership for a base class.

Input: A graph $G = (V, E)$ and an integer k.
Output: A tuple (X, Y) justifying either that $G \in \mathcal{C}{+}k\mathsf{v}$ or $G \notin \mathcal{C}{+}k\mathsf{v}$. If $G \in \mathcal{C}{+}k\mathsf{v}$ then X is True and $Y = (A, B)$ where A is a set of at most k vertices and B is the certificate of $G - A \in \mathcal{C}$. If $G \notin \mathcal{C}{+}k\mathsf{v}$ then X is False and Y is an embedding of a minimal forbidden graph.
Data: A procedure `CertifyC`, returning a certificate of membership of $\mathcal{C}$.

```
procedure CertifyCkvEnhanced(G, k)
    (O, U) = certifyCkv (G, k, k, ∅)
    if O then
        (A, B) = CertifyC (G − U);
        return (A, (U, B))
    else
        return (O, U)
    end
end
```

Verifier

The verifier for Algorithm 10 has two possible types of certificate to check, either a set U of k vertices or an index and an embedding. To check if the k vertices are a valid output the checker should first check that the set U contains k vertices and the result of removing U from G is a member of the class $\mathcal{C}$. Recall we assume that either the membership of the base class is trivial and therefore does not need a certificate or there exists a certifying algorithm for the base class itself. The certificate of membership can be checked in $O\left(T(n)\right)$ time, the algorithm needs only to remove the k vertices which can be achieved in constant time.

In the case where the prover returns an index i and an embedding the verifying algorithm will generate the minimal forbidden set in the same order as the prover discarding the first i generated graphs and check that the function is a valid embedding. The minimal forbidden graphs can be generated in constant time for each fixed $\mathcal{C}$ and k. To check that the embedding is valid the algorithm must check the adjacency configuration. The embedding contains at most $f(k)$ vertices therefore the embedding can be checked naively in $O\left(f(k)^2\right)$ time. Therefore the certificate for non-membership can be checked in time independent of the input size.

For Algorithm 11, where the algorithm returns a certificate that $G - U \in \mathcal{C}$, the verifying algorithm must have access to a verifier for the base class certificate. The certificate returned from the algorithm is of one of the following forms:

- $(\mathsf{True}, (U, B))$ where U is a set of at most k vertices such that $G - U \in \mathcal{C}$ and B is a certificate that $G - U \in \mathcal{C}$
- $(\mathsf{False}, (U, B))$ where U is a set of at most k vertices and B is a certificate that $G - U \notin \mathcal{C}$
- (False, B) where B is a certificate that $G \notin \mathcal{C}{+}k\mathsf{v}$

where G is the input to the algorithm.

6.3 Summary

In this chapter two generic algorithms have been presented that provide an application for the upper bound on the maximum order of a minimal forbidden set for the class $\mathcal{C}{+}k\mathsf{v}$. It has been demonstrated that the minimal forbidden set for a class $\mathcal{C}{+}k\mathsf{v}$ can be enumerated in constant time. The recognition of the class $\mathcal{C}{+}k\mathsf{v}$ has been made certifying and an argument has been put forward to promote the development of certifying fixed-parameter algorithms. Although the examples and pseudocode explicitly refer to the class $\mathcal{C}{+}k\mathsf{v}$ the techniques used to generate the minimal forbidden set and to certifying the recognition of the graph class can easily to adapted to the class $\mathcal{C}{+}k\mathsf{e}$. The practical running time for the algorithms presented here may be improved by considering specific graph classes and specific partial orders. For the induced subgraph relation it is noteworthy that the asymptotic complexity of the generic algorithms for certifying the recognition of the class $\mathcal{C}{+}k\mathsf{v}$ is tight. If there were to exist a tighter asymptotic bound it would imply that the recognition of an induced subgraph can be achieved in less that $O(n^c)$ where c is the order of the pattern graph.

Chapter 7

Partial orders relating to edge contraction

7.1 Contraction minors

Partial orders which include the operation of contracting edges are an interesting part of the lattice structure defined in Chapter 4. Partial orders such as the minor and topological minor relations have well known results concerning characterising graphs classes, *e.g.*, planar graphs, but results concerning characterising graph classes with respect to the contraction minor relation are less common. In this chapter we provide a set of alternative characterisations for some well studied graph classes with respect to the contraction minor relation and a partial order that is defined for the first time here.

It is alluded to that the graph contractibility problem is $\mathbb{NP}$-complete [64, GT51] so it is unlikely that characterising graph classes with respect to the contraction minor relation will provide efficient algorithms for recognising graph classes but it may provide insight into other partial orders which include the edge contraction operation. The reference provided for the complexity of the contractibility problem in [64, GT51] refers the reader to private communication. We provide a proof for the contractibility problem, reducing from 1-in-3SAT (defined in Section 7.3) . The contractibility problem when parameterized by a fixed graph H, often referred to as the H-CONTRACTION problem, has also been shown to be $\mathbb{NP}$-complete for many fixed graphs. It is known that on any connected graph of order at most 4, with the exception of C_4 and P_4 the problem is polynomial time solvable [18]. It is also noted in [18] that for any graph H that is triangle free with respect to the induced subgraph relation and is not a star then the H-CONTRACTION problem is $\mathbb{NP}$-complete.

Let us first define a witness structure for the contraction minor relation. In this context, two sets A and B *touch* if and only if $A, B \subseteq V(G)$ and there exists a vertex $u \in A$ and a vertex $v \in B$ such that $uv \in E(G)$.

Definition 75. An *H-contraction-witness structure* of graph G is a bijective function f between the vertices of H and a partition of $V(G)$ into $|V(H)|$ parts such that:

- each part induces a connected subgraph in G, and
- $uv \in E(H)$ if and only if $f(u)$ and $f(v)$ touch.

The relationship between the contraction minor relation and contraction witness structures is stated by the following lemma.

Lemma 76. *$H \leqslant_{\mathrm{c}} G$ if and only is there exists an H-contraction-witness structure in G.*

We provide a set of results for characterising a set of well studied graph classes with respect to the contraction minor relation. We also motivate and introduce a new partial order for which many of the considered classes have a finite forbidden set. In the following sections we restrict the set of graphs to the set of connected graphs unless otherwise stated, this restriction is reasonable as the number of connected components is preserved by edge contraction.

For all integers $n \geq 0$ we define:

$$\mathcal{C}_n = \{C_k \mid k \geq n\} \qquad \mathcal{D}_n = \{K_{2,k} \mid k \geq n\}$$
$$\mathcal{W}_n = \{C_4 \bowtie kK_1 \mid k \geq n\} \qquad \mathcal{P}_n = \{(V_k, E_k) \mid k \geq n\}$$

where $V_k = \{u, v, y, x_0, \ldots, x_k\}$ and $E_k = \{uv, vx_i, x_iy \mid 0 \leq i \leq k\}$ and $k \geq n$.

Subclasses of perfect graphs

Chordal graphs

Chordal graphs are a well studied subclass of perfect graphs. The class is closed with respect to the induced subgraph, induced minor, and induced topological minor relations. We provide a characterisation of chordal graphs with respect to the contraction minor relation. Recall that a graph is chordal if and only if every cycle of length four or more contains a chord, that is an edge connecting two non-consecutive vertices in the cycle.

Theorem 77. *Let G be a connected graph, the following conditions are equivalent:*

(i) G is a chordal graph.

(ii) G does not contain a graph in $\{C_n \mid n \geq 4\}$ with respect to $\leqslant_{\mathrm{i}}$ [78].

(iii) G does not contain a graph in $\mathcal{D}_2 \cup \{W_4, C_4 \bowtie 2K_1\}$ with respect to $\leqslant_{\mathrm{c}}$.

Proof. Recall that condition (ii) is the classical characterisation of chordal graphs [78]. We prove that if a graph contains a chordless cycle with respect to the induced subgraph relation then the graph contains a graph in the set $\mathcal{D}_2 \cup \{W_4, C_4 \bowtie 2K_1\}$ with respect to the contraction minor relation. Let G be a connected graph containing a chordless cycle as an induced subgraph and let $C \subseteq V(G)$ such that $G[C]$ is a chordless cycle. We partition the vertices of G into two parts $S = V(G) \setminus C$ and C. Observe the following rules preserve a chordless cycle in G. The rules should be applied in order, not progressing to the next rule until the current rule can no longer be applied.

1. While $|C| > 4$ contract an edge $uv \in E(G)$ where $u, v \in C$.

2. While there is an edge $uv \in E(G)$ such that $u, v \in S$, contract uv.

3. While there is an edge uv where $v \in S$ with $\deg(v) = 1$, contract uv.

4. While there is a vertex $v \in S$ with $\deg(v) = 2$ and an edge uw where $u, w \in C$ and $uv, vw \in E(G)$, contract uv (see Figure 7.1).

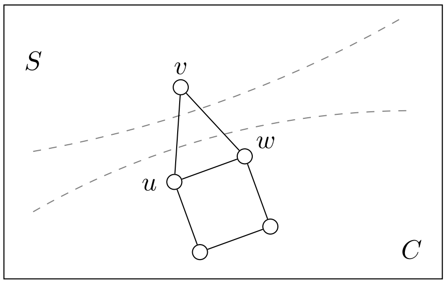


Figure 7.1: Precondition for rule 4.

5. While there is a vertex $v \in S$ with $\deg(v) = 3$ and edges $uw, wx \in E(G)$ where $u, w, x \in C$ and $vu, vw, vx \in E(G)$, contract vw (see Figure 7.2).

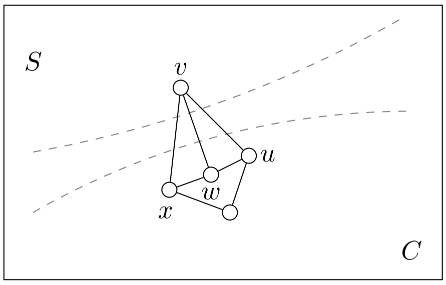


Figure 7.2: Precondition for rule 5.

6. While there are two vertices $u, v \in S$ with $\deg(v) = 2$ and $\deg(u) = 2$ and $N_G(v) \cap N_G(u) = \varnothing$, let u_1, u_2 be the neighbours of u and v_1, v_2 be the neighbours of v. Then $\{u, u_1, v_1, u_2\}$ induces a cycle of length 4. Let $C = \{u, u_1, v_1, u_2\}$ and $S = V(G) \setminus C$, continue to apply rule 2 (see Figure 7.3).

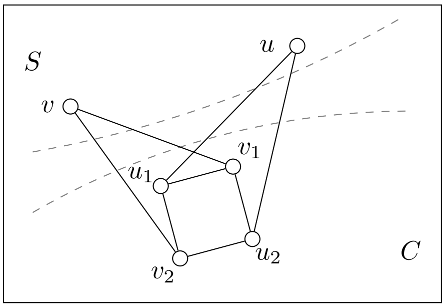


Figure 7.3: Precondition for rule 6.

7. While there are two vertices $u, v \in S$ with $\deg(v) = 4$ and $\deg(u) = 2$, let u_1, u_2 be the neighbours of u and $N(v) \setminus N(u) = \{v_1, v_2\}$ then $\{u, u_1, v_1, u_2\}$ induces a cycle of length 4. Let $C = \{u, u_1, v_1, u_2\}$ and $S = V(G) \setminus C$, continue to apply rule 2 (see Figure 7.4).

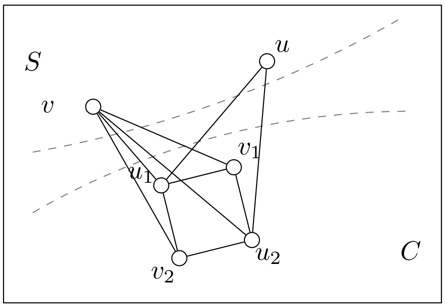


Figure 7.4: Precondition for rule 7.

8. While there are at least three vertices $u, v, x \in S$ such that $\deg(u) = 4$, $\deg(v) = 4$ and $\deg(x) = 4$, let $\{a, b, c, d\}$ be the neighbours of u (also the neighbours of v and x) such that a and c are not adjacent. Then $\{u, v, a, c\}$ induces a cycle of length 4. Let $C = \{u, v, a, c\}$ and $S = V(G) \setminus C$, continue to apply rule 2 (see Figure 7.5).

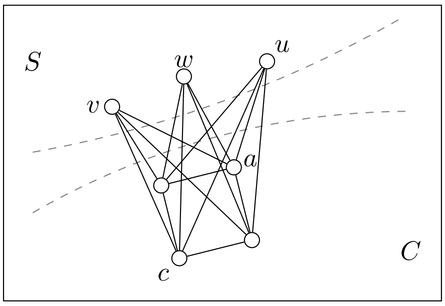


Figure 7.5: Precondition for rule 8.

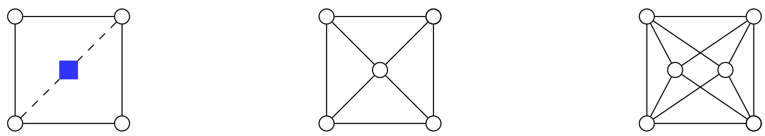


Figure 7.6: Minimal forbidden graphs for the class of connected chordal graphs with respect to the contraction minor relation, $\mathcal{D}_2, W_4, C_4 \bowtie 2K_1$ (respectively). Solid vertices indicate a possible empty set of vertices with the neighbours identified by the broken edges.

When the reduction rules can no longer be applied the result is one of the following forms:

- all vertices $v \in S$ have the same neighbourhood and $\deg(v) = 2$ then the graph is isomorphic to a graph in $\mathcal{D}_2$, or
- all vertices $v \in S$ have $\deg(v) = 4$ and $|S| \leq 2$. The graph is isomorphic to a graph in the set $\{W_4, C_4 \bowtie 2K_1\}$.

From this we obtain that every graph that contains a chordless cycle as an induced subgraph is contractible to a graph in the set $\mathcal{D}_2 \cup \{W_4, C_4 \bowtie 2K_1\}$ (see Figure 7.6). Observe that all graphs in the set $\mathcal{D}_2 \cup \{W_4, C_4 \bowtie 2K_1\}$ are minimal non-chordal graphs with respect to the contraction minor relation, that is, the contraction of any edge yields a chordal graph.

Lastly we show that every graph that can be contractible to W_4, $C_4 \bowtie 2K_1$ or a graph in $\mathcal{D}_2$ contains a chordless cycle. We show this by first observing that each graph in $\mathcal{D}_2 \cup \{W_4, C_4 \bowtie 2K_1\}$ contains a chordless cycle and that by contracting an edge in a graph it is not possible to construct a chordless cycle if one did not already exist.

Let G be a chordal graph and let G' be the graph after contracting the edge $uv \in E(G)$. Let w is the new vertex introduced from the contraction of the edge uv. Let us assume that G' is not chordal, therefore there exists a cycle $C = [w, c_0, \ldots, c_n]$ where $n \geq 2$. It is sufficient to show that $[u, v, c_0, \ldots, c_n]$ or $[u, c_0, \ldots, c_n]$ where $n \geq 2$ is a chordless cycle in G, contradicting that G is chordal. Consider expanding the vertex w into the vertices u and v, let C' be the graph induced by $\{u, v, c_0, \ldots c_n\}$ on G. As C is a chordless cycle then $N_{C'}(u) \subseteq \{v, c_0, c_n\}$ and $N_{C'}(v) \subseteq \{u, c_0, c_n\}$ (see Figure 7.7). If this was not the case then C would not be chordless cycle. Now assume that the edges $\{uc_0, vc_n\}$ are not present then $[u, v, c_0, \ldots, c_n]$ is a chordless cycle in G. Otherwise assume without loss of generality that the edge uc_0 is present then $[u, c_0, \ldots, c_n]$ is a chordless cycle in G. Therefore if there is a chordless cycle after contracting an edge then the graph contained a chordless cycle before the contraction.

Concluding the proof, we have shown the equivalence of a graph containing a chordless cycle as an induced subgraph and a graph containing a graph in the set $\mathcal{D}_2 \cup \{W_4, C_4 \bowtie 2K_1\}$ with respect to the contraction minor relation. This provides a characterisation of connected chordal graphs with respect to the contraction minor relation. □

For a disconnected graph to not be chordal it is sufficient for one component to contain a chordless cycle, all remaining components can be contracted to a single isolated vertex. A

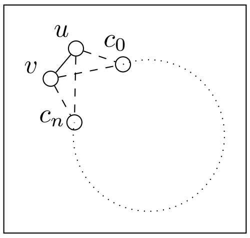


Figure 7.7: Adjacency of u, v to vertices in C.

corollary to Theorem 77 provides a characterisation of chordal graphs, relaxing the restriction from only connected graphs, with respect to the contraction minor relation.

Corollary 78. The set of minimal forbidden graphs for the class of chordal graphs with respect to the contraction minor relation is $\{\mathcal{D}_2 \uplus lK_1, W_4 \uplus lK_1, (C_4 \bowtie 2K_1) \uplus lK_1\}$ for $l \geq 0$.

Split graphs

Split graphs are the intersection of the class of chordal graphs and the class of co-chordal graphs. Recall from Chapter 2 that the vertices of a split graph can be partitioned into two parts; one which induces a complete graph and one which induces an independent set [58]. Many of the classical problems on split graphs can be solved in polynomial time.

$$\text{Split graphs} = \text{Chordal graphs} \cap \text{Co-Chordal graphs} \; [72, \text{Theorem 6.3.II}]$$

We provide a characterisation of split graphs with respect to the contraction minor relation. We first require the following lemmas.

Lemma 79. *If G is a connected graph containing $2K_2$ as an induced subgraph then it is possible to contract G to either $2K_2 \bowtie K_1, \overline{P}, P_5$ (see Figure 7.9) or a graph in $\mathcal{D}_2$.*

Proof. Let G be a connected graph where $u, v, x, y \in V(G)$ and $uv, xy \in E(G)$ such that $G[\{u, v, x, y\}] \simeq 2K_2$. Let $G' = G - \{u, v, x, y\}$ and let C be the set of connected components of G'.

Case 1. $|C| = 1$

Contract the edges of $\{ab \mid a, b \in V(G) \setminus \{u, v, x, y\}\}$ in G to a single vertex s. There are three configurations of adjacency between s and the set $\{u, v, x, y\}$. Each K_2 has either one or two edges incident to s (see Figure 7.9).

Case 2. $|C| > 1$

Case 2.1. *Some component has only neighbours in either $\{u, v\}$ or $\{x, y\}$.*
Without loss of generality assume the component has neighbours in $\{u, v\}$, then by contracting all edges of the component to a single vertex s forms either a K_3 or P_3 between the vertices

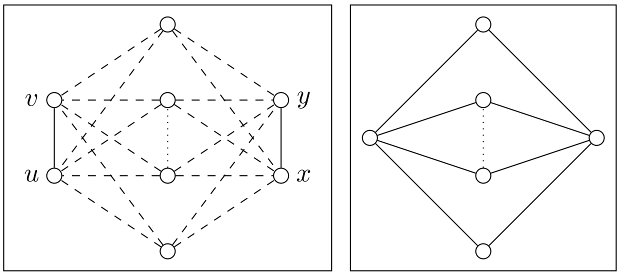


Figure 7.8: Adjacency configuration for Case 2.2. Broken lines indicate possible edges (Left). Result of contracting the edges uv and xy (Right).

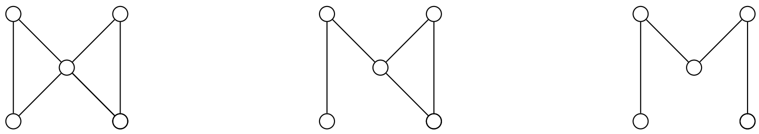


Figure 7.9: $2K_2 \bowtie K_1, \overline{P}, P_5$ (respectively)

$\{u, v, s\}$. In either case an edge incident with s can be contracted while still preserving a $2K_2$ in G.

Case 2.2. *Where Case 2.1 does not apply, each component has neighbours in both $\{u, v\}$ and $\{x, y\}$. Contracting each component in C to a single vertex results in all paths between $\{u, v\}$ and $\{x, y\}$ being of length 2. Each component, which has been contracted to a single vertex, has at least one neighbour in $\{u, v\}$ and at least one neighbour in $\{x, y\}$. By contracting uv and xy the resulting graph is isomorphic to a graph in $\mathcal{D}_2$ (see Figure 7.8).*

□

Lemma 80. *If G is a connected graph which is contractible to $2K_2 \bowtie K_1, \overline{P}$ or P_5 then G contains $2K_2$ as an induced subgraph.*

Proof. Let G be a connected graph that is contractible to $2K_2 \bowtie K_1, \overline{P}$ or P_5, then there exists an H-contraction-witness structure where $H \in \{2K_2 \bowtie K_1, \overline{P}, P_5\}$, let W denote the H-contraction-witness structure. Observe that each graph in $\{2K_2 \bowtie K_1, \overline{P}, P_5\}$ contains $2K_2$ as an induced subgraph. Let u, v, x, y, z be the vertices of H, such that the vertices in $W(u) \cup W(v)$ have no neighbours in $W(x) \cup W(y)$ (see Figure 7.10). Then by selecting any edge with endpoints in $W(u)$ and $W(v)$ and an edge with endpoints in $W(x)$ and $W(y)$ a $2K_2$ is obtained. This $2K_2$ is induced in G. □

Observe that a graph containing an induced cycle of length six or more contains an induced $2K_2$ and therefore contains a graph in $2K_2 \bowtie K_1, \overline{P}, P_5$ or $\mathcal{D}_2$ as a contraction minor.

Theorem 81. *Let G be a connected graph, the following conditions are equivalent:*

(i) G is a split graph.

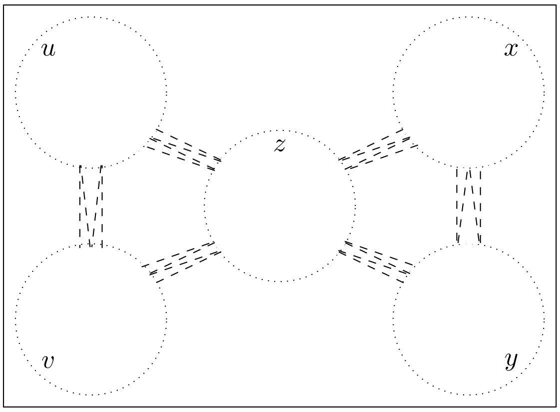


Figure 7.10: Adjacency of connected subgraphs for Lemma 80.

(ii) G does not contain a graph in $\{2K_2, C_4, C_5\}$ *with respect to* $\leqslant_{\mathrm{i}}$ [58].

(iii) G does not contain a graph in $\mathcal{D}_2 \cup \{W_4, C_4 \bowtie 2K_1, 2K_2 \bowtie K_1, \overline{P}, P_5\}$ *with respect to* $\leqslant_{\mathrm{c}}$.

Proof. We recall that condition (ii) is the classical characterisation of split graphs [58]. We prove next that condition (ii) holds if and only if condition (iii) holds. From Lemma 79 we obtain that if a graph contains $2K_2$ as an induced subgraph then the graph contains a graph in $\{2K_2 \bowtie K_1, \overline{P}, P_5\} \cup \mathcal{D}_2$ as a contraction minor. Observe C_4 and C_5 are chordless cycles, from Theorem 77 we have that if there exists a chordless cycle as an induced subgraph then there exists a graph in $\mathcal{D}_2 \cup \{W_4, C_4 \bowtie 2K_1\}$ as a contraction minor. Therefore if a graph contains a graph in $\{2K_2, C_4, C_5\}$ as an induced subgraph then that implies that the graph contains a graph in $\mathcal{D}_2 \cup \{W_4, C_4 \bowtie 2K_1, 2K_2 \bowtie K_1, \overline{P}, P_5\}$ as a contraction minor. To prove the opposite direction observe from Lemma 80 we get that if a graph contains a graph in $\{K_2 \bowtie K_1, \overline{P}, P_5\}$ as a contraction minor then G contains $2K_2$ as an induced subgraph. From Theorem 77 we obtain that if a graph contains a graph in $\mathcal{D}_2 \cup \{W_4, C_4 \bowtie 2K_1\}$ as a contraction minor then the graph contains a chordless cycle, hence containing either a C_4, C_5 or $2K_2$ (which is contained in any cycle of length greater than 5) as an induced subgraph. □

Cographs

Recall a cograph is a graph that is $\{P_4\}$-free$_{\mathrm{i}}$ (see Chapter 2).

Theorem 82. *Let G be a connected graph, the following conditions are equivalent:*

(i) G is a cograph.

(ii) G does not contain P_4 *with respect to* $\leqslant_{\mathrm{i}}$ [17, Theorem 11.3.3].

(iii) G does not contain a graph in $\{\mathcal{P}_0, P_4 \bowtie kK_1, \overline{P_5}, C_5, \overline{C_6}\}$ *with respect to* $\leqslant_{\mathrm{c}}$ *where* $k \geq 1$ (see Figure 7.16).

Proof. Recall that condition (ii) is the classical characterisation of cographs [27]. It remains to show the equivalence between conditions (ii) and (iii). Let G be a connected graph containing P_4 as an induced subgraph and let $P = \{u, v, x, y\}$ such that $P \subseteq V(G)$ and $G[P] \simeq P_4$ (and $uv, vx, xy \in E(G)$). Let $S = V(G) \setminus P$. Observe the following rules preserve a P_4. The rules should be applied in order, not progressing to the next rule until the current rule can no longer applied.

1. While there exists an edge $ab \in E(G)$ such that $a, b \in S$ contract the edge ab.

2. While there exists a pendent vertex $a \in S$ with neighbour $b \in P$ contract the edge ab.

3. While there exists a vertex $a \in S$ of degree two with adjacent neighbours b and c on P contract the edge ab.

4. While there exists a vertex $a \in S$ with $\deg(a) = 3$ and edges $bc, cd \in E(G)$ where $b, c, d \in P$ and $ab, ac, ad \in E(G)$ then contract the edge ac.

5. While there exists two vertices $a, b \in S$ such that $\deg(a) = 2$ and $\deg(b) = 2$, $N(a) = \{a_1, a_2\}$ and $N(b) = \{b_1, b_2\}$ and $a_1b_1 \in E(G)$ and $N(a) \cap N(b) = \varnothing$ then $\{a, a_1, b_1, b\}$ induces a P_4. Let $P = \{a, a_1, b_1, b\}$ and $S = V(G) \setminus P$, continue to apply rule 1.

6. While there exists two vertices $a, b \in S$ such that $\deg(a) = 2$, $\deg(b) = 2$, $N(a) = N(b)$ and $N(a) = \{u, y\}$ (see Figure 7.11) then $\{a, u, v, x\}$ induces a P_4. Let $P = \{a, u, v, x\}$ and $S = V(G) \setminus P$, continue to apply rule 1.

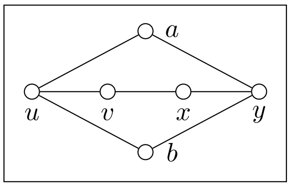


Figure 7.11: Adjacency for rule 6.

7. While there exists two vertices $a, b \in S$ such that $\deg(a) = 2$, $\deg(b) = 2$ and $|N(a) \cap N(b)| = 1$. Without loss of generality let $N(a) = \{u, y\}$ and $N(b) = \{u, x\}$ then $\{a, u, b, x\}$ induces a P_4. Let $P = \{a, u, b, x\}$ and $S = V(G) \setminus P$, continue to apply rule 1.

8. While there exists two vertices $a, b \in S$ such that $\deg(a) = 3$, $\deg(b) = 2$ and $|N(a) \cap N(b)| = 2$ we apply the following reduction. Without loss of generality let $N(a) = \{u, v, y\}$, then either $N(b) = \{u, y\}$ (see Figure 7.12 Left) or $N(b) = \{v, y\}$ (see Figure 7.12 Right). Assume $N(b) = \{u, y\}$ then $\{b, y, x, v\}$ induces a P_4 in G. Let $P = \{b, y, x, v\}$ and $S = V(G) \setminus P$, continue to apply rule 1. Otherwise let $N(b) = \{v, y\}$ then $\{u, a, y, b\}$ induces a P_4. Let $P = \{u, a, y, b\}$ and $S = V(G) \setminus P$, continue to apply rule 1.

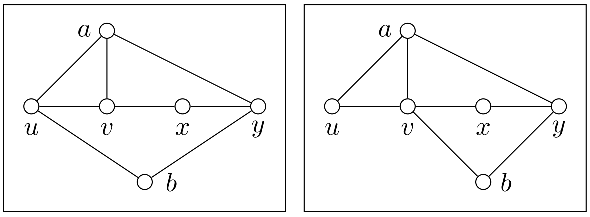


Figure 7.12: Adjacency for rule 8.

9. While there exists two vertices $a, b \in S$ such that $\deg(a) = 3$, $\deg(b) = 2$ and $|N(a) \cap N(b)| = 1$ (see Figure 7.13) we apply the following reduction . Without loss of generality let $N(b) = \{u, x\}$ then the set $\{b, x, y, a\}$ induces a P_4 in G. Let $P = \{b, x, y, a\}$ and $S = V(G) \setminus P$, continue to apply rule 1.

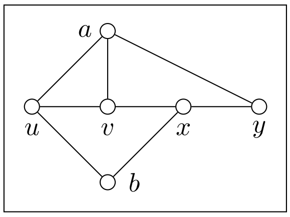


Figure 7.13: Adjacency for rule 9.

10. While there exists two vertices $a, b \in S$ such that $\deg(a) = 3$, $N(a) = N(b)$ and without loss of generality let $N(a) = \{u, v, y\}$ then $\{u, a, y, x\}$ induces a P_4 (see Figure 7.14). Let $P = \{u, a, y, x\}$ and $S = V(G) \setminus P$, continue to apply rule 1.

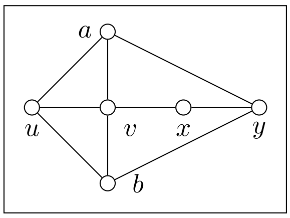


Figure 7.14: Adjacency for rule 10.

11. While there exists two vertices $a, b \in S$ such that $\deg(a) = 4$ and $\deg(b) = 2$. Without loss of generality let $N(a) = \{u, v, x, y\}$ and $N(b) = \{u, x\}$ then the set $\{y, a, u, b\}$ induces a P_4 in G. Let $P = \{y, a, u, b\}$ and $S = V(G) \setminus P$, continue to apply rule 1.

12. While there exists two vertices $a, b \in S$ such that $\deg(a) = 4$, $\deg(b) \neq 4$ and without loss of generality let $\{v\} \subset N(a) \setminus N(b)$ then $\{b, c, a, v\}$ induces a P_4 where $c \notin N(v)$ and $bc \in E(G)$. Let $P = \{b, c, a, v\}$ and $S = V(G) \setminus P$, continue to apply rule 1.

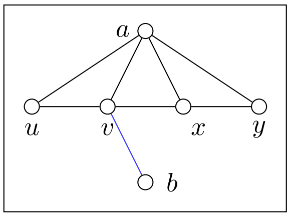


Figure 7.15: Adjacency for rule 12. Non-black edges indicate two non-adjacent vertices.

The result of the above rules leave the following configurations:

1. $S = \{a\}$ such that $\deg(a) = 2$ and $N(a) = \{u, y\}$. This configuration is isomorphic to C_5.
2. $S = \{a\}$ such that $\deg(a) = 3$. This configuration is isomorphic to $\overline{P_5}$.
3. $S = \{a, b\}$ such that $\deg(a) = 3$, $\deg(b) = 3$ and $|N(a) \cap N(b)| = 2$. This configuration is isomorphic to $\overline{C_6}$.
4. For all $a \in S$ we have $\deg(a) = 2$ and there exists an element $b \in P$ such that $N(b) = N(a)$. This configuration is isomorphic to a graph in $\mathcal{P}_0$.
5. For all $a \in S$ $\deg(a) = 4$. This configuration is isomorphic to $P_4 \bowtie kK_1$ where $|S| = k$.

Therefore if a connected graph contains P_4 as an induced subgraph then the graph is contractible to a graph in $\mathcal{P}_0 \cup \{P_4 \bowtie kK_1, \overline{P_5}, C_5, \overline{C_6}\}$ where $k \geq 1$.

To prove the opposite direction, assume there exists a graph $H \in \{P_4 \bowtie kK_1, \overline{P_5}, C_5, \overline{C_6}\} \cup \mathcal{P}_0$ where $k \geq 1$ such that $H \leqslant_c G$ then we show that $P_4 \leqslant_i G$. Observe that $P_4 \leqslant_i H$. As G is contractible to H there must exists an H-contraction-witness structure W, let $u, v, x, y \in V(H)$ and $P_4 \simeq H[\{u, v, x, y\}]$ such that $\{uv, vx, xy\} \subseteq E(H)$. Let $a \in W(u)$ and $b \in W(y)$. As G is connected and W is an H-contraction-witness structure then there must exist an ab-path in $G[W(u) \cup W(v) \cup W(x) \cup W(y)]$, let P denote this path. The path P must contain at least 4 vertices as it crosses 4 witness sets. Therefore G must contain an induced P_4. Concluding, a graph is a cograph if and only if the graph does not contain a graph in $\mathcal{P}_0 \cup \{P_4 \bowtie kK_1, \overline{P_5}, C_5, \overline{C_6}\}$ with respect to $\leqslant_c$. □

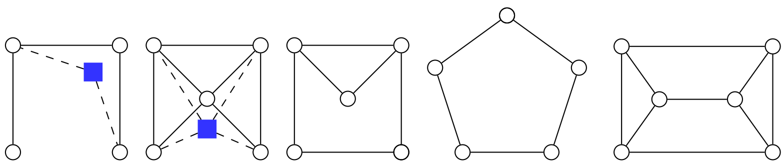


Figure 7.16: Minimal forbidden graphs for the class of connected cographs with respect to the contraction minor relation; $\mathcal{P}_0$, $P_4 \bowtie kK_1$, $\overline{P_5}$, C_5 and $\overline{C_6}$ (respectively) where $k \geq 1$. Solid vertices indicate a possibly empty set of vertices with adjacency given by the broken edges.

7.2 False twin minors

We define a new partial order, similar to contraction minor, which allows the additional operation of removing false twins. The removal of a false twin is the operation of deleting a vertex if there exists another vertex with an identical neighbourhood, that is, if there exists vertices $u, v \in V(G)$ such that $N_G(u) = N_G(v)$ then we may delete the vertex u. The newly defined partial order is called *false twin minor* and is denoted $\leqslant_{\mathrm{ftm}}$. The false twin minor relation may seem arbitrary but it overcomes some of the difficulties encountered when working with contraction minors. For instance the forbidden sets when considering the contraction minor partial order are often infinite due to the addition of false twins. This can easily be observed when considering a class of graphs which allows graphs with differing numbers of components, let $\mathcal{C}$ be a class closed with respect to the contraction minor partial order and let $\mathcal{C}'$ be a restriction of $\mathcal{C}$ to the set of connected graphs. The forbidden set for $\mathcal{C}$ will contain $\{H \uplus kK_1 \mid 0 \leq k \wedge H \in \mathrm{Forb}(\mathcal{C}')\}$ and these graphs will be minimal. This idea is explored in more detail in Section 7.4.

The false twin minor relation is an extension of the contraction minor relation and is a restriction of induced minor relation, that is the relation is sandwiched between the contraction minor and induced minor relation in the lattice defined in Chapter 4. The relation is not a well-quasi ordering on the set $\mathcal{G}$ as the set $\{nK_2 \mid n \geq 1\}$ is an infinite antichain. Alternatively, as the property of a partial order possessing an infinite antichain is inherited by the ideal then any antichain from the induced minor relation is also an antichain with respect to the false twin minor relation. The motivation for the false twin minor relation is derived from the requirement for a finite minimal forbidden set. For several interesting graph classes $\mathrm{Forb}(\mathcal{C})_{\mathrm{ftm}}$ is finite where $\mathrm{Forb}(\mathcal{C})_{\mathrm{i}}$ and $\mathrm{Forb}(\mathcal{C})_{\mathrm{c}}$ are infinite.

As with the contraction minor partial order there exists a witness structure for false twin minors. We require first a property of the false twin minor relation before we can define a witness structure.

Lemma 83. *If $H \leqslant_{\mathrm{ftm}} G$ then there exists a graph H' such that $H' \leqslant_{\mathrm{c}} G$ and $H \leqslant^* H'$ where $\leqslant^*$ denotes the removal of false twins.*

Proof. Let G, H be two graphs such that $H \leqslant_{\mathrm{ftm}} G$. Then there exists a tight chain with respect to the partial order such that $H \leqslant_{\mathrm{ftm}} H_1 \leqslant_{\mathrm{ftm}} \ldots \leqslant_{\mathrm{ftm}} H_k \leqslant_{\mathrm{ftm}} G$. Let l be the maximum valued index such that H_l is obtained from H_{l+1} by deleting a false twin. Let u, v be false twins in H_{l+1} and let H_l be the graph after removing the vertex v. Then one of the three cases apply:

1. all false twin deletions occur before index $l + 1$ and for all $j < l + 1$, H_j is obtained by deleting a false twin from H_{j+1}.
2. let $uu_a \in E(H_{l+1})$, for some $p < l$ the graph H_p is obtained from H_{p+1} by contracting the edge uu_a.
3. for all edges $uu_a \in E(H_{l+1})$ we have $uu_a \in E(H)$.

First assume the first case then all edge contractions occur after the index $l+1$ therefore if there are any false twin deletions then they must occur before index $l+1$, hence the theorem is correct and $H' = H_{l+1}$.

Assume the second case then the edge uu_a has been contracted. As u and v are false twins then v is adjacent to u_a by contracting the edge vu_a we obtain the same graph as if we deleted v and contracted the edge uu_a. Therefore if we are in Case 2 then we may replace the deletion of a false twin by an edge contraction.

Assume the third case. Let u, v be false twins in H_{l+1} as no edge incident to u is contracted then u and v remain false twins until v is deleted. This deletion can be moved to any position in the sequence before $l+1$.

Concluding we may modify the chain such that all edge contractions happen before the removal of false twins, therefore there exists a graph H' such that $H \leqslant^* H'$ where $\leqslant^*$ denotes the removal of false twins. □

Corollary 84. There exists a H'-contraction-witness structure in G by Lemma 76.

Definition 85. Let $H \leqslant_{\mathrm{ftm}} G$ then the false twin witness structure is a pair (H', U) such that $H' \leqslant_{\mathrm{c}} G$ and $U \subset V(H')$ whose removal from H' yields a graph isomorphic to H.

Theorem 86. *If $H \leqslant_{\mathrm{ftm}} G$ then there exists a false twin witness structure.*

Proof. The proof follows easily from Corollaries 84 and Definition 85. □

We require the following statement before we continue to classify hereditary graph classes with respect to the false twin minor relation.

Claim 87. Let G be a member of an hereditary graph class $\mathcal{C}$ then by the application of the false twin deletion operation on G the resulting graph G' is also a member of $\mathcal{C}$.

Proof. Let G be a member of an hereditary graph class $\mathcal{C}$ and let $G' = G - u$ where u is a false twin of a vertex v and $u, v \in V(G)$. Let $\leqslant^*$ denote the partial order defined by the operation of deleting a false twin. Observe that for any two graphs $G, H \in \mathcal{G}$ if $G \leqslant^* H$ then $G \leqslant_{\mathrm{i}} H$. The class $\mathcal{C}$ is closed with respect to $\leqslant_{\mathrm{i}}$ by its definition and is therefore closed with respect to $\leqslant^*$. □

.

Subclasses of perfect graphs

Chordal graphs

Theorem 88. *A connected graph is chordal if and only if it does not contain C_4 or W_4 as a false twin minor.*

Proof. From Theorem 77 it has been established that if a graph contains a chordless cycle with respect to the induced subgraph relation then the graph contains a graph in $\mathcal{D}_2 \cup \{W_4, C_4 \bowtie 2K_1\}$ with respect to the contraction minor relation. Observe that if $G \leqslant_c H$ then $G \leqslant_{\text{ftm}} H$ therefore if a graph contains a chordless cycle with respect to the $\leqslant_i$ then it contains a graph in $\mathcal{D}_2 \cup \{W_4, C_4 \bowtie 2K_1\}$ with respect to $\leqslant_{\text{ftm}}$. Observe that for all $G \in \mathcal{D}_2$ we have $C_4 \leqslant_{\text{ftm}} G$ and for all $G' \in \{W_4, C_4 \bowtie 2K_1\}$ we have $W_4 \leqslant_{\text{ftm}} G'$. Both C_4 and W_4 are minimal non-chordal graphs with respect to $\leqslant_{\text{ftm}}$. We now show that if a graph contains C_4 or W_4 with respect to $\leqslant_{\text{ftm}}$ then the graph contains a chordless cycle. From Theorem 77 it has been established that by contracting edges it is not possible to construct a chordless cycle in a chordal graph, it remains to show that the removal of a false twin cannot construct a chordless cycle in a chordal graph.

Note that the class of chordal graphs is an hereditary graph class, therefore by Claim 87 the removal of a false twin from G to obtain the graph G' then the G' will also be a chordal graph. The operations that define the $\leqslant_{\text{ftm}}$ partial order are edge contraction and false twin deletion. Neither operation can construct a chordless cycle. Therefore if a chordless cycle is contained within a graph G with respect to $\leqslant_{\text{ftm}}$ then a chordless cycle is contained in G with respect to $\leqslant_i$. Observe both C_4 and W_4 contain a chordless cycle as an induced subgraph, consequently if G contains C_4 or W_4 with respect to $\leqslant_{\text{ftm}}$ then G is not chordal. □

Split graphs

Theorem 89. *A connected graph is a split graph if and only if it does not contain C_4, W_4, $2K_2 \bowtie K_1$, $\overline{P}$ or P_5 as a false twin minor.*

Proof. Recall from Theorem 81 that the class of split graphs forbids the graphs $\mathcal{D}_2 \cup \{W_4, C_4 \bowtie 2K_1, P_5, \overline{P}, 2K_2 \bowtie K_1\}$ as contraction minors. Observe that if $G \leqslant_c H$ then $G \leqslant_{\text{ftm}} H$ therefore if G is not a split graph it must contain a graph $H \in \mathcal{D}_2 \cup \{W_4, C_4 \bowtie 2K_1, P_5, \overline{P}, 2K_2 \bowtie K_1\}$ with respect to $\leqslant_{\text{ftm}}$. From Theorem 88 recall that if a graph contains a graph in $\mathcal{D}_2 \cup \{W_4, C_4 \bowtie 2K_1\}$ as a contraction minor then it contains either C_4 or W_4 with respect to $\leqslant_{\text{ftm}}$. Observe that $P_5, \overline{P}$ and $2K_2 \bowtie K_1$ are $\leqslant_{\text{ftm}}$-minimal non-split graphs therefore the $\leqslant_{\text{ftm}}$-minimal non-split graphs are $\{W_4, C_4, 2K_2 \bowtie K_1, \overline{P}, P_5\}$. We now prove the reverse direction. From Theorem 88 we obtain that if a graph contains C_4 or W_4 with respect to $\leqslant_{\text{ftm}}$ then the graph contains a chordless cycle with respect to the induced subgraph relation and is therefore not a split graph. It remains to show that if a graph is a member of the class $\{2K_2\}$-free$_i$ then it is not possible to construct a $2K_2$ by contracting edges.

Claim 90. Let G be a member of the class $\{2K_2\}$-free$_i$ then by contracting any edge in G the resulting graph is a member of the class $\{2K_2\}$-free$_i$.

Proof. Let G be a member of the class $\{2K_2\}$-free$_i$ and $uv \in E(G)$. Let $G' = G/uv$ and w be the new vertex introduced by the contraction of uv. Assume $G' \notin \{2K_2\}$-free$_i$, let $\{ab, cd\}$ be an induced $2K_2$ in G'. If w is distinct from a, b, c and d then G is not a member of $\{2K_2\}$-free$_i$.

Therefore without loss of generality assume $w = a$. Consider the graph prior to the contraction of uv. The vertex b is adjacent to u or v and $\{c, d\} \nsubseteq N_G(u)$, $\{c, d\} \nsubseteq N_G(v)$ and $\{c, d\} \nsubseteq N_G(b)$ else $\{ab, cd\}$ would not induce a $2K_2$ in G'. This yields that $\{ub, cd\}$, $\{vb, cd\}$ or $\{uv, cd\}$ was an induced $2K_2$ in G, which is a contradiction that $G \in \{2K_2\}$-free$_\mathrm{i}$. Concluding, the assumption that $G' \notin \{2K_2\}$-free$_\mathrm{i}$ was wrong and therefore contracting an edge can not construct a $2K_2$ (see Figure 7.17). □

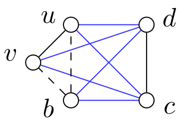


Figure 7.17: Adjacency for Claim 90. Blue edges indicate non-edges and dashed edges indicate the possible adjacency.

Observe from Claim 87, 90 that it is not possible to construct a $2K_2$ from a graph in $\{2K_2\}$-free$_\mathrm{i}$ by contracting edges and deleting false twins. Therefore any graph that contains $P_5, \overline{P}$ or $2K_2 \bowtie K_1$ with respect to $\leqslant_{\mathrm{ftm}}$ must contain $2K_2$ with respect to $\leqslant_\mathrm{i}$.

□

Cographs

Theorem 91. *A connected graph is a cograph if and only if it does not contain P_4, $P_4 \bowtie K_1$, $\overline{P_5}$, C_5 or $\overline{C_6}$ as a false twin minor.*

Proof. Recall that a graph is a cograph if and only if it does not contain P_4 as an induced subgraph. Observe from Theorem 82 if a graph contains P_4 as an induced subgraph then it contains a graph in $\{\mathcal{P}_0, P_4 \bowtie kK_1, \overline{P_5}, C_5, \overline{C_6}\}$ as a contraction minor (where $k \geq 1$). Note that if $G \leqslant_\mathrm{c} H$ then $G \leqslant_{\mathrm{ftm}} H$ therefore if a graph contains a P_4 as an induced subgraph then the graph contains a graph in $\{\mathcal{P}_0, P_4 \bowtie kK_1, \overline{P_5}, C_5, \overline{C_6}\}$ where $k \geq 1$ with respect to $\leqslant_{\mathrm{ftm}}$. Observe $\mathcal{P}_0$ and $P_4 \bowtie kK_1$ contain false twins, for all $k > 1$. If $G \in \mathcal{P}_0$ then $P_4 \leqslant_{\mathrm{ftm}} G$ or if $G \in P_4 \bowtie kK_1$ then $P_4 \bowtie K_1 \leqslant_{\mathrm{ftm}} G$. Therefore if a graph contains a P_4 with respect to $\leqslant_\mathrm{i}$ then the graph contains a graph in $\{P_4, P_4 \bowtie K_1, \overline{P_5}, C_5, \overline{C_6}\}$ with respect to $\leqslant_{\mathrm{ftm}}$.

To prove the opposite direction, assume $H \leqslant_{\mathrm{ftm}} G$ and $H \in \{P_4, P_4 \bowtie K_1, \overline{P_5}, C_5, \overline{C_6}\}$. As $H \leqslant_{\mathrm{ftm}} G$ then there exists a false twin witness structure (H', U) such that $H' \leqslant_\mathrm{c} G$. Observe that for all $H \in \{P_4, P_4 \bowtie K_1, \overline{P_5}, C_5, \overline{C_6}\}$ $P_4 \leqslant_\mathrm{i} H$ and that it is not possible to construct an induced P_4 by removing false twins therefore $P_4 \leqslant_\mathrm{i} H'$. From Theorem 82 we know that if $P_4 \leqslant_\mathrm{i} H'$ then there exists a graph $J \leqslant_\mathrm{c} H'$ where $J \in \{\mathcal{P}_0, P_4 \bowtie kK_1, \overline{P_5}, C_5, \overline{C_6}\}$ where $k \geq 1$. From the transitivity of the partial order we have that $J \leqslant_\mathrm{c} G$. Consequently from Theorem 82 we know that a graph contains a graph in $\{\mathcal{P}_0, P_4 \bowtie kK_1, \overline{P_5}, C_5, \overline{C_6}\}$ where $k \geq 1$ if and only if it contains a P_4 as an induced subgraph therefore G must contains a P_4 as an induced subgraph.

Concluding the proof that a connected graph is a cograph if and only if the graph does not contain a graph in $\{P_4, P_4 \bowtie K_1, \overline{P_5}, C_5, \overline{C_6}\}$ with respect to $\leqslant_{\mathrm{ftm}}$. □

General construction

From the results of the previous sections we distil a succinct theorem relating the size of a minimal forbidden set with respect to the induced subgraph and false twin minor relations. This provides a method for generating a minimal forbidden set for any class $\mathcal{C}$ which satisfies a set of simple properties.

Theorem 92. *For any hereditary class $\mathcal{C}$ closed under edge contraction where $\mathrm{Forb}(\mathcal{C})_{\mathrm{i}}$ is finite then $\mathrm{Forb}(\mathcal{C})_{\mathrm{ftm}}$ is finite too. Furthermore,*

$$\forall H \in \mathrm{Forb}(\mathcal{C})_{\mathrm{ftm}} \quad |V(H)| \leq 2^k + k$$

where $k = \max\{|V(G)| \mid G \in \mathrm{Forb}(\mathcal{C})_{\mathrm{i}}\}$

Proof. Let $\mathcal{C}$ be a hereditary class closed under edge contraction and let $\mathcal{F}_{\mathrm{i}}, \mathcal{F}_{\mathrm{ftm}}$ denote the sets of minimal forbidden graphs. Let $H \notin \mathcal{C}$ which implies there exists an element $F \in \mathcal{F}_{\mathrm{i}}$ such that $F \leqslant_{\mathrm{i}} H$. Observe that if $\mathcal{C}$ is closed with respect to $\leqslant_{\mathrm{i}}$ and $\leqslant_{\mathrm{c}}$ then $\mathcal{C}$ is also closed with respect to $\leqslant_{\mathrm{ftm}}$. Contracting the edges $\{uv \in E(H) \mid u, v \in V(H) \setminus V(F)\}$ leaves an independent set S and an induced subgraph of F with additional edges between S and $V(F)$. As $\leqslant_{\mathrm{ftm}}$ allows the removal of false twins the number of additional vertices is equal to at most the number of subsets of vertices in H. As $\mathcal{F}_{\mathrm{i}}$ is finite there is an upper bound on the maximum order of a graph in the set, let $k = \max\{|F| \mid F \in \mathcal{F}_{\mathrm{i}}\}$. Then the maximum number of vertices of a graph in $\mathcal{F}_{\mathrm{ftm}}$ is at most $2^k + k$. □

The upper bound on the size of the forbidden set for a hereditary graph class closed with respect to edge contractions can be applied to prove that trivially perfect and threshold graphs can be characterised by finite forbidden sets. With the previously stated characterisations of graph classes we are able to provide characterisations for trivially perfect and threshold graphs.

Theorem 93. *A connected graph is trivially perfect if and only if it does not contain C_4, W_4, P_4 or $P_4 \bowtie K_1$ as a false twin minor.*

Proof. Trivially perfect graphs are the intersection of chordal graphs and cographs therefore

$$\mathrm{Forb}(\text{trivially perfect}) = \mathrm{minimal}(\mathrm{Forb}(\text{cographs}) \cup \mathrm{Forb}(\text{chordal graphs}))$$

with respect to $\leqslant_{\mathrm{ftm}}$ (Theorem 39). □

Theorem 94. *A connected graph is a threshold graph if and only if it does not contain C_4, W_4, $2K_2 \bowtie K_1$, P_4 or $P_4 \bowtie K_1$ as a false twin minor.*

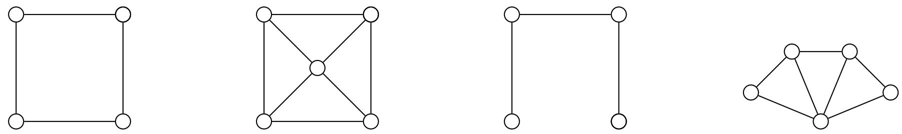


Figure 7.18: Minimal non-trivially perfect graphs with respect to $\leqslant_{\text{ftm}}$

Proof. Threshold graphs are the intersection of split graphs and trivially perfect graphs therefore Forb(threshold graphs) = minimal(Forb(trivially perfect) $\cup$ Forb(split graphs)) with respect to $\leqslant_{\text{ftm}}$ (Theorem 39). □

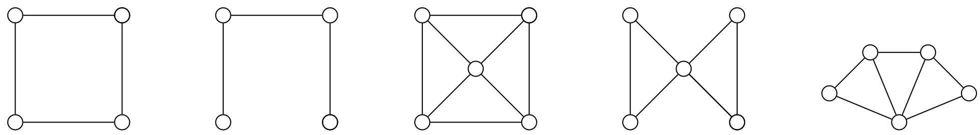


Figure 7.19: Minimal non-threshold graphs with respect to $\leqslant_{\text{ftm}}$

7.3 Containment Complexity

7.3.1 Contractibility

The graph contractibility problem is listed as an $\mathbb{NP}$-complete problem in [64], however the reference provided refers to private communication. The work in [115] also shows that the problem is $\mathbb{NP}$-complete. We present an $\mathbb{NP}$-completeness proof transforming 1-in-3SAT to graph contractibility. The motivation for the $\mathbb{NP}$-completeness proof presented here is that the same construction can be used to show that the false twin containment problem is also $\mathbb{NP}$-complete. The graph contractibility problem is defined as follows, given a graph G and a graph H, is there a sequence of edge contractions in G that yields a graph isomorphic to H. Note that edge contraction is commutative [166].

An instance of 1-in-3SAT consists of a set of variables $X = \{x_1, \ldots, x_n\}$ and a set of clauses $C = \{c_1, \ldots, c_m\}$ in conjunctive normal form each with exactly three literals. An instance of 1-in-3SAT is satisfiable if and only if there exists a truth assignment $\varphi : \{x_1, \ldots, x_n\} \to \{\text{T}, \text{F}\}$ such that exactly one literal in each clause is set to true. The 1-in-3SAT problem is $\mathbb{NP}$-complete [64, LO4].

Theorem 95. *Graph contractibility is* $\mathbb{NP}$*-complete.*

Proof. Firstly we show that graph contractibility is in $\mathbb{NP}$. It is easy to observe that we may guess an H-contraction-witness structure in G. The H-contraction-witness structure can be verified in polynomial time.

We next show that graph contractibility is $\mathbb{NP}$-hard, we prove this by reducing 1-in-3SAT into graph contractibility. Let $X = \{x_1, \ldots, x_n\}$ and $C = \{c_1, \ldots, c_m\}$ be the variables and clauses (respectively) of a 1-in-3SAT instance where each variable appears in at least one clause, let $c_i = \{c_i^0, c_i^1, c_i^2\}$. We construct the graphs $G = (V, E)$ and $H = (U, F)$ as follows (shown in Figure 7.20 and Figure 7.21 respectively);

$$\begin{aligned} V = &\{x_i, x_i^{\text{a}}, \overline{x}_i \mid 1 \leq i \leq n\} \cup \\ &\{d_i^k \mid 1 \leq i \leq m, 1 \leq k \leq 7\} \cup \\ &\{w_i \mid 0 < i \leq 5\} \cup \\ &\{u_i \mid 0 < i \leq 6\} \cup \\ &\{y_i \mid 0 < i \leq 6\} \cup \\ &\{u, v, w, y\} \end{aligned}$$

$$\begin{aligned} E = &\{ux_i, u\overline{x}_i, wx_i, w\overline{x}_i \mid 1 \leq i \leq n\} \cup \\ &\{x_i^{\text{a}}x_i, x_i^{\text{a}}\overline{x}_i \mid 1 \leq i \leq n\} \cup \\ &\{u_iu_{i+1}, w_iw_{i+1} \mid 1 \leq i \leq 4\} \cup \\ &\{y_iy_{i+1} \mid 1 \leq i \leq 3\} \cup \\ &\{uu_1, ww_1, yy_1, u_4u_6, y_3y_5, y_3y_6\} \cup \\ &\{x_i^{\text{a}}y \mid 1 \leq i \leq n\} \cup \\ &\{vd_i^1, vd_i^3, vd_i^5 \mid 1 \leq i \leq m\} \cup \\ &\{d_i^1d_i^2, d_i^2d_i^3, d_i^3d_i^4, d_i^4d_i^5, d_i^5d_i^6, d_i^6d_i^1 \mid 1 \leq i \leq m\} \cup \\ &\{d_i^7d_i^2, d_i^7d_i^4, d_i^7d_i^6 \mid 1 \leq i \leq m\} \cup \\ &\{d_i^{2k+1}c_i^k \mid 1 \leq i \leq m, k \in \{0, 1, 2\}\} \cup \\ &\{d_j^{2k+1}x_i \mid c_j^k = x_i, 1 \leq j \leq m, k \in \{0, 1, 2\}\} \cup \\ &\{d_j^{2k+1}\overline{x_i} \mid c_j^k = \overline{x_i}, 1 \leq j \leq m, k \in \{0, 1, 2\}\} \cup \end{aligned}$$

Each variable in the 1-in-3SAT instance is represented by three vertices $x_i, \overline{x}_i$ denoting the two literals and a marker x_i^{a}. Every clause is represented by a group of seven vertices where each group has four private vertices. Each clause has three distinguished vertices that are all adjacent to a common vertex, these vertices represent the literals in each clause and are adjacent to the corresponding literal vertices, *i.e.*,$x_i, \overline{x}_i$. The vertices w, u and y represent the assignment of false, the assignment of true and a marker to ensure a variable is not removed from the instance respectively. The vertices u_i, y_i and w_k are markers in order to distinguish certain vertices in the graph H where $1 \leq i \leq 6$ and $1 \leq k \leq 5$. For clarity the edges incident to the vertices representing the variable and the vertices representing the literals in the clauses have been omitted in Figure 7.20.

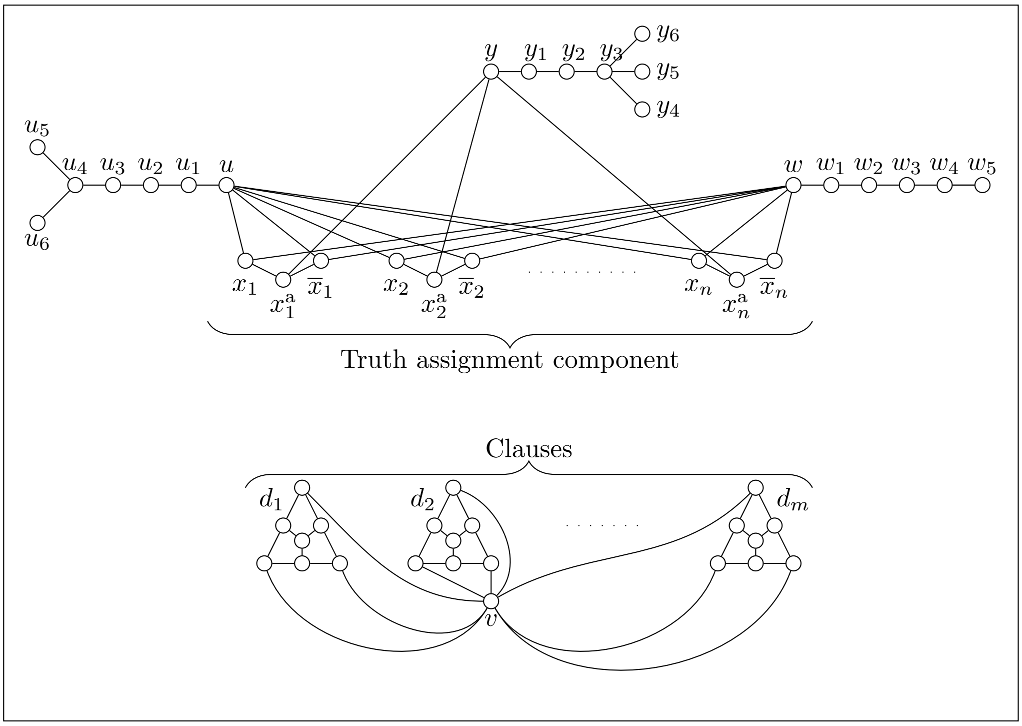


Figure 7.20: Construction of G for $\mathbb{NP}$-completeness proof of the general contractibility problem. Note that the edges between the vertices of d_i and the vertices of $\{x_j, \overline{x}_j, x_j^a\}$ where $0 \leq j \leq n$ and $0 \leq i \leq m$ are omitted for clarity.

$$\begin{aligned}U = &\{a_i \mid 1 \leq i \leq n\} \cup \\ &\{d_i'^k \mid 1 \leq i \leq m, 1 \leq k \leq 7\} \cup \\ &\{w_i' \mid 0 < i \leq 5\} \cup \\ &\{u_i' \mid 0 < i \leq 6\} \cup \\ &\{y_i' \mid 0 < i \leq 6\} \cup \\ &\{u', v', w', y'\}\end{aligned}$$

$$\begin{aligned}F = &\{d_i'^1 u' \mid 1 \leq i \leq m\} \cup \\ &\{d_i'^3 w', d_i'^5 w' \mid 1 \leq i \leq m\} \cup \\ &\{v' d_i'^1, v' d_i'^3, v' d_i'^5 \mid 1 \leq i \leq m\} \cup \\ &\{d_i'^1 d_i'^2, d_i'^2 d_i'^3, d_i'^3 d_i'^4, d_i'^4 d_i'^5, d_i'^5 d_i'^6, d_i'^6 d_i'^1 \mid 1 \leq i \leq m\} \cup \\ &\{d_i'^7 d_i'^2, d_i'^7 d_i'^4, d_i'^7 d_i'^6 \mid 1 \leq i \leq m\} \cup \\ &\{a_i u', a_i w' \mid 1 \leq i \leq n\} \cup \\ &\{a_i y' \mid 1 \leq i \leq n\} \cup \\ &\{u_i' u_{i+1}', w_i' w_{i+1}' \mid 1 \leq i \leq 4\} \cup \\ &\{y_i' y_{i+1}' \mid 1 \leq i \leq 3\} \cup \\ &\{u' u_1', w' w_1', y' y_1', u_4' u_6', y_3' y_5', y_3' y_6'\} \cup \\ &\{u' w'\}\end{aligned}$$

Figure 7.21 depicts the construction of H, each clause clearly has two vertices adjacent to w' and one vertex adjacent to u'. The clauses are labelled d_i', as in the figure, and refer to the group of seven vertices. We show that $H \leqslant_c G$ if and only if there exists a satisfying truth assignment. First assume there is a satisfying truth assignment $\varphi : X \to \{\mathrm{T}, \mathrm{F}\}$. Contract the edges in G according to the following rule; if $\varphi(x_i) = \mathrm{T}$ then contract the edges $\overline{x}_i w$ and $x_i u$ else contract the edges $x_i w$ and $\overline{x}_i u$. This shows that $H \leqslant_c G$.

Now assume that H is a contraction of G. Observe that by contracting edges the number of vertices and edges only reduces and that $|E(G)| - |E(H)| = 4n$ and $|V(G)| - |V(H)| = 2n$. As H is a contraction of G there must exist a H-contraction-witness structure. Let W denote a function that defines the H-contraction-witness structure.

Observation 96. Let G be a graph contractible to H with H-contraction-witness structure W. If $a, b \in V(G)$ and $c, d \in V(H)$ where $a \in W(c)$, $b \in W(d)$ then $\mathrm{dist}_G(a, b) \geq \mathrm{dist}_H(c, d)$.

Proof. The shortest path between a and b must pass through at least as many witness sets as there are vertices on the shortest path between c and d. □

Using the observation above it can easily be seen from the construction of G that $W(w_5') = \{w_5\}$, $W(u_5') \cup W(u_6') = \{u_5, u_6\}$ and $W(y_4') \cup W(y_5') \cup W(y_6') = \{y_4, y_5, y_6\}$. As $W(z) \neq \varnothing$ for all $z \in V(H)$ then $|W(z)| = 1$ for all $z \in \{w_5', u_5', u_6', y_4', y_5', y_6'\}$. Observe that u_5, u_6 and

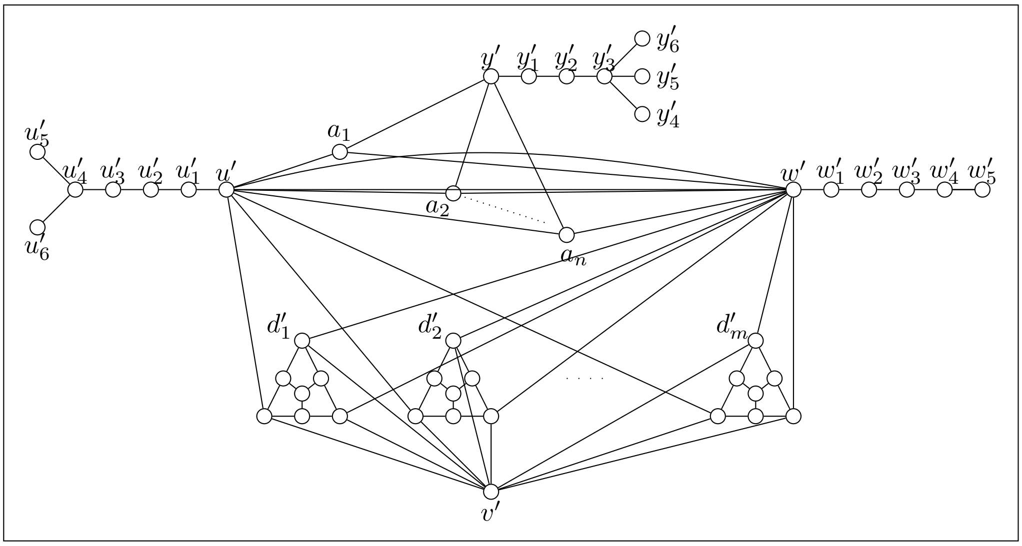


Figure 7.21: Construction of H

y_4, y_5, y_6 are indistinguishable (respectively) up to isomorphism. Without loss of generality assume $W(w'_5) = w_5$, $W(u'_5) = u_5$, $W(u'_6) = u_6$, $W(y'_4) = y_4$, $W(y'_5) = y_5$ and $W(y'_6) = y_6$. It follows that; $w_i \in W(w'_i)$, $w \in W(w')$, $u_j \in W(u'_j)$, $u \in W(u')$, $y_j \in W(y'_j)$ and $y \in W(y')$ where $1 \leq i \leq 5$, $1 \leq j \leq 6$. Next let us define a function $\varphi' : X \to \{\mathrm{T}, \mathrm{F}\}$ as follows;

$$\varphi'(x_i) = \begin{cases} \mathrm{F} & \text{if } x_i \in W(w') \\ \mathrm{T} & \text{otherwise} \end{cases} \qquad 0 \leq i \leq n$$

Observe that, for all x_i, $\varphi'(x_i)$ is well defined because either $x_i \in W(w')$ or $x_i \notin W(w')$. It remains to show that this assignment is a satisfying truth assignment.

We demonstrate a bijection between the vertices d_i^n and $d_i'^n$ and hence $d_i'^n$ represent clauses from the 1-in-3SAT instance. Let us show that $W(y') = \{y\}$. If this is not the case then either $y_1 \in W(y')$ or there exists an i such that $x_i^{\mathrm{a}} \in W(y')$ where $1 \leq i \leq n$. Suppose the former then W is not a valid H-contraction-witness structure as the image of W is a partition of $V(G)$ and it has already been established that $y_1 \in W(y'_1)$. Now suppose the latter case, $x_i^{\mathrm{a}} \in W(y')$, then there must be two vertices $l, k \in V(H)$ such that $x_i \in W(l)$ and $\overline{x}_i \in W(k)$ and both $W(l), W(k)$ touch $W(y')$. From the assumption that each variable appears in some clause and from the construction of G then there is a vertex $a \in \{d_i^n \mid 1 \leq i \leq m, 1 \leq n \leq n\}$ that is adjacent to x_i or $\overline{x}_i$. Without loss of generality assume a is adjacent to x_i, as G is contractible to H then x_i must be in $W(w')$ or $W(u')$ either leads to a contradiction that W is an H-contraction-witness structure as $W(y')$ does not touch either $W(w')$ or $W(u')$. The conclusion is that $W(y') = \{y\}$. With that in mind it follows, and without loss of generality, that $x_i^{\mathrm{a}} \in W(a_i)$ where $1 \leq i \leq n$. As $a_i w', a_i u' \in E(H)$ then $W(a_i)$ must touch $W(w')$ and

$W(u')$. It takes at least two edge contractions for each witness set to touch, totalling at least $2n$ edge contractions. From the construction of G and H, we may contract at most $2n$ edges therefore the edge contractions required to make $W(a_i)$ touch $W(w')$ and $W(u')$ exhausts the available edge contractions implying the remainder of the witness sets must be singletons and hence represent a bijection. Without loss of generality assume $v \in W(v')$, $d_i^n \in W(d_i'^n)$ for $1 \leq i \leq m$ and $1 \leq n \leq 7$. We refer to the set $\{d_i'^n \mid 1 \leq n \leq 6\}$ for some fixed i as a clause.

We next show that $|W(w') \cap \{x_i, \overline{x}_i\}| \leq 1$. Assume $|W(w') \cap \{x_i, \overline{x}_i\}| \not\leq 1$ then this implies that $|W(w') \cap \{x_i, \overline{x}_i\}| = 2$ therefore $\{x_i, \overline{x}_i\} \subseteq W(w')$. There are two cases to consider, either $x_i^{\mathrm{a}} \in W(w')$ in which case $W(y')$ touches $W(w')$ which is a contradiction that W is an H-witness structure or $x_i^{\mathrm{a}} \notin W(w')$ then there exists a path of length three between $W(w')$ and $W(y')$ with a vertex not adjacent to $W(u')$ which is a contradiction that W is an H-contraction-witness structure. Therefore $|W(w') \cap \{x_i, \overline{x}_i\}| \leq 1$, the same argument can be applied to show $|W(u') \cap \{x_i, \overline{x}_i\}| \leq 1$.

Each clause has two vertices adjacent to w' and one vertex adjacent to u', as each vertex adjacent to w' has been set to false and a variable and its complement cannot both be in $W(w')$ it follows that each clause has exactly two literals set to false and one literal set to true therefore φ' is a satisfying truth assignment.

□

7.3.2 False twin minors

Theorem 97. *Given H and G, determining if $H \leqslant_{\mathrm{ftm}} G$ is $\mathbb{NP}$-complete.*

Proof. Observe that determining if $H \leqslant_{\mathrm{ftm}} G$ is in $\mathbb{NP}$. If $H \leqslant_{\mathrm{ftm}} G$ then by Theorem 86 there is a false twin witness structure. We may verify the false twin witness structure in polynomial time. We next show that $H \leqslant_{\mathrm{ftm}} G$ is $\mathbb{NP}$-hard, by reducing from 1-in-3SAT. Let $X = \{x_1, \ldots, x_n\}$ and $C = \{c_1, \ldots, c_m\}$ be an instance of 1-in-3SAT where each variable appears in at least one clause. Construct the graphs $G = (V, E)$ and $H = (U, F)$ as in Theorem 95.

We show that $H \leqslant_{\mathrm{ftm}} G$ if and only if there is a satisfying truth assignment. Assume there is a satisfying truth assignment $\varphi : X \to \{\mathrm{T}, \mathrm{F}\}$. Contract the edges of G according to the following rule; if $\varphi(x_i) = \mathrm{T}$ then contract the edges $\overline{x}_i w$ and $x_i u$ else contract the edges $\overline{x}_i u$ and $x_i w$. This demonstrates that $H \leqslant_{\mathrm{ftm}} G$.

Now assume that $H \leqslant_{\mathrm{ftm}} G$, then from Theorem 86 there exists a false twin minor witness structure (H', U) such that $H' \leqslant_{\mathrm{c}} G$ and $H \leqslant^* H'$ where $\leqslant^*$ denotes the removal of false twins. Let W denote the H'-contraction witness structure in G. Using Observation 96 it is easily seen from the construction of G that $W(w_5') = \{w_5\}$, $W(u_5') \cup W(u_6') = \{u_5, u_6\}$ and $W(y_4') \cup W(y_5') \cup W(y_6') = \{y_4, y_5, y_6\}$. Observe that u_5, u_6 and y_4, y_5, y_6 are indistinguishable (respectively) up to isomorphism. Without loss of generality assume $W(w_5') = w_5$, $W(u_5') = u_5$, $W(u_6') = u_6$, $W(y_4') = y_4$, $W(y_5') = y_5$ and $W(y_6') = y_6$. It follows that; $w_i \in W(w_i')$, $w \in W(w')$, $u_j \in W(u_j')$, $u \in W(u')$, $y_j \in W(y_j')$ and $y \in W(y')$ where $1 \leq i \leq 5$, $1 \leq j \leq 6$.

Let us show that $W(y') = \{y\}$. If this is not the case then either $y_1 \in W(y')$ or there exists an i such that $x_i^{\mathrm{a}} \in W(y')$ where $0 \leq i \leq n$. Suppose the former then W is not a valid H'-contraction-witness structure as the image of W is a partition of $V(G)$ and it has already been established that $y_1 \in W(y_1')$. Now suppose the latter case, $x_i^{\mathrm{a}} \in W(y')$, then there must be two vertices $l, k \in V(H')$ such that $x_i \in W(l)$ and $\overline{x}_i \in W(k)$ and both $W(l), W(k)$ touch $W(y')$. From the assumption that each variable appears in some clause and from the construction of G then there is a vertex $a \in \{d_i^n \mid 1 \leq i \leq m, 1 \leq n \leq n\}$ that is adjacent to x_i or $\overline{x}_i$. Without loss of generality assume a is adjacent to x_i, as G is contractible to H' then x_i must be in $W(w')$ or $W(u')$ either leads to a contradiction that W is an H'-contraction-witness structure as $W(y')$ does not touch either $W(w')$ or $W(u')$. The conclusion is that $W(y') = \{y\}$. With that in mind it follows without loss of generality that $x_i^{\mathrm{a}} \in W(a_i)$. As $a_i w', a_i u' \in E(H)$ then $W(a_i)$ must touch $W(w')$ and $W(u')$. It takes at least two edge contractions for each set $W(a_i)$ to touch $W(w')$ and $W(u')$, this is a total of $2n$ edge contractions. Observe that $|V(G)| - |V(H)| = 2n$. Each edge contraction or removal of a false twin reduces the number of vertices by one therefore there are at most $2n$ modifications available. Consequently the edge contractions required to make $W(a_i)$ touch $W(w'), W(u')$ exhausts the available modifications. This leads to the conclusion that $U = \varnothing$ and therefore $H = H'$. This implies that the remainder of the H'-contraction witness structure must be singletons and hence represent a bijection. Without loss of generality assume $v \in W(v')$, $d_i^n \in W(d_i'^n)$ for $1 \leq i \leq m$ and $1 \leq n \leq 7$. We refer to the set $\{d_i'^n \mid 1 \leq n \leq 6\}$ for some fixed i as a clause. Let us define a function $\varphi' : X \to \{\mathrm{T}, \mathrm{F}\}$ as follows;

$$\varphi'(x_i) = \begin{cases} \mathrm{F} & \text{if } x_i \in W(w') \\ \mathrm{T} & \text{otherwise} \end{cases} \qquad 0 \leq i \leq n$$

We show that this function is a satisfying truth assignment, to do this we show that at most one of x_i or $\overline{x}_i$ is in $W(w')$. On the contrary assume $\{x_i, \overline{x}_i\} \subseteq W(w')$. There are two cases to consider, either $x_i^{\mathrm{a}} \in W(w')$ in which case $W(y')$ touches $W(w')$ which is a contradiction that W is an H'-witness structure or $x_i^{\mathrm{a}} \notin W(w')$ then there exists a path of length three between $W(w')$ and $W(y')$ with a vertex not adjacent to $W(u')$ which is a contradiction that W is an H'-contraction-witness structure. Therefore $|W(w') \cap \{x_i, \overline{x}_i\}| \leq 1$, the same argument can be applied to show $|W(u') \cap \{x_i, \overline{x}_i\}| \leq 1$.

Each clause has two vertices adjacent to w' and one vertex adjacent to u'. As each vertex adjacent to w' has been set to false by the function φ' and by the assertion that $|W(w') \cap \{x_i, \overline{x}_i\}| = 1$ then it follows that each clause has exactly two literals set to false and one literal set to true therefore φ' is a satisfying truth assignment.

□

7.4 Edge contraction and well-quasi ordering

Recall that $\mathcal{G}$ is well-quasi ordered by $\leqslant$ if $\mathcal{G}$ does not contain an infinite antichain or an infinite strictly descending chain. Since we define orders on all graphs, we should refer to orders on special classes as restrictions of orders. If a partial order has the property outlined in Equation (7.1) then $\mathcal{G}$ cannot contain an infinite strictly descending chain, *i.e.*, $\leqslant$ is well-founded on the set $\mathcal{G}$. Therefore to prove that a partial order is well-quasi ordered it is sufficient to show that the partial order satisfies the property outlined in Equation (7.1) and that the there exists no infinite antichains. For the minor and induced subgraph relation on the set of all finite unlabelled graphs the minimal element is K_0, the *null* graph, which is also the minimum element. For the contraction minor partial order the minimal elements are kK_1 for $k \geq 0$.

$$G \leqslant H \implies |G| \leq |H| \tag{7.1}$$

There exist classes of graphs that are well-quasi ordered with respect to the induced subgraph and partial subgraph relations [34, 39]. It is interesting to consider the classes of graphs that are well-quasi ordered by some partial order and further it is interesting to consider the well-quasi ordered classes that can be characterised by a finite forbidden set.

The works of Damaschke [34] and Ding [39] show necessary conditions for a class to be well-quasi ordered with respect to induced subgraphs and partial subgraphs respectively. We show that similar results for the contraction minor relation are not possible, that is, there are no well-quasi ordered classes that are characterised by a finite forbidden set with respect to the contraction minor relation. Further to this we show a general property of a partial order such that the property excludes the possibility of well-quasi ordered classes being characterised by a finite forbidden set.

Observe that with respect to the contraction minor relation the number of components is an invariant, *i.e.*, $\forall H, G \in \mathcal{G}$ $(H \leqslant_{\mathrm{c}} G)$ implies $C(H) = C(G)$ (where $C(G)$ denotes the number of connected components in G). Therefore the number of minimal elements is infinite with respect to the contraction minor relation, when the ground set is the set of all graphs. The minimal elements are kK_1 for $k \geq 0$. This demonstrates that the set of all graphs is not well-quasi ordered with respect to contraction minors as the minimal elements form an antichain.

Theorem 98. *For any class $\mathcal{C} \subseteq \mathcal{G}$ that is closed and well-quasi ordered with respect to $\leqslant_{\mathrm{c}}$, the set of minimal forbidden graphs is infinite.*

Proof. From the assertion that $\mathcal{C}$ is well-quasi ordered then $\mathcal{C}$ contains no infinite antichains. With respect to the contraction minor relation the set of graphs with differing numbers of components forms an antichain, therefore the class $\mathcal{C}$ can only contain a finite number of graphs with a differing number of components. This observation leads to the conclusion that the set of forbidden graphs must contain an infinite set of graphs with a differing number of components. □

Remark 99. The converse of Theorem 98 is also true, if a class $\mathcal{C}$ is closed with respect to the contraction minor relation and has a finite set of minimal forbidden graphs then $\mathcal{C}$ is not well-quasi ordered with respect to $\leqslant_c$.

This idea may be expressed more generally in terms of invariants with respect to a partial order. A parameter p is invariant with respect to a partial order if $G \leqslant H$ implies $p(G) = p(H)$.

Theorem 100. *Let p be a parameter of a graph such that p is a surjective function $p : \mathcal{G} \to \mathbb{N}$ and p is an invariant with respect to a partial order $\leqslant$ then $\mathcal{G}$ has an infinite number of minimal elements with respect to $\leqslant$.*

Proof. The invariance of p means that two graphs G and H are comparable only if $p(G) = p(H)$. This equivalence relation partitions the set $\mathcal{G}$ into an infinite number of equivalence classes. The minimal elements of each equivalence class form the minimal elements of $\mathcal{G}$ with respect to $\leqslant$, therefore as there are an infinite number of equivalence classes the number of minimal elements is also infinite. □

Theorem 101. *Let p be a parameter of a graph that is a surjective function $p : \mathcal{G} \to \mathbb{N}$ and p is an invariant with respect to a partial order $\leqslant$ then any well-quasi ordered class has an infinite minimal forbidden set.*

Proof. Let $\mathcal{C}$ be a well-quasi ordered class with respect to $\leqslant$. Two graphs G and H are equivalent if and only if $p(G) = p(H)$. This equivalence relation partitions the set $\mathcal{G}$ into an infinite number of equivalence classes, consequently there is an infinite set of minimal elements in $\mathcal{G}$ with respect to $\leqslant$. Any two elements from different equivalent classes are incomparable. The class $\mathcal{C}$ can only contain elements from a finite number of equivalence classes, otherwise $\mathcal{C}$ would contain an infinite antichain. Therefore as $\mathcal{C}$ is well-quasi ordered the forbidden set for $\mathcal{C}$ must contain an infinite number of elements. □

7.5 Summary

In this chapter we have given a set of alternative characterisations for a number of subclasses of perfect graphs. We have demonstrated that with respect to the contraction minor relation the classes of chordal graphs, split graphs, threshold graphs and trivially perfect graphs are closed and we have provided a description of the minimal forbidden set for each of these classes. We have introduced the false twin minor relation which is closely related to the contraction minor and induced subgraphs relations. We have motivated the definition of this partial order by demonstrating that a number of well-studied graph classes are closed with respect to it and moreover have a finite minimal forbidden set. This has particular importance when charactering the classes $\mathcal{C}{+}k\mathsf{v}$, $\mathcal{C}{+}k\mathsf{e}$ and $\mathcal{C}{-}k\mathsf{e}$, as it is a requirement that the base class should have a finite characterisation if the parameterized classes are to be characterised by a finite minimal forbidden set. The introduction of the lattice in Chapter 4 is motivated by showing that there are many interesting partial orders, other than those currently defined in the literature, which are useful in providing finite characterisations of established graph classes. We have provided an alternative $\mathbb{NP}$-completeness proof for the CONTRACTIBILITY problem than that presented in [115]. The $\mathbb{NP}$-completeness proof is then modified to provide show that the FALSE-TWIN-MINOR problem is $\mathbb{NP}$-complete. In addition an observation is formalised regarding necessary requirements for a class to be well-quasi ordered with respect to the contraction minor relation.

Chapter 8

Topological Minors

The topological minor relation is a member of the lattice structure defined in Chapter 4. The partial order is a restriction of the minor partial order and an extension of the partial subgraph partial order, *i.e.*, $\leqslant_{\mathrm{s}} \subset \leqslant_{\mathrm{t}} \subset \leqslant_{\mathrm{m}}$. The partial order is interesting as it is the last bastion of the problem which has resisted efforts to provide a proof or a counterexample as to whether the class $\mathcal{C}+k\mathsf{v}$ is characterised by a finite forbidden set with respect to the topological minor relation if the class $\mathcal{C}$ is. Although we do not have a counterexample nor a proof that covers all cases we have a collection of results which offer promising glimmer of hope that suggest the statement is true. We are inclined to believe that the statement is correct and that for any graph class $\mathcal{C}$ closed with respect to the topological minor relation and has a finite minimal forbidden set then the class $\mathcal{C}+k\mathsf{v}$ also have a finite minimal forbidden set.

The topological minor order is not a bounded expansion partial order. This can be seen by observing that $C_4 \leqslant_{\mathrm{t}} C_k$ where $k \geq 4$ (note that C_k has unbounded size) and no proper subgraph of C_k is a topological minor of C_4 (an alternative example can be seen in Figure 4.3 on page 73). A consequence of the topological minor not having the bounded expansion property is that we are unable to apply the bound on the maximum order of a minimal forbidden graph established in Chapter 5. The inability to apply the same technique as for partial orders that have the bounded expansion property is due to the the construction of the hypergraph. The bound relies on being able to construct a uniform critical hypergraph, this is not possible for the topological minor relation using the established technique. Attempting to construct a bound using the techniques in Chapter 5 results in a consistent statement that states the maximum order of a minimal forbidden graph is greater than or equal to the maximum order of a minimal forbidden graph.

Unlike the minor relation, the topological minor relation is not a well-quasi ordering on the set of all graphs and therefore the meta-theorems that are applied to characterising graph classes closed with respect to the minor relation do not apply. Because the relation is not a well-quasi ordering there exist graph classes closed with respect to the topological minor relation that have infinite minimal forbidden sets, for example the class that forbids the antichain shown in Table 3.2 on page 43.

Let σ be a function between a well-quasi ordered set L of labels and the vertices of a graph G, *i.e.*, $\sigma : L \to V(G)$. If G is a topological minor of H and for all vertices in G we have $\sigma_G(v) \leqslant \sigma_H(f(v))$ where f is the embedding of G in H then we say that $G \leqslant_t^\sigma H$. We say that there is a label order preserving topological embedding of G in H. It has been shown by Fellows *et al.* in [49] that if for every labelling σ a graph class $\mathcal{C}$ is well-quasi ordered with respect to $\leqslant_t^\sigma$ then the class $\mathcal{C}{+}k\mathsf{v}$ is well-quasi ordered by $\leqslant_t$. Consequently $\mathcal{C}{+}k\mathsf{e}$ and $\mathcal{C}{-}k\mathsf{e}$ are well-quasi ordered as well, on the account that $\mathcal{C}{+}k\mathsf{e} \subseteq \mathcal{C}{+}k\mathsf{v}$ and $\mathcal{C}{-}k\mathsf{e} \subseteq \mathcal{C}{+}k\mathsf{v}$. An implication of the result of Fellows, although unmentioned in the publication, is that every class $\mathcal{C}$ that is well-quasi ordered for any labelling σ and has a finite minimal forbidden set with respect to the topological minor relation then the class $\mathcal{C}{+}k\mathsf{v}$ must have a finite minimal forbidden set. The implication follows from the result of Fellows, due to the minimality of a minimal forbidden set the elements of the minimal forbidden set for the class $\mathcal{C}$ are members of the class $\mathcal{C}{+}1\mathsf{v}$, if we assume that $\mathcal{C}$ is well-quasi ordered then $\mathcal{C}{+}1\mathsf{v}$ is well-quasi ordered and hence $\mathrm{Forb}(\mathcal{C})$ is an antichain in a well-quasi ordered class and therefore must be finite. To highlight the reverse of this implication does not hold, not every graph class characterised by a finite minimal forbidden set with respect to the topological minor relation is well-quasi ordered. If this implication were true then it would be necessary that the topological minor relation was a well-quasi ordering on the set of all graphs which has been shown not to be the case.

The topological minor relation is sandwiched between the partial subgraph relation and the minor relation in the lattice of partial orders defined in Chapter 4. The characterisation of the classes $\mathcal{C}{+}k\mathsf{v}$, $\mathcal{C}{+}k\mathsf{e}$ and $\mathcal{C}{-}k\mathsf{e}$ is well known for the minor relation due the graph minor theorem and the characterisation of those classes has been resolved with respect to any partial order that has the bounded expansion property, including the partial subgraph relation. This leaves a small number of partial orders that where defined in Chapter 2 where the characterisation of the parameterized graph classes remains an open problem. The topological minor relation is of particular interest because of the impact such a result would have. It has been shown in [75] that the topological minor containment problem is fixed-parameter tractable, like the minor relation, and therefore given a finite characterisation of a graph class closed with respect to the topological minor relation the class can be recognised in polynomial time. The combination of the containment complexity result and the result for characterising the classes with respect to a finite set would yield a polynomial time algorithm for recognising each parameterized graph class and hence solve the vertex deletion problem for a large set of graph classes. Because of its applications we consider the class $\mathcal{C}{+}k\mathsf{v}$ in the remainder of this chapter.

The general case has eluded a complete characterisation. Instead there are a number of special cases which can be handled by a set of different techniques. Each technique is limited by a different factor. The techniques that are used for the special cases can be categorised into three distinct categories:

– the coincidence of the class with an already characterised class,

– the class under consideration is well-quasi ordered (by a label order preserving topological embedding),

– the class under consideration has a single minimal forbidden graph.

The first consideration is if the class $\mathcal{C}+k\mathsf{v}$ is closed with respect to the topological minor relation. It is easily observed that if $\mathcal{C}$ is closed with respect to the topological minor relation then $\mathcal{C}+k\mathsf{v}$ is also closed (see Theorem 52).

8.1 Coincidence with an alternatively characterised graph class

For some graph classes there exist many alternative characterisations, an example of this is the class of graphs of bounded treewidth. The class is closed with respect to all of the partial orders given in Figure 4.2 on page 72, this is easily confirmed by observing that the class of bounded treewidth graphs is closed with respect to the minor relation and that all partial orders in Figure 4.2 are in the ideal of the minor relation. This is equivalent to the statement in Theorem 34 on page 68. As such, the class of graphs of bounded treewidth have a characterisation with respect to each partial order in the ideal of the minor relation; however, these characterisations may not be finite.

With respect to the topological minor relation there are a number of classes which coincide with alternatively characterised graph classes. The alternative characterisations can often imply a finite characterisation with respect to the topological minor relation. Consider a graph class $\mathcal{C}$ closed with respect to the minor relation, the class is also closed with respect to the topological minor relation. The class $\mathcal{C}+k\mathsf{v}$ is also closed with respect to the minor and topological minor relations. From the graph minor theorem it is known that the minimal forbidden set for the class $\mathcal{C}+k\mathsf{v}$ is finite. From this finite minimal set for the class $\mathcal{C}+k\mathsf{v}$ it is possible to construct the set of minimal forbidden graphs with respect to the topological minor relation. The construction involves replacing every vertex of degree k where $k > 3$ with every tree of maximum degree 3 with k leaves.

Although the technique outlined above produces a finite minimal forbidden set it relies on the ability to compute the minimal forbidden set with respect to the minor relation. This has been shown to be computable for the class $\mathcal{C}+k\mathsf{v}$ but no bound on the maximum order of a minimal forbidden graph is given. This construction can be considered an existential proof of a minimal forbidden set but does not provide a mechanism to construct the set. This drawback limits its applications for graph class recognition.

Where the class $\mathcal{C}$ is closed with respect to the topological minor relation and $\mathcal{C}$ has a finite characterisation with respect to the induced subgraph relation then it is possible to use the bound given in Chapter 5. The algorithms given in Chapter 6 can be used to generate the minimal forbidden set for the class $\mathcal{C}+k\mathsf{v}$ with respect to the induced subgraph relation. As the induced subgraph relation is in the ideal of the topological minor relation then it is possible to apply Corollary 36 to obtain the result;

$$\mathrm{Forb}(\mathcal{C})_{\mathrm{t}} = \mathrm{minimal}(\mathrm{Forb}(\mathcal{C})_{\mathrm{i}})_{\mathrm{t}}.$$

As $\mathrm{Forb}(\mathcal{C})_{\mathrm{i}}$ is finite it must be that $\mathrm{Forb}(\mathcal{C})_{\mathrm{t}}$ is finite because $\mathrm{Forb}(\mathcal{C})_{\mathrm{t}}$ contains the minimal elements of $\mathrm{Forb}(\mathcal{C})_{\mathrm{i}}$. This technique not only provides a proof of a finite minimal forbidden set but also provides a mechanism for its construction. This technique may be applied generally to any partial order that has the bounded expansion property and is in the ideal of the topological minor relation.

However the application of this technique is limited to the coincidence of the graph classes closed with respect to the topological minor relation and a partial order with the bounded expansion property such that the minimal forbidden set with respect to the partial order with the bounded expansion property is finite. Some examples of such coincidences are the following classes;

Lemma 102. *For all $n \geq 0$ the class $\{K_{1,n}\}$-free$_{\mathrm{s}}$ and $\{K_{1,n}\}$-free$_{\mathrm{t}}$ coincide, i.e., $\{K_{1,n}\}$-free$_{\mathrm{s}}$ = $\{K_{1,n}\}$-free$_{\mathrm{t}}$.*

Proof. To prove the statement we show that (1) $G \leqslant_{\mathrm{s}} H$ implies $G \leqslant_{\mathrm{t}} H$ and (2) $K_{1,n} \leqslant_{\mathrm{t}} H$ implies $K_{1,n} \leqslant_{\mathrm{s}} H$. (1) As $\leqslant_{\mathrm{s}} \subseteq \leqslant_{\mathrm{t}}$ we have that $G \leqslant_{\mathrm{s}} H$ implies $G \leqslant_{\mathrm{t}} H$. (2) Observe that the class of bounded degree graphs is closed with respect to the topological minor relation. Let H be a graph containing $K_{1,n}$ with respect to $\leqslant_{\mathrm{t}}$ then there exists a vertex v of degree at least n in H. The graph induced by the closed neighbourhood of v is a subgraph of K_{n+1} with at least one vertex of degree greater than or equal to n. Therefore H contains a $K_{1,n}$ as a partial subgraph. □

Lemma 103. *For all $n \geq 0$ the class $\{P_n\}$-free$_{\mathrm{s}}$ and $\{P_n\}$-free$_{\mathrm{t}}$ coincide, i.e., $\{P_n\}$-free$_{\mathrm{s}}$ = $\{P_n\}$-free$_{\mathrm{t}}$.*

Proof. To prove the statement we show that (1) $(G, H) \in \leqslant_{\mathrm{s}}$ implies $(G, H) \in \leqslant_{\mathrm{t}}$ and (2) $(P_n, H) \in \leqslant_{\mathrm{t}} \implies (P_n, H) \in \leqslant_{\mathrm{s}}$. (1) As $\leqslant_{\mathrm{s}} \subseteq \leqslant_{\mathrm{t}}$ we have that $(G, H) \in \leqslant_{\mathrm{s}} \implies (G, H) \in \leqslant_{\mathrm{t}}$. (2) Let H be a graph that contains a P_n with respect to $\leqslant_{\mathrm{t}}$ then there exists an alternating sequence of vertices and edges $v_0, e_0, v_1, \ldots, e_i, v_{i+1}$ where $i \geq n$. This sequence contains a P_n as a partial subgraph. □

As is evident from the two examples above the structure of the forbidden graphs where the two classes coincide is fairly restrictive. The technique in Section 8.3 provides a slight generalisation, however, it is limited by other factors.

8.2 $\mathcal{C}$ is well-quasi ordered by $\leqslant_{\mathrm{t}}$

Well-quasi orderings have had an important role in characterising graph classes in the past. For the topological minor relation it was shown in [49] that if a class $\mathcal{C}$ is well-quasi ordered for every label assignment function σ then the class $\mathcal{C}+k\mathsf{v}$ is also well-quasi ordered. The implication of this result is that each class $\mathcal{C}+k\mathsf{v}$ the class can be characterised by a finite minimal forbidden set.

When this is considered with the knowledge that the containment problem for the topological minor relation is fixed-parameter tractable results in a polynomial time algorithm to recognise each class $\mathcal{C}+k\mathsf{v}$ where $\mathcal{C}$ is well-quasi ordered by every labelling with respect to the topological minor relation. The polynomial time algorithm is a direct application of Algorithm 4 on page 78. The limiting factor of this technique is the requisite that the class $\mathcal{C}$ is well-quasi ordered for every labelling. Although there are some well studied graph classes that are well-quasi ordered with respect to the topological minor relation it is not generally the case that all classes closed with respect to the topological minor relation are well-quasi ordered. Classes where this technique can be applied are some natural subclasses of bounded treewidth graphs. Such natural classes include graphs of bounded feedback vertex set, that is, the class of graphs where there exists a set of at most k vertices whose removal yields a forest. The class of bounded feedback vertex set graphs have become a topic of study in the field of fixed-parameter tractable problems as many problems that do not admit an algorithm when parameterized by treewidth admit a solution when parameterized by feedback vertex set.

8.3 $\mathcal{C}$ has a single minimal forbidden graph

This section relates to those graph classes, $\mathcal{C}$, that are closed with respect to the topological minor relation and have a single minimal forbidden topological minor, *i.e.*, $\mathcal{C} = \{H\}\text{-free}_{\mathrm{t}}$. We provide a proof for a bound on the size of the forbidden graphs of the class $\mathcal{C}+k\mathsf{v}$ where $\mathcal{C}$ has a single forbidden graph with respect to the topological minor relation. Recall that a graph H is a topological minor of G, denoted $H \leqslant_{\mathrm{t}} G$, if a subdivision of H is isomorphic to a partial subgraph of G. Recall that $\mathcal{C}+k\mathsf{v}$ denotes the class of graphs

$$\{G \mid \exists U \subseteq V(G)\ (|U| \leq k \wedge (G - U) \in \mathcal{C})\}.$$

Theorem 104. *For any graph class $\mathcal{C}$ where the class $\mathcal{C}$ has a single forbidden graph with respect to the topological minor relation the class $\mathcal{C}+k\mathsf{v}$ has a finite number of forbidden graphs with respect to the topological minor relation.*

Proof. Let $\mathcal{C}$ be a graph class closed with respect to the topological minor relation and $\mathrm{Forb}(\mathcal{C})_{\mathrm{t}} = \{L\}$. Observe the class $\mathcal{C}$ is closed with respect to the partial subgraph relation. Consequently there exists a set of graphs $\mathcal{F}$ such that $\mathcal{C} = \mathcal{F}\text{-free}_{\mathrm{s}}$. Let the graphs in $\mathcal{F}$ be minimal with this property. Let $\mathcal{F} = \{F_0, \ldots\}$ and let $L = F_0$. If $\mathcal{F}$ is finite then;

$$\mathrm{Forb}(\mathcal{C}{+}k\mathsf{v})_{\mathrm{t}} = \mathrm{minimal}(\mathrm{Forb}(\mathcal{C}{+}k\mathsf{v})_{\mathrm{s}})_{\mathrm{t}}.$$

From the argument in Chapter 5 the set $\mathrm{Forb}(\mathcal{C}{+}k\mathsf{v})_{\mathrm{s}}$ is finite if $\mathrm{Forb}(\mathcal{C})_{\mathrm{s}}$ is finite. From the assumption that $\mathcal{F}$ is finite then $\mathrm{Forb}(\mathcal{C}{+}k\mathsf{v})_{\mathrm{s}}$ and consequently $\mathrm{Forb}(\mathcal{C}{+}k\mathsf{v})_{\mathrm{t}}$ are finite. It is equivalent to state that a class $\mathcal{C}$ is $\mathcal{F}$-free and a class $\mathcal{C}$ is the intersection of the set of classes that forbid an element of $\mathcal{F}$. We formalise this concept in Claim 105.

Claim 105. Let $\mathcal{C} = \{H_0, \ldots, H_k\}$-free then $\mathcal{C} = \bigcap_{i\geq 0}^{k}\{H_i\}$-free.

Proof. Let $G \notin \mathcal{C}$ then there exists a graph H_i such that $H_i \leqslant G$ therefore $G \notin \{H_i\}$-free. Consequently G can not be in $\bigcap_{i\geq 0}^{k}\{H_i\}$-free. In the opposite direction let $G \notin \bigcap_{i\geq 0}^{k}\{H_i\}$-free then there exists an index i such that $G \notin \{H_i\}$-free therefore $H_i \leqslant G$. This implies that $G \notin \mathcal{C}$ as G contains one of the forbidden graphs. ■

The remaining case is when $\mathcal{F}$ is infinite. Clearly for all $i \geq 0$ we have $F_i \notin \mathcal{C}$ and therefore we have $L \leqslant_{\mathrm{t}} F_i$ for all $i \geq 0$. Let $\mathcal{C}_i = \{F_i\}\text{-free}_{\mathrm{s}}$ for all $i \geq 0$ then

$$\mathcal{C} \;= \bigcap_{i\geq 0} \mathcal{C}_i \;= \bigcap_{i\geq 0}\{F_i\}\text{-free}_{\mathrm{s}} \;= \{L\}\text{-free}_{\mathrm{t}}.$$

There is an inclusion relation between $\mathcal{C}{+}k\mathsf{v}$ and the intersection of each $\mathcal{C}_i{+}k\mathsf{v}$, *i.e.*,

$$\mathcal{C}{+}k\mathsf{v} \subseteq \bigcap_{i\geq 0} \mathcal{C}_i{+}k\mathsf{v}.$$

The forbidden set for the class $\mathcal{C}_i{+}k\mathsf{v}$ is finite for each $i \geq 0$. The graph in $\bigcup_{i\geq 0} \mathrm{Forb}(\mathcal{C}_i{+}k\mathsf{v})$ are forbidden for the class $\mathcal{C}{+}k\mathsf{v}$ but may not be minimal with respect to the topological minor relation.

Let $\mathcal{F}_i = \mathrm{Forb}(\mathcal{C}_i{+}k\mathsf{v})_{\mathrm{s}}$ and $\mathcal{X}_i = \mathrm{minimal}(\mathcal{F}_i)_{\mathrm{t}}$. Note that $\mathcal{X}_i$ is finite for all $i \geq 0$ and $\mathcal{X}_i$ is an antichain with respect to the topological minor relation. We require the following claim to continue.

Claim 106. For all $i \geq 0$, $\mathcal{X}_i\text{-free}_{\mathrm{t}} = \mathcal{F}_i\text{-free}_{\mathrm{t}}$ and $\mathcal{X}_i\text{-free}_{\mathrm{t}} \subseteq \mathcal{F}_i\text{-free}_{\mathrm{s}}$.

Proof. First we show for all $0 \leq i$, $\mathcal{X}_i\text{-free}_{\mathrm{t}} = \mathcal{F}_i\text{-free}_{\mathrm{t}}$. We show equality by showing the subset relation in both directions. Let us show that $\mathcal{F}_i\text{-free}_{\mathrm{t}} \subseteq \mathcal{X}_i\text{-free}_{\mathrm{t}}$, suppose $G \notin \mathcal{X}_i\text{-free}_{\mathrm{t}}$ then there exists a graph $H \in \mathcal{X}_i$ such that $H \leqslant_{\mathrm{t}} G$, from the definition of $\mathcal{X}_i$ we have that $\mathcal{X}_i \subseteq \mathcal{F}_i$ therefore $H \in \mathcal{F}_i$ and consequently $G \notin \mathcal{F}_i\text{-free}_{\mathrm{t}}$. For the opposite direction let $G \notin \mathcal{F}_i\text{-free}_{\mathrm{t}}$ then there exists a graph $H \in \mathcal{F}_i$ such that $H \leqslant_{\mathrm{t}} G$. From the definition of $\mathcal{X}_i$ we have that for all $H \in \mathcal{F}_i$ there exists a $H' \in \mathcal{X}_i$ such that $H' \leqslant_{\mathrm{t}} H$. By transitivity we have that $H \leqslant_{\mathrm{t}} G$ and $H' \leqslant_{\mathrm{t}} H$ therefore $H' \leqslant_{\mathrm{t}} G$. As $H' \in \mathcal{X}_i$ then $G \notin \mathcal{X}_i\text{-free}_{\mathrm{t}}$.

Secondly we show that $\mathcal{X}_i\text{-free}_{\mathrm{t}} \subseteq \mathcal{F}_i\text{-free}_{\mathrm{s}}$. Observe that for all $H, G \in \mathcal{G}$, $H \leqslant_{\mathrm{s}} G \implies$

$H \leqslant_t G$. Let $G \notin \mathcal{F}_i$-free$_s$ then there exists a graph $H \in \mathcal{F}_i$ such that $H \leqslant_s G$ which implies that $H \leqslant_t G$. From the definition of $\mathcal{X}_i$, for all $H \in \mathcal{F}_i$ there exists a $H' \in \mathcal{X}_i$ such that $H' \leqslant_t H$, therefore there exists a graph $H' \in \mathcal{X}_i$ such that $H' \leqslant_t H$. By transitivity $H' \leqslant_t G$. Consequently if $G \notin \mathcal{F}_i$-free$_s$ then $G \notin \mathcal{X}$-free$_t$. ■

Claim 107. For all $i \geq 0$, $\mathcal{X}_0$-free$_t \subseteq \mathcal{X}_i$-free$_t$.

Proof. Let $G \in \mathcal{X}_0$-free$_t$ and suppose $G \notin \mathcal{X}_i$-free$_t$ then there exists a graph $H \in \mathcal{X}_i$ such that $H \leqslant_t G$. For all $H' \in \mathcal{F}_i$ and for all $U \subseteq V(H')$ where $|U| \leq k$ we have $F_i \leqslant_s (H' - U)$ therefore for all $H'' \in \mathcal{X}_i$ and for all $U \subseteq V(H'')$ where $|U| \leq k$ we have $F_i \leqslant_s (H'' - U)$ as $\mathcal{X}_i = \text{minimal}(\mathcal{F}_i)_t$, this is trivial to observe as a consequence of $\mathcal{X}_i \subseteq \mathcal{F}_i$. Therefore;

$$F_i \leqslant_s H \leqslant_t G.$$

For all $i \geq 0$ we have $F_0 \leqslant_t F_i$, from the construction of $\mathcal{F}$. Therefore;

$$F_0 \leqslant_t F_i \leqslant_s H \leqslant_t G$$

which implies that $G \notin \mathcal{X}_0$-free$_t$. As U was chosen without discrimination then;

$$\forall U \in V(G)\ F_0 \leqslant_t (G - U).$$

This contradicts the statement that $G \in \mathcal{X}_0$-free$_t$, concluding that $G \notin \mathcal{X}_0$-free$_t$. ■

Claim 108. $\mathcal{C}{+}k\mathsf{v} = \bigcap_{i\geq 0} \mathcal{X}_i$-free$_t$

Proof. We prove Claim 108 by proving the subset relation in both directions. Firstly we prove $\mathcal{C}{+}k\mathsf{v} \subseteq \bigcap_{i\geq 0} \mathcal{X}_i$-free$_t$. Let $G \notin \bigcap_{i\geq 0} \mathcal{X}_i$-free$_t$ then there exists an index i such that $G \notin \mathcal{X}_i$-free. There must exist a graph $H \in \mathcal{X}_i$ such that $H \leqslant_t G$. From the definition of $\mathcal{X}_i$ and $\mathcal{F}_i$ we have that for all $H' \in \mathcal{X}_i$ and for all $U \subseteq V(H')$ where $|U| \leq k$ we have $F_i \leqslant_s (H' - U)$ and for all $i \geq 0$ we have $F_0 \leqslant_t F_i$, therefore;

$$F_0 \leqslant_t F_i \leqslant_s (H - U) \leqslant_s H \leqslant_t G.$$

As U was chosen without discrimination then $G \notin \mathcal{C}{+}k\mathsf{v}$. Secondly we prove $\bigcap_{i\geq 0} \mathcal{X}_i$-free$_t \subseteq \mathcal{C}{+}k\mathsf{v}$. Let $G \notin \mathcal{C}{+}k\mathsf{v}$ then for all $U \subseteq V(G)$ where $|U| \leq k$ we have $F_0 \leqslant_t (G - U)$ which implies $G \notin \mathcal{X}_0$-free$_t$ therefore $G \notin \bigcap_{i\geq 0} \mathcal{X}_i$-free$_t$. In conclusion as $\bigcap_{i\geq 0} \mathcal{X}_i$-free$_t \subseteq \mathcal{C}{+}k\mathsf{v}$ and $\mathcal{C}{+}k\mathsf{v} \subseteq \bigcap_{i\geq 0} \mathcal{X}_i$-free$_t$ then it must be the case that $\mathcal{C}{+}k\mathsf{v} = \bigcap_{i\geq 0} \mathcal{X}_i$-free$_t$ ■

Observe from Claim 107 that $\bigcap_{0\leq i} \mathcal{X}_i$-free$_t = \mathcal{X}_0$-free$_t$ and from Claim 108 that $\mathcal{C}{+}k\mathsf{v} = \bigcap_{i\geq 0} \mathcal{X}_i$-free$_t$ therefore $\mathcal{C}{+}k\mathsf{v} = \mathcal{X}_0$-free$_t$. From the definition of $\mathcal{F}_i$ each $\mathcal{X}_i$ is finite for all $i \geq 0$. It is therefore clear that $\mathcal{C}{+}k\mathsf{v}$ has a finite forbidden set, more specifically $\text{Forb}(\mathcal{C}{+}k\mathsf{v})_t = \mathcal{X}_0$. □

Limitations

The limitations of this technique are evident from the theorem statement. The technique can only be applied to classes closed with respect to the topological minor relation that have a single minimal forbidden graph. Unfortunately the technique does not generalise easily to other partial orders. The technique does not work for the induced topological minor relation, the technique breaks down when attempting to prove Claim 106. The operation of removing edges is vital in order to perform topological contractions.

The extension of this technique to the general case, that is for any class closed with respect to the topological minor relation and has a finite minimal forbidden set is more difficult. The difficulty arises when trying to establish a relationship between the class $\mathcal{C}+k\mathsf{v}$ and the class $\bigcap_{H \in \mathrm{Forb}(\mathcal{C})}(\{H\}\text{-free}_{\mathrm{t}}+k\mathsf{v})$. For partial orders that have the bounded expansion property this is achieved by abstracting away from the idea of classes and instead reasoning about the maximum order of a critical uniform hypergraph. This abstraction does not apply for the topological minor relation as the order of the partial subgraph that the pattern graph is embedded into cannot be bounded in size. An alternative approach would be to express the graph class with respect to the partial subgraph relation, using a similar techniques that is used when $|\,\mathrm{Forb}(\mathcal{C})| = 1$. Expanding each minimal forbidden topological minor to an infinite series of forbidden partial subgraphs. However, in general this will not yield a finite obstruction set and consequently the abstraction to reasoning about critical uniform hypergraphs can not be successfully achieved, where successful means that the approach yields a finite bound.

Despite the difficulties in providing the generalisation for the class $\mathcal{C}+k\mathsf{v}$ where $\mathrm{Forb}(\mathcal{C})$ is finite there are a number of avenues of research that would yield the desired result. If it could be established that $\bigcap_{H \in \mathrm{Forb}(\mathcal{C})}(\{H\}\text{-free}_{\mathrm{t}}+k\mathsf{v} \setminus \mathcal{C}+k\mathsf{v})$ is finite then the result would yield;

$$\mathrm{Forb}(\mathcal{C}+k\mathsf{v}) = \mathrm{minimal}\Big(\bigcup_{0 \le i \le n} \mathrm{Forb}(\{H_i\}\text{-free}_{\mathrm{t}}+k\mathsf{v}) \\ \cup \big((\bigcap_{0 \le i \le n} \{H_i\}\text{-free}_{\mathrm{t}}+k\mathsf{v}) \setminus \mathcal{C}+k\mathsf{v}\big)\Big)_{\mathrm{t}}.$$

Alternatively it may be possible to show that the critical hypergraph can be restricted to those hyperedges of a bounded size and therefore apply the same abstraction and reasoning about the maximum order of a critical uniform hypergraph.

8.4 Summary

We have provided a set of special cases where the class $\mathcal{C}+k\mathsf{v}$ can be characterised by a finite minimal forbidden set with respect to the topological minor relation providing the class $\mathcal{C}$

is closed with respect to the topological minor relation and the class has a single forbidden topological minor. It is the opinion of the author that the general case is also true however no proof is given. Although the general case of proving when $\mathcal{C}+k\mathsf{v}$ has a finite minimal forbidden set with respect to the topological minor relation is not complete, a set of special cases has been established which allow the techniques of earlier chapters to be applied to the topological minor relation. For each of the techniques the limitations have been discussed and a boundary has been indicated as to where each technique can be applied. Of the techniques exposed it is most likely that the technique expressed in Section 8.3 will bear fruit in providing insight or a solution to solving the general case. It is likely that a proof for the general case would take into account topological properties which would not generalise easily to other partial orders.

Chapter 9

Conclusions

9.1 Summary

We have conducted research into characterising parameterized graph classes relating to the graph modification problems. We have provided a set of tools and techniques for determining when the parameterized graph classes $\mathcal{C}+k\mathsf{v}$ and $\mathcal{C}+k\mathsf{e}$ are characterised by a finite minimal forbidden set with respect to some partial order. We have established a set of sufficient properties for a partial order to posses such that the classes $\mathcal{C}+k\mathsf{v}$ and $\mathcal{C}+k\mathsf{e}$ have a finite minimal forbidden set if the class $\mathcal{C}$ has a finite minimal forbidden set. These general techniques are an improvement on the current state as up to now each class has been considered on a class by class basis. This set of tools has led to the development of the first certifying algorithm that solves a fixed-parameter tractable problem.

We have introduced a mathematical structure which provides a mechanism to reason about the relationships between partial orders. Using this tool it is possible to define types of inheritance that apply to properties of partial orders. This tool allows results which have been proved for a specific partial order to be lifted into a more abstract setting and applied to other partial orders.

Using the tools defined in Chapter 4 a property of a partial order is defined that is sufficient to prove that if a graph class $\mathcal{C}$ is characterised by a finite minimal forbidden set then so are the classes $\mathcal{C}+k\mathsf{v}$ and $\mathcal{C}+k\mathsf{e}$ provided that the classes are also closed with respect to the partial order under consideration. This general characterisation of parameterized graph classes is a signification contribution as previous to this thesis it was unknown whether such graph classes have a finite characterisation. This has led to the development of certifying algorithms for the class membership problem of these parameterized graph classes.

We have demonstrated that the partial orders that allow edge contraction, including topological edge contraction, cannot be characterised using the general technique developed in Chapter 5. We have demonstrated that the partial orders that allow edge contractions, such as the contraction minor relation can be used to provide alternative characterisatio thatns of graph classes. We have provided alternative characterisations of a number of well studied graph classes (Chapter 7).

For the topological minor relation we have highlighted a set of special cases where it is possible to prove that the class $\mathcal{C}+k\mathsf{v}$ has a finite characterisation. The consequence of this result is that a number of graph classes can be recognised in polynomial time.

Contributions to the field

The main contributions made in this thesis are:

- A tool to explore the relationships between partial orders.
- A constructive bound on the maximum order of a minimal forbidden graph for the classes $\mathcal{C}+k\mathsf{v}$ and $\mathcal{C}+k\mathsf{e}$ where the class $\mathcal{C}$ has a finite minimal forbidden set and both classes are closed with respect to a partial order satisfying a set of properties.
- The introduction and motivation of certifying algorithms that run in fixed-parameter time.
- A generic construction for certifying the recognition of the class $\mathcal{C}+k\mathsf{v}$.
- An alternative set of characterisations of a set of well studied graph classes with respect to partial orders that include edge contraction.
- A collection of partial results for characterising the class $\mathcal{C}+k\mathsf{v}$ with respect to the topological minor relation.

9.2 Future work

Although the results of this thesis contribute to the field of theoretical computer science there are a number of questions left open. The open questions fall into three categories:

1. properties of partial order that imply results,
2. generalisation of the techniques developed, and
3. improvements for specific classes.

As has been demonstrated by the results in Chapter 4 and Chapter 5 there are benefits from abstracting away from specific partial order and instead considering properties of a partial order that imply the desired results. The research has raised a number of interesting problems including:

- Is there a property of a partial order that implies the class $\mathcal{C}+k\mathsf{v}$ or $\mathcal{C}+k\mathsf{e}$ is closed with respect to a partial order if $\mathcal{C}$ is?
- Is there a property of a partial order that implies the class $\mathcal{C}+k\mathsf{v}$ ($\mathcal{C}+k\mathsf{e}$) is characterised by a finite minimal forbidden set if $\mathcal{C}$ is, even if generally the class $\mathcal{C}+k\mathsf{v}$ ($\mathcal{C}+k\mathsf{e}$) is not closed?
- Is there a property of a partial order that implies that the complexity of the containment problem is polynomial or fixed-parameter tractable?

The techniques developed to obtain the results of the previous chapters are general, however they do not apply to all partial orders. The cases where the techniques can be applied cover a number of interesting cases but it would be a contribution to the field if a general technique was developed that applied to all partial orders. The following questions are left open:

- Is the class $\mathcal{C}+k\mathsf{v}$ characterised by a finite minimal forbidden set with respect to the topological minor relation if the class $\mathcal{C}$ is characterised by a finite minimal forbidden set with respect to the topological minor relation?
- Is it possible to construct a bound for the maximum size of a graph in the minimal forbidden set for the class $\mathcal{C}+k\mathsf{v}$ where $\mathcal{C}$ is closed with respect to the minor relation?
- Is it possible to weaken the condition on the partial order where the arguments of Chapter 5 can be applied?

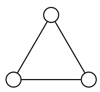
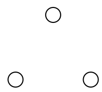
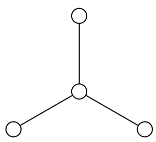
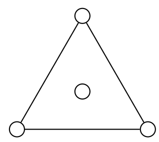
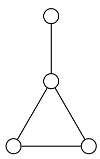
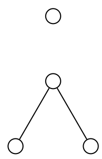
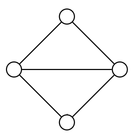
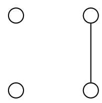
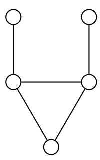
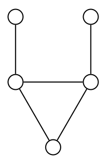
Of course the risk of abstraction and generality is the compromise of tightness of the bounds. Therefore the following question is of interest:

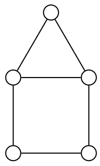
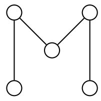
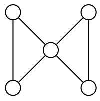
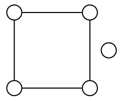
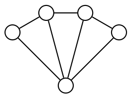
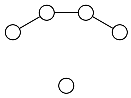
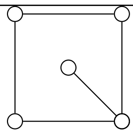
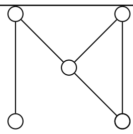
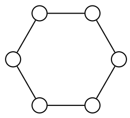
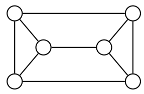
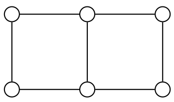
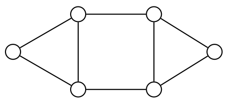
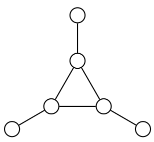
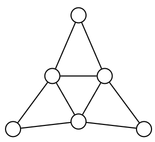
- Can the bound on the maximum order of a minimal forbidden graph be improved, either in general or for specific classes?

Appendices

Appendix A

Small graphs

Graph	Complement
 triangle, C_3, K_3	 $3K_1$
 claw, $K_{1,3}$	 $K_1 \uplus K_3$
 paw, $K_1 \bowtie (K_1 \uplus K_2)$	 co-paw , $K_1 \uplus P_3$
 diamond	 co-diamond, $2K_2 \uplus P_2$, $2K_2 \uplus K_2$
 bull	 bull

 House	 P_5
 butterfly, bowtie	 co-butterfly, co-bowtie, $K_1 \uplus C_4$
 gem, $P_4 \bowtie K_1$	 $K_1 \uplus P_4$
 P	 $\overline{P}$
 C_6	 prism , $\overline{C_6}$
 domino	 co-domino
 net	 S_3

Appendix B

Psuedocode

Algorithm 12: Certifying split graph recognition algorithm [83]

Data: A graph $G = (V, E)$
Result: $\{\mathrm{T}, \mathrm{F}\}$ and either a partition of the vertex set into a clique and an independent set or an induced subgraph isomorphic to a graph in $\{2K_2, C_4, C_5\}$.

Compute a non-decreasing degree sequence of $\alpha = (v_1, \ldots, v_n)$ of G
if *α is not a perfect elimination ordering of G* **then**
    Let $v_i, v_j, v_k \in V$ where $v_iv_j, v_iv_k \in E$ such that $v_jv_k \notin E$ and $i < j < k$
    if *there exists a $z \in N_G(v_j) \cap N_G(v_k)$ and $zv_i \notin E$* **then**
        return F and $\{v_i, v_j, z, v_k\}$ `//` $\{v_i, v_j, z, v_k\}$ `induces a` C_4
    else
        Let $x, y \in V$ such that $v_jx, v_ky \in E$ and $v_jy, v_kx, v_ix, v_iy \notin E$
        if $xy \in E$ **then**
            return F and $\{v_i, v_j, z, v_k\}$ `//` $\{v_i, v_j, x, v_k, y\}$ `induces a` C_5
        else
            return F and $\{v_j, x, v_k, y\}$ `//` $\{v_j, x, v_k, y\}$ `induces a` $2K_2$
        end
    end
end
Let k denote the order of the largest clique in G
$K = \varnothing$, $I = \varnothing$, $i = n$
while $|K| \leq k - 1$ **do**
    $A = N_G(v_i) \cap K$
    if $|A| = |K|$ **then**
        $K = K \cup \{v_i\}$
    else
        Let $x \notin K$ and $y \in K$ be neighbours of v_i
        Let $z \in V$ be a neighbour of y where $v_iz, xz \notin E$
        return F and $\{v_i, x, y, z\}$ `//` $\{v_i, x, y, z\}$ `induces` $2K_2$
    end
    $i = i + 1$
end
while $i \geq 1$ **do**
    $A = N_G(v_i) \cap (K \cup I)$
    if $A \subseteq K$ **then**
        $I = I \cup \{v_i\}$
    else
        Let $x = A \cap I$
        Let $y \in K$ such that $xy, v_iy \notin E$
        Let z be a neighbour of y with $xz \notin E$
        return F and $\{v_i, x, y, z\}$ `//` $\{v_i, x, y, z\}$ `induces` $2K_2$
    end
    $i = i - 1$
end
return T and (K, I)

Algorithm 13: Certifying threshold graph recognition algorithm [83]

Data: A graph $G = (V, E)$
Result: $\{\mathrm{T}, \mathrm{F}\}$ and either a partition of the vertex set into a clique and a nested neighbour ordered independent set or an induced subgraph isomorphic to a graph in $\{2K_2, C_4, P_4\}$.

if `Certifying-Split`(G) *returns F and U* **then**
    Let $U' \subseteq U$ where $|U'| = 4$
    return F and U' `//` U' `induces a graph in` $\{2K_2, C_4, P_4\}$ `in` G
else
    Let K, I be the clique and independent set return from `Certifying-Split`(G)
    Let $\alpha = \{v_1, \ldots, v_n\}$ be a non-decreasing degree sequence of G
    let $\beta = (v_1, \ldots, v_{|I|})$
    $V = V \setminus \{x \mid \deg(x) = 0 \wedge x \in V\}$
    Let $member = \mathrm{T}$, $i = n$
    while $v_i \in V$ **do**
        if v_i *is universal* **then**
            $V = V \setminus \{v_i\}$
            $V = V \setminus \{x \mid \deg(x) = 0 \wedge x \in V\}$
        else
            $member = \mathrm{F}$
        end
        $i = i - 1$
    end
    if $member = T$ **then**
        return T and β
    else
        repeat
            $V = V \setminus \{x \mid xu \notin E \text{ and } x \in K,\ u \in I\}$
            $V = V \setminus \{x \mid \forall u \in K\ ux \in E \text{ and } x \in I\}$
        until V *is unchanged*;
        Let $v \in I$ be the vertex of highest degree
        Let $y \in K$ such that $yv \notin E$
        Let $z \in I$ be a neighbour of y
        Let $w \in K$ such that $wv \in E$ and $zw \notin E$
        return F and $\{v, w, y, z\}$ `//` $\{v, w, y, z\}$ `indcues` P_4
    end
end

Bibliography

[1] I. Adler, M. Grohe, and S. Kreutzer. Computing excluded minors. In *Proceedings of the 19th Symposium on Discrete Algorithms*, pages 641–650. Society for Industrial and Applied Mathematics, 2008.

[2] J. R. Alfonsin. Spatial graphs and oriented matroids: the trefoil. *Discrete & Computational Geometry*, 22(1):149–158, 1999.

[3] N. Alon, R. Yuster, and U. Zwick. Finding and counting given length cycles. *Algorithmica*, 17(3):209–223, 1997.

[4] M. I. Andreica. A dynamic programming framework for combinatorial optimization problems on graphs with bounded pathwidth. *arXiv preprint arXiv:0806.0840*, 2008.

[5] S. Arnborg. Efficient algorithms for combinatorial problems on graphs with bounded decomposabilitya survey. *BIT Numerical Mathematics*, 25(1):1–23, 1985.

[6] S. Arnborg, A. Proskurowski, and D. G. Corneil. Forbidden minors characterization of partial 3-trees. *Discrete Mathematics*, 80(1):1–19, 1990.

[7] A. Asratian, T. Denley, and R. Häggkvist. *Bipartite Graphs and Their Applications*. Cambridge Tracts in Mathematics. Cambridge University Press, 1998.

[8] L. Babai, D. Y. Grigoryev, and D. M. Mount. Isomorphism of graphs with bounded eigenvalue multiplicity. In *Proceedings of the 14th Symposium on Theory of Computing*, pages 310–324. ACM, 1982.

[9] R. Belmonte, P. Golovach, P. Heggernes, P. van 't Hof, M. Kamiński, and D. Paulusma. Finding contractions and induced minors in chordal graphs via disjoint paths. *Proceedings of Algorithms and Computation*, pages 110–119, 2011.

[10] R. Belmonte, P. A. Golovach, P. Heggernes, P. van 't Hof, M. Kamiński, and D. Paulusma. Detecting fixed patterns in chordal graphs in polynomial time. *Algorithmica*, 69(3):501–521, 2014.

[11] R. Belmonte, P. Heggernes, and P. van 't Hof. Edge contractions in subclasses of chordal graphs. *Discrete Applied Mathematics*, 160(7):999–1010, 2012.

[12] S. Benzer. On the topology of the genetic fine structure. *Proceedings of the National Academy of Sciences of the United States of America*, 45(11):1607, 1959.

[13] C. Berge and E. Minieka. *Graphs and Hypergraphs*, volume 7. North-Holland publishing company Amsterdam, 1973.

[14] H. L. Bodlaender. Polynomial algorithms for graph isomorphism and chromatic index on partial k-trees. *Journal of Algorithms*, 11(4):631–643, 1990.

[15] B. Bollobás. On generalized graphs. *Acta Mathematica Hungarica*, 16(3):447–452, 1965.

[16] K. S. Booth and C. J. Colbourn. *Problems polynomially equivalent to graph isomorphism.* Computer Science Department, University of Waterloo, 1979.

[17] A. Brandstädt, V. Le, and J. Spinrad. *Graph classes: A Survey.* Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1999.

[18] A. Brouwer and H. Veldman. Contractibility and NP-completeness. *Journal of Graph Theory*, 11(1):71–79, 1987.

[19] L. Cai. Fixed-parameter tractability of graph modification problems for hereditary properties. *Information Processing Letters*, 58(4):171–176, 1996.

[20] Y. Cao. Linear recognition of almost interval graphs. *CoRR*, abs/1403.1515, 2014.

[21] Y. Cao and D. Marx. Interval deletion is fixed-parameter tractable. *ACM Transactions on Algorithms (TALG)*, 11(3):21, 2015.

[22] M. Chudnovsky, N. Robertson, P. Seymour, and R. Thomas. The strong perfect graph theorem. *Annals of Mathematics*, pages 51–229, 2006.

[23] K. Cirino, S. Muthukrishnan, N. Narayanaswamy, and H. Ramesh. Graph editing to bipartite interval graphs: exact and asymptotic bounds. In *Proceedings of Foundations of Software Technology and Theoretical Computer Science*, pages 37–53. Lecture notes in Computer Science Springer, 1997.

[24] W. Cook and P. Seymour. Tour merging via branch-decomposition. *INFORMS Journal on Computing*, 15(3):233–248, 2003.

[25] D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progressions. *Journal of Symbolic Computation*, 9(3):251–280, 1990.

[26] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms.* MIT press, 3rd edition, 2009.

[27] D. G. Corneil, H. Lerchs, and L. Stwart Burlingham. Complement reducible graphs. *Discrete Applied Mathematics*, 3(3):163–174, 1981.

[28] D. G. Corneil, Y. Perl, and L. K. Stewart. A linear recognition algorithm for cographs. *SIAM Journal on Computing*, 14(4):926–934, 1985.

[29] D. G. Corneil and U. Rotics. On the relationship between clique-width and treewidth. *SIAM Journal on Computing*, 34(4):825–847, 2005.

[30] B. Courcelle. The monadic second-order logic of graphs. I. recognizable sets of finite graphs. *Information and Computation*, 85(1):12–75, 1990.

[31] B. Courcelle and S. Olariu. Upper bounds to the clique width of graphs. *Discrete Applied Mathematics*, 101(13):77–114, 2000.

[32] B. Courcelle and S. Oum. Vertex-minors, monadic second-order logic, and a conjecture by Seese. *Journal of Combinatorial Theory, Series B*, 97(1):91–126, 2007.

[33] R. Courcelle, B. Downey and M. Fellows. A note on the computability of graph minor obstruction sets for monadic second order ideals. *Journal of Universal Computer Science*, 3(11):1194–1198, 1997.

[34] P. Damaschke. Induced subgraphs and well-quasi-ordering. *Journal of Graph Theory*, 14(4):427–435, 1990.

[35] C. M. H. de Figueiredo, L. Faria, S. Klein, and R. Sritharan. On the complexity of the sandwich problems for strongly chordal graphs and chordal bipartite graphs. *Theoretical Computer Science*, 381(1):57–67, 2007.

[36] H. de Ridder et al. Information system on graph classes and their inclusions (ISGCI). http://www.graphclasses.org, March 2015.

[37] J. Dean and S. Ghemawat. MapReduce: Simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.

[38] R. Diestel. *Graph Theory*, volume 173 of *Graduate Texts in Mathematics*. Springer-Verlag, Heidelberg, third edition, 2005.

[39] G. Ding. Subgraphs and well-quasi-ordering. *Journal of Graph Theory*, 16(5):489–502, 1992.

[40] G. Ding. Excluding a long double path minor. *Journal of Combinatorial Theory, Series B*, 66(1):11–23, 1996.

[41] G. Dirac. On rigid circuit graphs. *Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg*, 25(1-2):71–76, 1961.

[42] F. Dorn. Dynamic programming and fast matrix multiplication. In *Proceeding of ESA 2006*, pages 280–291. Springer, 2006.

[43] R. Downey and M. Fellows. *Parameterized complexity*, volume 3. Springer New York, 1999.

[44] R. G. Downey and M. R. Fellows. *Fundamentals of parameterized complexity*, volume 4. Springer-Verlag, 2013.

[45] E. S. El-Mallah and C. J. Colbourn. The complexity of some edge deletion problems. *IEEE Transactions on Circuits and Systems*, 35(3):354–362, 1988.

[46] D. Eppstein. Subgraph isomorphism in planar graphs and related problems. In *Proceedings of Symposium on Discrete Algorithms 1995*, pages 632–640. Society for Industrial and Applied Mathematics, 1995.

[47] Erdős and R. Rado. Intersection theorems for systems of sets. *Journal of the London Mathematical Society*, 35:85–90, 1960.

[48] P. Erdős and T. Gallai. On the minimal number of vertices representing the edges of a graph. *Magyar Tud. Akad. Mat. Kutató Int. Közl*, 6:181–203, 1961.

[49] M. Fellows, D. Hermelin, and F. Rosamond. Well quasi orders in subclasses of bounded treewidth graphs and their algorithmic applications. *Algorithmica*, 64(1):3–18, 2012.

[50] M. Fellows and B. Jansen. FPT is characterized by useful obstruction sets. In A. Brandstädt, K. Jansen, and R. Reischuk, editors, *Proceedings of Graph-Theoretic Concepts in Computer Science*, volume 8165 of *Lecture notes in Computer Science*, pages 261–273. Springer-Verlag Berlin Heidelberg, 2013.

[51] M. Fellows, J. Kratochvíl, M. Middendorf, and F. Pfeiffer. The complexity of induced minors and related problems. *Algorithmica*, 13(3):266–282, 1995.

[52] M. R. Fellows, D. Hermelin, and F. A. Rosamond. Well-quasi-orders in subclasses of bounded treewidth graphs. In *Parameterized and Exact Computation*, pages 149–160. Springer-Verlag, 2009.

[53] M. R. Fellows and M. A. Langston. Nonconstructive tools for proving polynomial-time decidability. *Journal of the ACM (JACM)*, 35(3):727–739, 1988.

[54] M. R. Fellows and M. A. Langston. An analogue of the Myhill-Nerode theorem and its use in computing finite-basis characterizations. In *Proceeding of the 54th Foundations of Computer Science 2013*, pages 520–525. IEEE Comput. Soc. Press, 1989.

[55] M. R. Fellows and M. A. Langston. On search decision and the efficiency of polynomial-time algorithms. In *Proceedings of the twenty-first annual Symposium on Theory of Computing*, pages 501–512. ACM, 1989.

[56] J. Fiala, M. Kamiński, B. Lidicky, and D. Paulusma. The k-in-a-path problem for claw-free graphs. *Algorithmica.*, 62(1-2):499–519, 2012.

[57] J. Fiala, M. Kamiński, and D. Paulusma. A note on induced tree minors, submitted.

[58] S. Foldes and P. Hammer. Split graphs. *Congr. Numer*, 19:311–315, 1977.

[59] L. R. Ford and D. R. Fulkerson. Maximal flow through a network. *Canadian Journal of Mathematics*, 8(3):399–404, 1956.

[60] P. Frankl. An extremal problem for two families of sets. *European Journal of Combinatorics*, 3(2):125–127, 1982.

[61] D. R. Fulkerson and O. A. Gross. Incidence matrices and interval graphs. *Pacific Journal of Mathematics*, 15(3):835–855, 1965.

[62] Z. Füredi. Geometrical solution of an intersection problem for two hypergraphs. *European Journal of Combinatorics*, 5(2):133–136, 1984.

[63] T. Gallai. Transitiv orientierbare Graphen. *Acta Mathematica Hungarica*, 18:25–66, 1967.

[64] M. R. Garey and D. S. Johnson. *Computers and Intractability, A Guide to the Theory of NP-Completeness*. W.H. Freeman and Company, New York, 1979.

[65] A. C. Giannopoulou, I. Salem, and D. Zoros. Effective computation of immersion obstructions for unions of graph classes. *Journal of Computer and System Sciences*, 80(1):207–216, 2014.

[66] P. Gilmore and A. Hoffman. A characterization of comparability graphs and of interval graphs. Technical report, DTIC Document, 1962.

[67] P. W. Goldberg, M. C. Golumbic, H. Kaplan, and R. Shamir. Four strikes against physical mapping of DNA. *Journal of Computational Biology*, 2(1):139–152, 1995.

[68] S. Goldwasser, S. Micali, and C. Rackoff. The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208, 1989.

[69] P. A. Golovach, M. Kamiński, D. Paulusma, and D. M. Thilikos. Containment relations in split graphs. *Discrete Applied Mathematics*, 160(1):155–163, 2012.

[70] P. A. Golovach, D. Paulusma, and E. J. van Leeuwen. Induced disjoint paths in AT-free graphs. In *Proceedings of Algorithm Theory–SWAT 2012 Lecture notes in Computer Science*, pages 153–164. Springer, 2012.

[71] M. Golumbic. Trivially perfect graphs. *Discrete Mathematics*, 24(1):105–107, 1978.

[72] M. C. Golumbic. *Algorithmic graph theory and perfect graphs*, volume 57 of *Annals of Discrete Mathematics*. Elsevier, 2004.

[73] M. C. Golumbic, H. Kaplan, and R. Shamir. Graph sandwich problems. *Journal of Algorithms*, 19(3):449–473, 1995.

[74] M. Grohe. Computing crossing numbers in quadratic time. In *Proceedings of the 33rd Symposium on Theory of Computing*, pages 231–236. ACM, 2001.

[75] M. Grohe, K. Kawarabayashi, D. Marx, and P. Wollan. Finding topological subgraphs is fixed-parameter tractable. In *Proceedings of the 43rd Symposium on Theory of Computing*, pages 479–488. ACM, 2011.

[76] A. Gyárfás, J. Lehel, and Z. Tuza. Upper bound on the order of τ-critical hypergraphs. *Journal of Combinatorial Theory, Series B*, 33(2):161–165, 1982.

[77] M. Habib and C. Paul. A simple linear time algorithm for cograph recognition. *Discrete Applied Mathematics*, 145(2):183–197, 2005.

[78] A. Hajnal and J. Surányi. Über die Auflösung von Graphen in vollständige Teilgraphen. *Univ Sci. Budapest, Eötvös Sect. Math. 1*, pages 113–121, 1958.

[79] R. Halin. S-functions for graphs. *Journal of Geometry*, 8:171–186, 1976.

[80] P. Hammer, T. Ibaraki, and U. N. Peled. Threshold numbers and threshold completions. *North-Holland Mathematics Studies*, 59:125–145, 1981.

[81] P. L. Hammer, T. Ibaraki, and B. Simeone. *Degree sequences of threshold graphs.* Department of Combinatorics and Optimization, University of Waterloo, 1978.

[82] P. L. Hammer and B. Simeone. The splittance of a graph. *Combinatorica*, 1(3):275–284, 1981.

[83] P. Heggernes and D. Kratsch. Linear-time certifying recognition algorithms and forbidden induced subgraphs. *Nordic Journal of Computing*, 14(1):87–108, 2007.

[84] P. Hell and J. Huang. Certifying LexBFS recognition algorithms for proper interval graphs and proper interval bigraphs. *SIAM J. Discrete Math*, 18:554–570, 2004.

[85] G. Higman. Ordering by divisibility in abstract algebras. *Proceedings of the London Mathematical Society*, 3(1):326–336, 1952.

[86] C. A. R. Hoare. An axiomatic basis for computer programming. *Communications of the ACM*, 12(10):576–580, 1969.

[87] J. Hopcroft and R. Tarjan. Efficient planarity testing. *Journal of the ACM*, 21:549–568, 1974.

[88] J. E. Hopcroft and J.-K. Wong. Linear time algorithm for isomorphism of planar graphs (preliminary report). In *Proceedings of the 6th Symposium on Theory of Computing*, pages 172–184. ACM, 1974.

[89] D. Johnson. The many faces of polynomial time. *Journal of Algorithms*, 8(2):285–303, 1987.

[90] M. Kamiński, D. Paulusma, and D. M. Thilikos. Contractions of planar graphs in polynomial time. In *Algorithms–ESA 2010*, pages 122–133. Springer, 2010.

[91] M. M. Kanté and O. Kwon. An upper bound on the size of obstructions for bounded linear rank-width. *arXiv preprint arXiv:1412.6201*, 2014.

[92] H. Kaplan and Y. Nussbaum. Certifying algorithms for recognizing proper circular-arc graphs and unit circular-arc graphs. In *Graph-Theoretic Concepts in Computer Science, Lecture notes in Computer Science*, pages 289–300. Springer, 2006.

[93] H. Kaplan, R. Shamir, and R. E. Tarjan. Tractability of parameterized completion problems on chordal, strongly chordal, and proper interval graphs. *SIAM Journal on Computing*, 28(5):1906–1922, 1999.

[94] R. Karp. Reducibility among combinatorial problems. In R. Miller, J. Thatcher, and J. Bohlinger, editors, *Complexity of Computer Computations*, pages 85–103. Springer, 1972.

[95] T. Kashiwabara and T. Fujisawa. An NP-complete problem on interval graphs. In *IEEE Symp. of Circuits and Systems*, pages 82–83, 1979.

[96] G. O. H. Katona. Solution of a problem of A. Ehrenfeucht and J. Mycielski. *Journal of Combinatorial Theory, Series A*, 17(2):265–266, 1974.

[97] K. Kawarabayashi, Y. Kobayashi, and B. Reed. The disjoint paths problem in quadratic time. *Journal of Combinatorial Theory, Series B*, 102(2):424–435, 2012.

[98] K. Kawarabayashi and B. Reed. Computing crossing number in linear time. In *Proceedings of the 39th Symposium on Theory of Computing*, pages 382–390. ACM, 2007.

[99] F. N. Khzam. A kernelization algorithm for d-hitting set. *Journal of Computer and System Sciences*, 76(7):524–531, 2010.

[100] T. Kloks. *Treewidth: Computations and Approximations, Lecture notes in Computer Science*, volume 842. Springer-Verlag Berlin, 1994.

[101] T. Kloks, D. Kratsch, and H. Müller. Finding and counting small induced subgraphs efficiently. *Information Processing Letters*, 74(3):115–121, 2000.

[102] N. Korpelainen and V. V. Lozin. Bipartite induced subgraphs and well-quasi-ordering. *Journal of Graph Theory*, 67(3):235–249, 2011.

[103] D. Kratsch, R. McConnell, K. Mehlhorn, and J. Spinrad. Certifying algorithms for recognizing interval graphs and permutation graphs. *SIAM Journal on Computing*, 36(2):326–353, 2006.

[104] S. Kratsch and P. Schweitzer. Isomorphism for graphs of bounded feedback vertex set number. In *Proceeding of Algorithm Theory-SWAT 2010, Lecture notes in Computer Science*, pages 81–92. Springer, 2010.

[105] J. B. Kruskal. Well-quasi-ordering, the tree theorem, and Vazsonyi's conjecture. *Transactions of the American Mathematical Society*, pages 210–225, 1960.

[106] C. Kuratowski. Sur le probleme des courbes gauches en topologie. *Fundamenta mathematicae*, 15(1):271–283, 1930.

[107] B. Lévêque, D. Y. Lin, F. Maffray, and N. Trotignon. Detecting induced subgraphs. *Discrete Applied Mathematics*, 157(17):3540–3551, 2009.

[108] A. Levin, D. Paulusma, and G. J. Woeginger. The computational complexity of graph contractions I: Polynomially solvable and NP-complete cases. *Networks*, 51(3):178–189, 2008.

[109] A. Levin, D. Paulusma, and G. J. Woeginger. The computational complexity of graph contractions II: two tough polynomially solvable cases. *Networks*, 52(1):32–56, 2008.

[110] J. M. Lewis and M. Yannakakis. The node-deletion problem for hereditary properties is NP-complete. *Journal of Computer and System Sciences*, 20(2):219–230, 1980.

[111] D. Lokshtanov. Wheel-free deletion is W[2]-hard. In *Proceeding of Parameterized and Exact Computation, Lecture notes in Computer Science*, pages 141–147. Springer, 2008.

[112] D. Lokshtanov, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. Fixed-parameter tractable canonization and isomorphism test for graphs of bounded treewidth. In *Proceeding of the 55th Foundations of Computer Science*, pages 186–195. IEEE, 2014.

[113] L. Lovász. Graph minor theory. *Bulletin of the American Mathematical Society*, 43(1):75–86, 2006.

[114] D. Marx. Chordal deletion is fixed-parameter tractable. *Algorithmica*, 57(4):747–768, 2010.

[115] J. Matoušek and R. Thomas. On the complexity of finding iso-and other morphisms for partial k-trees. *Discrete Mathematics*, 108(1):343–364, 1992.

[116] R. M. McConnell, K. Mehlhorn, S. Näher, and P. Schweitzer. Certifying algorithms. *Computer Science Review*, 5(2), 2011.

[117] B. D. McKay. Nauty users guide (version 2.4). *Computer Science Dept., Australian National University*, pages 225–239, 2007.

[118] K. Mehlhorn and S. Näher. LEDA: A library of efficient data types and algorithms. In *Proceeding of Mathematical Foundations of Computer Science 1989*, pages 88–106. Springer, 1989.

[119] K. Mehlhorn and S. Näher. *LEDA: A platform for combinatorial and geometric computing.* Cambridge University Press, 1999.

[120] D. Meister. Recognition and computation of minimal triangulations for AT-free claw-free and co-comparability graphs. *Discrete Applied Mathematics*, 146(3):193–218, 2005.

[121] K. Menger. Zur allgemeinen Kurventheorie. *Fundamenta Mathematicae*, 10(95-115):5, 1927.

[122] A. Natanzon, R. Shamir, and R. Sharan. Complexity classification of some edge modification problems. *Discrete Applied Mathematics*, 113(1):109–128, 2001.

[123] G. Necula. *Proof-carrying code.* Springer, 2011.

[124] S. Oum. Rank-width and vertex-minors. *Journal of Combinatorial Theory, Series B*, 95(1):79–100, 2005.

[125] S. Oum. Rank-width and well-quasi-ordering. *SIAM Journal on Discrete Mathematics*, 22(2):666–682, 2008.

[126] S. Oum. Rank-width is less than or equal to branch-width. *Journal of Graph Theory*, 57(3):239–244, 2008.

[127] S. Oum and P. Seymour. Approximating clique-width and branch-width. *Journal of Combinatorial Theory, Series B*, 96(4):514–528, 2006.

[128] S.-L. Peng, T. Kloks, and C.-M. Lee. The maximum interval graphs on distance hereditary graphs. In *Proceedings of the 9th Joint International Conference on Information Sciences.* Atlantis Press, 2006.

[129] A. Pnueli, A. Lempel, and S. Even. Transitive orientation of graphs and identification of permutation graphs. *Canad. J. math*, 23(1):160–175, 1971.

[130] V. Raghavan and J. Spinrad. Robust algorithms for restricted domains. In *Proceedings of the 12th Symposium on Discrete Algorithms*, pages 460–467. Society for Industrial and Applied Mathematics, 2001.

[131] S. Ramachandramurthi. The structure and number of obstructions to treewidth. *SIAM Journal on Discrete Mathematics*, 10(1):146–157, 1997.

[132] N. Robertson and P. Seymour. Graph minors. I. Excluding a forest. *Journal of Combinatorial Theory, Series B*, 35(1):39–61, 1983.

[133] N. Robertson and P. Seymour. Graph minors. II. Algorithmic aspects of tree-width. *Journal of Algorithms*, 7(3):309–322, 1986.

[134] N. Robertson and P. Seymour. Graph minors. V. Excluding a planar graph. *Journal of Combinatorial Theory, Series B*, 41(1):92–114, 1986.

[135] N. Robertson and P. Seymour. Graph minors. IV. Tree-width and well-quasi-ordering. *Journal of Combinatorial Theory, Series B*, 48(2):227–254, 1990.

[136] N. Robertson and P. Seymour. Graph minors. X. Obstructions to tree-decomposition. *Journal of Combinatorial Theory, Series B*, 52(2):153–190, 1991.

[137] N. Robertson and P. Seymour. Graph minors. XIII. The disjoint paths problem. *Journal of Combinatorial Theory, Series B*, 63(1):65–110, 1995.

[138] N. Robertson and P. Seymour. Graph minors. XVI. Excluding a non-planar graph. *Journal of Combinatorial Theory, Series B*, 89(1):43–76, 2003.

[139] N. Robertson and P. Seymour. Graph minors. XX. Wagner's conjecture. *Journal of Combinatorial Theory, Series B*, 92(2):325–357, 2004.

[140] N. Robertson and P. Seymour. Graph minors. XXI. Graphs with unique linkages. *Journal of Combinatorial Theory, Series B*, 99(3):583–616, 2009.

[141] N. Robertson and P. Seymour. Graph minors. XXIII. Nash-Williams' immersion conjecture. *J. Comb. Theory Ser. B*, 100(2):181–205, 2010.

[142] N. Robertson and P. Seymour. Graph minors. XXII. Irrelevant vertices in linkage problems. *Journal of Combinatorial Theory, Series B*, 102(2):530–563, 2012.

[143] D. J. Rose, R. E. Tarjan, and G. S. Lueker. Algorithmic aspects of vertex elimination on graphs. *SIAM Journal on Computing*, 5(2):266–283, 1976.

[144] B. Schröder. *Ordered sets: An Introduction.* Springer, 2003.

[145] P. Seymour and R. Thomas. Graph searching and a min-max theorem for tree-width. *Journal of Combinatorial Theory, Series B*, 58(1):22 – 33, 1993.

[146] P. D. Seymour and R. Thomas. Call routing and the ratcatcher. *Combinatorica*, 14(2):217–241, 1994.

[147] V. Strassen. Gaussian elimination is not optimal. *Numerische Mathematik*, 13(4):354–356, 1969.

[148] E. Szemerédi and G. Petruska. On a combinatorial problem I. *Studia Sci. Math. Hungar*, 7:363–374, 1972.

[149] R. E. Tarjan and M. Yannakakis. Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs. *SIAM Journal on Computing*, 13(3):566–579, 1984.

[150] S. Thomassé. A quadratic kernel for feedback vertex set. In *Proceedings of the 20th Symposium on Discrete Algorithms*, pages 115–119. Society for Industrial and Applied Mathematics, 2009.

[151] S. Thomassé. A $4k^2$ kernel for feedback vertex set. *ACM Trans. Algorithms*, 6(2):32:1–32:8, Apr. 2010.

[152] C. Thomassen. Kuratowski's theorem. *Journal of Graph Theory*, 5(3):225–241, 1981.

[153] W. T. Trotter. *Combinatorics and partially ordered sets: Dimension theory*, volume 6. JHU Press, 2001.

[154] Z. Tuza. Critical hypergraphs and intersecting set-pair systems. *Journal of Combinatorial Theory, Series B*, 39(2):134–145, 1985.

[155] R. Van Bevern, C. Komusiewicz, H. Moser, and R. Niedermeier. Measuring indifference: Unit interval vertex deletion. In *Proceeding of Graph Theoretic Concepts in Computer Science,Lecture notes in Computer Science*, pages 232–243. Springer, 2010.

[156] P. van 't Hof, M. Kamiński, D. Paulusma, S. Szeider, and D. Thilikos. On graph contractions and induced minors. *Discrete Applied Mathematics.*, 160(6):799–809, April 2012.

[157] P. van 't Hof and Y. Villanger. Proper interval vertex deletion. *Algorithmica*, 65(4):845–867, 2013.

[158] Y. Villanger, P. Heggernes, C. Paul, and J. A. Telle. Interval completion is fixed parameter tractable. *SIAM Journal on Computing*, 38(5):2007–2020, 2009.

[159] K. Wagner. Über eine Eigenschaft der ebenen Komplexe. *Mathematische Annalen*, 114(1):570–590, 1937.

[160] C. Wang. A subgraph problem from restriction maps of DNA. *Journal of Computational Biology*, 1(3):227–234, 1994.

[161] T. Watanabe, T. Ae, and A. Nakamura. On the removal of forbidden graphs by edge-deletion or by edge-contraction. *Discrete Applied Mathematics*, 3(2):151–153, 1981.

[162] D. B. West. *Introduction to Graph Theory (2nd Edition)*. Prentice Hall, Aug. 2000.

[163] V. V. Williams. Multiplying matrices faster than Coppersmith-Winograd. In *Proceedings of the forty-fourth annual ACM symposium on Theory of computing*, pages 887–898. ACM, 2012.

[164] S. G. Williamson. Depth-first search and Kuratowski subgraphs. *J. ACM*, 31(4):681–693, Sept. 1984.

[165] S. Wilson. Open source graph theory java library. http://annas.gt4j.org/, apr 2015.

[166] T. Wolle and H. L. Bodlaender. A note on edge contraction. Technical report, Utrecht Technical Report UU-CS-2004, 2004.

[167] J.-H. Yan, J.-J. Chen, and G. J. Chang. Quasi-threshold graphs. *Discrete applied mathematics*, 69(3):247–255, 1996.

[168] M. Yannakakis. Node-and edge-deletion NP-complete problems. In *Proceedings of the 10th Symposium on Theory of Computing*, pages 253–264. ACM, 1978.

[169] M. Yannakakis. The effect of a connectivity requirement on the complexity of maximum subgraph problems. *Journal of the ACM (JACM)*, 26(4):618–630, 1979.

[170] M. Yannakakis. Computing the minimum fill-in is NP-complete. *SIAM Journal on Algebraic Discrete Methods*, 2(1):77–79, 1981.

[171] P. Zhang, E. A. Schon, S. G. Fischer, E. Cayanis, J. Weiss, S. Kistler, and P. E. Bourne. An algorithm based on graph theory for the assembly of contigs in physical mapping of DNA. *Computer applications in the biosciences : CABIOS*, 10(3):309–317, 1994.