Clique-width for hereditary graph classes

Konrad K. Dabrowski, Matthew Johnson and Daniël Paulusma

Abstract

Clique-width is a well-studied graph parameter owing to its use in understanding algorithmic tractability: if the clique-width of a graph class $\mathcal{G}$ is bounded by a constant, a wide range of problems that are NP-complete in general can be shown to be polynomial-time solvable on $\mathcal{G}$. For this reason, the boundedness or unboundedness of clique-width has been investigated and determined for many graph classes. We survey these results for hereditary graph classes, which are the graph classes closed under taking induced subgraphs. We then discuss the algorithmic consequences of these results, in particular for the Colouring and Graph Isomorphism problems. We also explain a possible strong connection between results on boundedness of clique-width and on well-quasi-orderability by the induced subgraph relation for hereditary graph classes.

1 Introduction

Many decision problems are known to be NP-complete [84], and it is generally believed that such problems cannot be solved in time polynomial in the input size. For many of these hard problems, placing restrictions on the input (that is, insisting that the input has certain stated properties) can lead to significant changes in the computational complexity of the problem. This leads one to ask fundamental questions: under which input restrictions can an NP-complete problem be solved in polynomial time, and under which input restrictions does the problem remain NP-complete? For problems defined on graphs, we can restrict the input to some special class of graphs that have some commonality. The ultimate goal is to obtain complexity dichotomies for large families of graph problems, which tell us exactly for which graph classes a certain problem is efficiently solvable and for which it stays computationally hard. Such dichotomies may not always exist if P$\neq$ NP [129], but rather than solving problems one by one, and graph class by graph class, we want to discover general properties of graph classes from which we can determine the tractability or hardness of families of problems.

1.1 Width Parameters

One way to define a graph class is to use a notion of "width" and consider the set of graphs for which the width is bounded by a constant. Though it will not be our focus, let us briefly illustrate this idea with the most well-known width parameter, *treewidth*. A *tree decomposition* of a graph $G = (V, E)$ is a tree T whose nodes are subsets of V and has the properties that, for each v in V, the tree nodes that contain v induce a non-empty connected subgraph, and, for each edge vw in E, there is at least one tree node that contains v and w. See Figure 1 for an illustration of a graph and one of its tree decompositions. The sets of vertices that form the nodes of the tree are called *bags* and the width of the decomposition is one less than the size of the largest bag. The treewidth of G is the minimum width of its tree decompositions. One can therefore define a class of graphs of bounded treewidth; that is, for some constant c, the collection of graphs that each have treewidth at most c. The example in Figure 1 has treewidth 2. Moreover, it is easy to see that trees form exactly the class of graphs with treewidth 1. Hence, the treewidth of a graph can be seen as a measure that indicates how close a graph is to being a tree. Many graph problems can be solved in

Cambridge University Press
978-1-108-74072-2 — Surveys in Combinatorics 2019
Edited by Allan Lo , Richard Mycroft , Guillem Perarnau , Andrew Treglown
Excerpt
More Information

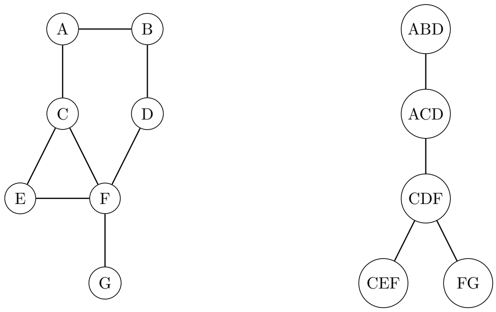


Figure 1: A graph, and a tree decomposition of the graph.

polynomial time on trees. For such problems it is natural to investigate whether restricting the problem to inputs that have bounded treewidth still yields algorithmic tractability. An approach that often yields polynomial-time algorithms is to apply dynamic programming over the decomposition tree. A disadvantage of this approach is that only sufficiently sparse graphs have bounded treewidth.

We further discuss reasons for focussing on width parameters in Section 1.2, but let us first note that there are many alternative width parameters, each of which has led to progress in understanding the complexity of problems on graphs.

Clique-width, the central width parameter in our survey, is another well-known example, which has received significant attention since it was introduced by Courcelle, Engelfriet and Rozenberg [56] at the start of the 1990s. Clique-width can be seen as a generalisation of treewidth that *can* deal with dense graphs, such as complete graphs and complete bipartite graphs, provided these instances are sufficiently regular. We will give explain this in Section 3, where we also give a formal definition, but, in outline, the idea is, given a graph G, to determine how it can be built up vertex-by-vertex using four specific graph operations that involve assigning labels to the vertices. The operations ensure that vertices labelled alike will keep the same label and thus, in some sense, behave identically. The clique-width of G is the minimum number of different labels needed to construct G in this way. Hence, if the clique-width of a graph G is small, we can decompose G into large sets of similarly behaving vertices, and these decompositions can be exploited to find polynomial-time algorithms (as we shall see later in this paper).

We remark that many other width parameters have been defined including boolean-width, branch-width, MIM-width, MM-width, module-width, NLC-width, path-width and rank-width, to name just a few. These parameters differ in strength, as we explain below; we refer to [95, 111, 116, 164] for surveys on width parameters.

Given two width parameters p and q, we say that p *dominates* q if there is a function f such that $p(G) \leq f(q(G))$ for all graphs G. If p dominates q but not the reverse, then p is *more general* than q, as p is bounded for larger graph classes: whenever q is bounded for some graph class, then this is also the case for p, but there exists an infinite family of graphs for which the reverse does not hold. If p dominates q and q dominates p, then p and q are *equivalent*. For instance, MIM-width is more general than boolean-width, clique-width, module-width, NLC-width and rank-width, all of which are equivalent [42, 114, 151, 154, 164]. The latter parameters are more general than the equivalent group of parameters branch-width, MM-width and treewidth, which are, in turn, more general than path-width [59, 155, 164]. To give a concrete example, recall that the treewidth of the class of complete graphs is unbounded, in contrast to the clique-width. More precisely, a complete graph on $n \geq 2$ vertices has treewidth $n-1$ but clique-width 2. As another example, the reason that rank-width and clique-width are equivalent is because the inequalities $\mathrm{rw}(G) \leq \mathrm{cw}(G) \leq 2^{\mathrm{rw}(G)+1} - 1$ hold for every graph G [151]. These two inequalities are essentially tight [150], and, as such, the latter example also shows that two equivalent parameters may not necessarily be linearly, or even polynomially, related.

1.2 Motivation for Width Parameters

The main computational reason for the large interest in width parameters is that many well-known NP-complete graph problems become polynomial-time solvable if some width parameter is bounded. There are a number of meta-theorems which prescribe general, sufficient conditions for a problem to be tractable on a graph class of bounded width. For treewidth and equivalent parameters, such as branch-width and MM-width, one can use the celebrated theorem of Courcelle [51]. This theorem, slightly extended from its original form, states that for every graph class of bounded treewidth, every problem definable in MSO_2 can be solved in time linear in the number of vertices of the graph.[1] In order to use this theorem, one can use the linear-time algorithm of Bodlaender [17] to verify whether a graph has treewidth at most c for any fixed constant c (that is, c is not part of the input). However, many natural graph classes, such as all those that contain graphs with arbitrarily large cliques, have unbounded treewidth.

We have noted that clique-width is more general than treewidth. This means that if we have shown that a problem can be solved in polynomial time on graphs of bounded clique-width, then it can also be solved in polynomial time on graphs of bounded treewidth. Similarly, if a problem is NP-complete for graphs of bounded treewidth, then the same holds for graphs of bounded clique-width. For graph classes of bounded clique-width, one can use several other meta-theorems. The first such result is due to Courcelle, Makowsky and Rotics [58]. They proved that graph problems that can be defined in MSO_1 are linear-time solvable on graph classes of bounded clique-width.[2] An example of such a problem is the well-known Dominating Set problem. This problem is to decide, for a graph $G = (V, E)$ and integer k, if G contains a set $S \subseteq V$ of size at most k such that every vertex of $G - S$ has at least one neighbour in S.[3]

1.3 Focus: Clique-Width

As mentioned, in this survey we focus on clique-width. Despite the usefulness of boundedness of clique-width, our understanding of clique-width itself is still very limited. For

[1] MSO_2 refers to the fragment of second order logic where quantified relation symbols must have arity at most 2, which means that, with graphs, one can quantify over both sets of vertices and sets of edges. Many graph problems can be defined using MSO_2, such as deciding whether a graph has a k-colouring (for fixed k) or a Hamiltonian path, but there are also problems that cannot be defined in this way.

[2] MSO_1 is monadic second order logic with the use of quantifiers permitted on relations of arity 1 (such as vertices), but not of arity 2 (such as edges) or more. Hence, MSO_1 is more restricted than MSO_2. We refer to [55] for more information on MSO_1 and MSO_2.

[3] Several other problems, such as List Colouring and Precolouring Extension are polynomial-time solvable on graphs of bounded treewidth [113], but stay NP-complete on graph of bounded clique-width; the latter follows from results of [113] and [20], respectively; see also [88].

example, although computing the clique-width of a graph is known to be NP-hard in general [77],[4] the complexity of computing the clique-width is open even on very restricted graph classes, such as unit interval graphs (see [107] for some partial results). To give another example, the complexity of determining whether a given graph has clique-width at most c is still open for every fixed constant $c \geq 4$. On the positive side, see [49] for a polynomial-time algorithm for $c = 3$ and [75] for a polynomial-time algorithm, for every fixed c, on graphs of bounded treewidth.

To get a better handle on clique-width, many properties of clique-width, and relationships between clique-width and other graph parameters, have been determined over the years. In particular, numerous graph classes of bounded and unbounded clique-width have been identified. This has led to several dichotomies for various families of graph classes, which state exactly which graph classes of the family have bounded or unbounded clique-width. However, determining (un)boundedness of clique-width of a graph class is usually a highly non-trivial task, as it requires a thorough understanding of the structure of graphs in the class. As such, there are still many gaps in our knowledge.

A number of results on clique-width are collected in the surveys on clique-width by Gurski [95] and Kamiński, Lozin and Milanič [116]. Gurski focuses on the behaviour of clique-width (and NLC-width) under graph operations and transformations. Kamiński, Lozin and Milanič also discuss results for special graph classes. We refer to a recent survey of Oum [150] for algorithmic and structural results on the equivalent width parameter rank-width.

1.4 Aims and Outline

In Section 2 we introduce some basic terminology and notation that we use throughout the paper. In Section 3 we formally define clique-width. In the same section we present a number of basic results on clique-width and explain two general techniques for showing that the clique-width of a graph class is bounded or unbounded. For this purpose, in the same section we also list a number of graph operations that preserve (un)boundedness of clique-width for hereditary graph classes.

A graph class is *hereditary* if it is closed under taking induced subgraphs, or equivalently, under vertex deletion. Due to its natural definition, the framework of hereditary graph classes captures many well-known graph classes, such as bipartite, chordal, planar, interval and perfect graphs; we refer to the textbook of Brandstädt, Le and Spinrad [34] for a survey. As we shall see, boundedness of clique-width has been particularly well studied for hereditary graph classes. We discuss the state-of-the-art and other known results on boundedness of clique-width for hereditary graph classes in Section 4. This is all related to our first aim: to update the paper of Kamiński, Lozin and Milanič [116] from 2009 by surveying, in a systematic way, known results and open problems on boundedness of clique-width for hereditary graph classes.

Our second aim is to discuss algorithmic implications of the results from Section 4. We do this in Section 5 by focussing on two well-known problems. We first discuss implications for the Colouring problem, which is well known to be NP-complete [133]. We focus on (hereditary) graph classes defined by two forbidden induced subgraphs. Afterwards, we consider the algorithmic consequences for the Graph Isomorphism problem. This problem can be solved in quasi-polynomial time [7]. It is not known if Graph Isomorphism

[4]It is also NP-hard to compute treewidth [4] and parameters equivalent to clique-width, such as NLC-width [98], rank-width (see [110, 149]) and boolean-width [159].

can be solved in polynomial time, but it is not NP-complete unless the polynomial hierarchy collapses [160]. As such, we define the complexity class GI, which consists of all problems that can be polynomially reduced to Graph Isomorphism and a problem in GI is GI-complete if Graph Isomorphism can be polynomially reduced to it. The Graph Isomorphism problem is of particular interest, as there are similarities between proving unboundedness of clique-width of some graph class and proving that Graph Isomorphism stays GI-complete on this class [161].

Our third aim is to discuss a conjectured relationship between boundedness of clique-width and well-quasi-orderability by the induced subgraph relation. If it can be shown that a graph class is well-quasi-ordered, we can apply several powerful results to prove further properties of the class. This is, for instance, illustrated by the Robertson-Seymour Theorem [157], which states that the set of all finite graphs is well-quasi-ordered by the minor relation. This result makes it possible to test in cubic time whether a graph belongs to some given minor-closed graph class [156] (see [112] for a quadratic algorithm). For the induced subgraph relation, it is easy to construct examples of hereditary graph classes that are not well-quasi-ordered. Take, for instance, the class of graphs of degree at most 2, which contains an infinite anti-chain, namely the set of all cycles.

If every hereditary graph class that is well-quasi-ordered by the induced subgraph relation also has bounded clique-width, then all algorithmic consequences of having bounded clique-width would also hold for being well-quasi-ordered by the induced subgraph relation. However, Lozin, Razgon and Zamaraev [142] gave a negative answer to a question of Daligault, Rao and Thomassé [69] about this implication, by presenting a hereditary graph class of unbounded clique-width that is nevertheless well-quasi-ordered by the induced subgraph relation. Their graph class can be characterized only by infinitely many forbidden induced subgraphs. This led the authors of [142] to conjecture that every finitely defined hereditary graph class that is well-quasi-ordered by the induced subgraph relation has bounded clique-width, which, if true, would still be very useful. All known results agree with this conjecture, and we survey these results in Section 6. In the same section we explain that the graph operations given in Section 3 do not preserve well-quasi-orderability by the induced subgraph relation. However, we also explain that a number of these operations can be used for a stronger property, namely well-quasi-orderability by the labelled induced subgraph relation.

In Section 7 we conclude our survey with a list of other relevant open problems. There, we also discuss some variants of clique-width, including linear clique-width and power-bounded clique-width.

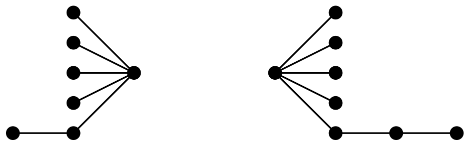
2 Preliminaries

Throughout the paper we consider only finite, undirected graphs without multiple edges or self-loops.

Let $G = (V, E)$ be a graph. The *degree* of a vertex $u \in V$ is the size of its neighbourhood $N(u) = \{v \in V \mid uv \in E\}$. For a subset $S \subseteq V$, the graph $G[S]$ denotes the subgraph of G *induced by* S, which is the graph with vertex set S and an edge between two vertices $u, v \in S$ if and only if $uv \in E$. If F is an induced subgraph of G, then we denote this by $F \subseteq_i G$. Note that $G[S]$ can be obtained from G by deleting the vertices of $V \setminus S$. The *line graph* of G is the graph with vertex set E and an edge between two vertices e_1 and e_2 if and only if e_1 and e_2 share a common end-vertex in G.

An *isomorphism* from a graph G to a graph H is a bijective mapping $f : V(G) \to V(H)$

Cambridge University Press
978-1-108-74072-2 — Surveys in Combinatorics 2019
Edited by Allan Lo , Richard Mycroft , Guillem Perarnau , Andrew Treglown
Excerpt
More Information



$K_{1,t}^{+}$ ($t = 5$ shown) $K_{1,t}^{++}$ ($t = 5$ shown)

Figure 2: The graphs $K_{1,t}^{+}$ and $K_{1,t}^{++}$.

such that there is an edge between two vertices u and v in G if and only if there is an edge between $f(u)$ and $f(v)$ in H. If such an isomorphism exists then G and H are said to be *isomorphic*. We say that G is H-*free* if G contains no induced subgraph isomorphic to H.

Let $G = (V, E)$ be a graph. A set $K \subseteq V$ is a *clique* of G and $G[K]$ is *complete* if there is an edge between every pair of vertices in K. If G is connected, then a vertex $v \in V$ is a *cut-vertex* of G if $G[V \setminus \{v\}]$ is disconnected, and a clique $K \subset V$ is a *clique cut-set* of G if $G[V \setminus K]$ is disconnected. If G is connected and has at least three vertices but no cut-vertices, then G is 2-*connected*. A maximal induced subgraph of G that has no cut-vertices is a *block* of G. If G is connected and has no clique cut-set, then G is an *atom*.

The graphs C_n, P_n and K_n denote the cycle, path and complete graph on n vertices, respectively. The *length* of a path or a cycle is the number of its edges. The *distance* between two vertices u and v in a graph G is the length of a shortest path between them. For an integer $r \geq 1$, the r-*th power* of G is the graph with vertex set $V(G)$ and an edge between two vertices u and v if and only if u and v are at distance at most r from each other in G.

If F and G are graphs with disjoint vertex sets, then the *disjoint union* of F and G is the graph $G+F = (V(F) \cup V(G), E(F) \cup E(G))$. The disjoint union of s copies of a graph G is denoted sG. A *forest* is a graph with no cycles, that is, every connected component is a *tree*. A forest is *linear* if it has no vertices of degree at least 3, or equivalently, if it is the disjoint union of paths. A *leaf* in a tree is a vertex of degree 1. In a *complete binary* tree all non-leaf vertices have degree 3.

Let S and T be disjoint vertex subsets of a graph $G = (V, E)$. A vertex v is *(anti-)complete* to T if it is (non-)adjacent to every vertex in T. Similarly, S is *(anti-)complete* to T if every vertex in S is (non-)adjacent to every vertex in T. A set of vertices M is a *module* of G if every vertex of G that is not in M is either complete or anti-complete to M. A module of G is *trivial* if it contains zero, one or all vertices of G, otherwise it is *non-trivial*. We say that G is *prime* if every module of G is trivial.

A graph G is *bipartite* if its vertex set can be partitioned into two (possibly empty) subsets X and Y such that every edge of G has one end-vertex in X and the other one in Y. If X is complete to Y, then G is *complete bipartite*. For two non-negative integers s and t, we denote the complete bipartite graph with partition classes of size s and t, respectively, by $K_{s,t}$. The graph $K_{1,t}$ is also known as the $(t+1)$-vertex *star*. The *subdivision* of an edge uv in a graph replaces uv by a new vertex w and edges uw and vw. We let $K_{1,t}^{+}$ and $K_{1,t}^{++}$ be the graphs obtained from $K_{1,t}$ by subdividing one of its edges once or twice, respectively.

A graph is *complete r-partite*, for some $r \geq 1$, if its vertex set can be partitioned into r independent sets $V_1, \ldots, V_r$ such that there exists an edge between two vertices u and v

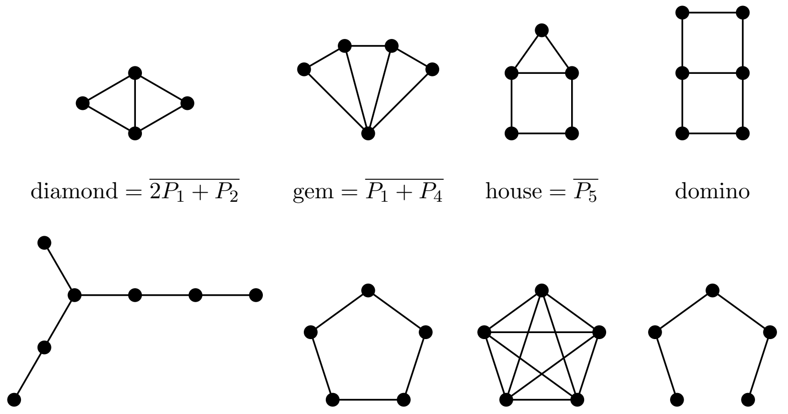


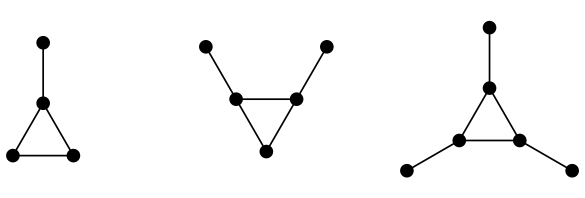
Figure 3: Some common graphs used throughout the paper.

if and only if u and v do not belong to the same set V_i. Note that a non-empty graph is complete r-partite for some $r \geq 1$ if and only if it is $(P_1 + P_2)$-free.

Let $G = (V, E)$ be a graph. Its *complement* $\overline{G}$ is the graph with vertex set V and an edge between two vertices u and v if and only if uv is not an edge of G. We say that G is *self-complementary* if G is isomorphic to $\overline{G}$. The complement of a bipartite graph is a *co-bipartite* graph.

The graphs $K_{1,3}$, $\overline{2P_1 + P_2}$, $\overline{P_1 + P_4}$, and $\overline{P_5}$ are also known as the *claw*, *diamond*, *gem*, and *house*, respectively. The latter three graphs are shown in Figure 3, along with the *domino*. The graph $S_{h,i,j}$, for $1 \leq h \leq i \leq j$, denotes the *subdivided claw*, which is the tree with one vertex x of degree 3 and exactly three leaves, which are of distance h, i and j from x, respectively. Note that $S_{1,1,1} = K_{1,3}$, $S_{1,1,2} = K_{1,3}^{+}$ and $S_{1,1,3} = K_{1,3}^{++}$. See Figure 3 for an example. We let $\mathcal{S}$ be the class of graphs every connected component of which is either a subdivided claw or a path on at least one vertex. The graph $T_{h,i,j}$ with $0 \leq h \leq i \leq j$ denotes the triangle with pendant paths of length h, i and j, respectively. That is, $T_{h,i,j}$ is the graph with vertices $a_0, \ldots, a_h$, $b_0, \ldots, b_i$ and $c_0, \ldots, c_j$ and edges a_0b_0, b_0c_0, c_0a_0, a_pa_{p+1} for $p \in \{0, \ldots, h-1\}$, b_pb_{p+1} for $p \in \{0, \ldots, i-1\}$ and c_pc_{p+1} for $p \in \{0, \ldots, j-1\}$. Note that $T_{0,0,0} = C_3 = K_3$. The graphs $T_{0,0,1} = \overline{P_1 + P_3}$, $T_{0,1,1}$, $T_{1,1,1}$ and $T_{0,0,2}$ are also known as the *paw*, *bull*, *net* and *hammer*, respectively; see also Figure 4. Also note that $T_{h,i,j}$ is the line graph of $S_{h+1,i+1,j+1}$. We let $\mathcal{T}$ be the class of graphs that are the line graphs of graphs in $\mathcal{S}$. Note that $\mathcal{T}$ contains every graph $T_{h,i,j}$ and every path (as the line graph of P_t is P_{t-1} for $t \geq 2$).

Let $G = (V, E)$ be a graph. For an induced subgraph $F \subseteq_i G$, the *subgraph complementation* operation, which acts on G with respect to F, replaces every edge in F by a non-edge, and vice versa. If we apply this operation on G with respect to G itself, then we obtain the complement $\overline{G}$ of G. For two disjoint vertex subsets S and T in G, the *bipartite complementation* operation, which acts on G with respect to S and T, replaces every edge



$T_{0,0,1}$ = paw $T_{0,1,1}$ = bull $T_{1,1,1}$ = net $T_{0,0,2}$ = hammer

Figure 4: Examples of graphs $T_{h,i,j}$.

with one end-vertex in S and the other one in T by a non-edge and vice versa. We note that applying a bipartite complementation is equivalent to applying a sequence of three consecutive subgraph complementations, namely on $G[S \cup T]$, $G[S]$ and $G[T]$.

Let $\mathcal{G}$ be a graph class. Denote the number of labelled graphs on n vertices in $\mathcal{G}$ by g_n. Then $\mathcal{G}$ is *superfactorial* if there does not exist a constant c such that $g_n \leq n^{cn}$ for every n.

Recall that a graph class is hereditary if it is closed under taking induced subgraphs. It is not difficult to see that a graph class $\mathcal{G}$ is hereditary if and only if $\mathcal{G}$ can be characterized by a unique set $\mathcal{F}_\mathcal{G}$ of minimal forbidden induced subgraphs. A hereditary graph class $\mathcal{G}$ is *finitely defined* if $\mathcal{F}_\mathcal{G}$ is finite. We note, however, that the set $\mathcal{F}_\mathcal{G}$ may have infinite size. For example, if $\mathcal{G}$ is the class of bipartite graphs, then $\mathcal{F}_\mathcal{G} = \{C_3, C_5, C_7, \ldots\}$. If $\mathcal{F}$ is a set of graphs, we say that a graph G is $\mathcal{F}$-free if G does not contain any graph in $\mathcal{F}$ as an induced subgraph. In particular, this means that if a graph class $\mathcal{G}$ is hereditary, then $\mathcal{G}$ is exactly the class of $\mathcal{F}_\mathcal{G}$-free graphs. If $\mathcal{F} = \{H_1, H_2, \ldots\}$ or $\{H_1, H_2, \ldots, H_p\}$ for some $p \geq 0$, we may also describe a graph G as being $(H_1, H_2, \ldots)$-free or $(H_1, H_2, \ldots, H_p)$-free, respectively, rather than $\mathcal{F}$-free; recall that if $\mathcal{F} = \{H_1\}$ we may write H_1-free instead.

Observation 2.1. *Let $\mathcal{H}$ and $\mathcal{H}^*$ be sets of graphs. The class of $\mathcal{H}$-free graphs is contained in the class of $\mathcal{H}^*$-free graphs if and only if for every graph $H^* \in \mathcal{H}^*$, the set $\mathcal{H}$ contains an induced subgraph of H^*.*

Suppose $\mathcal{H}$ and $\mathcal{H}^*$ are sets of graphs such that for every graph $H^* \in \mathcal{H}^*$, the set $\mathcal{H}$ contains an induced subgraph of H^*. Observation 2.1 implies that any graph problem that is polynomial-time solvable for $\mathcal{H}^*$-free graphs is also polynomial-time solvable for $\mathcal{H}$-free graphs, and any graph problem that is NP-complete for $\mathcal{H}$-free graphs is also NP-complete for $\mathcal{H}^*$-free graphs.

We define the *complement* of a hereditary graph class $\mathcal{G}$ as $\overline{\mathcal{G}} = \{\overline{G} \mid G \in \mathcal{G}\}$. Then $\mathcal{G}$ is *closed under complementation* if $\mathcal{G} = \overline{\mathcal{G}}$. As $\mathcal{F}_\mathcal{G}$ is the unique minimal set of forbidden induced subgraphs for $\mathcal{G}$, we can make the following observation.

Observation 2.2. *A hereditary graph class $\mathcal{G}$ is closed under complementation if and only if $\mathcal{F}_\mathcal{G}$ is closed under complementation.*

Let G be a graph. The *contraction* of an edge uv replaces u and v and their incident edges by a new vertex w and edges wy if and only if either uy or vy was an edge in G (without creating multiple edges or self-loops). Let u be a vertex with exactly two neighbours v, w, which in addition are non-adjacent. The *vertex dissolution* of u removes u, uv and uw, and adds the edge vw. Note that vertex dissolution is a special type of edge contraction, and it is the reverse operation of an edge subdivision (recall that the latter operation replaces an edge uv by a new vertex w with edges uw and vw).

Let G and H be graphs. The graph H is a *subgraph* of G if G can be modified into H by a sequence of vertex deletions and edge deletions. We can define other containment relations using the graph operations defined above. We say that G contains H as a *minor* if G can be modified into H by a sequence of edge contractions, edge deletions and vertex deletions, as a *topological minor* if G can be modified into H by a sequence of vertex dissolutions, edge deletions and vertex deletions, as an *induced minor* if G can be modified into H by a sequence of edge contractions and vertex deletions, and as an *induced topological minor* if G can be modified into H by a sequence of vertex dissolutions and vertex deletions. Let $\{H_1, \ldots, H_p\}$ be a set of graphs. If G does not contain any of the graphs $H_1, \ldots, H_p$ as a subgraph, then G is $(H_1, \ldots, H_p)$*-subgraph-free*. We define the terms $(H_1, \ldots, H_p)$*-minor-free*, $(H_1, \ldots, H_p)$*-topological-minor-free*, $(H_1, \ldots, H_p)$*-induced-minor-free*, and $(H_1, \ldots, H_p)$*-induced-topological-minor-free* analogously. Note that graph classes defined by some set of forbidden subgraphs, minors, topological minors, induced minors, or induced topological minors are hereditary, as they are all closed under vertex deletion.

Example 2.3. A graph is *planar* if it can be embedded in the plane in such a way that any two edges only intersect with each other at their end-vertices. It is well known that the class of planar graphs can be characterized by a set of forbidden minors: Wagner's Theorem [165] states that a graph is planar if and only if it is $(K_{3,3}, K_5)$-minor-free.

We will also need the following folklore observation (see, for example, [90]).

Observation 2.4. *For every $F \in \mathcal{S}$, a graph is F-subgraph-free if and only if it is F-minor-free.*

A k*-colouring* of a graph G is a mapping $c : V \to \{1, \ldots, k\}$ such that $c(u) \neq c(v)$ whenever u and v are adjacent vertices. The *chromatic number* of G is the smallest k such that G has a k-colouring. The *clique number* of G is the size of a largest clique of G.

A graph G is *perfect* if, for every $H \subseteq_i G$, the chromatic number of H is equal to the clique number of H. The Strong Perfect Graph Theorem [45] states that G is perfect if and only if G is $(C_5, C_7, C_9, \ldots)$-free and $(\overline{C_7}, \overline{C_9}, \ldots)$-free. A graph G is *chordal* if it is $(C_4, C_5, C_6, \ldots)$-free and *weakly chordal* if it is $(C_5, C_6, C_7, \ldots)$-free and $(\overline{C_6}, \overline{C_7}, \ldots)$-free. A graph G is a *split graph* if it has a *split partition*, that is, a partition of its vertex set into two (possibly empty) sets K and I, where K is a clique and I is an independent set. It is known that a graph is split if and only if it is $(C_4, C_5, 2P_2)$-free [78]. A graph G is a *permutation graph* if line segments connecting two parallel lines can be associated to its vertices in such a way that two vertices of G are adjacent if and only if the two corresponding line segments intersect. A graph G is a *permutation split graph* if it is both permutation and split, and G is a *permutation bipartite graph* if it is both permutation and bipartite. A graph G is *chordal bipartite* if it is $(C_3, C_5, C_6, C_7, \ldots)$-free. A graph G is *distance-hereditary* if the distance between any two vertices u and v in any connected induced subgraph of G is the same as the distance of u and v in G. Equivalently, a graph is distance-hereditary if and only if it is $(\text{domino}, \text{gem}, \text{house}, C_5, C_6, C_7, \ldots)$-free [9]. A graph is *(unit) interval* if it has a representation in which each vertex u corresponds to an interval I_u (of unit length) of the line such that two vertices u and v are adjacent if and only if $I_u \cap I_v \neq \emptyset$.

We make the following observation. A number of inclusions in Observation 2.5 follow immediately from the definitions and the Strong Perfect Graph Theorem. For the remaining inclusions we refer to [34].

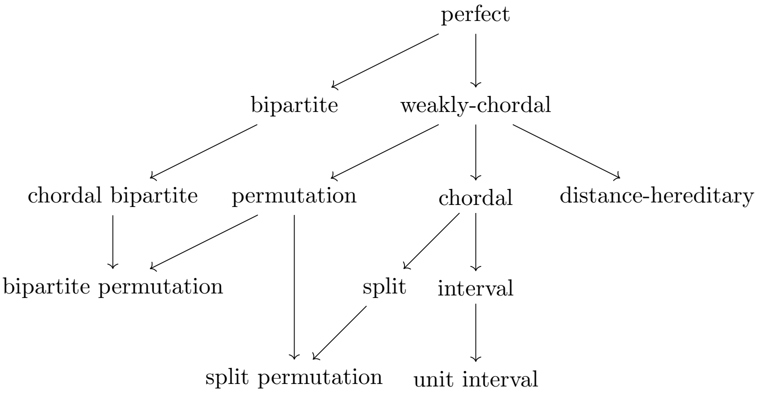


Figure 5: The inclusion relations between well-known classes mentioned in the paper. An arrow from one class to another indicates that the first class contains the second.

Observation 2.5. *The following statements hold:*

1. *every split graph is chordal,*
2. *every (unit) interval graph is chordal,*
3. *every chordal graph is weakly chordal,*
4. *every (bipartite or split) permutation graph is weakly chordal,*
5. *every distance-hereditary graph is weakly chordal,*
6. *every weakly chordal graph is perfect,*
7. *every bipartite permutation graph is chordal bipartite, and*
8. *every (chordal) bipartite graph is perfect.*

The containments listed in Observation 2.5 (and those that follow from them by transitivity) are also displayed Figure 5. It is not difficult to construct counterexamples for the other containments. Indeed, for pairs of classes above for which we have listed the minimal forbidden induced subgraph characterizations, these characterizations immediately provide such counterexamples.

We now introduce the notion of treewidth formally. Recall from Section 1 that treewidth expresses to what extent a graph is "tree-like". A *tree decomposition* of a graph G is a pair $(T, \mathcal{X})$ where T is a tree and $\mathcal{X} = \{X_i \mid i \in V(T)\}$ is a collection of subsets of $V(G)$, such that the following three conditions hold:

(i) $\bigcup_{i \in V(T)} X_i = V(G)$

(ii) for every edge $xy \in E(G)$, there is an $i \in V(T)$ such that $x, y \in X_i$ and

(iii) for every $x \in V(G)$, the set $\{i \in V(T) \mid x \in X_i\}$ induces a connected subtree of T.

The *width* of the tree decomposition $(T, \mathcal{X})$ is $\max\{|X_i| - 1 \mid i \in V(T)\}$, and the *treewidth* $\mathrm{tw}(G)$ of G is the minimum width over all tree decompositions of G. If T is a path, then (X, T) is a *path decomposition* of G. The *path-width* $\mathrm{pw}(G)$ of G is the minimum width over all path decompositions of G.