University of Bergen
Department of Informatics

MASTER THESIS

Róbert Sasák

Comparing 17 graph parameters

Supervisor: Jan Arne Telle

2010

I would like to thank my supervisor Jan Arne Telle for introducing me to the subject of my thesis, for many helpful discussions and valuable advices. I am indebted to Martin Vatshelle for his comments that always point me in right direction.

I hereby proclaim that I worked out this thesis on my own, using only the resources stated. I agree that the thesis may be publicly available.

Bergen, 1st of August 2010 Róbert Sasák

Contents

Title: Comparing 17 graph parameters
Author: Róbert Sasák, Robert.Sasak@student.uib.no
Department: Department of Informatics
Supervisor: Jan Arne Telle, Jan.Arne.Telle@ii.uib.no

Abstract: Many parametrized problems were decided to be FPT or W-hard. However, there is still thousands of problems and parameters for which we do not know yet whether are FPT or W-hard. In this thesis, we provide a tool for extending existing results to additional parametrized problems.

We use the comparison relation for comparing graph parameters, i.e. we say that parameter p_1 is bounded by parameter p_2 if exists a function f that for every graph G holds $p_1(G) \leq f(p_2(G))$. This allows us to extend results for parametrized problem π in two ways. If problem π parametrized by p_1 is FPT then also π parametrized by p_2 is FPT. Or wise-versa, if problem π parametrized by p_2 is W-hard then also π parametrized by p_1 is W-hard.

Moreover, we show whether is a parameter bounded by another parameter for the 17 graph parameters: path-width, tree-width, branch-width, clique-width, rank-width, boolean-width, maximum independent set, minimum dominating set, vertex cover number, maximum clique, chromatic number, maximum matching, maximum induced matching, cut-width, carving-width, degeneracy and tree-depth. To avoid all 272 different comparisons, we introduce methodology for examining graph parameters which rapidly reduce number of comparisons. And finally, we provide comparison diagram for all 17 parameters.

Keywords: comparison, width parameter, FPT, W-hard

Chapter 1

Introduction

Since Stephen Cook and Leonid Levin in 1970s proved that the boolean satisfiability problem is NP-Complete, many other problems were decided to be either P or NP-hard. When dealing with problems in real world we might get lucky and our problem could be polynomial time solvable. Often thus, there is only an exponential algorithm (unless P=NP). However, in practice, even NP-hard problem has to be solved. So we revisit our problem and if we are lucky we may find out that our input is restricted in some way.

Example Travelling salesman is a famous problem where the salesman wants to find a shortest tour that visits each city exactly once. Trivial depth-first search would lead to exponential complexity $O(n^n)$. However in real world we can assume that each city has at most k outgoing roads with $k \ll N$. This additional restriction gives us a running time $O(n^k)$. The running time is still exponential. However the problem can now be solved on a computer for some small value k.

1.1 Parametrized problems

It comes very natural to classify NP-hard problems on a finer scale based on an additional parameter k.

Formally, we define a parametrized problem as a language $L \subseteq \Sigma^* \times \mathbb{N}$, where Σ is a finite alphabet and $\mathbb{N}$ is an integer number (parameter):

Input: (x, k)
Problem: Decide if $(x, k) \in L$

Now we can analyse running time not only as a function of the input size $f(|x|)$, but as a function of the input size and the parameter $f(|x|, k)$. For example, it is easy to see the difference between these two running times: $O(|x|^k)$ and $O(2^k \cdot |x|^3)$. The complexity classes FPT and W-hierarchy refines parametrized problems according to their running time functions.

1.2 FPT and W-hard

In previous section we introduced parametrized problems. Analogical as we partition general problems into the classes P and NP, we can partition parametrized problems into FPT and W-hard. Class FPT corresponds to class P in a sense that problems in those classes are practically solvable on computer. In contrast with W-hard and NP-hard problems which are, due to exponential running time, impractical.

Definition FPT is a class of all parametrized problems (x, k) that can be decided in running time $f(k) \cdot |x|^{O(1)}$. In other words, the running time can be any (exponential) function of the parameter k but polynomial in the input size. FPT is an abbreviation for fixed-parameter tractable.

Example Parametrized version of a Vertex Cover problem.

Input: Graph $G = (V, E)$, positive integer k
Problem: Is there a subset $X \subseteq V$ of cardinality k s.t.

$$\forall v \in V \setminus X \quad \exists u \in X : (u, v) \in E$$

W-hard is a class of all parametrized problems believe not to be in FPT.

1.3 Comparing parameters

In this thesis we focus on 17 graph parameters. However, the introduced methodology can be used to systematize any parameters.

While designing a new parametrized graph algorithm, there is the crucial part of choosing a right parameter. In general, a graph parameter can be any function that assigns a non-negative number to a graph.

To make some system in parameters, we define partial order relation $\preceq$ on all parameters.

Definition *($\preceq$)* Let p_1 and p_2 be two graph parameters. Then $p_1 \preceq p_2$ if and only if

$$\exists f \, \forall G : \quad p_1(G) \leq f(p_2(G))$$

We also say that p_1 is bounded by (some function of) p_2. It is easy to see that the relation $\preceq$ is a partial order.

To simplifies the notation, we also intuitively define relations $\approx$, $\not\preceq$ and $|$.

Definition *($\approx$)* We write $p_1 \approx p_2$ if and only if $p_1 \preceq p_2 \wedge p_2 \preceq p_1$.

Definition *($\not\preceq$)* We write $p_1 \not\preceq p_2$ if and only if $\neg(p_1 \preceq p_2)$.

Definition *($\prec$)* We write $p_1 \prec p_2$ if and only if $p_1 \preceq p_2 \wedge p_2 \not\preceq p_1$.

Definition *($|$)* We write $p_1 | p_2$ if and only if $p_1 \not\preceq p_2 \wedge p_2 \not\preceq p_1$.

The following two theorems show why the relation $\preceq$ plays an important role when examining FPT problems.

Theorem 1.1 *Let p and q be graph parameters. If $p \preceq q$ and problem π parametrized by p is in FPT then π parametrized by q is also in FPT i.e.*

$$p \preceq q \wedge (\pi, p) \in FPT \implies (\pi, q) \in FPT$$

Proof Recall definition of $p \preceq q$:

$$\exists g \, \forall G : \quad p(G) \leq g(q(G))$$

From definition of FPT, we know there exists an algorithm A that can decide problem (π, p) for any graph G with fixed parameter p in time $h(p) \cdot n^{O(1)}$. Since $p \leq g(q)$ for any graph then we can use the algorithm A and decide $< \pi, q >$ in running time $h(g(q)) \cdot n^{O(1)}$. Let f be $h \circ g$ and we get running time $f(q) \cdot n^{O(1)}$. Therefore problem π parametrized by q is in FPT.

Theorem 1.2 *Let p and q be parameters. If $p \preceq q$ and problem π parametrized by q is W-hard then π parametrized by p is also W-hard i.e.*

$$p \preceq q \wedge (\pi, q) \in W\text{-}hard \implies (\pi, p) \in W\text{-}hard$$

Proof Assume for contradiction that $(\pi, p) \in$ FPT. Since $p \preceq q$ we can apply previous theorem and get $(\pi, q) \in$ FPT. That is contradiction.

However, proper proof requires deep knowledges from Turing machines and complexity theory. For further reading, we refer to [20].

Comparison diagram

To visualize relation $\preceq$ we use comparison diagram. In a comparison diagram, two parameters p and q are connected by an oriented edge from q to p if and only if $p \preceq q$.

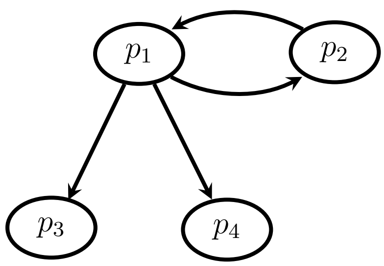


Figure 1.1: Comparison diagram of relations $p_1 \approx p_2$, $p_3 \prec p_1$, $p_4 \prec p_1$.

Example Consider 4 parameters p_1, p_2, p_3, p_4 and following relations: $p_1 \approx p_2$, $p_3 \prec p_1$, $p_4 \prec p_1$. The comparison diagram is in figure 1.1. The comparison diagram is very useful not only to visualize but further more to explore transitive relations. For example, p_3 and p_4 are also bounded by p_2. Moreover when ever $(\pi, p_4) \in$ FPT then $(\pi, p_1), (\pi, p_2) \in$ FPT.

Relation $\preceq$ gives us a powerful tool to decide whether a parametrized problem is fixed-parameter tractable. In this thesis, we give a complete comparison diagram for 17 graph parameters. Many parametrized problems are already known to be in FPT or W-hard. The diagram can be used to extend results for more parameters without actually examining each pair consisting of problem and parameter.

1.4 Notation

Unless specified differently, in this work we consider by G simple (no loops and multiple edges), undirected, connected graph with vertex set V and edge set E. n stands for size of a graph, the number of vertices in a graph.

Even though we focus only on connected graphs it is easy extend the given results for disconnected graphs by appropriately combining results for each component of graph.

While we use notion of vertex for original graph, nodes refer to vertices in decomposition tree. Furthermore, decomposition usually maps one or more vertices or edges to bags.

Sub-cubic tree is a tree in which every vertex has degree 1 or 3.

Formally, we define paths, stars, trees, cliques and grids in the following way. In all definitions, we assume graph G on n vertices with vertices labelled from v_1 to v_n.

Definition *(Path)* Graph G is path if and only if $E = \bigcup_{i \in \{1,\dots,n-1\}} (v_i, v_{i+1})$

Definition *(Star)* Graph G is a start if ant only if $E = \bigcup_{i \in \{2,\dots,n\}} (v_1, v_i)$

Definition *(Tree)* Graph G is a tree if and only if has no cycles.

Definition *(Clique)* Graph G is a clique if and only if $\forall u \in V\ \forall v \in V\ u \neq v : (u, v) \in E$.

Definition *(Grid)* Graph G is a (square) grid if and only if $n = k^2$ where k is an integer and

$$E = \bigcup_{i \in \{1,\dots,k-1\}} \bigcup_{j \in \{1,\dots,k-1\}} \{(u_{i,j}, u_{i+1,j}), (u_{i,j}, u_{i,j+1})\}$$

where $u_{i,j}$ correspond to vertex $v_{(i-1)*k+j}$.

1.5 Thesis outline

The aim of the thesis is to provide complete comparison diagram of all 17 parameters listed in Table 1.1. This task involves to decide for every two parameters p, q whether:

1. $p \approx q$

Section	Parameter	Notation
2.1	Path-width	pw
2.2	Tree-width	tw
2.3	Branch-width	bw
2.4	Clique-width	cw
2.5	Rank-width	rw
2.6	Boolean-width	bw
2.7	Maximum independent set	IS
2.8	Minimum dominating set	γ
2.9	Vertex cover number	VCN
2.10	Maximum clique	ω
2.11	Chromatic number	χ
2.12	Maximum matching	MM
2.13	Maximum induced matching	MiM
2.14	Cut-width	cutw
2.15	Carving-width	carw
2.16	Degeneracy	δD
2.17	Tree-depth	td

Table 1.1: List of all 17 parameters and their abbreviations. Definitions are in corresponding sections.

2. $p \prec q$

3. $q \prec p$

4. $p \mid q$

That leads to 17×16 different comparisons. However, transitivity decreases the number of necessary comparisons. In order to systematize comparison of parameters we introduce the following process:

Chapter 2 First, we examine each parameter on five graph classes i.e. paths, stars, trees, cliques and grids. For each parameter and each class we show whether the parameter is bounded by a constant or not (unbounded). We choose those graph classes for two reasons. Parameters are easy to examine on those classes. The classes represent different basic structures of a graph.

Chapter 3 Consider two parameters p_1 and p_2. Let p_1 be bounded on grids and p_2 be unbounded. This provides the useful information $p_2 \not\preceq p_1$.

Therefore we identify parameters that behave similar on the five graph classes and place them in groups. We consider two parameters as similar if for each of the 5 classes both parameters are either bounded nor unbounded.

So we can generalized previous fact to the level of the groups.

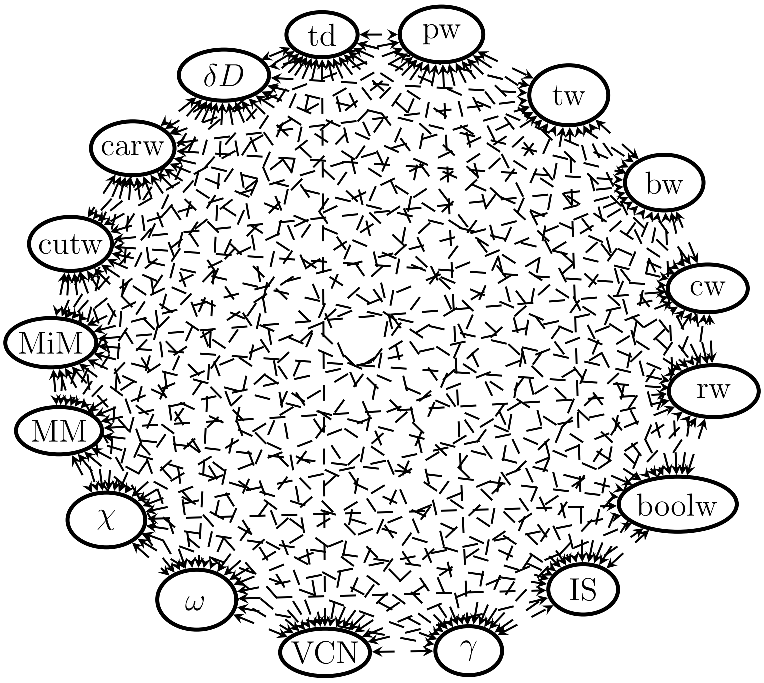


Figure 1.2: 17 parameters and all 17×16 different comparisons that we show in this thesis.

Chapter 4 We compare all parameters within groups and get partial comparison diagram for each group of parameters. As it turns out later, these partial comparison diagrams correspond to parts of the complete comparison diagram.

Chapter 5 In the previous step, we identified partial comparison diagrams inside the groups. However, there is still a lot of remaining comparisons to examine in this chapter.

Chapter 2

Parameters overview

In this chapter we introduce 17 graph parameters. We determine the size of each parameter on five graph classes: paths, stars, trees, cliques and grids.

We are interested in whether a parameter is bounded on the graph class by a constant or if it depends on the graph size, in other words unbounded on the graph class.

To determine that a parameter is unbounded we need to show a lower bound; all graphs from a particular sub class have parameter size bigger than some function of $|V|$. It is usually easier to show that a parameter is bounded. We simply provide a description of how to obtain a representation with bounded parameter for any graph from the class.

For each parameter p and each graph class $\mathcal{C}$ we show one of the two options:

- Parameter p is bounded by a constant e.g. $\exists k\ \forall G \in \mathcal{C} :\ p(G) \leq k$. We also claim that k is a tight bound (except some trivial cases).

- Parameter p is not bounded by constant e.g. $\forall k\ \exists G \in \mathcal{C} :\ p(G) > k$. In other words, for some graphs parameter p is bigger than some function of $|V(G)|$. To show this, we provide sub class $\mathcal{C}' \subseteq \mathcal{C}$ and prove $\exists f\ \forall G \in \mathcal{C}' :\ p(G) \geq f(|V(G)|)$. Moreover, in most cases we show that this is a tight lower bound.

The chapter ends with a table that summarized the behaviour of the 17 parameters on the five graph classes.

2.1 Path-width

Definition *(Path-width)* Path decomposition of a graph G is a pair (P, X) where P is a path, $X = \{X_1, X_2, ..., X_q\}$ is a family of vertex subsets and satisfies:

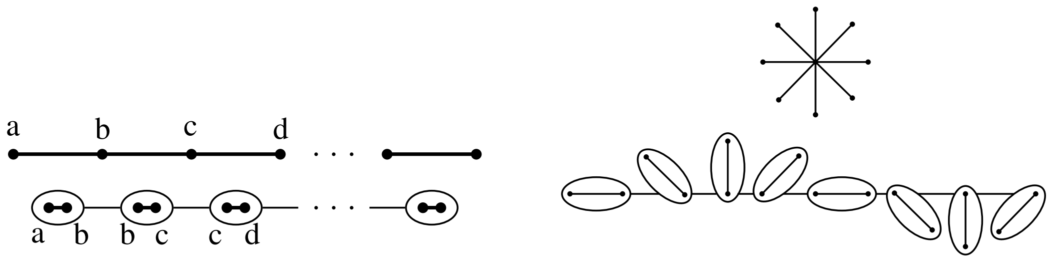
- $\forall (u, v) \in E(G) \quad \exists p : u, v \in X_p$

- $\forall v \in V(G)$ set of vertices $\{p | v \in X_p\}$ is a connected sub path of P.

- $\bigcup_{p \in \{1,...,q\}} X_p = V(G)$

Width of the path decomposition (P, X) is $\max(|X_p| - 1)_{p \in \{1,\ldots,q\}}$. Path-width is a minimal width among all possible path decomposition of graph G, denoted $\mathrm{pw}(G)$.

Paths, Stars



(a) Path decomposition of a path.

(b) Path decomposition of a star.

Figure 2.1a shows path decomposition of a path. All bags contain exactly two vertices therefore path-width is 1.

Similar construction works for decomposition of a star shown in figure 2.1b. Path width of any star is also 1.

Trees

Theorem 2.1 *[1] Path of a complete binary tree with a height h is at least $\frac{h}{2}$.*

Therefore path-width is unbounded on a class of trees.

Cliques

Theorem 2.2 shows tree-width of a clique. Similar approach can be applied for path-width and show that path-width of clique on n vertices is $n - 1$.

Grids

Theorem 5.3 shows that $\forall G\ tw(G) \leq pw(G)$. Since tree-width of a grid on n vertices is $\sqrt{n}$ then path-width is at least $\sqrt{n}$.

2.2 Tree-width

Definition *(Tree-width)* Tree decomposition of a graph G is a pair (T, X) where T is a tree and $X = \{X_1, X_2, \ldots, X_q\}$ is a family of vertices and satisfied:

- $\forall (u, v) \in E(G) \quad \exists t : u, v \in X_t$
- $\forall v \in V(G)$ set of vertices $\{t | v \in X_t\}$ is a connected sub tree of T
- $\bigcup_{t \in \{1,\ldots,q\}} X_t = V(G)$

Width of the tree decomposition (T, X) is $\max(|X_t| - 1)_{t \in \{1,\dots,q\}}$. The tree-width is a minimal width among all possible tree decomposition of the graph G, denoted $\mathrm{tw}(G)$.

Paths, Stars, Trees

Tree-width measure how graph "looks" like tree and it is not hard to see that tree-width of tree is one.

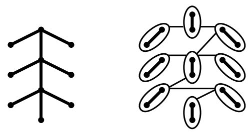


Figure 2.1: Example of a tree decomposition of a tree.

Cliques

Theorem 2.2 *Let G be a graph that contains a clique C. Any tree decomposition (T, X) contains a bag X_i which contain all vertices of the clique C.*

Proof We prove it by induction on the clique size.

$k = 1, 2$ Trivial.

$k > 2$ Let a be a vertex of the clique C. By induction we know that there exists bag X_a that contains all vertices of clique except a $(V(C)\setminus\{a\})$. Since k is at least 3, we can similarly find vertices b, c and corresponding bags X_b, X_c.

In a tree T, there exists unique path between any two bags. Let X_v be a bag where intersect all tree paths between bags X_a, X_b, X_c. From third condition, we know that X_v has all vertices that have X_a and X_b in common:

$$X_v \supseteq X_a \cap X_b \supseteq V(C)\setminus\{a, b\}$$

Similarly it works for pairs X_a, X_c and X_b, X_c.

$$X_v \supseteq X_a \cap X_c \supseteq V(C)\setminus\{a, c\}$$

$$X_v \supseteq X_b \cap X_c \supseteq V(C)\setminus\{b, c\}$$

It is easy to see that X_v contains all vertices the clique.

Corollary 2.3 *Tree-width of clique on n vertices is $n - 1$.*

Grids

Theorem 2.4 *[2] Tree-width of a grid ($\sqrt{n} \times \sqrt{n}$) on n vertices is $\sqrt{n}$.*

2.3 Branch-width

Definition *(Branch-width)* Branch decomposition of a graph G is a pair (T, χ). Where:

- T is a sub-cubic tree (all nodes have degree 1 or 3).
- $\chi : A(T) \to V$ is a bijection, mapping leaves of T to into vertices of G.

Any edge (u, v) of the tree divides the tree into two components and divides the set of edges of G into two parts X, $E \backslash X$, consisting of edges mapped to the leaves of each component. Width of the edge (u, v) is the connectivity value of $\lambda_G(X)$. The connectivity value $\lambda_G(X)$ is defined as the number of vertices of G that is incident both with an edge in X and with an edge in $E \backslash X$.

The width of the decomposition (T, χ) is maximum width of its edges. Branch-width of the graph G is the minimum over all branch-decompositions of G, denoted $\mathrm{bw}(G)$.

Paths, Stars and Trees

Following theorem gives full description of graphs with small branch-width.

Theorem 2.5 (Robertson and Seymour) *[21] A graph G has branch-width of size:*

- *0 if and only if every component of G has at most one edge.*
- *at most 1 if and only if every component of G has at most one vertex of degree at least two.*
- *at most 2 if and only if G has no K_4 minor.*

Corollary 2.6 *Branch-width of a path on at least 4 vertices is 2.*

Corollary 2.7 *Branch-width of a star is 1.*

Corollary 2.8 *Branch-width of a tree on at least two internal vertices is 2.*

Cliques

Theorem 2.9 *[21] Clique on n vertices with at least 3 vertices has branch-width $\lceil \frac{2}{3}n \rceil$.*

Grids

Theorem 2.10 *[21] Branch-width of grid $\sqrt{n} \times \sqrt{n}$ on n vertices is $\sqrt{n}$.*

2.4 Clique-width

Definition *(Clique-width)* Clique-width of a graph G is the minimum number of labels needed to construct G with the following operations:

- Creation of a new vertex with label i, denoted $\bullet_i$.
- Disjoint union of two labelled graphs G and H, denoted $G \oplus H$.
- Joining by an edge every vertex labelled i to every vertex labelled j, denoted $\eta_{i,j}(G)$. Note that $i \neq j$.
- Renaming all vertices with label i to label j, denoted $\rho_{i \to j}(G)$.

Sequence of operations that use at most k labels to build a graph is called k-expression.

Paths

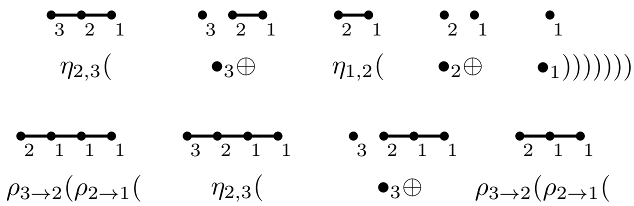


Figure 2.2: 3-expression for building path P_4.

It is obvious that graphs with a clique-width 1 do not have edges. Graphs with clique-width 2 do not have P_4 as sub graph [8]. However any arbitrary long path can be built with a 3-expression as follows:

$$\rho_{3\to 2}(\rho_{2\to 1}(\eta_{2,3}(\bullet_3 \oplus ...\rho_{3\to 2}(\rho_{2\to 1}(\eta_{2,3}(\bullet_3 \oplus \eta_{1,2}(\bullet_1 \oplus \bullet_2))))...)))$$

Stars

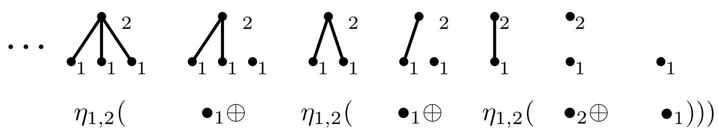


Figure 2.3: 2-expression for building star.

Any star with more than two vertices can be built with the 2-expression as follows: $\eta_{1,2}(\bullet_1 \oplus ...\eta_{1,2}(\bullet_1 \oplus \eta_{1,2}(\bullet_1 \oplus \bullet_2))...)$

Trees

We show construction of a tree with 3-expression by induction on number of vertices.

$k = 1$ We label one vertex tree with $\bullet_2$.

$k > 1$ Let T be a tree with k vertices and $v \in T$. Denote neighbours of v as $t_1, ..., t_p$ and $T_1, ..., T_p$ components of $T \backslash v$ where $\forall i \in \{1, ..., p\}$ $t_i \in T_i$. By induction step, we know that there exists 3-expression for tree T_i. Moreover, t_i is labelled with 2 and remaining vertices in t_i are labelled with 1. It is easy to see that following 3-expression create the tree T:

$$\rho_{3\to2}(\rho_{2\to1}(\eta_{2,3}(\bullet_3 \oplus T_1 \oplus ... \oplus T_p)))$$

Finally, it is important that to relabel vertices, so they meet the induction step.

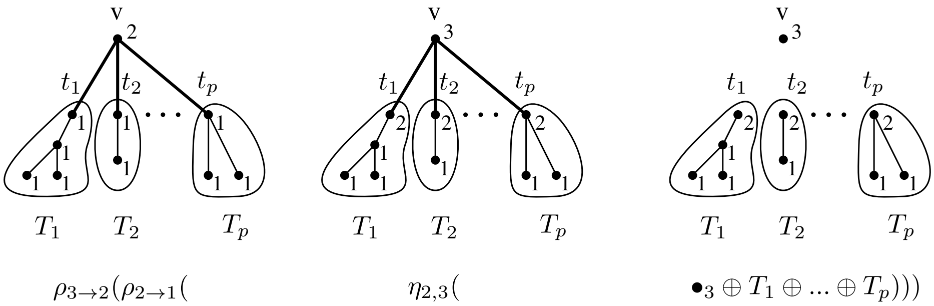


Figure 2.4: Induction step of building a tree.

Cliques

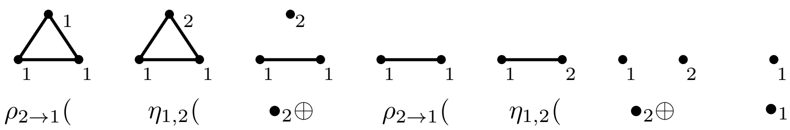


Figure 2.5: 2-expression for building clique K_3.

Arbitrary large clique can be built with the following 2-expression:

$$\rho_{2\to1}(\eta_{1,2}(\bullet_2 \oplus ...\rho_{2\to1}(\eta_{1,2}(\bullet_2 \oplus (\rho_{2\to1}(\eta_{1,2}(\bullet_2 \oplus \bullet_1)))))...))$$

Grids

Theorem 2.11 *[10] Clique-width of a grid* $\sqrt{n} \times \sqrt{n}$ *on* $n \geq 9$ *vertices is* $\sqrt{n} + 1$.

2.5 Rank-width

Definition *(Rank-width)* Let cut-rank of set $A \subseteq V(G)$ be defined as rank of matrix M where M is a $A \times (V(G) \backslash A)$ matrix over $\mathbf{Z_2}$ s.t.

$$M_{xy} = \begin{cases} 1 & \text{if } xy \in E(G) \\ 0 & \text{otherwise} \end{cases}$$

$$\operatorname{cutrk}_G(A) = \operatorname{rank}(M)$$

The rank-decomposition of a graph G is a pair (T, L) where T is a sub-cubic tree and L is a bijection from $V(G)$ to leaves of the tree T.

Width of the rank-decomposition (T, L) is a maximum width of e, over $e \in E(T)$. For each edge $e \in E(T)$, width of e is cutrank $\operatorname{cutrk}_G(A_e)$, where (A_e, B_e) are partitions of $V(G)$ given by $T \backslash e$. Any edge in a tree T split all leaves $(V(G))$ into two partitions A_e, B_e.

Rank-width of the graph G is a minimum width of rank-decompositions of G.

Paths, start, trees and cliques

Rank of any non empty matrix is non zero and therefore rank-width is 0 if and only if a graph has no edges. Class of graphs with rank-width 1 is well described be the following definition and theorem.

Definition *(Distance-hereditary graphs)* A graph is called distance-hereditary if and only if for every connected induced sub graph H of G, the distance between every pair of vertices of H is the same as in G.

Theorem 2.12 *[15] G is a distance-hereditary if and only if the rank-width of G is at most 1.*

Corollary 2.13 *It is not hard to see that trees and cliques are distance-hereditary graphs and hence have rank-width 1.*

Grids

However there is still a class of graphs where the rank-width remains unbounded.

Theorem 2.14 *[12] Rank-width of a grid $\sqrt{n} \times \sqrt{n}$ on n vertices is $\sqrt{n} - 1$.*

2.6 Boolean-width

Definition *(Boolean-width [4])* Decomposition of a graph G is a pair (T, L) where T is a sub-cubic tree and L is a bijection from $V(G)$ to leaves of the tree T.

The function cut-bool : $2^{V(G)} \to R$ is defined as

$$\text{cut-bool}(A) = log_2|\{S \subseteq \overline{A} : \exists X \subseteq A \wedge S = \overline{A} \cap \bigcup_{x \in X} N(x)\}|$$

Every edge $e \in T$ partition vertices $V(G)$ into $(A_e, \overline{A_e})$. Boolean-width of tree decomposition (T, L) is

$$\max_{e \in E(T)} \{\text{cut-bool}(A_e)\}$$

Boolean-width of graph G is a minimal boolean-width tree decomposition over all boolean-width tree decompositions of G, denoted boolw(G).

Paths, Stars, Trees, Cliques

Instead of examining class of trees and cliques, we rather recall that boolean-width is bounded by rank-width [4]. Since rank-width is one, boolean-width is at most 1.5. Boolean-width of a size 0 correspond to graph with no edge. Finally, we end up with boolean-width 1.

Grids

Similarly to rank-width, grids are also unbounded.

Theorem 2.15 *[4] Boolean-width of a grid $\sqrt{n} \times \sqrt{n}$ on n vertices lies between $\frac{1}{6}\sqrt{n}$ and $\sqrt{n} + 1$.*

2.7 Maximum Independent Set

Definition *(Maximum independent set)* An independent set of a graph $G = (V, E)$ is a set of vertices X such that no two vertices in X are adjacent $(\forall u, v \in X : (u, v) \notin E)$. The largest independent set of the graph G is maximum independent set, denoted $\alpha(G)$.

Paths, Stars and Trees

Lemma 2.16 *Let us look at graphs with leaves. Consider a leaf u and it's neighbour vertex v. Only one of u and v can be placed into an independent set. By choosing vertex v we exclude all neighbours vertices of v. Therefore it is always wise to include a leaf into the independent set.*

Previous lemma gives us straightforward algorithm for finding maximum independent set for trees. It is easy to see that size of maximum independent set for path on n vertices is $\lceil \frac{n}{2} \rceil$ and for star $n - 1$.

Algorithm

1. Remove all leaves and place them into the independent set.

2. Once again delete all newly formed leaves. All newly formed leaves were joined to some vertex already included into the independent set and therefore cannot be included in the independent set.

3. Repeat from first step until the remaining graph is not empty.

Since star is also tree then in an extreme case maximum independent set of a tree can be also $n-1$.

Cliques

For a clique, the situation is opposite and only one vertex can be placed in the independent set.

Grids

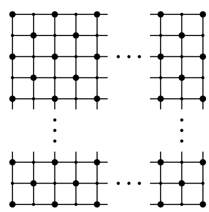


Figure 2.6: The maximum independent set of a grid with odd number of vertices.

We can follow a chessboard pattern (Figure 2.6) to obtain an independent set of the size $\lceil \frac{n}{2} \rceil$. If we delete all vertical edges then we get forest of n paths with maximum possible independent set. Hence a whole grid have maximum possible independent set.

2.8 Minimum Dominating Set

Definition *(Minimum dominating set)* For a graph $G = (V, E)$ a dominating set is a set of vertices X such that every vertex outside of X has at least one neighbour vertex inside of a set X ($\forall v \in V \setminus X \quad \exists u \in X : (u, v) \in E$). The dominating set with the smallest number of vertices is minimum dominating set. The size of minimum dominating set is called a domination number, denoted $\gamma(G)$.

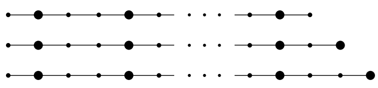


Figure 2.7: Minimum dominating set of a path with $3k$, $3k+1$ and $3k+2$ vertices. Bold vertices represent dominating set.

Paths, Trees

Let us examine class of paths. Maximum degree is 2. For $v \in X$ there is at most 2 vertices outside of X that are adjacent to v. In other words one vertex can dominate at most two other vertices. Therefore minimum dominating set of path is at least $\lceil \frac{n}{3} \rceil$ vertices and this is also necessary. On the figure 2.7 is shown pattern for choosing dominating vertices. This is also an extreme case for trees and we can assume the same bound.

Stars, Cliques

It is easy to see that middle vertex of a star dominates all vertices. And one vertex can dominate any clique as well.

Grids

We will use the same approach as for paths. Maximum degree in a grid is 4 and therefore domination number is at least $\lceil \frac{n}{5} \rceil$. Better result can be found in [5].

2.9 Vertex Cover Number

Definition *(Vertex cover number)* Vertex cover of a graph $G = (V, E)$ is a set of vertices $S \in V$ such that every edge has either one or both vertices in S. Vertex cover number is minimum vertex cover, a minimum number of vertices that cover all edges of a graph.

Paths, Trees

Maximum degree of a path is 2. Therefore any vertex covers at most two edges. Since there are $n-1$ edges in a graph, the vertex cover number is $\lceil \frac{n-1}{2} \rceil = \lfloor \frac{n}{2} \rfloor$. Path is an extreme case of a tree, hence vertex cover number of tree is at least $\lfloor \frac{n}{2} \rfloor$ as well.

Stars

All edges of a star are covered by the middle vertex. Hence vertex cover number of a star is 1.

Cliques

Lemma 2.17 *Vertex cover number of clique is $n-1$.*

Proof It is easy to see that such a vertex cover of size $n-1$ exists. Assume for a contradiction that the vertex cover number of a clique is less than $n-1$ and let vertices $u, v \notin S$. Since it is a clique, the edge $(u, v) \in E$ is not covered by either u or v. This is a contradiction.

Grids

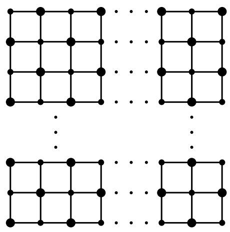


Figure 2.8: Chessboard pattern for vertex cover of a grid.

Since maximum degree of a grid is 4 then any vertex can cover at most 4 edges. There is $2 \times (\sqrt{n}-1) \times \sqrt{n} = 2n - 2\sqrt{n}$ edges in a graph and therefore vertex cover number of grid is at least $\lceil \frac{2n-2\sqrt{n}}{4} \rceil$. However, in figure is shown chessboard pattern for vertex cover of a grid of size $\lfloor \frac{n}{2} \rfloor$. Note that non of edges is covered be two vertices.

2.10 Maximum Clique

Definition *(Maximum clique)* For a graph $G = (V, E)$ a clique is a sub set of vertices X s.t. there is an edge between all pairs of vertices in X $(\forall u, v \in X : (u, v) \in E)$. Clique with largest number of vertices is maximum clique of a graph G.

Paths, Stars, Trees, Cliques, Grids

Paths, stars and trees contains maximum clique of size 2, the edge. Maximum clique size for graph class clique is obviously n. For a grid the situation is the same as for trees. Grids have maximum clique of size 2 as well.

2.11 Chromatic Number

Definition *(Chromatic number)* For a graph $G = (V, E)$, let $\varphi : V \to \{1..k\}$ be a one-to-one mapping function that assign to each vertex a number (colour). Minimum number of colours needed to label vertices in such a way that no two vertices are connected by edge, is called chromatic number of graph, denoted $\chi(G)$.

Paths, Stars, Trees

It is easy to see that chromatic number of paths, stars and trees is 2.

Grids

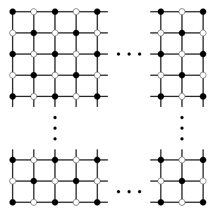


Figure 2.9: Vertex colouring of a grid using 2 colours.

Two colours are also sufficient to colour grids. To achieved it lets colour vertices in a way similar to chessboard pattern (Figure 2.9).

Cliques

For cliques the situation is opposite. Each vertex is adjacent to all other vertices and therefore all vertices need to have different colour. Chromatic number of a clique is n.

2.12 Maximum Matching

Definition *(Maximum matching)* For a graph G let us define matching as subset of disjoint edges (any two edges that do not share an endpoint). Largest matching of a graph G is maximum matching. Hence each edge consist of two different vertices, maximum matching is at most $\lfloor \frac{n}{2} \rfloor$. Matching that matched all vertices is called perfect matching.

Paths

For even paths, there always exists perfect matching and for general paths exist maximum matching of size $\lfloor \frac{n}{2} \rfloor$.

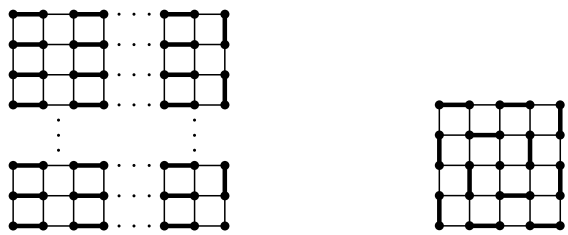
Stars, Trees

In a star graph, all edges share one vertex and therefore matching can never be more than 1. In worst case graph of a tree looks like a path and therefore graph class of trees have maximum matching of size $\lfloor \frac{n}{2} \rfloor$.

Cliques

Cliques contains as a sub graph path on all vertices. Matching along edges on this path leads to maximum matching of size $\lfloor \frac{n}{2} \rfloor$ as well.

Grids



(a) Path pattern (b) Spiral

Figure 2.10: Maximum matching of a grid with odd number of vertices.

Similar approach with a path can be used to show that maximum matching of grids is of size $\lfloor \frac{n}{2} \rfloor$. Pattern for choosing edges is in figure 2.10.

2.13 Maximum Induced Matching

Definition *(Maximum induced matching)* For a graph G let us define induced matching as an edge subset $M \subseteq E$ that satisfies two conditions:

- M is a matching of graph G.
- There is no other edge connecting any two vertices belonging to edges of the matching M.

Largest matching of the graph G is maximum matching.

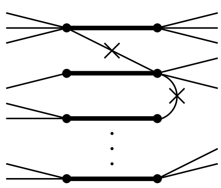


Figure 2.11: Induced matching forbids edges between already matched vertices.

Paths

In figure 2.12 is shown induced matching of a path. It is easy to see that the maximum induced matching of a path has $\lfloor \frac{n+1}{3} \rfloor$ edges.

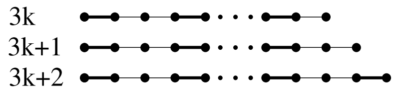


Figure 2.12: Maximum induced matching of path of lengths $3k$, $3k+1$ and $3k+2$. Bold edges belong to matching.

Stars, Clique

In contrast, star and clique can only contribute by one edge to an induced matching.

Grids

Roughly lower bound for grid on n vertices is $\lfloor \frac{n}{5} \rfloor$, because each matched edge forbids at least 5 vertices (corner edge).

Trees

Almost perfect matching can be achieved for a tree in figure 2.13.

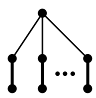


Figure 2.13: Tree with maximum induced matching (bold).

2.14 Cut-width

Definition *(Cut-width)* Denoted all vertices of a graph $G = (V, E)$ by numbers $1, ..., n$. Let π be a permutation of set $\{1, ..., n\}$ that place vertices in order on the line. Any cut of the line divide vertices into two partitions A and $V \backslash A$. We define width of this cut as number of edges that goes from A to $V \backslash A$.

Width of a permutation π is then defined as maximum width over all possible cuts. And finally, cut-width of G is defined as the minimum width over all possible permutations π, denoted cutw(G).

Paths

It is obvious that any path has a cut-width of size one.

Stars, Trees

Lemma 2.18 *For any graph G and any vertex $v \in V(G)$, $cutw(G) \geq \lceil \frac{deg(v)}{2} \rceil$.*

Proof Consider any permutation π. All edges of vertex v rather point to previous vertices nor next vertices in order of permutation π. Hence cut right before or after vertex v cross at least $\lceil \frac{deg(v)}{2} \rceil$ edges.

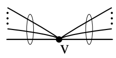


Figure 2.14: $\mathrm{cutw}(G) \geq \lceil \frac{deg(v)}{2} \rceil$

Corollary 2.19 *Cut-width of a star is* $\lceil \frac{n-1}{2} \rceil$*, and hence there exists a tree with a cut-width of size* $\lceil \frac{n-1}{2} \rceil$*.*

Cliques

For cliques, any permutation leads to the same vertex placement on the line. Any cut among this line divide vertices into two partitions of size k and $n-k$ and have size of $k(n-k)$. Maximum number of crossing edges is among two equally big partitions. Hence cut-width of a clique is $\lceil \frac{n}{2} \rceil \lfloor \frac{n}{2} \rfloor$.

Grids

Theorem 5.6 shows that for any graph path-width is smaller than cut-width. Since path-width of a grid on n vertices is at least $\sqrt{n}$ then cut-width is of grid is at least $\sqrt{n}$ as well.

2.15 Carving-width

Definition *(Curving-width)* Consider a decomposition (T, χ) of a graph G where T is a sub-cubic tree with $|V(G)|$ leaves and $\chi : A(T) \to V(G)$ is bijection mapping the leaves of a tree into vertices of G. Every edge of tree T, $e \in E(T)$, partition vertices of a graph into two partitions V_e and $V \backslash V_e$. Define a width of the edge of the tree as number of edges of a graph G that have exactly one endpoint in V_e and another endpoint in $V \backslash V_e$ e.g.

$$\mathrm{width}_e = |\{e = (u, v) \in V(G) : \quad u \in V_e \wedge v \in V \backslash V_e\}|$$

Furthermore let the width of the decomposition be the maximal width over all edges of tree T. And finally, carving-width of the graphs is minimal width over all decompositions, denoted $\mathrm{carw}(G)$.

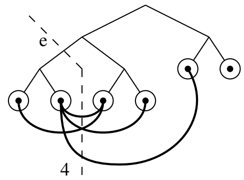


Figure 2.15: Example of width decomposition and width along edge e.

Stars, Trees

Before examining graph classes let us discuss the relation between maximum degree and carving-width.

Lemma 2.20 *Carving-width of a graph G is at least* $\Delta(G)$ *where* $\Delta(G)$ *is the maximum degree of a graph* G.

Proof Consider any decomposition (T, χ). Let v be a vertex with maximum degree and e an edge connecting directly to vertex v in T, i.e., e partition vertices into $\{v\}$ and $V \backslash \{v\}$.

Corollary 2.21 *Carving-width of a star is* $n-1$.

Corollary 2.22 *Since star is also a tree, carving-width of tree is also* $n-1$. *Tree have* $n-1$ *edges and therefore it is maximum possible carving-width.*

Cliques

Carving-with of a clique is at least $n-1$ since that is a maximum degree of a clique.

Paths

Path can be easily decomposed as shown in figure 2.16. Bold line represents the path itself and dashed line shows the maximal width along one of the edges of tree. The figure shows that exists carving decomposition of path with carving-width 2. Lemma 2.20 imply that this is tight estimation.

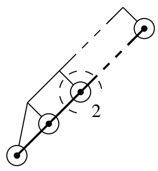


Figure 2.16: Carving decomposition of a path (bold), carving-width is 2.

Grids

Theorem 5.5 shows that for every graph G is $\mathrm{tw}(G) \leq 3 \cdot \mathrm{carw}(G) - 1$. Since a tree-width of a grid on n vertices is $\sqrt{n}$ due to [2] then carving-width is at least $3\sqrt{n} - 1$ and therefore unbounded.

2.16 Degeneracy

Definition *(Degeneracy)* Let G be a graph and consider the following algorithm:

- Find vertex v with smallest degree.
- Delete vertex v and connected edges.
- Repeat until graph is not empty.

Define degeneracy of a graph G (denoted $\delta D(G)$) as maximum degree over all deleted vertices. Sometimes we are talking k-degenerate graphs, i.e. graphs with degeneracy of size k.

Alternatively degeneracy can be define as maximum minimum degree of all sub graphs i.e.

$$\delta D(G) = \max_{G'}\{\min_{v \in V(G')} d_{G'}(v) \mid G' \text{ is a sub graph of } G\}$$

Pats, Stars, Trees

The previous algorithm when applied on trees would give a degeneracy of size one. The same applies for stars and paths as well.

Cliques

Since the minimum degree of a clique is already $n-1$ also degeneracy is the same.

Grids

And finally, it is easy to see that degeneracy of a grid is 2.

2.17 Tree-depth

Definition *(Tree-depth)* Tree-depth decomposition of a graph $G = (V, E)$ is a rooted tree T with the same vertex set V. In additional, for every edge $(u, v) \in E$, u is an ancestor of v or v is an ancestor of u in the tree T. We define depth of T as maximum number of vertices from the root to any leaf. And finally, tree-depth of the graph G is minimum depth among all tree-depth decompositions.

Paths, Trees

Consider the following construction of tree-depth decomposition for a path. For simplicity assume that the path has $2^k - 1$ vertices. Mark the middle vertex as root and denote remaining paths as p_1 and p_2. Once again for both remaining paths choose middle vertices r_1 and r_2 and connect them to the previous middle vertex. Construction step is shown in figure 2.17.

It is not hard to see that by applying recursion give us tree-depth decomposition of a path with height $\lceil log_2(n+1) \rceil$. Rather then proving previous construction for general case we refer to following theorem that gives us more general lower bound and upper bound.

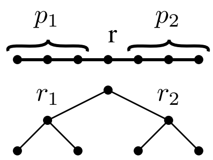


Figure 2.17: Construction of a tree-depth decomposition tree of a path.

Theorem 2.23 *[13] Let l be the length of the longest path in a graph G. Then the tree-depth of G is bounded as follows:*

$$\lceil \log_2 (l+2) \rceil \leq td(G) \leq \binom{l+3}{2} - 1$$

Note that path of length l consist of $l-1$ vertices.

Corollary 2.24 *Tree-depth of a grid is at least $\lceil \log_2 (n+1) \rceil$.*

Stars

In contrast, tree-depth of a star is 2 since tree-decomposition is the same as the star itself.

Cliques

It is easy to see that the only tree-depth decomposition of a clique is a path and therefore the tree-depth of the clique is n.

2.18 Summary

We end this chapter with a summary table. There is no parameter that is bounded on all graph classes. As expected most parameters are bounded on graphs with simpler structure such as path and star. On the other hand only three parameters are bounded on grid.

There are tree possible values for a parameter p and a class $\mathcal{C}$ in the Table 2.1:

constant k Any graph from class $\mathcal{C}$ has the parameter of size c i.e. $\forall G \in \mathcal{C}$: $p(G) = k$.

	Path	Star	Tree	Clique	Grid
Path-width	1	1	$\geq \frac{log_2 n}{2}$	$n-1$	$\geq \sqrt{n}$
Tree-width	1	1	1	$n-1$	$\sqrt{n}$
Branch-width	2	1	2	$\lceil \frac{2}{3}n \rceil$	$\sqrt{n}$
Clique-width	2	2	3	2	$\sqrt{n}+1$
Rank-width	1	1	1	1	$\sqrt{n}-1$
Boolean-width	1	1	1	1	$\geq \frac{1}{6}\sqrt{n}$
Max. indep. set	$\lceil \frac{n}{2} \rceil$	$n-1$	$n-1$	1	$\lceil \frac{n}{2} \rceil$
Min. dom. set	$\lceil \frac{n}{3} \rceil$	1	$\lceil \frac{n}{3} \rceil$	1	$\geq \lceil \frac{n}{5} \rceil$
Vertex cover #	$\lfloor \frac{n}{2} \rfloor$	1	$\lfloor \frac{n}{2} \rfloor$	$n-1$	$\lfloor \frac{n}{2} \rfloor$
Max. clique	2	2	2	n	2
Chromatic #	2	2	2	n	2
Max. matching	$\lfloor \frac{n}{2} \rfloor$	1	$\lfloor \frac{n}{2} \rfloor$	$\lfloor \frac{n}{2} \rfloor$	$\lfloor \frac{n}{2} \rfloor$
Max. ind. mat.	$\lfloor \frac{n+1}{3} \rfloor$	1	$\frac{n-1}{2}$	1	$\geq \lfloor \frac{n}{5} \rfloor$
Cut-width	1	$\lceil \frac{n-1}{2} \rceil$	$\lceil \frac{n-1}{2} \rceil$	$\lceil \frac{n}{2} \rceil \lfloor \frac{n}{2} \rfloor$	$\geq \sqrt{n}$
Carving-width	2	$n-1$	$n-1$	$\geq n-1$	$n+1$
Degeneracy	1	1	1	$n-1$	2
Tree-depth	$\lceil \log_2 (n+1) \rceil$	2	$\lceil \log_2 (n+1) \rceil$	n	$\geq \lceil \log_2 (n+1) \rceil$

Table 2.1: Parameter size for five different graph classes.

$\geq$ **f(n)** There is no such constant k i.e. $\forall k \; \exists G \in \mathcal{C} : \; p(G) > k$. We showed that there exists a graph G s.t. $p(G) \geq f(n)$ where n is the number of vertices of a graph G.

f(n) The same as previous case. In additional, we showed that $\exists G \in \mathcal{C}$: $p(G) = f(n)$ so $f(n)$ is a tight upper bound.

Note that bounded and unbounded values may differ for trivial graphs e.g. one vertex graph.

Chapter 3

Parameter groups

In this short chapter, we place similar parameters into groups. We consider two parameters p_1, p_2 similar if for every of the five graph classes the parameters are either both bounded or unbounded. Note that it does not mean that $p_1 \approx p_2$. Parameter groups are shown in Table 3.1.

Even if it is not important, we ordered groups in the table by increasing complexity of structure of graph classes. We consider paths and stars as graphs with very simple structure. On the other side, cliques and specially grids we consider as graphs with more complex structure. However, first two groups are unbounded on both simple or more complex classes.

	Group	Path	Star	Tree	Clique	Grid
1.	Maximum independent set	U	U	U	B	U
2.	Minimum dominating Set, Maximum induced matching	U	B	U	B	U
3.	Cut-width, Carving-width	B	U	U	U	U
4.	Vertex cover number, Tree-depth, Maximum matching	U	B	U	U	U
5.	Path-width	B	B	U	U	U
6.	Tree-width, Branch-width	B	B	B	U	U
7.	Clique-width, Rank-width, Boolean-width	B	B	B	B	U
8.	Maximum clique, Chromatic number, Degeneracy	B	B	B	U	B

Table 3.1: Parameters are grouped when ever they behave the same on all five graph classes. “B” means that all parameters are **B**ounded by a constant for a specific class. On the other hand, “U” stands for **U**nbounded i.e. the parameter size depends on the graph size n.

Whenever parameter p_1 is bounded on a class of graphs C and parameter p_2 is unbounded then $p_2 \not\preceq p_1$.

From Table 3.1, we extract information about inbound relations and obtain following list. Parameters in brackets belongs to the same group.

IS $\not\preceq$ (cutw, carw), (VCN, td, MM), pw, (tw, bw), (ω, χ, δD)

(γ, MiM) $\not\preceq$ IS, (cutw, carw), (VCN, td, MM), pw, (tw, bw), (ω, χ, δD)

(cutw, carw) $\not\preceq$ IS, (γ, MiM), (VCN, td, MM)

(VCN, td, MM) $\not\preceq$ IS, (cutw, carw)

pw $\not\preceq$ IS, (γ, MiM), (cutw, carw), (VCN, td, MM)

(tw,bw) $\not\preceq$ IS, (γ, MiM), (cut, carw), (VCN, td, MM), pw

(cw, rw, bw) $\not\preceq$ IS, (γ, MiM), (cut, carw), (VCN, td, MM), pw, (tw, bw), (ω, χ, δD)

(ω, χ, δD) $\not\preceq$ IS, (γ, MiM), (cut, carw), (VCN, td, MM), pw, (tw, bw), (cw, rw, bw)

To make it more clear, we draw Figure 3.1.

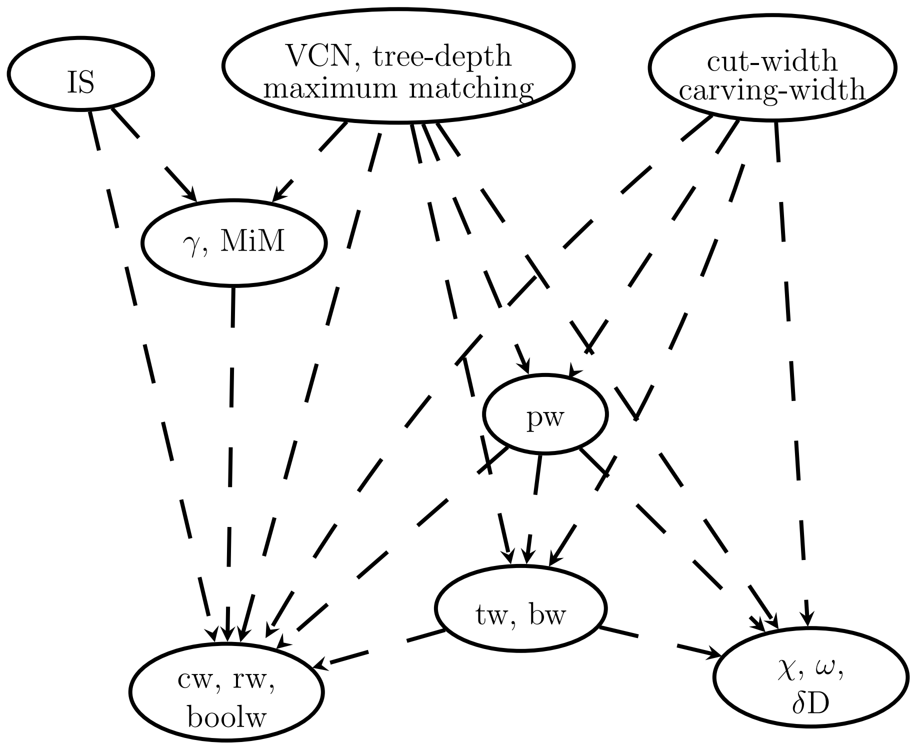


Figure 3.1: Partial order of parameter groups. Dashed line represent possible relation which have to be explore in more details. Consider a dashed arrow from group G_1 to group G_2. For every parameter p_1 from G_1 and for every parameter p_2 from G_2 we have to further examine whether $p_2 \prec p_1$ or $p_1|p_2$. No line from a group G_1 to G_2 means that for any parameter p_1 from G_1 is not bounded by any parameter p_2 from group G_2 i.e. $p_1|p_2$.

Chapter 4

Comparing similar parameters

In the previous chapter, we identified groups with similar parameters. Since parameters in a group behave similar on the five main graph classes, we are expecting that parameters might be mutually bounded.

In each of the following sections we examine one group of parameters except those groups with only one parameter. For every pair of parameters in the group, we show one of the four possible relations ($\prec, \succ, \approx, |$). Moreover, each section ends with a partial comparison diagram.

4.1 Relations between minimum dominating set and maximum induced matching

Theorem 4.1 (MiM $\not\preceq \gamma$) *Maximum induced matching is not bounded by minimum dominating set.*

Proof Consider the graph shown in figure 4.1. It is obvious that the middle vertex dominates all vertices. On the other hand, the bold edges represent a maximum induced matching of size $\frac{n-1}{2}$.

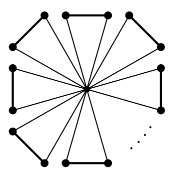


Figure 4.1: Minimum dominating set is bounded, but maximum induced matching can be arbitrary large.

Theorem 4.2 ($\gamma \not\preceq$ MiM) *Minimum dominating set is not bounded by maximum induced matching.*

Proof In this proof, consider the graph in figure 4.2. It consists of a clique on $\frac{n}{2}$ vertices and additional $\frac{n}{2}$ leaves. Each leaf corresponds to a vertex of the complete graph.

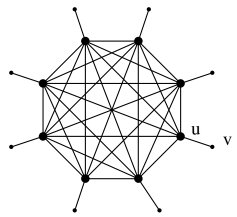


Figure 4.2: Maximum induced matching is bounded, but minimum dominating set can be arbitrary large.

Only one vertex from the clique can be included in an induced a matching and therefore maximum induced matching has size one.

In contrast, from pair u, v at least one vertex have to be chosen into a dominating set. Therefore at least $\frac{n}{2}$ vertices have to be chosen. It is easy to see that $\frac{n}{2}$ is also sufficient. In figure 4.2 bold vertices represent minimum dominating set, but all leaves would dominate the graph as well.

Figure 4.3: Relations between minimum dominating set, maximum induced matching. They are mutually unbounded, $\gamma \mid$ MiM.

4.2 Relations between cut-width and carving-width

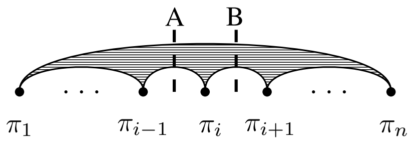
Cut-width and carving-width are very similar. Both count number of crossing edges. Main difference is how they partition vertices. Following theorem show that cut-width is bounded by carving-width. However, opposite relation is unbounded.

Theorem 4.3 (carw $\preceq$ cutw) *Carving-width is bounded by cut-width.*

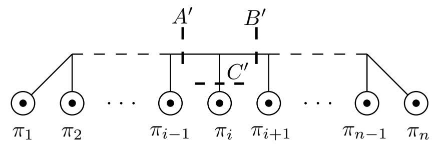
Proof In this proof, we show that for any graph with bounded cut-width there exists carving decomposition with carving-width no more than twice as big. Let G be a graph with cut-width of size k and π corresponds to permutation of vertices. We fix order of vertices and build a sub cubic tree above it as it is shown in figure 4.4b. For any vertex π_i there are three types of carving cuts i.e. A', B' and C'. Let us examine them.

Cut A' splits vertices into two partitions $\{\pi_1, ..., \pi_{i-1}\}$ and $\{\pi_i, ..., \pi_n\}$. This partitioning corresponds to cut A in cut decomposition in figure 4.4a. Therefore cuts A and A' have the same size. Similar approach applies for carving cut B' and corresponding cut B.

Cut C' splits vertices into two partitions $\{\pi_i\}$ and $V\backslash\{\pi_i\}$. However, size of C' is no more than $A' + B' = A + B = 2k$. Since cut-width of G is k then carving-width is no more than $2k$.



(a) Cut decomposition



(b) Carving decomposition

Figure 4.4: Cut A' correspond to A, B' to B and size of cut C' correspond to sum of A and B.

Theorem 4.4 (cutw $\not\preceq$ carw) *Cut-width is* <u>*not*</u> *bounded by carving-width.*

Proof We do not prove this directly. Bodlaender proved in [3] that for all graphs pw$(G) \leq$ cutw(G). We prove in Theorem 5.4 that path-width is not bounded by carving-width and therefore also cut-width is not bounded by carving-width.

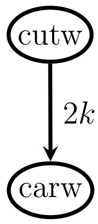


Figure 4.5: Relations between cut-width and carving-width.

4.3 Relations between vertex cover number, tree-depth and maximum matching

Theorem 4.5 (MM $\preceq$ VCN) *Maximum matching is bounded by vertex cover number.*

Proof Let S be a vertex cover of size k. Since all edges contain at least one vertex from S ($\forall(u, v) \in E : u \in S \vee v \in S$), there is at most k disjoint edges and therefore maximum matching is never more than k. Furthermore this is a tight bound, we can easily check vertex cover number and maximum matching of complete bipartite graph $K_{n,n}$.

Theorem 4.6 (VCN $\preceq$ MM) *Vertex cover number is bounded by maximum matching.*

Proof Let $G = (V, E)$ be a graph and $M \subseteq E$ is maximum matching of size k. For convenience, let us define $V(M)$ as set of all vertices that are matched, i.e.

$$V(M) = \{u | \exists v \in V \quad (u, v) \in M\}$$

It is not hard to see that all edges of G either intersect in set M or intersect with some of edge in M, i.e. $\forall (u, v) \in E : u \in V(M) \vee v \in V(M)$. Otherwise, we can add such an edge to M to get matching of bigger size and that is contradiction with chosen maximum matching.

We use this fact to build a vertex cover by choosing all vertices from $V(M)$ to vertex cover S. Therefore the vertex cover number of G is at most $2k$.

Theorem 4.7 (td $\preceq$ VCN) *Tree-depth is bounded by vertex cover number.*

Proof First, we show construction of tree with depth $k + 1$ and then we examine all edges that satisfied definition of tree-depth.

Let S be a vertex cover of size k of graph G. Consider the following construction of tree T from vertices $V(G)$. Construct path from vertices S in arbitrary order and denote endpoints as r and t. Append all remaining vertices $V \backslash S$ as direct ancestors of vertex t. If we consider r as root of tree then we get the tree T as shown in figure 4.6.

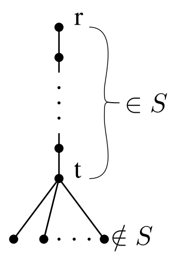


Figure 4.6: On original vertices of the a graph G, we construct the tree T that satisfy definition of tree-depth. Set S represent vertex cover.

We examine edges $V(G)$ and show that the tree T satisfies the definition of tree-depth. For any edge $e = (u, v)$ of G, consider 3 cases:

1. Both endpoints are in S, i.e. $u, v \in S$. Since all vertices from S lie on path from r to t then either u is an ancestor of v or v is an ancestor of u in tree T.

2. Exactly one endpoint is in S, i.e. $u \in S$ and $v \notin V \backslash S$. Vertex v is a leaf in T and therefore v is an ancestor of u.

3. None of the endpoints is in S, i.e. $u, v \notin S$. Since vertices in S cover all edges of graph G this case can never happen. In other words, there cannot be edge edge in the original graph G between two leaves of the tree T.

We have shown that for any graph with bounded vertex cover number k, we can construct tree-depth tree T of depth no more that $k + 1$ and therefore tree-depth of G is at most $k + 1$ as well.

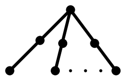


Figure 4.7: Construction of a graph with arbitrary large vertex cover number or maximum matching and fixed tree-depth.

Theorem 4.8 (VCN $\not\preceq$ td) *Vertex cover number is* <u>*not*</u> *bounded by tree-depth.*

Proof Consider the graph in figure 4.7. By adding double leaves we get a graph with arbitrary large vertex cover number, but tree-depth remains fixed at size 3.

Corollary 4.9 (MM $\not\preceq$ td) *Maximum matching is* <u>*not*</u> *bounded by tree-depth.*

Theorem 4.10 (td $\preceq$ MM) *Tree-depth is bounded by maximum matching.*

Proof Let G be a graph with maximum matching of size k. Using theorem 4.6 we get that the graph G has a vertex cover of size at most $2k$. And for a graph with bounded vertex cover number theorem 4.7 gives upper bound for tree-depth. In total, tree-depth of G is at most $2k + 1$.

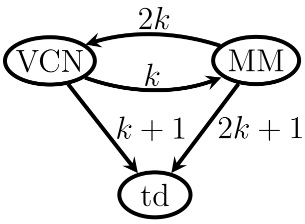


Figure 4.8: Relations between vertex cover number, maximum matching and tree-depth.

4.4 Relations between tree-width and branch-width

We show that for any graph G with branch-width equal to k there exists a tree decomposition with tree-width at most $\lfloor \frac{3}{2}k \rfloor - 1$. Any branch decomposition T_b already satisfies these two tree decomposition conditions:

- $\forall (u, v) \in E(G) \quad \exists t : u, v \in X_t$
- $\cup_{t \in T} X_t = V(G)$

For every two leaves in the branch decomposition T_b that contains vertex x exist a unique path in the tree T. Adding vertex x to every node on the path (that not already include it) we get a tree decomposition T_t of a graph G that also satisfies the third condition. It remains to determine maximal size of bag.

Theorem 4.11 *[21] Any bag u of tree decomposition T_t have size no more than $\lfloor \frac{3}{2}k \rfloor$.*

Proof If bag u have a degree 1 then u is a leaf and size of the bag is 2. Otherwise u have a degree 3 and let denote $W_u^1, W_u^2, W_u^3 \subseteq V(G)$ the guts of separation for three incident edges of u. It is easily seen that a vertex of G occurring in one of W_u^1, W_u^2, W_u^3 has to appear also in another one of those, therefore the cardinality of $B_u = W_u^1 \cup W_u^2 \cup W_u^3$ is at most $\lfloor \frac{3}{2}k \rfloor$.

Corollary 4.12 (tw $\preceq$ bw) *[21] Tree-width is bounded by branch-width.*

Theorem 4.13 (bw $\preceq$ tw) *[21] Branch-width is bounded by tree-width, i.e. $\forall G\ bw(G) \leq tw(G) + 1$.*

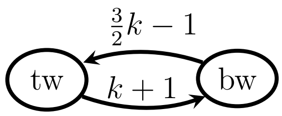


Figure 4.9: Relations between tree-width and branch-width. Arrow from branch-width to tree-width means that when ever branch-width is bounded by constant k then also tree-width is bounded by constant. Function above this arrow means that the tree-width is no more than $\frac{3}{2}$ of the branch-width. Similarly it works for opposite direction or any other comparison diagram in this thesis.

4.5 Relations between clique-width, rank-width and boolean-width

The following theorems give a full overview of relations. All three parameters are mutually bounded. However, there is a difference between them. Boolean-width is bounded by some polynomial function and comes as "smallest" among them. On the other hand, clique-width is bounded exponentially.

Theorem 4.14 (boolw $\preceq$ rw) *[4] Boolean-width is bounded by rank-width, i.e. $\forall G\ boolw(G) \leq \frac{1}{4}rw(G)^2 + \frac{5}{4}rw(G) + \log rw(G)$.*

Theorem 4.15 (rw $\preceq$ boolw) *[4] Rank-width is bounded by boolean-width, i.e. $\forall G\ rw(G) \leq 2^{boolw(G)}$.*

Theorem 4.16 (boolw $\preceq$ cw) *[4] Boolean-width is bounded by clique-width, i.e. $\forall G\ boolw(G) \leq cw(G)$.*

Theorem 4.17 (cw $\preceq$ boolw) *[4] Clique-width is bounded by boolean-width, i.e. $\forall G\ cw(G) \leq 2^{boolw(G)+1}$.*

Theorem 4.18 (rw $\preceq$ cw) *[18] Rank-width is bounded by clique-width, i.e. $\forall G\ rw(G) \leq cw(G)$.*

Theorem 4.19 (cw $\preceq$ rw) *[18] Clique-width is bounded by rank-width, i.e. $\forall G\ cw(G) \leq 2^{rw(G)+1} - 1$.*

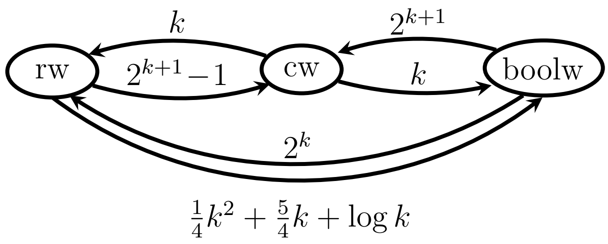


Figure 4.10: Relations between clique-width, rank-width and boolean-width.

4.6 Relations between maximum clique, chromatic number and degeneracy

Theorem 4.20 ($\chi \preceq \delta D$) *Chromatic number is bounded by degeneracy.*

Proof Let $G = (V, E)$ be a graph with degeneracy k and $v_n, v_{n-1}, ..., v_1 \in V$ a sequence of vertices as they were processed by the degeneracy algorithm in definition 2.16. We show colouring of vertices with $k + 1$ colours by induction on sequence of vertices $v_1, v_2, ..., v_n$.

- One vertex graph can be easily coloured by any colour.
- Let G' be coloured sub graph of G with $i-1$ vertices and v_i a vertex that is going to be added and coloured. It is important to see that v_i have at most k neighbours in the graph G'. So among $k + 1$ colours there is always one colour that can be used to colour vertex v_i.

Therefore chromatic number of G is at most $k + 1$.

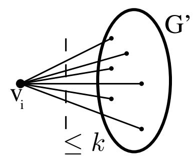


Figure 4.11: Induction step of colouring of a graph with a degeneracy k.

Theorem 4.21 ($\omega \preceq \chi$) *Maximum clique is bounded by chromatic number.*

Proof It is obvious that to colour K_k we need k colours. Therefore any graph with fixed chromatic number k cannot contain clique of size more than k.

Theorem 4.22 ($\delta D \not\preceq \omega$) *Degeneracy is not bounded by chromatic number.*

Proof We are looking for a graph class in which for the fixed chromatic number there exists arbitrary large degeneracy. Let us examine class of bipartite graphs $K_{n,n}$. Chromatic number of $K_{n,n}$ is 2, one colour for each partition. In contrast, degeneracy of $K_{n,n}$ is n, since minimal degree across all vertices is n. Note that number of vertices in the graph is $2n$.

Theorem 4.23 *(Mycielski's theorem [6]) For every $k \geq 1$ there exists a triangle free graph with chromatic number k.*

Corollary 4.24 ($\chi \not\preceq \omega$) *Chromatic number is not bounded by maximum clique.*

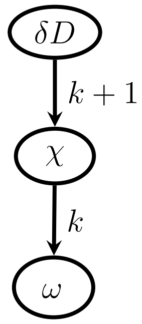


Figure 4.12: Relations between degeneracy (δD), chromatic number (χ) and maximum clique (ω).

4.7 Summary

In this chapter, we investigated relations inside of groups. Figure 4.13 updates Figure 3.1 from previous chapter for partial comparison diagrams.

As we can see, tree-width and branch-width are mutually bounded. This saves us some work. Whenever parameter p is bounded by tree-width then p is bounded also by branch-width. Similarly it works for clique-width, rank-width and boolean-width.

On the other hand, minimum dominating set and maximum independent set split into two mutually unbounded parameters. Therefore both parameters have to be investigate separately.

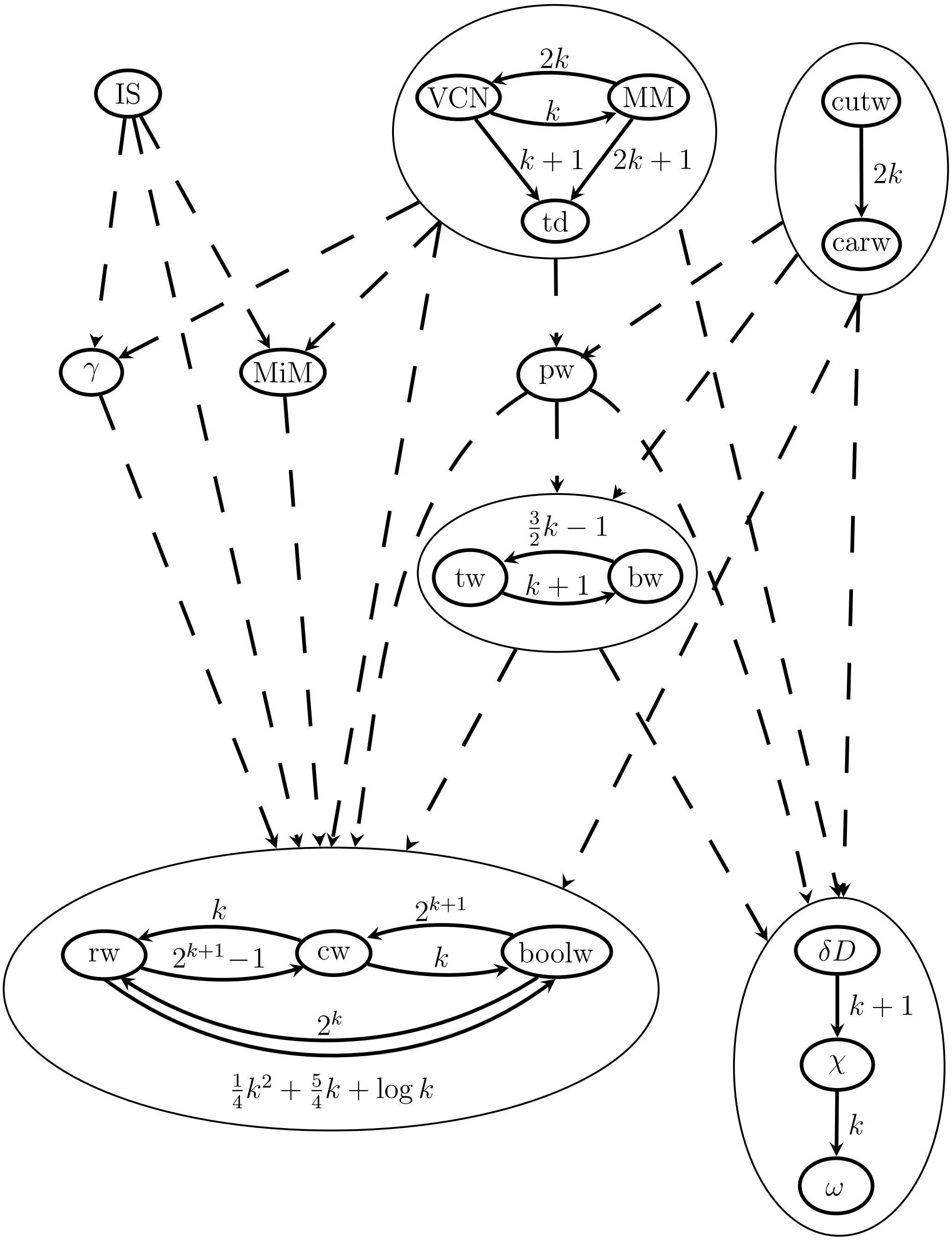


Figure 4.13: This figure extend figure 3.1 for partial comparison diagrams from chapter 4. Note that group with dominating set (γ) and minimum induced matching (MiM) pull apart into two separate parameters. Meaning of dashed arrow remain the same as in figure 3.1. Suppose a dashed arrow from group 1 to group 2. Any parameter p_1 from group 1 can be a bound for any parameter p_2 from group 2 ($p_1 \preceq p_2$) or unbounded ($p_1 \not\preceq p_2$). We will further investigate dashed relations in the following chapter.

Chapter 5

Relations between groups

The previous chapter, we finished with Figure 4.13 with many dashed arrows that have to be examine. In this chapter we resolve all of them and bring a complete comparison diagram.

We build complete comparison diagram bottom-up. We start with rank-width, clique-width, boolean-width, degeneracy, chromatic number and maximum clique. In the following sections, we gradually remaining parameters and create the complete comparison diagram.

Adding tree-width and branch-width

Next, we add tree-width and branch-width. Theorem 5.1 shows that clique-width is bounded by tree-width. Therefore rank-width, clique-width and boolean-width are bounded by tree-width and branch-width. Similarly, Theorem 5.2 shows that degeneracy, chromatic number and maximum clique are bounded by tree-width and branch-width.

Theorem 5.1 (cw $\preceq$ tw) *[7] Clique-width is bounded by tree-width, i.e.* $\forall G\ cw(G) \leq 3 \times 2^{tw(G)-1}$.

Theorem 5.2 ($\boldsymbol{\delta D} \preceq$ tw) *Degeneracy is bounded by tree-width.*

Proof Let G be a graph with tree-width k. There exists a chordal graph G' such that G is sub graph of G'. Moreover, G' has a perfect elimination ordering. Since tree-width is k then maximal clique of G is at most $k+1$ (Theorem 2.2). Therefore any vertex in elimination ordering has at most k neighbours. And finally, degeneracy of G is at most k.

Adding path-width

Theorem 5.3 shows that tree-width is bounded by path-width and therefore rank-width, clique-width, boolean-width, degeneracy, chromatic number and maximum clique are also bounded by path-width.

Theorem 5.3 (tw $\preceq$ pw) *Tree-width is bounded by path-width.*

Proof Let (X, P) be a path decomposition of graph G. However, any path is also a tree, and therefore (X, P) is also tree decomposition of G.

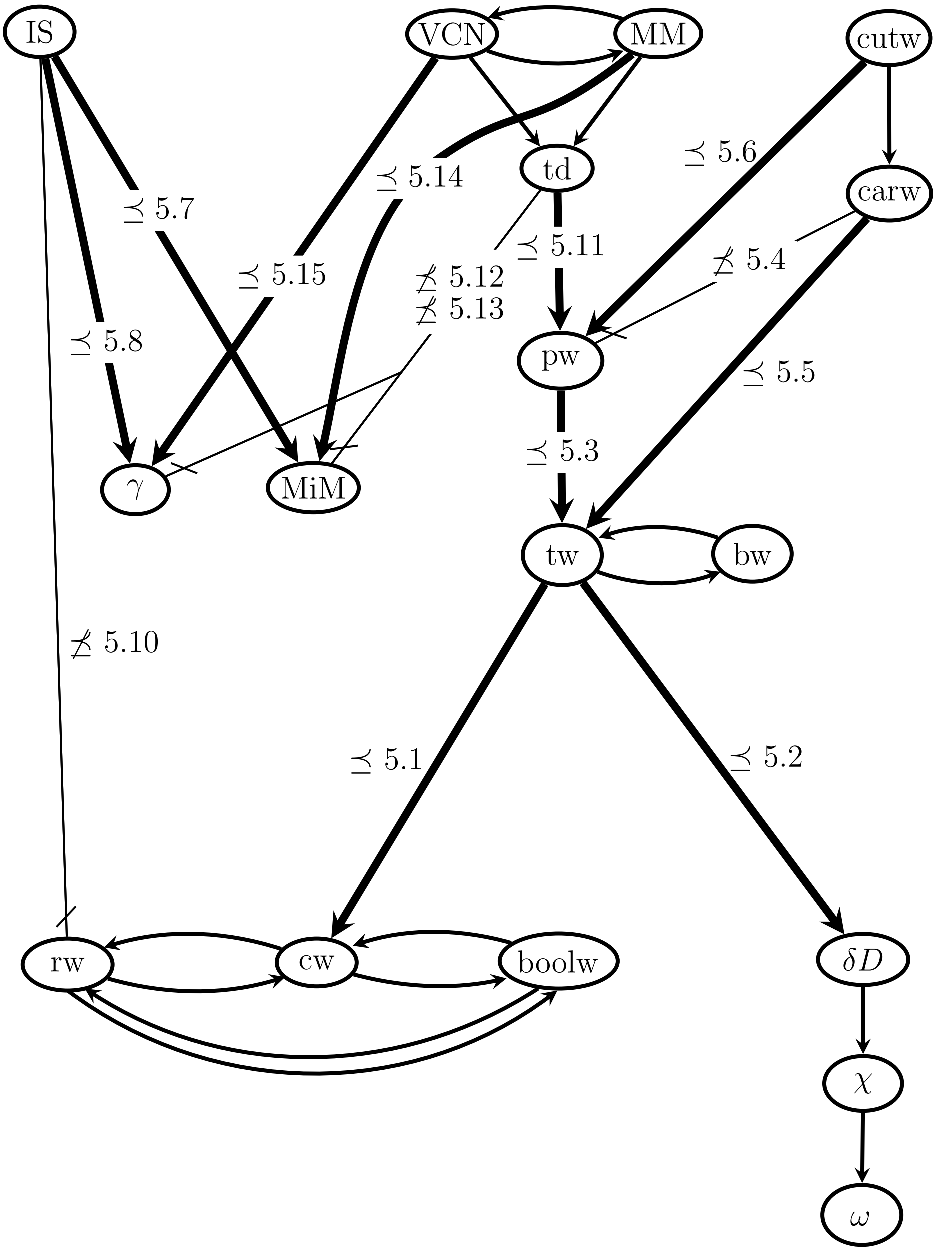


Figure 5.1: Comparison diagram that resolves all dashed arrows from Figure 4.13. Number next to the line refers to the theorem where it is the relation discuss. Bold arrow from a parameter p_1 to a parameter p_2 marked with $\preceq$ means that $p_2 \preceq p_1$. Thin crossed arrow from p_1 to p_2 ($p_1 \not\longrightarrow p_2$) marked with $\not\preceq$ means that $p_2 \not\preceq p_1$.

Adding carving-width and cut-width

In next step, we add cut-width and carving-width. While path-width is not bounded (Theorem 5.4), tree-width is already bounded by carving-width (Theorem 5.5). To complete the picture, Theorem 5.6 shows that path-width is bounded by cut-width.

Theorem 5.4 (pw $\not\preceq$ carw) *Path-width is not bounded by carving-width.*

Proof Let us examine class of a complete binary trees. Path-width is unbounded on complete binary trees [1] and we show that carving-width of a complete binary trees is bounded.

Let T be a complete binary tree with a height h and a root r. We alter the tree T in tree steps and obtain carving decomposition (T, χ):

1. We replace every leaf v with a node n_v and mapped n_v to v, $\chi(n_v) \to v$.
2. Let v be an internal vertex except root and e_v be a first edge on the path from v to r in T. We replace v with a node n_v^1. Then divide edge e_v width a new node n_v^2. And finally, add new node n_v^3 with with mapped vertex v and connect the node v_n^3 to the node v_n^2.
3. In the last step, we replace the root r with node n_r'. And for the root r, we create a new node n_r and connect it to n_r'.

It is not hard to see that maximum width of the decomposition is along the edge (v_n^3, v_n^2) and has the size 3. Figure 5.2 shows an example of a carving decomposition of a binary tree on 7 vertices.

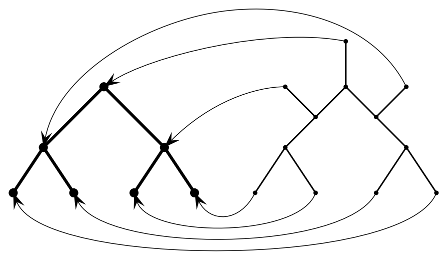


Figure 5.2: Carving-width of a complete binary tree on 7 vertices. The original binary tree is bold, carving decomposition is drawn with thin lines and arrows show mapping nodes to vertices.

Theorem 5.5 (tw $\preceq$ carw) *Tree-width is bounded by carving-width.*

Proof Let (T, χ) be a carving-decomposition of a graph G with carving-width k. If $|V(G)| = 2$ then we place both vertices into one bag and obtain tree-width of size 1.

Assume that G has at least 3 vertices. We create a tree-decomposition (T, X) following way. We use the tree T from carving-decomposition and for each leaf with mapped vertex v, we create a bag X_v that contains only v. Any internal node n splits leaves of T and also vertices of original graph into three disjoint sets B_1^n, B_2^n, B_3^n. Each node n, we replace with bag X_n s.t.

$$\begin{aligned} X_n = &\{(u,v) \in E : \ u \in B_1^n \wedge v \notin B_1^n\} \cup \\ &\{(u,v) \in E : \ u \in B_2^n \wedge v \notin B_2^n\} \cup \\ &\{(u,v) \in E : \ u \in B_3^n \wedge v \notin B_3^n\} \end{aligned}$$

It is not hard to see that (T, X) is a tree decomposition. Since for any edge in carving decomposition there is at most k crossing edges then any bag X_n has cardinality at most $3k$. Finally, tree-width of G is $3k - 1$.

Theorem 5.6 (pw $\preceq$ cutw) *[3] Path-width is bounded by cut-width, i.e. $\forall G$ $pw(G) \leq cutw(G)$.*

Adding maximum independent set, minimum dominating set and maximum induced matching.

We prove that minimum dominating set and maximum induced matching are bounded by independent set (Theorems 5.7 and 5.8). Remains a question whether rank-width, clique-width and boolean-width are bounded or not. In theorem 5.10, we show that rank-width is not bounded by independent set. As direct consequence, minimum dominating set and maximum independent set are not bounded by rank-width, clique-width and boolean-width, i.e.

$$\text{IS}, \gamma, \text{MiM} \mid \text{rw}, \text{cw}, \text{bw}$$

Theorem 5.7 (MiM $\preceq$ IS) *Maximum induced matching is bounded by maximum independent set.*

Proof Let M be a maximum induced matching of size k. For every edge (u, v) from M, place u into S. Directly from definition of maximum independent matching, we can see that S is an independent set of size k. Therefore size of maximum induced matching is always smaller than size of maximum independent set.

Theorem 5.8 ($\gamma \preceq$ IS) *Minimum dominating set is bounded by maximum independent set.*

Proof Let S be a maximum independent set of graph $G = (V, E)$. We can easily check that all vertices from $V \backslash S$ have a neighbour vertex in S i.e. $\forall u \in V \backslash S \ \exists v \in S \ : (u, v) \in E$. Suppose there is vertex u that have no neighbour in S. Them $\{u\} \cup S$ is independent set and that is contradiction with chosen maximum independent set. Thus, S is a dominating set.

Lemma 5.9 *[16] Let G be a graph. Then $|rw(G) - rw(\overline{G})| \leq 1$, where $\overline{G}$ is a complement graph of G i.e. $\overline{G} = (V(G), E')$, $E' = \{(u,v) : u, v \in V(G) \wedge (u,v) \notin E(G)\}$.*

Theorem 5.10 (rw $\not\preceq$ IS) *Rank-width is not bounded by maximum independent set.*

Proof Consider a $n \times n$ grid H. Let $\overline{H}$ be a complement graph of H. Let v be vertex from maximum independent set of $\overline{H}$. Then there are at most 4 vertices that are not neighbours of v. However, it is not hard to see that only one can be picked to independent set. Therefore size of maximum independent set of $\overline{H}$ is 2.

On the other hand, rank-width of $\overline{H}$ is unbounded. Rank-width of the grid H is $n-1$ [12]. And by applying previous lemma, rank-width of $\overline{H}$ is at least $n-2$.

Adding tree-depth, vertex cover number and maximum induced matching

Theorem 5.11 give us all following relations:

$$\mathrm{pw}, \mathrm{tw}, \mathrm{bw}, \mathrm{rw}, \mathrm{cw}, \mathrm{bw}, \delta D, \chi, \omega \prec \mathrm{td}, \mathrm{VCN}, \mathrm{MM}$$

Finally, it remains to examine relations to minimum dominating set and maximum induced matching. Both are not bounded by tree-depth (Theorems 5.12 and 5.13), but they are bounded by vertex cover number and maximum matching (Theorems 5.14 and 5.15).

Theorem 5.11 (pw $\preceq$ td) *Path-width is bounded by tree-depth.*

Proof Let T be a tree-depth decomposition tree with depth k. Consider depth-first search algorithm for visiting vertices of T. Each time algorithm visit leaf form a bag of vertices from root to actual leaf. We keep order of bags and place them on the path P (Figure 5.3). We have to ensure that mentioned construction gives path decomposition.

- Every path from root to leaf is in some bag hence every edge is inside some bag as well.

- Consider any vertex v from T and denote T_v sub tree rooted at vertex v. Vertex v occurs only in bags $\mathbb{B}_v$ that correspond to leaves in sub tree T_v. Depth-first search make sure that all leaves in sub tree T_v are proceed before any other leaf from tree T. Therefore bags $\mathbb{B}_v$ induce sub path of P.

- Since all edges belong to some bag, also each vertex belongs to at least one bag.

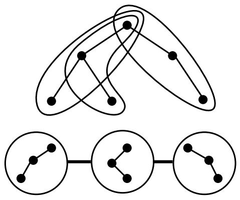


Figure 5.3: Construction of path-width decomposition from tree-depth decomposition.

Since T has depth k then any bag has at most k vertices. Therefore path-width of G is at most $k-1$.

Theorem 5.12 (MiM $\not\preceq$ td) *Maximum induced matching is <u>not</u> bounded by tree-depth.*

Proof Consider the graph in Figure 5.4. It has unbounded maximum induced matching while tree-depth is 3.

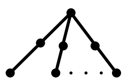


Figure 5.4: Graph with arbitrary large dominating set or maximum induced matching and fixed tree-depth.

Theorem 5.13 ($\gamma \not\preceq$ td) *Minimum dominating set is <u>not</u> bounded by tree-depth.*

Proof Once again, consider the graph in Figure 5.4. It has unbounded minimum dominating set while tree-depth is 3.

Theorem 5.14 (MiM $\preceq$ MM) *Maximum induced matching is bounded by maximum matching.*

Proof Inequality $\mathrm{MiM}(G) \leq \mathrm{MM}(G)$ comes directly from the definitions of maximum matching and it's restricted version maximum induced matching.

Theorem 5.15 ($\gamma \preceq$ VCN) *Minimum dominating set is bounded by vertex cover number.*

Proof It is easy to see that any vertex cover dominates all vertices. And therefore any vertex cover is also a dominating set.

5.1 Improving exponential relations

By his point, all relations between all 17 parameters are known. However, when designing an algorithm based on relations from comparison diagram, starts to be important, whether is bounding function linear or exponential.

In this section we recall two theorems that improve exponential bounding function between tree-width and boolean-width; tree-width and rank-width.

Theorem 5.16 (boolw $\preceq$ tw) *[19] Boolean-width is bounded by tree-width, i.e.* $\forall G$ $boolw(G) \leq tw(G) + 1$.

Theorem 5.17 (rw $\preceq$ tw) *[17] Rank-width is bounded by tree-width, i.e.* $\forall G$ $rw(G) \leq tw(G) + 1$.

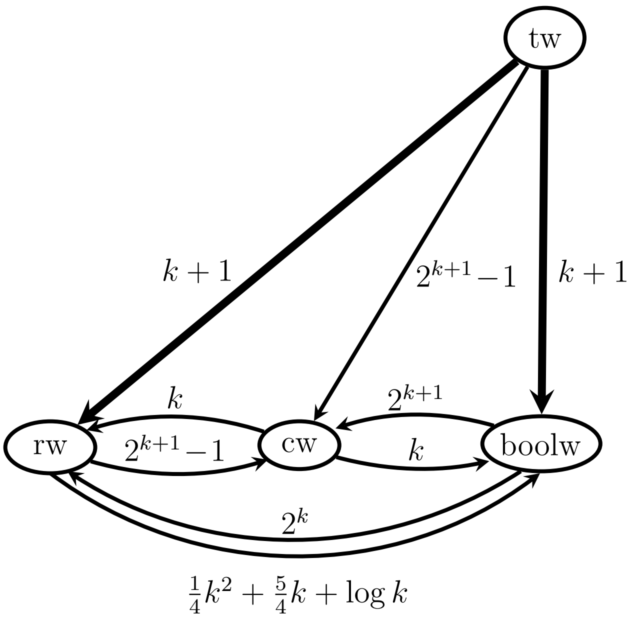


Figure 5.5: Transitive relation between tree-width and boolean-width was an exponential, boolw $\preceq 2^{\mathrm{tw}+1} - 1$. We improve it by showing linear bound bw $\preceq$ tw + 1. This also leads to improving bounds for all parameters above tree-width e.g. path-width, tree-depth, carving-width. We achieve similar improvements with rank-width.

Chapter 6

Conclusion

The previous chapter covers all remaining relations and completed comparison diagram (Figure 6.1). The longest chain of parameters starts at vertex cover number or maximum matching and ends at maximum clique. All together consists of 7 parameters. Another interesting chain goes from cut-width to group rank-width, clique-width and boolean-width. There is a possible detour through carving-width instead of path-width. Tree-width and branch-width play a role of bottleneck in the comparison diagram.

In additional to the thesis, we also created a web site at internet address `http://robert.sasak.sk/gpp` which summaries all results and presents comparison diagram in interactive way.

6.1 Correctness

Comparison diagram is the main result of the thesis. It is therefore important to ensure about it's correctness. During all four steps we always kept the list of relations that have to be examined and sequentially decreased it. This ensures correctness of the comparison diagram.

6.2 Discussion

The aim of the thesis was to provide the comparison diagram for 17 parameters without actually examining all pairs of parameters. We succeeded, our method rapidly reduced number of comparisons. Especially, unbound relations were obtained very easy. However, this required lot of preprocessing work in form of examining parameters on different graph classes.

Selection of graph classes partitioned parameters into reasonably big classes. That made examining parameters inside groups as expected.

6.3 Practical impact

Recall Theorems 1.1 and 1.2:

$$p \preceq q \wedge (\pi, p) \in \mathrm{FPT} \implies (\pi, q) \in \mathrm{FPT}$$

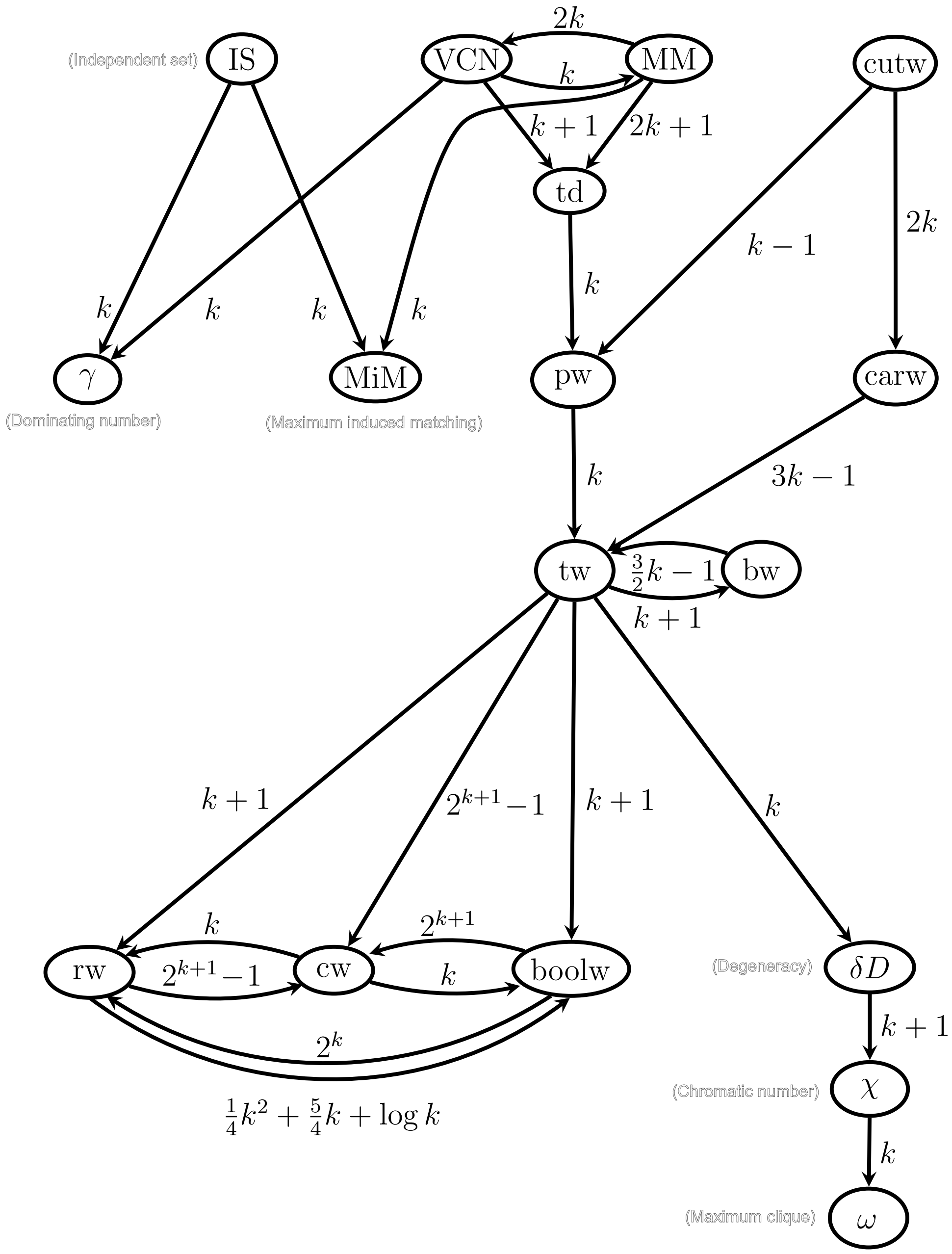


Figure 6.1: Complete comparison diagram of the 17 graph parameters. Provide a powerful tool for extending FPT or W-hard results. For example, if problem π parametrized by boolean-width is FPT then problem π is also FPT when parametrized by rank-width, clique-width, tree-width, path-width, carving-width, cut-width, tree-depth, maximum matching and vertex cover number (all "above" parameters). In opposite direction, if π parametrized by cut-width is W-hard then π is W-hard when parametrized by carving-width, path-width, tree-width and so on.
Arrow from the parameter p_2 to p_1 marked $f(k)$ means $p_1 \preceq p_2$, i.e. $\forall G : p_1(G) \leq f(p_2(G))$. List of all parameters with their abbreviations is in Table 1.1.

$$p \preceq q \land (\pi, q) \in \text{W-hard} \implies (\pi, p) \in \text{W-hard}$$

These theorems give us a tool to easily extend results for parametrized problems.

Example Problems Edge Dominating Set, Hamiltonian Cycle and Graph Coloring are W-hard parametrized by clique-width [9]. From our comparison diagram, it is easy to see those problems remains W-hard when parametrized also by rank-width or boolean-width.

Example Chordal Sandwich problem was not known to be FPT for any parameter. Paper [11] shows that it is FPT parametrized by VCN. From comparison diagram, we can see that Chordal Sandwich parametrized by maximum matching is FPT as well.

Example Independent Set problem is W[1]-hard parametrized by maximum independent set [14]. Therefore Independent Set is W-hard when parametrized by dominating set or maximum induced matching.

And many other results follow from $\preceq$ relation.

6.4 Further improvements

When designing an algorithm, multiplicative constants does matter. Therefore it is important whether $\forall G\ p_1 \leq p_2$ or $\forall G\ p_1 \leq 3 \cdot p_2$. Especially when running time is exponential in a parameter.

All relations were shown among all 17 parameters and therefore comparison diagram is complete. However, there are thousands known parameters. Basically, any graph property can be turn into parameter. By extending comparison diagram for more parameters, more results can be obtain.

Bibliography

[1] Dan Bienstock, Neil Robertson, Paul Seymour, and Robin Thomas. Quickly excluding a forest. *J. Comb. Theory Ser. B*, 52(2):274–283, 1991.

[2] Csaba Biró. Tree-width and grids. 2005.

[3] Hans L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. pages 1–16, 1998.

[4] B. M. Bui-Xuan, J. A. Telle, and M. Vatshelle. Boolean-width of graphs. pages 61–74, 2009.

[5] Tony Yu Chang and W. Edwin Clark. The domination numbers of the 5 x n and 6 x n grid graphs. *J. Graph Theory*, 17(1):81–107, 1993.

[6] John Clark and Derek Allan Holton. *A first look at graph theory*. World Scientific, Singapore, 1991.

[7] Derek G. Corneil and Udi Rotics. On the relationship between clique-width and treewidth. *SIAM J. Comput.*, 34(4):825–847, 2005.

[8] Bruno Courcelle and Stephan Olariu. Upper bounds to the clique width of graphs. *Discrete Appl. Math.*, 101(1-3):77–114, 2000.

[9] Fedor V. Fomin, Petr A. Golovach, Daniel Lokshtanov, and Saket Saurabh. Clique-width: on the price of generality. In *SODA '09: Proceedings of the twentieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 825–834, Philadelphia, PA, USA, 2009. Society for Industrial and Applied Mathematics.

[10] Martin Charles Golumbic and Udi Rotics. On the clique-width of perfect graph classes. In *WG '99: Proceedings of the 25th International Workshop on Graph-Theoretic Concepts in Computer Science*, pages 135–147, London, UK, 1999. Springer-Verlag.

[11] Pinar Heggernes, Federico Mancini, Jesper Nederlof, and Yngve Villanger. A parameterized algorithm for chordal sandwich. In *CIAC*, pages 120–130, 2010.

[12] Vít Jelínek. The rank-width of the square grid. *Discrete Appl. Math.*, 158(7):841–850, 2010.

[13] Jaroslav Nešetřil and Patrice Ossona de Mendez. Grad and classes with bounded expansion ii. algorithmic aspects. *Eur. J. Comb.*, 29(3):777–791, 2008.

[14] Niedermeier. *Invitation to Fixed Parameter Algorithms (Oxford Lecture Series in Mathematics and Its Applications).* Oxford University Press, USA, March 2006.

[15] Sang-il Oum. Rank-width and vertex-minors. *J. Comb. Theory Ser. B*, 95(1):79–100, 2005.

[16] Sang-il Oum. Rank-width and vertex-minors. *J. Comb. Theory Ser. B*, 95(1):79–100, 2005.

[17] Sang-il Oum. Rank-width is less than or equal to branch-width. *J. Graph Theory*, 57(3):239–244, 2008.

[18] Sang-il Oum and Paul Seymour. Approximating clique-width and branch-width. *J. Comb. Theory Ser. B*, 96(4):514–528, 2006.

[19] Y Rabinovich and J A Telle. On the boolean-width of a graph: structure and applications. Technical Report arXiv:0908.2765, Aug 2009.

[20] M.R. Fellows R.G. Downey. *Parameterized Complexity.* Springer, Berlin, 1999.

[21] Neil Robertson and P. D. Seymour. Graph minors. x. obstructions to tree-decomposition. *Journal of Combinatorial Theory, Series B*, 52(2):153 – 190, 1991.