



UNIVERSITY OF BERGEN

DEPARTMENT OF INFORMATICS

ALGORITHMS

---

**Exploring graph parameters  
similar to tree-width and  
path-width**

---

*Student:*  
Joakim Alme Nordstrand

*Supervisor:*  
Jan Arne Telle

Master Thesis  
June 2017

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Definitions</b>	<b>5</b>
<b>3</b>	<b>Tree-like parameters</b>	<b>7</b>
3.1	Tree-width . . . . .	7
3.2	Branch-width . . . . .	9
3.3	Maximum matching-width . . . . .	11
3.4	Subtree representation . . . . .	12
<b>4</b>	<b>The fourth tree-like parameter</b>	<b>17</b>
4.1	x-width is tree-width . . . . .	17
<b>5</b>	<b>Defining the tree-like parameters with chordal supergraphs</b>	<b>19</b>
5.1	Tree-width . . . . .	20
5.2	Branch-width . . . . .	20
5.3	Maximum matching-width . . . . .	21
<b>6</b>	<b>Path-like parameters</b>	<b>26</b>
6.1	Path-width . . . . .	26
6.2	Linear branch-width . . . . .	27
6.3	Linear maximum matching-width . . . . .	27
6.4	Comparing the different path-like parameters . . . . .	28
<b>7</b>	<b>Minimal forbidden minors</b>	<b>33</b>
7.1	MFM for tree-like parameters . . . . .	33
7.2	MFM path-width . . . . .	35
7.3	MFM linear branch-width . . . . .	36
7.4	MFM linear maximum matching-width . . . . .	38
<b>8</b>	<b>Conclusion</b>	<b>42</b>

## Abstract

In a recent paper appearing at IPEC 2015, "Maximum matching width: new characterization and fast algorithms for dominating set" [12], three similar tree-like parameters, tree-width, branch-width and maximum matching-width, are studied using a common framework. Here we extend the work of that paper in several ways. First we will answer one of the open problems by proving that the "last parameter" from the framework defining the three parameters is equal to tree-width. Second we fill out the details in one of the proofs to ensure its correctness. Third we define two new parameters, linear branch-width and linear maximum matching-width. These are the "path-like" variants of branch-width and maximum matching-width respectively in the same sense that path-width is the "path-like" variant of tree-width. We show that linear branch-width is almost identical to path-width, for any graph  $G$  we have  $pw(G) \leq lbw(G) \leq pw(G) + 1$ , and that when a graph has linear maximum matching-width  $k$  then its path-width is somewhere between  $k$  and  $2k$ . We also explore the minimal forbidden minors of graphs with linear branch-width less than 1,2 and 3 and with linear maximum matching-width less than 1,2 and 3.

## Acknowledgements

Thank you to all my fellow student and the algorithms group that provided with interesting discussion and helpful insight into the different aspects of algorithms and academia. And of course thank you to my family for all the support throughout this thesis and for proofreading. Special thanks to my spouse for all your support on top taking care of our two lively kids when I was busy with the thesis. And finally thank you Jan Arne for helping me navigate though this thesis, from finding what the thesis should be about both before and after we realized that my first assignment was a bit to much.

## 1 Introduction

When solving problems we often distil them down to get to the core of the problem. This can be done by considering only certain parts of the problem or represent the problem in a specific way to make it more comprehensive. A well known such representation method is the graph representation, which saw its rise after Euler's "Seven bridges of Königsberg". The paper that is often attributed to be the start of graph theory, a now important and well studied field of mathematics. One important aim in graph theory is to explore and show how structural properties in graphs behave and relate, and there are a plethora of different graph properties relating to structure, for instance planar, connected, dense, acyclic or bipartite to name a few. When trying to solve an important problem through a graph representation pinning down which structural properties that applies to the graph is often a crucial key, since we then can use all we know from graph theory to get more insight into the problem at hand

Name	Function name	Subtree rep.
Tree-width	tw	node con./node weight
Branch-width	bw	edge con./edge weight
Maximum matching-width	mmw	node con./edge weight
X-width	xw	edge con./node weight

Figure 1: Table of all the tree-like parameters discussed in this thesis. First column is the parameter name, second column is the short version, third column is the short hand definition with respect to subtree representation.

Some structural properties are quantifiable by a parameter. For graphs it can for instance be the number of vertices, the largest clique or the chromatic number etc. Tree-width is a well know graph parameter. It tells us how tree-like a graph is and is a fundamental parameter in the field of fixed parametrized complexity where for low tree-width values we know efficient algorithms for many different problems. Also trees has many interesting results tied to it that makes it an important theoretical tool too. Branch-width is closely related to tree-width and is always within a constant factor from tree-width for any given graph, as we will see. Another closely related parameter of tree-width, introduced by Vatschelle, is maximum matching-width which is within a constant factor of tree-width for any given graph as well. Jeong, Sæther and Telle shows that tree-width, branch-width and maximum matching-width can be defined through subtree representations of similar structure [5]. The subtree representation differ only on whether subtree pairs corresponding to edges has to overlap at a node or an edge and whether we are capacity bound at nodes or at edges. Choosing between node and edge at the two places gives rise to the four parameters listed in figure 1. In light of this studying the different relations between these parameters might give a deeper understanding of them. In figure 2 we can see how they all relate. x-width is not included since it is equal to tree-width, as we will see in section 4.

In section 3 we define the three tree-like parameter and show how they relate, see table 2, and how they fit into the subtree representation framework. We will use what is called subtree representation.

Another definition of tree-width is through chordal supergraphs with bounded maximal cliques size equal to the tree-width of the graph plus one. Since the subtree representation so nicely could define the three tree-like parameters, this

	tw	bw	mmw
tw	1	1	1
bw	3/2	1	1
mmw	3	2	1

Figure 2: A table that shows how the different tree-like parameters relate to each other. The constant  $c$  in a cell  $(Aw, Bw)$  means than if a graph  $G$  has  $Aw \leq k$  then it has  $Bw \leq ck$  plus some constant.

Name	Function name
Path-width	pw
Linear branch-width	lbw
Linear maximum matching-width	lmmw

Figure 3: Table of the path-like parameters and their function names.

chordal supergraph structure might also define them with only slight differences between the three. Section 5 delves into this.

A more restrictive version of tree-width is the path-width. Path-width (pw) measures how path-like a graph is and can be defined similar to tree-width but with the decomposition tree restricted to be a path. This linearisation of tree-width to get path-width can be done with any parameter that utilizes a decomposition tree, e.g. branch-width and maximum matching-width. We call these linear versions the linear branch-width and the linear maximum matching-width respectively. The table in figure 3 lists them alongside their function names. Section 6 defines these three path-like parameters and prove how they relate for a given graph, see the table in figure 4.

$H$  is a minor from  $G$  if  $H$  can be formed from  $G$  by deleting edges and vertices and by contracting edges. Robertson and Seymour's graph minor project proves that graphs are well-quasi-ordered under the minor relation [10]. Which means that any family of graphs  $F$  that is minor closed can be defined by a finite set of graphs  $M$  that upper bounds  $F$  with respect to the minor relation, i.e. iff a graph does not have any of the graphs in  $M$  as a minor, then it is part of  $F$ . We call  $M$  a set of forbidden minors, and it is minimal if no element in  $M$  is a minor of another element in  $M$ . Kuratowski's theorem tells us that the minimal forbidden minors of planar graphs is the complete graph  $K_5$  and the complete bipartite graph  $K_{3,3}$ . This gives us great insight to the structure of planar graphs, and also a short and nice way to define them without sphere embedding and edge crossings. The graph classes with tree-width or branch-width bounded by 1,2,3 and 4 also has few minimal forbidden minors, and for a given bound. Additionally the minimal forbidden minors of graphs with  $tw(G) \leq k$  and the minimal forbidden minors of  $bw(G) \leq k$  always has some similarities. For instance graphs of tree-width less than 3 only has one minimal forbidden minor, the  $K_4$  graph. This is also the only minimal forbidden minor of graphs

	pw	lbw	lmmw
pw	1	1	1
lbw	1	1	1
lmmw	2	2	1

Figure 4: A table that shows how the different path-like parameters relate to each other. The constant  $c$  in a cell  $(Aw, Bw)$  means that if a graph  $G$  has  $Aw \leq k$  then it has  $Bw \leq ck$  plus some constant.

with branch-width less than 3. Section 7 discusses the minimal forbidden minors of bounded values of the three tree-like parameters and the three path-like parameters. We show the set of minimal forbidden minors for linear maximum matching-width for the values 1 and 2.

## 2 Definitions

As the title gives away this section is just to define the terminology used in this thesis.

A graph  $G = (V, E)$  is a pair. The first part is a set of vertices  $V$  (or nodes) and the other is a set of edges  $E \subseteq \{\{u, v\} \mid u, v \in V, u \neq v\}$ . To ease notation we write  $uv$  in place of  $\{u, v\}$  for edges. Whenever we write  $V(G)$  and  $E(G)$  we mean the set of vertices and the set of edges in  $G$  respectively. All graphs in this thesis are finite, i.e.  $V(G)$  and  $E(G)$  are finite sets.

Interpreting graphs just from the vertex and edge sets is in most cases very hard, so we usually depict graphs as figures with a point for every vertex and lines between points for the edges. In figure 5 we have depicted graph  $G$  with  $V(G) = \{a, b, c, d, e, f, g\}$  and  $E(G) = \{ab, ac, bc, bd, be, ce, cf, de, ef, fg\}$ .

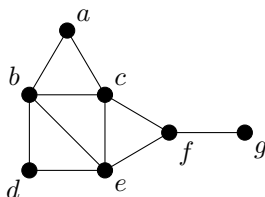


Figure 5: A graph depicted with dots and lines

The empty graph is the graph with empty vertex set and empty edge set, and the trivial graph is the graph with one vertex and empty edge set. The ends of an edge are the two vertices composing the edge. Vertices are incident to an edge if it is one of the ends of the edge and vice versa. Two vertices are incident to an edge if they are incident to the same edge. The set of all neighbours of a given vertex  $x$  is called the neighbourhood of  $x$ , and the neighbourhood of a set of vertices are all the vertices outside the set with a neighbour in the set. The degree of a vertex is the size of its neighbourhood. A vertex is independent if it has degree 0 and a set of vertices is independent if none of the pairs in the set is neighbours.

A graph  $H$  is a subgraph of a graph  $G$  if  $V(H) \subseteq V(G)$  and  $E(H) \subseteq E(G)$ , we say that  $H$  is in  $G$ . Also  $G$  is a supergraph of  $H$ . A subgraph is said to cover a vertex/edge if it contains it. If two subgraphs of a graph contain the same vertex/edge we say that they overlap, more specifically they overlap at that vertex/edge.

A path is a graph  $P_n = (v_1, v_2, \dots, v_n)$  with  $V(P_n) = \{v_1, v_2, \dots, v_n\}$  and  $E(P_n) = \{v_1v_2, v_2v_3, \dots, v_{n-1}v_n\}$ . A cycle is a graph  $C_n = (v_1, v_2, \dots, v_n)$  that is a path  $P_n$  with the addition of the edge  $v_nv_1$ . The connected graphs are the graphs  $G$  where for every pair of vertices there exists path in  $G$  covering both of them. Unless otherwise specified all the graphs in this thesis are connected.  $H$  is a component of  $G$  if it is a connected subgraph which is maximal with respect to both vertex and edge sets. A tree  $T$  is a (connected) graph without cycles. All degree one vertices in a tree are called the leaves with  $L(T)$  being the set of all the leaves in a tree  $T$ . Subtrees are the connected subgraphs of a tree. A graph

is complete if all its vertex pairs are edges of the graph, usually denoted  $K_n$ . If a subgraph is complete we call it a clique. A bipartite graph  $B$  is a graph with  $V(B) = (A_1 \cup A_2)$  where  $A_1 \cap A_2 = \emptyset$  and both  $A_1$  and  $A_2$  are independent sets. Both the length of a path and a cycle and the size of a clique is the size of the vertex set involved.

We say that a cycle in a graph  $G$  has a chord if there exists an edge in  $G$  where both ends are in the cycle but the edge is not. Further a graph is a chordal graph if all its cycles of length at least 4 has a chord.

Deleting an edge is to remove it from the edge set. Deleting a vertex is to remove it from the vertex set together with all edges it is incident to. Contracting an edge  $uv$  is to make, without loss of generality,  $u$  neighbour all the neighbours of  $v$  and then delete  $v$ .

Given a set of vertices  $S$  in a graph  $G$ , if there exists a pair of vertices in a component  $G$  but not in  $S$  that is not in the same component after deleting  $S$  from  $G$ , then  $S$  is a vertex separator. The vertex subset  $U \subseteq V(G)$  of a graph  $G$  is said to (vertex) induce the graph  $G[U] = (U, \{uv | u \in U, v \in U, uv \in E(G)\})$ . The edge subset  $W \subseteq E(G)$  of a graph  $G$  is said to (edge) induce the graph  $G[W] = (\{v | uv \in W\}, W)$ .

We will use the word bag in some specific cases instead of set. This is for both historical purposes and this also makes reading easier. And unless otherwise specified the constants on this thesis will be non-negative integers.



### 3 Tree-like parameters

In this section we review some earlier results on the tree parameter  $tw$ ,  $bw$  and  $mmw$ . In section 3.4 we discuss the subtree representation, which will be used later.

Trees are an important algorithmic tool, they are a go to base case for many problems since they, among other things, separate easily. This is why decomposing graphs into trees is of interest, which in layman's terms is a projection of a graph onto some tree. This can be done in many different ways, but with the different decompositions we will look at we are interested in a decomposition trees with nodes that separates the original graph in some way. Tree decomposition is the first one we will define, which since its use in the graph minor project [1] has proven to be a very useful when trying to understand underlying structures of problems. Alongside tree decomposition there is branch decomposition and maximum matching decomposition. It is well know that tree-width and branch-width is within a constant factor from each other and maximum matching-width falls into this grouping as well. Studying the way these parameters interplay might cast some new light onto structural properties of graphs.

#### 3.1 Tree-width

We start out with tree-width since it is both the historically first one and arguably the most important. It rose to popularity after its appearance in the Graph Minor Project [11]. Tree-width has multiple ways of defining it, many giving good insight to how tree-width behaves. The cops and robber definition is maybe the most visual. In short imagine a graph as a city grid, with edges as roads and vertices as intersections. If one (infinitely) fast robber runs along the streets, how many slow (finitely fast) cops does it take to catch the robber. A cop catches the robber if they cross paths. The amount of cops needed will be the same as the tree-width of the graph. Its easy to see the how the cop squad relates to separator of the graph. In the standard definition it is not as clear, but this definition is more convenient mathematically. We will do a slight "cosmetic" alteration to the traditional definition where we bound the degree of the decomposition tree by three.

**Definition 3.1.** *Tree decomposition*

*Given a graph  $G$  then a tree decomposition of  $G$  is a tree  $T$  of maximum degree 3 with nodes  $X_1, X_2, \dots, X_n$  where every node is a bag containing vertices from  $G$  if it satisfies the following:*

1. *The union of every bag in  $T$  is equal to  $V(G)$ .*
2. *For all edges  $uv \in E(G)$ , there is a bag  $X_i \in V(T)$  containing  $u$  and  $v$ .*
3. *For every two bags  $X_i$  and  $X_l$  in  $V(T)$ , all vertices that are both in  $X_i$  and  $X_l$  must be in all bags  $X_j$  on the path from  $X_i$  to  $X_l$ .*

Constructing a tree decomposition of a graph is not hard. The easiest tree decomposition for any graph  $G$  is the trivial tree decomposition consisting of one bag that contains every vertex in  $G$ . The challenge with tree decomposition is to find a decomposition that has as small bags as possible. This is what quantifies the tree-width of a graph.

**Definition 3.2.** *Tree-width*

If  $T$  is a tree decomposition of a graph  $G$  then the width of  $T$  is equal to the size of its largest bag minus 1. The tree-width of  $G$  ( $tw(G)$ ) equals the smallest tree decomposition of  $G$  with respect to width.

An example of a tree decomposition  $T$  of a graph  $G$  can be seen in figure 6. We can see that all vertices are present in at least one of the bags. Same for all the vertex pairs composing the edges, thus satisfying the first and second criterion of tree decomposition. For every vertex  $v \in V(G)$  the induced subgraph of  $T$  consisting off all the bags containing  $v$  is connected, satisfying the third criterion. We can see that the width of  $T$  is 2. The question is whether there exists a tree decomposition of  $G$  where all bags in the decomposition are of size less than 3. If a decomposition has the same width as the width of the graph it decomposes, we call it optimal.

Our first theorem is a well know result when it comes to tree decomposition, and is good for lower bounding the tree-width of graphs as well as more abstract results.

**Theorem 3.1.** *If  $T$  is a tree decomposition of a graph  $G$  and  $Q$  is a clique in  $G$  then there is a bag in  $T$  containing all of  $V(Q)$*

*Proof.* For any bag  $X_i$  containing a strict subset  $A$  of  $V(Q)$  and  $y \in V(Q)$ ,  $y \notin A$  we can find a bag containing  $A \cup y$ . This will prove the lemma. If  $X_i$  is a bag containing  $y$  and  $X_j$  is the last bag on the path from  $X_i$  to  $X_l$  containing all of  $A$ , then by property three of tree decomposition there must be a vertex  $z \in V(Q)$  that is not part of the remaining path. Since  $yz \in E(G)$  there must be a bag containing them both, by property two, and by property three this must be the case in  $X_j$ .  $\square$

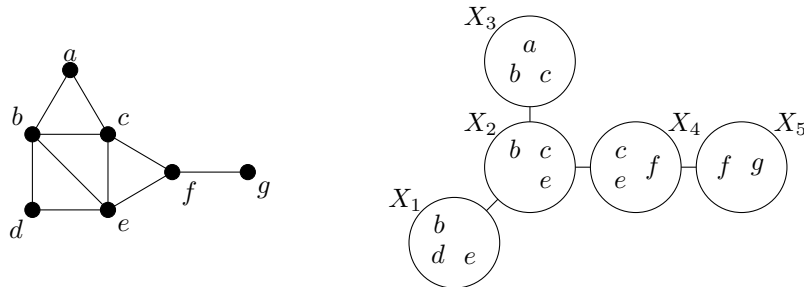


Figure 6: The graph on the right is a tree decomposition of the graph on the left. Inside each node is written the vertices from its bag.

Since the original graph  $G$  from figure 6 has a clique of size 3 it must have tree-width at least 2 by theorem 3.1 making the tree decomposition from the same figure an optimal tree decomposition and thus  $G$  has tree-width 2.

The elusive group "keen readers" might have noticed that we used node in place of vertex in the tree decomposition definition. This we will do when dealing with decomposition trees to more clearly distinguish between the original graph and the decomposition tree. Not having the degree bound on the decomposition will make our proofs longer and it makes no difference with respect to tree-width. Because for any bag  $X_i$  in a decomposition tree of a tree decomposition with degree more than 3 we can split it up into two bags, both containing  $X_i$ , with an edge between them and distribute the neighbours between the two. Property one and two is not tempered with and property three still holds since for every bag we split the content of the two new bags is precisely the content of the split bag. The "minus 1" in the definition of tree-width might seem arbitrary, but doing so makes trees have tree-width 1, which is nice.

Before moving onto branch-width we will define clique decomposition, a special type of tree decomposition on chordal graphs. This decomposition will come in handy when proving relations between the different parameters.

**Definition 3.3.** *Clique decomposition*

*Given a chordal graph  $G$  where  $Q_1, Q_2, \dots, Q_m$  are all the maximal cliques in  $G$  then a tree  $T$  with node set  $V(T) = \{X_1, X_2, \dots, X_m\}$  is a clique decomposition if*

1. *For every maximal clique  $Q_i$  in  $G$ ,  $X_i = V(Q_i)$ .*
2. *For all vertices  $v_i$  in  $G$  the subgraph of  $T$  induced by the bags containing  $v_i$  is connected.*

Figure 6 also serves as a clique decomposition since the decomposed graph is chordal. To see that clique decomposition is a variant of tree decomposition notice that property three of a tree decomposition is the same as property two of a clique decomposition and property one and two of a tree decomposition follows from property one in clique decomposition, so every clique decomposition is a tree decomposition with the exception that it might not have maximum degree 3, but as we have discussed this is not an issue with respect to tree-width. A clique decomposition is even an optimal (non-degree bounded) tree decomposition because its largest bag contains a maximum clique from the graph, and by 3.1 it must be optimal.

### 3.2 Branch-width

Our second parameter is branch-width. Branch-width is sometimes dubbed the cousin of tree-width. Whereas tree decomposition, in a sense, traverses graphs using vertices, branch decomposition traverses graphs using edges. The structure of branch decomposition is different from that of tree decomposition, but we are still using a decomposition tree.

**Definition 3.4.** *Branch decomposition*

Given graph  $G$  then a branch decomposition of  $G$  is a pair  $(T, \delta)$  where  $T$  is a tree of maximum degree 3 and  $\delta$  is a bijection  $\delta : E(G) \rightarrow L(T)$ , i.e. from the edges in  $G$  to the leaves in  $T$ .

As with tree decomposition, branch decompositions are not hard to construct, but again we generally want the branch decomposition to be "small" with respect to some separators. The separators in question here are the middle sets. Given a branch decomposition  $(T, \delta)$  of a graph  $G$ , then for each  $e_i \in E(T)$  we get two components,  $T_1$  and  $T_2$ , when we delete  $e_i$  from  $T$ . Using the two edge sets corresponding to all the leaves on each of the components we get two edge induced graphs  $G_1$  and  $G_2$  respectively.  $G_1$  and  $G_2$  might have some vertices in common, the middle set is precisely these vertices, i.e.  $mid(e_i) = V(G_1) \cap V(G_2)$ .

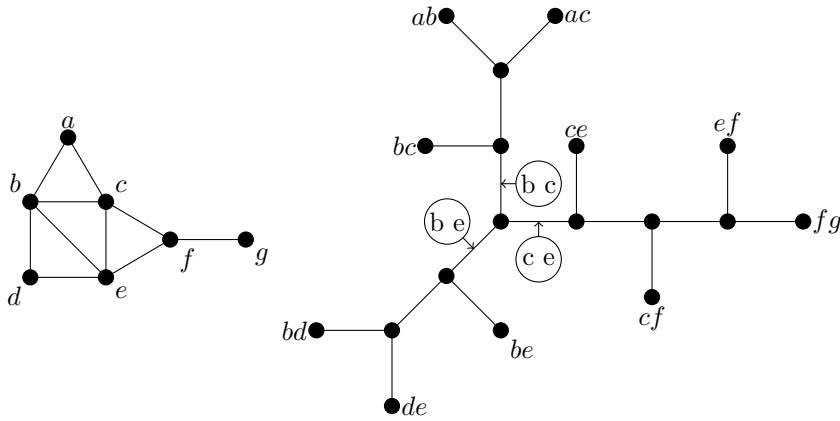


Figure 7: The right hand part is a branch decomposition of the graph in the left hand part. The three bubbles are the middle sets of the edges they point to.

**Definition 3.5.** *Branch-width*

Given a graph  $G$  and a branch decomposition  $(T, \delta)$  of  $G$ , then the width of  $(T, \delta)$  is equal to the size of the largest middle set over all the edges of  $T$ . The branch-width of  $G$  is equal to the width of the smallest branch decomposition, with respect to its width. Since a branch decomposition of them is not well defined, the branch-width of the empty graph and the trivial graph is 0 and the single edge graph has branch-width 1.

An example of a branch decomposition  $(T, \delta)$  is illustrated in figure 7. The three middle sets shown in the figure are of size 2, and a quick check shows that the other middle sets are all of size less or equal to 2. If a graph has an edge  $e$  where both ends have degree larger than 1, then its branch-width must be at least 2, by the middle set of the leaf corresponding with  $e$ . Thus graph  $G$  from figure 5 has branch-width 2.

Tree-width and branch-width for a given graph  $G$  is within a constant factor of each other, specifically:  $bw(G) \leq tw(G) + 1 \leq \max(2, (3/2)bw(G))$  and this relation is tight [9].

In that respect tree-width and branch-width are very close. Additionally if a family of graphs, for instance all outer planar graphs, is upper bounded by one of the parameters by a constant factor it will also be upper bounded by the other parameter by some other constant factor. For all trees with more than one vertex tree-width is 1 and branch-width is 1 or 2.

### 3.3 Maximum matching-width

The third tree-like width parameter we will look into is maximum matching-width, first defined by Vatschelle [12]. He used it mainly as a stepping stone when comparing tree-width with other parameters, but it has been shown to have very similar structural properties in relation with both branch-width and tree-width.

**Definition 3.6.** *Maximum matching decomposition*

*A maximum matching decomposition of a given graph  $G$  is a pair  $(T, \partial)$  consisting of a tree  $T$  of maximum degree 3, and a bijection  $\partial : V(G) \rightarrow L(T)$ , i.e. from the vertices in  $G$  to the leaves in  $T$ .*

We can see that this decomposition is quite similar to branch decomposition with the difference that the bijection is between the vertices and leaves instead of edges and leaves. For the width of a maximum matching decomposition  $(T, \partial)$  we also look at separator sets generated from cuts in the decomposition graph but this time around it is a bit more involved. For each  $e_i \in E(T)$  we get two components,  $T_1$  and  $T_2$  by deleting  $e_i$  from  $T$ . These components in turn generate two vertex sets  $D_1$  and  $D_2$  by taking the vertices corresponding to the leaves of each component respectively. The graph  $bip(e_i) = B_i$  is then the bipartite subgraph of the original graph induced by the set of all edges with one endpoint in  $D_1$  and the other in  $D_2$ .

**Definition 3.7.** *Maximum matching-width*

*Given a graph  $G$  and a maximum matching decomposition  $(T, \partial)$  of  $G$ , then the width of  $(T, \partial)$  is equal to the size of the largest maximum matching over all bipartite subgraphs  $bip(e_i) = B_i$  for all  $e_i \in T$ . The maximum matching width of  $G$  ( $mmw(G)$ ) is the width of the smallest maximum matching decomposition of  $G$  with respect to its width. Since a maximum matching decomposition of the empty graph is not well defined the maximum matching-width of the empty graph is defined as 0.*

Figure 8b above shows a maximum matching decomposition  $(T, \partial)$  of the graph  $G$  in figure 8a. The bipartite subgraph  $bip(e_i) = B_i$  from  $(T, \partial)$  is shown in figure 8c.  $B_i$  has a maximum matching of size 2. Checking the maximum matchings of the other bipartite subgraphs of  $G$  induced by  $(T, \partial)$  reveals that  $(T, \partial)$  has width 2. Now let us prove that  $(T, \partial)$  is optimal. Any maximum matching decomposition of  $C_4$  must have an edge that gives us a bipartite graph

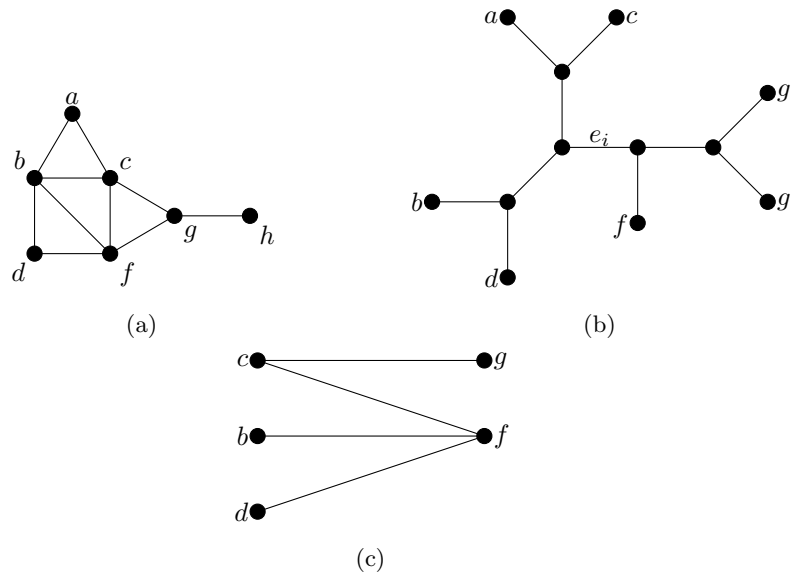


Figure 8

with two vertices from  $C_4$  on each side. Let  $\{a, b, c, d\}$  be the four vertices of  $C_4$  within a graph  $G$  and  $(T, \partial)$  any decomposition of  $G$  where  $\{\nu_a, \nu_b, \nu_c, \nu_d\}$  are the leaf nodes in  $T$  corresponding with the vertices of  $C_4$ . Denote the path from  $\nu_a$  to  $\nu_b$  as  $P_1$  and the path from  $\nu_c$  to  $\nu_d$  as  $P_2$ . If  $P_1$  and  $P_2$  intersect at an edge, then this edge yields a bipartite graph with two vertices from  $C_4$  on each side. If  $P_1$  and  $P_2$  do not intersect at an edge they do not intersect at all, since  $T$  has maximum degree 3. This means that there is an edge from  $\nu_a$  to  $\nu_d$  that will yield a bipartite graph with  $\nu_a$  and  $\nu_b$  on one side and  $\nu_c$  and  $\nu_d$  on the other. Now we can say that since  $G$  contains a 4 cycle  $mmw(G) \geq 2$ , but we know that  $(T, \partial)$  from figure 8b that  $mmw(G) \leq 2$  and we have that  $(T, \partial)$  is optimal.

Vatshelle shows that for any graph  $(tw(G)+1) \leq 3*mmw(G)$  and  $mmw(G) \leq \max(bw(G), 1)$  and that these relations are tight [12]. Also  $bw(G) \leq 2*mmw(G)$  as we will show in the next subsection.

### 3.4 Subtree representation

Since the three tree-like parameters upper bound the same graph classes, one can hope they are similar in other aspects as well. In [5] Telle et. al. gives alternative definitions for the three parameters using subtree representation.

**Definition 3.8.** *Subtree representation*

Given a graph  $G$  with vertex set  $v_1, v_2, \dots, v_n$  then the subtree representation  $R = (T, \{T_1, T_2, \dots, T_n\})$  of  $G$  is a pair where  $T$  is a tree of maximum degree 3 and  $\{T_1, T_2, \dots, T_n\}$  a set of  $n$  non-trivial subtrees of  $T$ .

$R$  is node connected with respect to  $G$  if for every edge  $v_i v_j \in E(G)$  the subtree

$T_i$  and  $T_j$  overlap at a node in  $T$ .  $R$  is edge connected with respect to  $G$  if for every edge  $v_i v_j \in E(G)$  the subtree  $T_i$  and  $T_j$  overlap at an edge in  $T$ . The node weight of a subtree representation is the maximum number of subtree that overlap at one node, over all nodes in  $T$ . The edge weight of a subtree representation is the maximum number of subtree that overlap at one edge, over all nodes in  $T$ .

When we write only "subtree representation of  $G$ " we will mean "subtree representation of  $G$  that is node connected with respect to  $G$ ". Notice that a subtree representation that is edge connected with respect to  $G$  is also a node connected with respect to  $G$ . The following three theorems serves as alternative definitions of the three tree-like parameters defined using subtree representations.

**Theorem 3.2.**

For  $0 \leq k$  and a graph  $G$  with vertices  $v_1, v_2, \dots, v_n$  then  $tw(G) \leq k - 1$  iff there exists a subtree representation  $R = (T, \{T_1, T_2, \dots, T_n\})$  that is **node** connected with respect to  $G$  and has **node** weight less or equal to  $k$ .

**Theorem 3.3.**

For  $2 \leq k$  and a graph  $G$  with vertices  $v_1, v_2, \dots, v_n$  then  $bw(G) \leq k$  iff there exists a subtree representation  $R = (T, \{T_1, T_2, \dots, T_n\})$  that is **edge** connected with respect to  $G$  and has **edge** weight less or equal to  $k$ .

**Theorem 3.4.**

For  $1 \leq k$  and a graph  $G$  with vertices  $v_1, v_2, \dots, v_n$  then  $mmw(G) \leq k$  iff there exists a subtree representation  $R = (T, \{T_1, T_2, \dots, T_n\})$  that is **node** connected with respect to  $G$  and has **edge** weight less or equal to  $k$ .

As we can see the second part of all these theorems only differ at two words, here in bold, between node and edge. This help us see the close relation between these three parameters. They also differ at which values of  $k$  they hold true, but the outlying graphs are just the star graphs with respect branch-width, and the trivial graph with respect to maximum matching-width. Another point is that the subtrees can be trivial in the subtree representation of theorem 3.2 and it will still hold, but we will keep them non-trivial for convenience.

Examples of subtree representations of the graph  $G$  in figure 9 can be seen in figure 10 and 11. Figure 10 has edge weight 2 and node weight 3, same for 11. Figure 10 is node connected with respect to  $G$  and 11 is edge connected with respect to  $G$ , which tells us that  $tw(G) = bw(G) = mmw(G) \leq 2$

Theorem 3.2 is proven by constructing a tree decomposition  $T'$  from a subtree representation  $R = (T, \{T_1, T_2, \dots, T_n\})$  in the backwards direction and doing the reverse in the forward direction. Backward: let  $R$  be any subtree representation with node weight  $k$  of a graph  $G$ . We construct a tree decomposition  $T'$  from  $R$  by letting  $T' = T$  and for every bag  $X_i \in V(T')$  add every vertex  $x_j$  iff  $T_j$  covers node  $v_i \in V(T)$ .  $T'$  will satisfy the three properties for tree decomposition and bags will not exceed size  $k$  since  $R$  has node weight no more than  $k$ . Forward: essentially the reverse of the backwards direction, but the tree might need to be slightly modified to ensure non-triviality of subtrees. Let  $T'$  be a tree

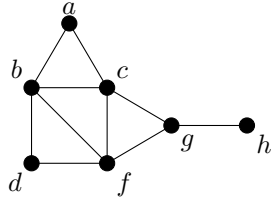


Figure 9

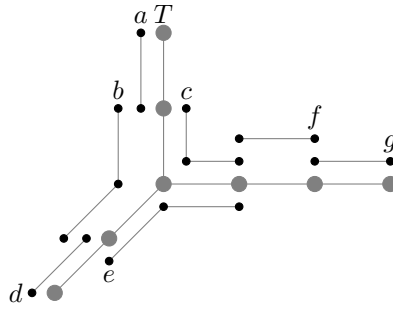


Figure 10: A subtree representation that is node connected with respect to the graph in figure 9. We see that the node weight is 3, whilst the edge weight is 2.

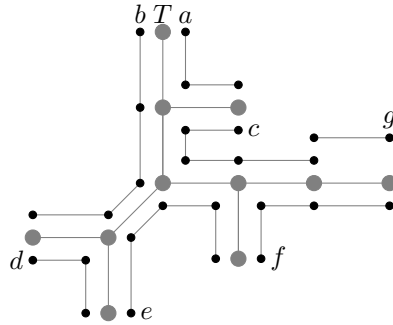


Figure 11: A subtree representation that is edge connected with respect to the graph in figure 9 with edge weight 2.



decomposition of a graph  $G$  with width  $k - 1$ . Extending every node in  $T'$  into two nodes with an edge between them and distributing the neighbours between the two in such a way that the resulting tree does not have a degree 4 vertex will suffice, see section 4 for more details. This extension will be our  $T$ . Then, for all  $x_i \in V(G)$ , let subtree  $T_i$  be the vertex induces subgraph using every node in  $T$  that comes from a bag in  $T'$  containing  $x_i$ . The extension ensures non-triviality and together with three tree decomposition properties this ensures that this is a subtree representation that is node connected with respect to  $G$  of node weight no more than  $k$ .

The proof of theorem 3.3 follows the same idea as the proof of 3.2 [7]. Proving theorem 3.4 is more involved, but again it uses the same idea [5]. From these three theorems the hierarchy between the tree parameters arises. As we will see in the next section section a subtree representation that is node connected with respect to some graph can be transformed so that is is edge connected with respect to the same graph, without increasing the node weight. Further the edge weight of a subtree representation can never exceed its node weight. Combining these two observations together with theorem 3.2 and 3.3 we get that branch-width is always below tree-width (+1) for branch-width values above 1. Comparing theorem 3.3 with theorem 3.4 we easily see that maximum matching-width is always below branch-width. because when a subtree representation is edge connected with respect to a graph it will also be node connected with respect to the same graph. So we have the tight [5] relation  $mmw(G) \leq bw(G) \leq tw(G) + 1$  for all non-star graphs from the subtree representation, but all star graphs  $S$  with at least two vertices has  $tw(S) = bw(S) = mmw(S) = 1$ , and for the trivial graph they are all 0, and the empty graph tree-width is -1 and branch-width and maximum matching-width is 0. Thus the inequality is holds for all graph.

Note that any subtree representation with edge weight  $k$  cant have node weight more then  $3k$ , for this reason we also have that  $tw(G) + 1 \leq 3mmw(G)$ , for all graph except the trivial graph. This is also a tight inequality [5].

To get the final inequality we need to alter a subtree representation  $R = (T, \{T_1, T_2, \dots, T_n\})$  that is node connected with respect to a graph, into a  $R' = (T', \{T'_1, T'_2, \dots, T'_n\})$  subtree representation that is edge connected whilst keeping the edge weight as low as possible. First subdivide every edge, i.e. for every edge  $e$  add a node, delete the  $e$  and make the new node neighbour both ends of  $e$ . Now for all nodes that does not stem from a subdivision, if it has degree one, do noting. If it has degree two extend every subtree to cover both edges incident to the node. If it has degree three let  $e_1, e_2$  and  $e_3$  be the three edges, extend all subtrees that cover  $e_1$  to cover  $e_2$ , all that cover  $e_2$  to cover  $e_3$  and all that cover  $e_3$  to cover  $e_1$ . Note that the subdivision makes sure that the weight of one edge is only affected by one node, and it at most it doubles. Since  $R$  is node connected with respect to the original graph  $R'$  will be edge connected with respect to the original graph by construction. We now have the final inequality  $bw(G) \leq 2mmw(G)$ , for non-star graph but a quick check shows that it will hold for all graphs. This inequality is also tight by the complete graphs.

Thus the subtree representation has given us all tight the relations between

the tree-like parameters without much effort.

## 4 The fourth tree-like parameter

### 4.1 x-width is tree-width

An open question from [5] is what type of parameter we would get if we looked at the node weight of a subtree representation that is edge connected with respect to a graph.

The following quote is taken from [5]: "There is also a fourth way of defining a parameter through these intersections of subtrees representation; where subtrees  $T_u$  and  $T_v$  must share an edge if  $uv \in E(G)$  (similar to branchwidth) and the width is defined by the maximum number of subtrees sharing a single vertex (similar to treewidth). This parameter will be an upper bound on all the other three parameters, but might it be that the structure this parameter highlights can be used to improve the runtime of Dominating Set beyond  $O^*(3tw(G))$  for even more cases than those shown using mm-width and branchwidth?"

We will call this parameter x-width and prove that it is equal to tree-width.

Name	Subtree rep.
Tree-width	node con./node weight
Branch-width	edge con./edge weight
Maximum matching-width	node con./edge weight
x-width	edge con./node weight

Figure 12: Table of all the tree-like parameters discussed in this thesis. First column is the parameter name, second column is the short hand definition with respect to subtree representation.

#### Definition 4.1.

For  $0 \leq k$  and a graph  $G$  with vertices  $v_1, v_2, \dots, v_n$  then  $xw(G) \leq k - 1$  iff there exists a subtree representation  $R = (T, \{T_1, T_2, \dots, T_n\})$  that is **edge** connected with respect to  $G$  and has **nodes** weight less or equal to  $k$ .

For convenience x-width is also defined with a cosmetic "minus one" since it will only be compared with tree-width. As stated in the open problem it will upper bound all the other parameters, which is why we compare it to the largest one first. It is easily proven that  $tw(G) \leq xw(G)$ , but we can do better with just small tweaks on any subtree representation that is node connected with respect to the graph  $G$  and also show that  $tw(G) \geq xw(G)$ .

**Claim 4.1.** For any graph  $G$   $tw(G) = xw(G)$ .

*Proof.* ( $tw(G) \leq xw(G)$ ) If  $xw(G) = k$  we have a subtree representation  $R$  of node weight  $k$  that is edge connected with respect to  $G$ . If two subtrees overlap at an edge, they also overlap at a node making  $R$  node connected with respect to  $G$ , i.e.  $tw(G) \leq k$ .

( $tw(G) \geq xw(G)$ ) Given a subtree representation  $R = (T, \{T_1, T_2, \dots, T_n\})$  that is node connected with respect to  $G$ , we want to tweak it into a subtree

representation  $R' = (T', \{T'_1, T'_2, \dots, T'_n\})$  that is edge connected with respect to  $G$ , without increasing node weight.

For every node  $\nu_i \in V(T)$  we add a new node  $\nu'_i$  and the edge  $\nu_i\nu'_i$ , do the same for all the nodes in each subgraph. with every subgraph for all the nodes it covers. Iteratively go though each node pair  $\nu_i\nu'_i$  in any order. If  $\nu_i$  has degree 4 delete any one of the edges  $\nu_i x \neq \nu_i\nu'_i$  and add the edge  $\nu'_i x$ . Do the same for every subgraph that covers  $\nu_i x$  and before moving onto the next node pair.

We have constructed no cycles and no disconnects so  $T'$  is still a tree. Same for the subtrees, they are also still non-trivial. No node in  $T'$  has degree more than 3 by construction, and since every pair of vertices forming an edge in  $G$  have corresponding subtrees overlapping at a node in  $T$ , there must be an edge that they overlap at in  $T'$ . The node weight of  $R'$  is the same as the node weight of  $R$  thus the claim is proven.  $\square$

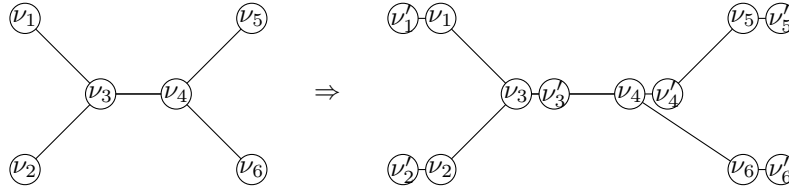


Figure 13

This means that x-width tells us nothing more than tree-width does about a graph, therefore needs no more studying as its own parameter.

## 5 Defining the tree-like parameters with chordal supergraphs

The subtree decomposition proved to be a good framework when comparing the three tree-like parameters. Maybe looking at another framework can yield additional information. In this section we will explore how branch-width and maximum matching-width looks using a framework based on a short and elegant definition of tree-width. It closely correlates with the subtree representation through the node intersection graph. Further we will give the full details for the proof of the maximum matching-width part of this framework, as it has only been outlined.

**Definition 5.1.** *Node intersection graph*

Given a graph  $G$  and a subtree representation  $R = (T, \{T_1, T_2, \dots, T_n\})$  of  $G$ , then  $NI(R) = H$  is the node intersection graph of  $R$  if  $V(H) = V(G)$  and  $v_i v_j \in E(H)$  iff  $V(T_i) \cap V(T_j) \neq \emptyset$ .

Relating to node intersection graphs there are two lemmas we will use. The first one is a key structural observation tying it together with the chordal supergraphs, and the second one is essentially the same as theorem 3.1 but for subtree representations.

**Lemma 5.1.** *The node intersection graph  $H$  of a subtree representation  $R = (T, \{T_1, T_2, \dots, T_n\})$  is chordal.*

*Proof.*  $H$  is chordal by contradiction. Assume  $H$  is not chordal, then  $H$  has some chordless cycle  $(x_1, x_2, x_3, \dots, x_m)$  of length at least 4.  $T_1$  must overlap  $T_2$  at some node  $\nu_p \in V(T)$  and  $T_2$  must overlap  $T_3$  at some node  $\nu_q \in V(T)$ . Since  $T_1$  and  $T_3$  cannot overlap at any node there is an edge  $e$  on the path from  $\nu_p$  to  $\nu_q$  that none of them overlap. If we remove  $e$  from  $T$  we get two components  $T_L$  and  $T_R$  with  $T_1$  and  $T_3$  contained in one each respectively. Observe that for any  $i > 3$ ,  $T_i$  cannot cover  $e$  since  $T_2$  must cover  $e$  by its connectivity, thus if  $T_{i-1}$  is contained in  $T_R$  then so must  $T_i$ . By induction this means that  $T_m$  is contained in  $T_R$  and cannot overlap with  $T_1$  and we have a contradiction.  $\square$

**Lemma 5.2.** *Consider a node intersection graph  $H$  of a subtree representation  $R = (T, \{T_1, T_2, \dots, T_n\})$ . For any maximal clique  $Q$  in  $H$  there exists a node  $\nu_i \in V(T)$  such that a vertex  $x_j$  is in  $Q$  iff  $T_j$  covers  $\nu_i$ .*

*Proof.* Let  $R = (T, \{T_1, T_2, \dots, T_n\})$  be a the subtree representation of a graph and  $H = NI(R)$ . For every node  $\nu_i \in T$  we define the bag  $X_i = \{x_j \mid x_j \in V(G); \nu_i \in V(T_j)\}$  i.e.  $x_j$  is in  $X_i$  if and only if  $T_j$  covers  $\nu_i$ .

Assume there is no  $X_i$  that contains all of  $V(Q)$ . Let  $D$  be a maximal subset of  $V(Q)$  that is contained in some  $X_i$  and let  $x_j$  be a vertex of  $V(Q)$  that is not in  $D$ . If  $T_D$  is the induced subgraph of the nodes in  $T$  that has bags containing all of  $D$ , then  $T_j$  does not intersect  $T_D$  by maximality of  $D$ . Since  $T_j$  is connected, it is contained in only one of the branches connected to  $T_D$ , and for this branch there must be at least one vertex  $x_l$  in  $D$  such that  $T_l$  does not intersect this

branch.  $x_j x_l$  is an edge in  $H$  and must have overlapping subtrees and we have a contradiction. Further, we know that  $Q$  can not be a strict subset of any bag, since it is a maximal clique in  $H$  and any bag induces a clique in  $H$ . This proves the lemma.  $\square$

The other important construct with chordal supergraphs is the relatively minimal separator. It will come into play with both branch-width and maximum matching-width. Note that the relatively minimal separator captures a larger set of separators than just the minimal separators.

**Definition 5.2.** *Relatively minimal separator*

*A separator  $S$  is relatively minimal if there exists a pair of vertices  $u$  and  $v$  separated by  $S$  such that no strict subset of  $S$  separates  $u$  and  $v$ .*

## 5.1 Tree-width

Let us now state the short tree-width definition before we move onto the other two parameters.

**Theorem 5.1.** *A graph  $G$  has  $tw(G) \leq k - 1$  iff  $G$  has a chordal supergraph  $H$  containing no cliques larger than  $k$ .*

To prove theorem 5.1 we can use theorem 3.2 together with lemma 5.1 and 5.2 in the forward direction. In the backward direction we use the fact that all chordal graphs have a clique decomposition from theorem 2 in [3] and as we discussed for any clique decomposition there is a tree decomposition of the same graph of maximum degree 3 with bags of size no larger than the bags in the clique decomposition. Now it only remains to trivially show that if  $G$  is a subgraph of  $H$  then any tree decomposition of  $H$  is a tree decomposition of  $G$  after removing the vertices that is not part of  $G$ .

One might hope that within the framework of theorem 5.1 branch-width and maximum matching-width will graciously fit in.

## 5.2 Branch-width

Since we know that branch-width behaves different than tree-width the supergraph bounding the branch-width must be different from the one bounding tree-width. A natural idea is to look at the edge intersection graph, i.e. the same as the node intersection graph but with the edges of the subtree representation of bounded edge weight. It turns out this will not give us a different set of graphs than if we were to look at the node intersection graphs, by lemma 1 in [8]. But we still need to capture the edge weight bound of the subtree representation, and one way is to just expand on theorem 5.1 by letting the bound be on a special division of the maximal cliques.

**Theorem 5.2.** *[7] A graph  $G$  and  $2 \leq k$  has  $bw(G) \leq k$  iff  $G$  has a chordal supergraph  $H$  where all maximal cliques  $X$  in  $H$  have three vertex subsets  $A$ ,  $B$  and  $C$  s.t.*

1.  $A \cup B = B \cup C = C \cup A = V(X)$ .
2.  $|A|, |B|, |C| \leq k$ .
3. All relatively minimal separator sets  $S$  of  $H$  contained in  $X$  must be contained in  $A$ ,  $B$  or  $C$ .

Theorem 5.2 is proven using the subtree representation given in theorem 3.3 [7]. The elephant in the room is that this theorem is not as short as 5.1, see figure , we will talk about this after the maximum matching part. We see that the lower bound of 2 on the width carries over from theorem 3.3 but it can be shown that theorem 5.2 is true for all graphs with more than two vertex.

### 5.3 Maximum matching-width

As with the subtree representation theorems we can get the maximum matching-width part of the chordal supergraph framework by just changing a small part of theorem 5.2, more specifically the first property.

**Theorem 5.3.** [5] *A non-trivial graph  $G$  has  $mmw(G) \leq k$  iff  $G$  has a chordal supergraph  $H$  where all maximal cliques  $X$  in  $H$  have three vertex subsets  $A$ ,  $B$  and  $C$  s.t.*

1.  $A \cup B \cup C = V(X)$ .
2.  $|A|, |B|, |C| \leq k$ .
3. All relatively minimal separator sets  $S$  of  $H$  contained in  $X$  must be contained in  $A$ ,  $B$  or  $C$ .

The proof of theorem 5.3 is only sketched in [5]. We will here give the full details of this proof, but first, to ease notation, we will introduce the following definition and some lemmas.

**Definition 5.3.** *k-tricover*

*(A, B, C) is a k-tricover of a set X if  $A \cup B \cup C = X$  and  $|A|, |B|, |C| \leq k$ . We say that the k-tricover respects a family of sets F if for any set in the family is contained entirely in A, B or C.*

The two following lemmas are well known properties of chordal graphs.

**Lemma 5.3.** *Given a chordal graph H, for any relative minimal separator S in H there exists two distinct maximal cliques  $Q_1$  and  $Q_2$  in H s.t.  $V(Q_1) \cap V(Q_2) = S$ .*

*Proof.* Let  $u$  and  $v$  be two vertices that  $S$  relatively minimally separates and  $G_1$  and  $G_2$  be the components with  $u$  and with  $v$  respectively after deleting  $S$  from  $G$ .

First we show that  $S$  is chordal by contradiction. Assume there are two vertices  $x$  and  $y$  in  $S$  that are not neighbours. We know that there is a path from  $x$  to  $u$  that does not use any other vertex from  $S$  than  $x$ , else  $S$  would not be

relativity minimal with respect to  $u$  and  $v$ . Same from  $y$ . Let  $P_1$  be the shortest path from  $x$  to  $y$  than only uses vertices from  $G_1$  between  $x$  and  $y$  and let  $P_2$  be the shortest path from  $y$  to  $x$  through  $G_2$ . Then the cycle cycling through  $P_1$ ,  $y$ ,  $P_2$  and  $x$ , in that order, will be a cycle of length at least 4 without chords, which is a contradiction to  $H$  being chordal. Thus  $S$  is chordal.

Second,  $G_1$  has at least one vertex that neighbours the whole of  $S$ . Assume it does not. Let  $a$  be a neighbour of  $S$  in  $G_1$  s.t. no other vertex in  $G_1$  has more neighbours with  $S$  than  $a$ . Since  $a$  does not neighbour every vertex in  $S$  let  $y$  be one such vertex and let  $d \neq a$  be a vertex in  $G_1$  neighbouring  $y$ , existing by relative minimality of  $S$ . If  $P$  is the shortest path from  $a$  to  $d$  in  $G_1$  let  $c$  be the first vertex in  $P$  that has  $y$  as a neighbour. The neighbourhood of  $c$  can not contain every vertex from the neighbourhood of  $a$  because it would be larger than the neighbourhood of  $a$ , so let  $x \in S$  be a neighbour of  $a$  but not  $c$ . If  $b$  is the last vertex in  $P$  before  $c$  that has  $x$  as a neighbour then with  $P'$  as the subpath of  $P$  from  $b$  to  $c$  we have that the cycle starting with  $P'$  and going through  $y$  then  $x$  has length at least 4 and has no chords, contradicting  $H$  being chordal. So there must be a vertex in  $G_1$  neighbouring all of  $S$ , the same argument hold for  $G_2$ .

Finally we have that since  $S$  is chordal and there exists at least one vertex  $u'$  in  $G_1$  neighbouring all of  $S$ , there must be a maximal clique containing  $S$  and  $u'$  that does not overlap with  $G_2$  since,  $S$  is a separator. Similarly we have a maximal clique containing  $S$  and at least one vertex  $v'$  from  $G_2$  that does not overlap  $G_1$ . This proves the lemma.  $\square$

**Lemma 5.4.** *The intersection of two neighbouring node sets in a clique decomposition of a chordal graph  $H$  is a relatively minimal separator of  $H$ .*

*Proof.* Let  $T$  be a clique decomposition of a chordal graph  $H$ . For every  $X_i X_j \in E(T)$  there is a  $u \in X_i, u \notin X_j$  and  $v \notin X_i, v \in X_j$  by maximality of the cliques corresponding with  $X_i$  and  $X_j$ . If  $X_i \cap X_j = S \neq \emptyset$  then  $S$  separates  $u$  and  $v$  by contradiction. Assume there is a path  $P$  from  $u$  to  $v$  that does not intersect  $S$ . Removing the edge  $X_i X_j$  from  $T$  gives us two trees  $T_1$  and  $T_2$  containing  $X_i$  and  $X_j$  respectively. Any vertex in  $P$  is not part of both  $X_i$  and  $X_j$ , since it does not intersect  $S$ . This means that the second vertex  $w$  in  $P$  must be entirely contained in  $T_1$  by the second property (connectivity) of clique decomposition and the fact that there must be at least one maximal clique containing both  $u$  and  $w$ . Induction on  $P$  shows that all vertices in  $P$  must be contained in  $T_1$ , contradicting the fact that  $v$  is in  $T_2$ . For any vertex  $z \in S$  we have the path  $(u, z, v)$ . This together with the fact that  $S$  separates  $u$  and  $v$  proves that  $S$  is a relatively minimal separator.  $\square$

We are now ready to prove the theorem.

*Proof.* (of theorem 5.3)

( $\Rightarrow$ ) In the forward direction we know from theorem 3.4 that for  $k \geq 1$  when we have a graph  $G$  with  $V(G) = \{x_1, x_2, \dots, x_n\}$  and  $mmw(G) \leq k$  there exist



a subtree representation  $R = (T, \{T_1, T_2, \dots, T_n\})$  that is node connected with respect to  $G$  with edge weight at most  $k$ . Let  $H$  be the node intersection graph of  $R$ . By lemma 5.1 we know that  $H$  is chordal. For every node  $\nu_i \in T$  we define the bag  $X_i = \{x_j \mid x_j \in V(G); \nu_i \in V(T_j)\}$  i.e.  $x_j$  is in  $X_i$  if and only if  $T_j$  covers  $\nu_i$ .

Now we only need to show that for the vertex set of any maximal clique  $Q$  in  $H$  we can find a  $k$ -tricover respecting all relatively minimal separators within  $Q$ . So given any maximal clique  $Q$  we know there exists a node  $\nu_i \in T$  with bag  $X_i = V(Q)$ , by lemma 5.2. Assuming  $\nu_i$  has degree 3 denote the three edges incident to  $\nu_i$   $e_A, e_B$  and  $e_C$ . Let  $A, B$  and  $C$  be subsets of  $V(H)$  s.t  $x_j$  is in  $A, B$  or  $C$  if  $T_j$  covers  $e_A, e_B$  or  $e_C$  respectively. If  $\nu_i$  has degree less than 3 we let some of the sets be empty accordingly.

$A \cup B \cup C = V(Q)$  follow from non-triviality of the subtrees.  $|A|, |B|, |C| \leq k$  follows directly from the edge weight  $R$ . From lemma 5.3 we know that every relatively minimal separator of a chordal graph is an intersection of two maximal cliques in the graph. Let  $S$  be a relatively minimal separator of  $H$  in  $Q$  and let  $\nu_j, \nu_l \in V(T)$  be two nodes with bags  $X_j$  and  $X_l$  such that  $V(X_j) \cap V(X_l) = S$ . Without loss of generality either  $\nu_i$  is equal to  $\nu_j$  or its unequal to both, and since all the subtrees corresponding to the vertices in  $S$  cover both  $\nu_i$  and  $\nu_l$ , they also cover the path between them, i.e. one of  $A, B$  or  $C$  contains all of them.

( $\Leftarrow$ ) All chordal graphs has a clique decomposition from theorem 2 in [3]. Let  $T$  be a clique decomposition of  $H$ , where  $H$  is the chordal supergraph of a graph  $G$  with  $V(G) = \{x_1, x_2, \dots, x_n\}$  and every maximal clique  $Q$  in  $H$  has a vertex set with a  $k$ -tricover respecting all relatively minimal separators of  $H$  contained in  $Q$ . We will construct a subtree representation that is node connected with respect to  $G$  of edge weight  $k$  using  $T, H$  and the  $k$ -tricovers. An example of this can be seen in figure 15.

$T$  has arbitrary maximum degree so we must construct a new decomposition tree for our subtree representation. Start with the empty  $R = (T', \{T_1, T_2, \dots, T_n\})$  where  $T'$  and all  $T_i$  are empty graphs. For every maximal clique  $Q$  in  $H$ , with  $X \in V(T)$  as a corresponding node in  $T$ , we add a (centre) node  $\nu$  to  $V(T')$  and if  $x_i \in X$  we let  $T_i$  cover  $\nu$ . Let  $(A, B, C)$  be a  $k$ -tricover of  $V(Q)$  respecting the relatively minimal separators of  $H$  contained in  $V(Q)$  and  $X_1, X_2, \dots, X_p$  be the neighbours of  $X$ . We know from lemma 5.4 that for all  $XX_i \in E(T)$  the set  $S_i = X \cap X_i$  is a relatively minimal separator. We use this fact and the  $k$ -tricover to partition  $\{1, 2, \dots, p\}$  into  $W_A, W_B, W_C$  such that if  $S_i \subseteq A$  then  $i \in W_A$  else if  $S_i \subseteq B$  then  $i \in W_B$  else if  $S_i \subseteq C$  then  $i \in W_C$ . Add three paths  $P_A, P_B$  and  $P_C$  to  $T'$ , of length  $|W_A|, |W_B|$  and  $|W_C|$  respectively, all staring at  $\nu$ . These we will call the branches connected to the centre node. We also extend the subtrees  $T_i$  to cover  $P_A$  if  $x_i \in A, P_B$  if  $x_i \in B$  and  $P_C$  if  $x_i \in C$ . For all  $\Theta \in \{A, B, C\}$  we add one (separator) node  $\nu_j$  to  $V(T')$  for every  $j \in W_\Theta$ , each uniquely neighbouring a node on the branch  $P_\Theta$ , and if  $x_i \in S_j$ , we let subtree  $T_i$  cover  $\nu_j$  along with its edge connecting it with  $P_\Theta$ . Finally for every two nodes  $X_a$  and  $X_b$  that are neighbours in the clique decomposition we need to "glue" the two part of  $T'$  they construct. We know their intersection is a

relatively minimal separator  $S$ , and for both of them a separator node has been added to  $T'$  because of  $S$ . Contract each such pair of nodes, altering the subtrees accordingly. The result of such a "gluing" can be seen in figure 15 where the white trees are "glued" to the grey tree at the rectangles.

The centre node for each maximal clique has degree at most 3. Same for the branch nodes, since we never add more than one neighbour to each path node. The separator nodes has degree two. The edge weight of each edge is at most  $k$  since subtrees only cover an edge in  $T'$  if it's in the part of the  $k$ -tricover that corresponds to the branch, and all parts in a  $k$ -tricover has bounded size  $k$ . For every edge in  $H$  there must exist a maximal clique containing the edge, so there must be a node in  $T'$  that is covered by both subtrees corresponding to the edge ends. The union of a  $k$ -tricover of a maximal clique is the whole clique vertex set, so every node must be in at least one of the  $k$ -tricover parts and the subtree of every node must go along one of the branches, making it nontrivial. If a subtree covers nodes in a branch, it also covers the centre node connected to the branch together with the path between. If it covers a separator node then we know that this separator node is neighbouring two branch nodes in different branches which the subtree also covers, including the two edges incident to the separator node, and by the above argument will cover the two centre nodes connected to the two branches. So all components in a subtree covers a centre node. If a subtree covers two distinct centre nodes we know its part of both corresponding maximal cliques, making it part of the relatively minimal separators corresponding to the separator nodes on the path between the two centre nodes. So the subtree must cover these nodes too and the whole path between the two centre nodes and thus the subtrees are connected.

This proves that  $R$  is a subtree representation that is node connected with respect to  $G$  of edge weight no more than  $k$  and since  $k \geq 1$  by 3.4 we get  $mmw(G) \leq k$ .  $\square$

The branch-width and the maximum matching-part of this framework did not turn out as elegant as the tree-width part, but between themselves they are almost identical. In maximum matching-width we require the union of all three subsets of clique vertex set to equal the whole set, while with branch-width we require them pairwise to be equal to the whole set. If we continue this trend and require every single subset to be equal the whole set, we actually arrive at tree-width, we can just write it up with fewer words since the other properties follows from the subsets being the entire clique.

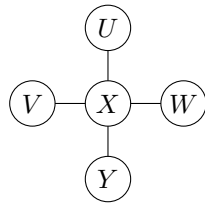


Figure 14: A clique decomposition.

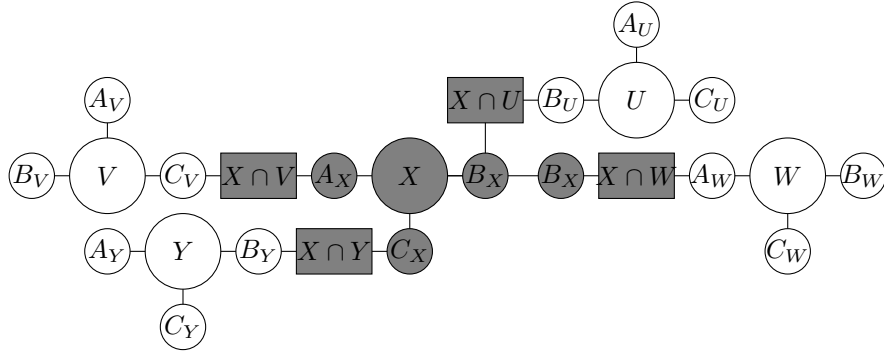


Figure 15: The resulting graph after taking the clique decomposition tree  $T$  from figure 14 making a subtree representation that is node connected with respect to original graph. The grey nodes are the nodes corresponding to the part constructed from node  $X$  in  $T$  and the rectangular nodes are the "glued" together separator nodes.

## 6 Path-like parameters

A closely related parameter to tree-width is path-width. In short it is a linearisation of tree-width i.e. instead of a decomposition tree it has a decomposition path. Similar linearisation can be done with any decomposition tree to get the path-like variant of it. For instance on the decomposition tree from branch-width and maximum matching-width. The path-like variants of these will be called linear branch-width and linear maximum matching-width. We will compare how they relate to each other, and in a later section explore there minimal forbidden minors for given bounds.

### 6.1 Path-width

We can get to a definition of path-width through definition 3.2 of tree decomposition by further restricting the decomposition tree to be a path. Then the path-width will be the width of this decomposition. Here path-width will be defined without a decomposition graph, since it will be superfluous.

**Definition 6.1.** *Path-width*

Given a graph  $G$ , a path decomposition is a sequence of bags  $X_1, X_2, \dots, X_m$  each a subset of  $V(G)$ , with these three properties:

1. The union of all the bags in the sequence is equal to  $V(G)$ .
2. For every edge  $e$  in  $G$  there must be a bag  $X_i$  in the sequence containing both ends of  $e$ .
3. For all three indices  $1 \leq i \leq j \leq l \leq m$  any  $x$  that is both in  $X_i$  and  $X_l$  must be in  $X_j$ .

The width of a path decomposition is equal to its largest bag minus 1, and the path-width of a graph  $G$  ( $pw(G)$ ) is equal to its smallest path decomposition.

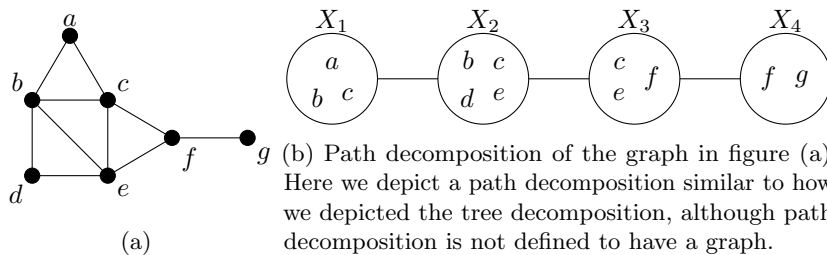


Figure 16

Since all path decompositions are linear tree decompositions, path-width will never be more than tree-width for a given graph. For our graph  $G$  in figure 16 the tree-width is 2, whereas the path-width of  $G$  is at least 3. This is because the

vertex set of a clique in a graph must be part of a bag in any tree decomposition by 3.1, and since the vertex sets of the 4 triangles on  $G$  cannot be ordered in any way without violating property 3 of path-decomposition at least one bag must contain a triangle and an additional vertex. The width of the path decomposition in figure 16 is 3 and we know it must be optimal and thus  $pw(G) = 2$

Path-width is quite handy since results relating to tree-width will often work with path-width as well, and in some cases we can even state stronger versions of the results. The drawback is that path-width can be vastly larger than tree-width, for instance on binary trees, where tree-width is always 1 but path-width is unbounded.

## 6.2 Linear branch-width

As with path-width we will omit the decomposition tree in the definition of linear branch-width. The alternative would have been to decompose it onto a caterpillar graph of maximum degree 3 and in most cases this would give the same width. The exceptions are, ironically enough, the caterpillar graphs that are not a star graphs, and then they will only differ by 1, but this is more a quirk with branch-width than with linear branch-width.

### Definition 6.2. Linear Branch-width

*Given a graph  $G$  with  $m$  edges, a linear branch decomposition is an ordering  $\sigma = (e_1, e_2, \dots, e_m)$  of the edges in  $G$ . For  $1 \leq i < m$  let  $cut(\sigma, i)$  be the (cut) set of vertices that are incident to at least one edge before  $e_{i+1}$  and at least one after  $e_i$  in  $\sigma$ ,  $cut(\sigma, m) = \emptyset$ . The width of a linear branch decomposition  $\sigma$  is the same as the size of its largest cut set, and the linear branch-width of a graph  $G$  ( $lbw(G)$ ) is the same as the width of its smallest linear branch decomposition with respect to its width.*

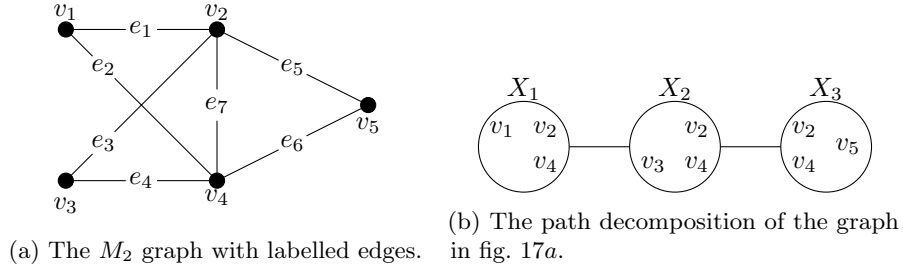
We can see an example of a linear branch decomposition.  $\sigma$  in figure 17c of the graph  $M_2$  in figure 17a. The width of  $\sigma$  is 3 and we will see that this is the best we can do with the graph  $M_2$  in section 6.4. A path decomposition of  $M_2$  is provided in 17b to compare. We can see that they are very similar, in fact path-width and linear branch-width of a graph are never more than one apart as we will prove.

## 6.3 Linear maximum matching-width

Following the same structure as with linear branch-width the path-like variant of maximum matching-width will also utilize orderings, but for linear maximum matching-width it makes no difference with respect to the width whether we use an ordering or a caterpillar graph of maximal degree three.

### Definition 6.3. Linear Maximum Matching-width

*Given a graph  $G$  with  $n$  vertices, a linear maximum matching decomposition is an ordering  $\omega = (v_1, v_2, \dots, v_n)$  of the vertices in  $G$ . For  $1 \leq i \leq n$  let  $B_{\omega, i}$  be a bipartite graph with one side consisting of  $v_i$  and all the vertices before  $v_i$  in  $\omega$*



$v_1, v_2$	$v_2, v_4$	$v_2, v_4$	$v_2, v_4$	$v_2, v_4$	$v_4, v_5$	$\emptyset$
$e_1,$	$e_2,$	$e_7,$	$e_3,$	$e_4,$	$e_5,$	$e_6$

(c) The linear branch decomposition of the graph in fig. 17a, with the cut sets depicted over each element in the ordering.

Figure 17

and the other side the rest of the vertices. So the edges of  $B_{\omega,i}$  are all the edges from  $G$  with one end on each side of  $B_{\omega,i}$ . The width of  $\omega$  is equal to the largest maximum matching over all bipartite graphs  $B_{\omega,i}$  for  $1 \leq i \leq n$ , and the linear maximum matching-width of a graph  $G$  ( $lmmw(G)$ ) is equal to its smallest linear maximum matching decomposition with respect to its width.

Figure 18 shows a linear maximum matching decomposition  $\omega$  with the bipartite subgraphs of our trusted graph  $G$  in the same figure. Its width is 2 and since  $G$  has maximum matching width 2 the linear maximum matching-width of  $G$  cannot be lower.  $G$  actually contains all the tree graphs that minimally upper bounds graphs with linear maximum matching-width less than 2, this will be explored further in section 7.4.

## 6.4 Comparing the different path-like parameters

The constant factor relation between the three tree-like parameters is a key point of interest since it highlights their relation to each other. So it is natural to ask what the quantitative relation between the different path-like parameters are. We will start with the relation between linear branch-width and path-width as they are the closest. As we have stated earlier for many graphs path-width is very close to linear branch-width, in fact:

**Theorem 6.1.** *If graph  $G$  has at least two edges,  $pw(G) \leq lbw(G) \leq pw(G) + 1$ .*

*Proof.* ( $pw(G) \leq lbw(G)$ )

If  $G$  has  $lbw(G) = k$ , let  $\sigma = \{e_1, e_2, \dots, e_m\}$  be a linear branch decomposition of  $G$  with width  $k$ . We will construct a path decomposition  $X_1, X_2, \dots, X_m$  of  $G$ . Bag  $X_i$  consists of the two incident vertices of  $e_i$  and every vertex in  $cut(\sigma, i)$ .

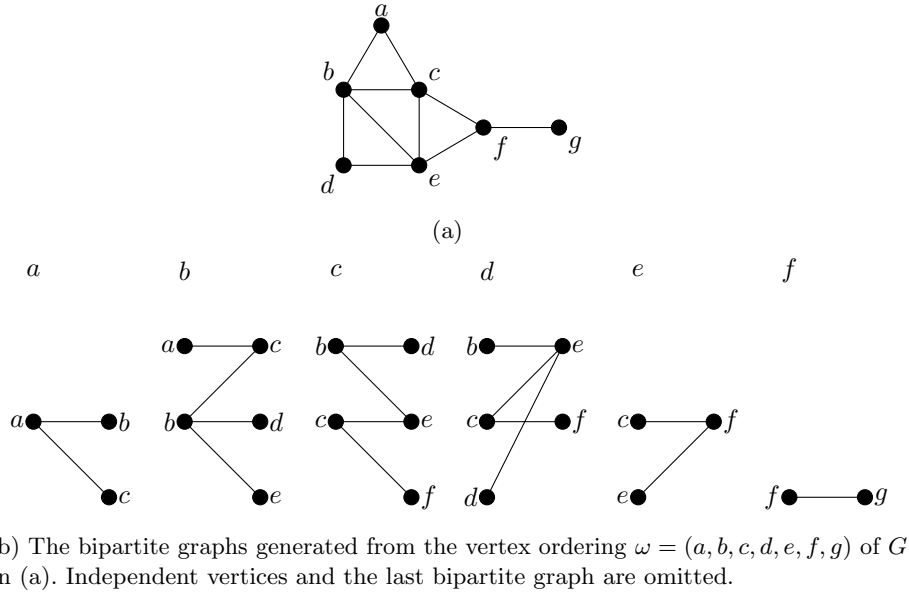


Figure 18

Note that for each  $i \in \{1, 2, \dots, m\}$ ,  $\text{cut}(\sigma, i)$  must contain at least one of the ends in  $e_i$ . If not there would be an edge  $e_i$  with no incident edges and our graph would be disconnected or just a single edge. So the size of the bags does not exceed  $k + 1$ .

Now to show that this is indeed a path decomposition. By construction property 1 and 2 of path decomposition is satisfied. If a vertex  $x \in V(G)$  is in  $X_i$  and in  $X_l$  then  $x$  must be incident to at least one edge  $e_p$ ,  $p \leq i$  and one edge  $e_q$ ,  $q \geq l$  then it must also be in any  $X_j$  whenever  $i < j < l$ , since  $p < j < q$ , so property 3 is also satisfied. Thus we have a path decomposition with bags of size at most  $k + 1$ , i.e.  $pw(G) \leq k = lbw(G)$ .

$$(lbw(G) \leq pw(G) + 1)$$

Given a path decomposition of  $G$  of width  $k$  we will construct a linear branch decomposition  $\sigma$  of width at most  $k + 1$ . Let  $X_1, X_2, \dots, X_t$  be the consecutive bags of any path decomposition of  $G$  of width  $k$  that is altered so that no bags is a another bag. As we have discussed, this will still be a valid path decomposition of the same graph and now each bag has at least one vertex that is not in the next bag, and at least one that is not in the previous bag. Start out with an empty  $\sigma$  then iteratively we go through the bags. If bag  $X_i$  has a pair of vertices that form an edge in  $G$  and has not yet been added to  $\sigma$ , add it at the end. Move on to the next bag when  $X_i$  has no more such pairs. In particular, in each bag if the bag has a vertex  $v$  that is not in any of the previous bags and different vertex  $u$  that is not in any of the later bags, start by checking all the vertex pairs that  $u$

is part of. Since every edge of  $G$  appears in at least one bag,  $\sigma$  will be a linear branch decomposition.

Given  $e_a \in E(G)$  let  $X_j$  be the bag that added  $e_a$  to  $\sigma$ . Then every  $x \in \text{cut}(\sigma, a)$  must be in  $X_j$ . This is because  $x$  is part of  $\text{cut}(\sigma, a)$  only when there exists a  $p$  and a  $q$  with  $p \leq a \leq q$  such that  $x$  is incident to both  $e_p$  and  $e_q$ . Let  $X_i$  and  $X_l$  be the bags that added  $e_p$  and  $e_q$  respectively. Since  $x$  is in both  $X_i$  and  $X_l$  and we know that  $i \leq j \leq l$  then by property 3 of path decomposition  $X_j$  must contain  $x$ . If  $X_j$  has size less than  $k + 1$  we are good. Further if  $X_j$  has a vertex  $v$  that is not in any of the previous bags and different vertex  $u$  that is not in any of the later bags,  $\text{cut}(\sigma, a)$  will only have one of the two vertices  $u$  and  $v$ , since all edges using  $u$  comes before all the edge using  $v$  in  $\sigma$ , thus the cut has size no more than  $k$ , one less than the largest bags. The only path decompositions we can not construct a linear branch decomposition where the size of every cut set is upper bounded by  $k$  are the ones with tree consecutive bags all of size  $k + 1$  and the intersection of them is a set of size  $k$ . In other words, all three are of size  $k + 1$  and all three have just have one vertex that the other two does not have. Then the unique vertex in the middle bag will be both the vertex that has its first appearance and the vertex that has its last appearance. If every path decomposition of width  $k$  has three such consecutive bags the cut sets of constructed linear branch decomposition will be upper bounded by  $k + 1$ . If not we can use the path decomposition without these three consecutive bags and bound the cut sets by  $k$ , since each bag has two distinct vertices one making its last appearance and one making its first. In the end this limits the size of the  $\sigma$  cuts to be no more than  $k + 1$ , i.e.  $lbw(G) \leq k + 1 = pw(G) + 1$ .  $\square$

This result is tight. Let us prove this by exhibiting  $K_n$  and  $M_n$ . For the complete graphs  $K_n$  path-width will be at least  $n - 1$  since  $tw(K_n) \geq n - 1$  by theorem 3.1, and the trivial path decomposition has width  $n - 1$ . The linear branch-width of  $K_n$  will also be  $n - 1$  with the optimal linear branch decomposition starting with all the incident edge of one vertex in  $K_n$ . It is optimal since path-width lower bounds linear branch-width by theorem 6.1. For the other bound let  $M_n$  be the graph with an  $n$  clique  $Q$  and three mutually independent vertices  $x_1, x_2$  and  $x_3$  all with neighbourhood  $V(Q)$ . The path decomposition is just three bags, all containing  $V(Q)$  and one of  $x_1, x_2$  and  $x_3$  distributively. Since this is also an optimal tree decomposition by theorem 3.1 we have  $pw(M_n) = n$ . For  $2 \leq n$  a linear branch decomposition of  $M_n$  of width  $n + 1$  is constructed by taking all the incident edges of  $x_1$  first, then all the incident edges of  $x_2$  and the remaining edges in any order of your liking. To prove its optimality let  $\sigma$  be any linear branch decomposition of  $M_n$ , with  $2 \leq n$ , and let  $i$  be the lowest  $i$  such that the size of  $\text{cut}(\sigma, i)$  is at least  $n$ , which always exists by theorem 6.1. Observe that since all vertices in  $M_n$  has degree at least  $n$   $\text{cut}(\sigma, i)$  must be a superset of all sets  $\text{cut}(\sigma, j)$  for  $j < i$ . If  $\text{cut}(\sigma, i) = V(Q)$  then the first  $i$  elements in  $\sigma$  are all within  $Q$ . Let element  $j$  in  $\sigma$  be the first element after  $i$  incident to  $x_1, x_2$  or  $x_3$ , then  $|\text{cut}(\sigma, j)| = n + 1$ . If  $\text{cut}(\sigma, i) \neq V(Q)$  then there exists an  $v \in V(Q)$  whose incident edges all appear after edge  $i$  in  $\sigma$ , since  $\text{cut}(\sigma, i)$  is a superset of all previous cut sets. Let element  $j$  in  $\sigma$  be the first



edge incident to  $v$ . Then  $cut(\sigma, j)$  will be of size  $n + 1$  unless, without loss of generality, all the incident edges of  $x_1$  appear before element  $j + 1$  in  $\sigma$ , but then  $cut(\sigma, j) = V(Q)$  and we can use the similar argument as in the previous case. In conclusion  $K_n$  and  $M_n$ , for  $2 \leq n$ , proves the tightness of theorem 6.1.

It can also be shown that for a given graph  $G$

**Theorem 6.2.** *Given a graph  $G$  with at last one edge then  $lmmw(G) \leq pw(G) \leq 2 * lmmw(G)$ .*

*Proof.* ( $lmmw(G) \leq pw(G)$ )

Given a graph  $G$  with a path decomposition  $X_1, X_2, \dots, X_m$  of width  $k$  we can construct a linear maximum matching decomposition  $\omega$  of width at most  $k$ . Let  $X'_1, X'_2 \dots X'_n$  be the path decomposition after removing the bags in  $X_1, X_2, \dots, X_m$  that is a subset of any other bag in  $X_1, X_2, \dots, X_m$  and preserving the bag ordering. This will not violate any of the path decomposition criteria, and note that every bag has at least one vertex that is not part of the next bag. The vertex ordering of  $\omega$  will be constructed by iterating through the bags  $X'_1, X'_2 \dots X'_n$  and greedily adding all the new vertices from each bag, first the ones that are not part of the next bag, then the rest. Now, for any given  $v_i \in \omega$  we need to show that  $B_{\omega, i}$  has maximum matching no more than  $k$ . Let  $X'_j$  be the bag that first contains  $v_i$ . By construction there is a vertex on the same side as  $v_i$  in  $B_{\omega, i}$  from  $X'_j$  that can only have neighbours within  $X'_j$ , since all its neighbours are in the bags before and in  $X'_j$  and these vertices has priority. Further, any edge in  $B_{\omega, i}$  has at least one end in  $X'_j$  because of property 3 of path decomposition. This means that since the size of  $X'_j$  is at most  $k + 1$  a maximum matching in  $B_{\omega, i}$  of size larger than  $k$  would have to use every vertex in  $X'_j$  where each matching edge used only one vertex from  $X'_j$  each. But this cannot be the case since there is at least one vertex with neighbours only within  $X'_j$ . And thus  $\omega$  can not have a width of no more than  $k$ .

( $pw(G) \leq 2 * lmmw(G)$ )

If a graph  $G$  has linear maximum matching-width  $k$ , then it has a linear maximum matching decomposition  $\omega = (x_1, x_2, \dots, x_n)$  of width  $k$ . From  $\omega$  we can easily create a subtree representation  $R = (T, \{T_1, T_2, \dots, T_n\})$  that is node connected with respect to  $G$  with edge width  $k$ . Start with a path  $P_n = (\nu_1, \nu_2, \dots, \nu_n)$  and add a leaf  $l_i$  to each node  $\nu_i$  to get  $T$ . Now a maximum matching decomposition  $(T, \partial)$  of  $G$ , where  $\partial$  is the bijection between each vertex  $x_i$  in  $\omega$  and leaf  $l_i$  from  $T$ . The bipartite subgraphs induced by  $(T, \partial)$  will be the same ones as in  $\omega$  with de addition of the ones induced by the leaf edges, but these will only have maximum matching of one, and since we are dealing with graphs with at most one edge it will not impact the width.

When we have a maximum matching decomposition we know from theorem 3.4 that there exists a subtree representation  $R = (T, \{T_1, T_2, \dots, T_n\})$  that is node connected with respect to  $G$  with edge width  $k$  of  $G$ . To get such a subtree representation we turn to the proof of theorem 3.4 from [5]. It is a constructive proof that uses the decomposition tree from a maximum matching decomposition as the decomposition tree in the subtree representation, also the leaf edges are

only covered by the subtree corresponding with the vertex paired with this leaf node from the maximum matching decomposition. For all non-leaf nodes  $\nu_i \in T$  let  $X_i$  be a bag containing all the vertices that corresponds to a subtree that covers  $\nu_i$ . Now  $X_1, X_2, \dots, X_n$  will be a path decomposition, since all path decomposition properties follow from the definition of a subtree representations that is node connected with respect to the graph in question. Since  $R$  has edge width  $k$ , any bag  $X_i$  has size at most  $2k + 1$ . The plus 1 comes from the leaf. This means that  $pw(G) \leq 2k = 2 * lmmw(G)$  and we are done. □

This inequality is tight on the complete graphs.

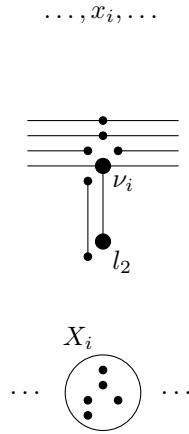


Figure 19: A three part figure demonstrating the transitions in the proof 6.4

## 7 Minimal forbidden minors

We know from definition that trees have no cycles in them, in short: a graph is a tree if it does not have a cycle of length three as a minor. Minors is another construct relating bigger graphs with smaller graphs that encapsulates subgraphs and edge contraction. With respect to forbidding certain graphs as minors the triangle graph upper bounds trees, we call it a forbidden minor. A graph class can have multiple forbidden minors, for instance the planar graphs, which has the complete graph  $K_5$  and the complete bipartite graph  $K_{3,3}$  as forbidden minors by Kuratowski's theorem. Together they are the minimal forbidden minors (MFM), since neither one of them is a minor of the other. These two minor upper bounds work because a minor of a tree and a planar graph is still a tree and a planar graph, tree-width and planarity are closed under minors. We say that a graph property is closed under minors if for a graph  $G$  that has this property, all its minors has the property.

One of the bigger result in graph theory is the graph minor theorem by Neil Robertson and Paul D. Seymour. The theorem states: "For infinite set of finite graphs, one of its members is isomorphic to a minor of another" [10]. The proof is derived at through 20 papers published over the span of 21 years from 1983 to 2004. What the graph minor theorem tells us is that for a graph class that is closed under minors there is a finite set of minimal forbidden minors, since any infinite set of forbidden minors will not be minimal. We will in this section show that linear branch-width and linear maximum matching-width is closed under minors and look into the minimal forbidden minors for different values of the tree-like and path-like parameters

### 7.1 MFM for tree-like parameters

For completeness sake, we will give the basic definitions of new terminology in this section then move onto the minimal forbidden minors of the tree-like parameters.

**Definition 7.1.** *Minor*

*A graph  $H$  is a minor of a graph  $G$  if you can get a graph that is isomorphic to  $H$  from  $G$  with the three minor operators vertex deletion, edge deletion and edge contraction.*

Note that a minor might not be connected, but we will not encounter them here. The figure 20 shows an example of a graph  $G$  and one of its minors  $H$ .

**Definition 7.2.** *A graph property is closed under minors if for any graph  $G$  with this property all its minors has the same property.*

We say that a graph parameter is closed under minors if all minors of any graph  $G$  has less or equal value in the parameter than  $G$ . Tree-width is closed under minors, as well as branch-width and maximum matching-width. We can see this by looking at each minor operator on a graph  $G$  through how we might alter a subtree representation that is node connected or edge connected with

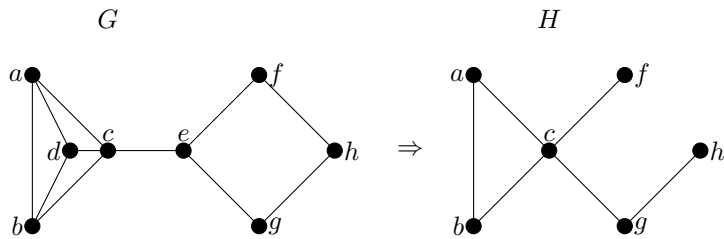


Figure 20: Here  $H$  is a minor of  $G$  since we can get  $H$  by removing vertex  $d$  and edge  $fh$  and contracting edge  $ce$ .

respect to  $G$  to become a fitting subtree representation of the resulting graph. With edge deletion no alteration is needed, with vertex deletion we remove the subtree corresponding to the deleted vertex and edge contraction just removes the two subtrees corresponding with the ends of the contracted edge and adds a new subtree that covers the union of the two deleted subtrees. Note that with edge contraction the two deleted subtrees overlap, so our new subtree is connected. Neither one of node weight or edge weight in any of the nodes or edges will increase under any of the minor operations. Thus tree-width, branch-width and maximum matching-width is minor closed.

**Definition 7.3.** *Minimal Forbidden Minors*

*Given a minor closed graph class  $A$  then a family of graphs  $F$  is a set of forbidden minors of  $A$  if every graph that has a minor in  $F$  is not part of  $A$ , and every graph that does not have a minor in  $F$  is part of  $A$ .  $F$  is minimal if none of the graphs in  $F$  is a minor of any other graph in  $F$ .*

For every minor closed graph classes the set of all the graphs not in the graph class is a set of forbidden minors. This set, however, tends to be infinite. Luckily for us the graph minor theorem tells us that in any infinite set of finite graphs there is at least one graph that is superfluous with respect to a set of forbidden minors set by transitivity of minors. In fact through the graph minor theorem we can deduce that any minor closed graph class has a unique and finite set of minimal forbidden minors. If  $A$  is a minor closed graph class and  $F$  is the set of all the graphs not  $A$  then let  $M$  be the set of graphs in  $M$  that has no other minors within  $M$  than itself. Note that  $M$  is not empty, it has to contain all the graph with the least amount of edges among the graphs with the least amount of vertices. Any graph  $G$  of  $F$  not in  $M$  has a minor in  $M$ , if not all its minors would need to have a minor, including the ones with the least amount of edges of the ones with the least amount of nodes, which can not be the case.  $M$  has to be finite, if not it would contradict the graph minor theorem, and is a unique set of minimal forbidden minors by construction.









	tw	bw	mmw
0			
1			
2			42 graphs [4]
3	4 graphs [1]	4 graphs [2]	?

Figure 21

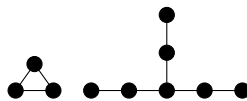
## 7.2 MFM path-width

Since path-width is a restricted form of tree-width it is easy to see that path width is closed under minors by the same argument as with all the tree-like parameters.

The minimal forbidden minors of graphs with path-width less than 1 is the single edge graph, since the only graph of path-width less than one is the empty graph and the trivial graph.



For graph with path-width less than 2 there two minimal forbidden minors, the complete  $K_3$  graph and the  $S_{3,2}$  graph which is a graph with a central vertex with three paths leading out from it of length 2. The graphs without  $K_3$  and  $S_{3,2}$  as a minor are the caterpillar graphs, trees where if we were to remove all degree one vertices we would end up with a path. A path decomposition of a caterpillar graph is constructed by iteratively going through the underlying path. Start with all the leaf edge pairs at the beginning of the path, then the first edge pair in the underlying path, then the leaf edge pairs of the second vertex in the underlying path, then the second edge pair in the underlying path, and so on. It is trivial to see that this decomposition has width 1. Theorem 3.1 is why  $pw(K_3) = 2$ . To see why  $pw(S_{3,2}) = 2$  note that for any path decomposition there is a left most bag  $X_L$  containing an edge pair  $ab$  and a right most bag  $X_R$  containing an edge pair  $cd$  which leaves us with at least one path  $(v_1, v_2, v_3)$ , where  $v_1$  is the centre vertex and  $a, b, c$  and  $d$  is neither  $v_2$  nor  $v_3$ . All bags between  $X_L$  and  $X_R$  has to contain at least one vertex from remaining path that does not include  $v_2$  and  $v_3$ , since every bag containing a given vertex is contiguous and every edge pair in the path has a bag in common. This together with the fact that the bag  $X_i$  containing the edge pair  $v_2v_3$  is some where between  $X_L$  and  $X_R$  means that there has to be a bag containing three vertices. A simple decomposition of  $S_{3,2}$  with width 2 is just having three bags each containing one of the three paths that starts at the centre vertex in any order.



When it comes to the minimal forbidden minors of graphs with path-width less than 3 the amount of forbidden minors drastically increases. A computer aided proof found all the 110 minimal forbidden minors through the gate matrix layout problem [6], since the two problems are equivalent in the sense that for every graph with path-width  $k$  we can create a gate matrix layout instance of value  $k + 1$  and vice versa. At the end in the same paper they cite that the minimal forbidden minors of graphs with path-width at most 4 is at least 122 million graphs, and that as the parameter grows the size of the minimal forbidden minors will not decrease. Although 110 elements seem a little much, it pales in comparison with 122 million.


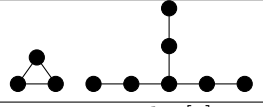
	pw
At most 0	
At most 1	
At most 2	110 graphs [6]

Figure 22: A table showing the different minimal forbidden minors for given path-width upper bounds

### 7.3 MFM linear branch-width

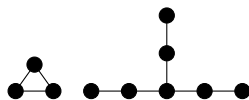
Since  $pw(G) \leq lbw(G) \leq 1 + pw(G)$  for every graph with more than two edges we would expect the minimal forbidden minors of bounded linear branch-width to be similar to the minimal forbidden minors of path-width with the same bound. We of course need to prove that linear branch-width is closed under minors. To prove that linear branch-width is closed under minors we again use the same argument as with the subtree representations. For vertex deletion and edge deletion the new edge ordering is the ordering after removing the vertices and edges in question from any linear branch decomposition of the original graph. Edge contraction removes the contracted edge and relabels every instance of the two ends to one label, for instance the lexicographical first one. If linear branch-width were to increase after an edge contraction on an edge  $ab$  there must be an  $i$  such that the cut set at position  $i$  of the new edge ordering would be of size one larger than the corresponding cut set  $j$  in the old edge ordering. The only "new" element  $a$  is the relabelled  $b$ . The cut set  $j$  in the old ordering can not have an  $a$  or a  $b$  in both sides, since we assumed the cut set  $i$  to have a "new" element. So there must be an  $a$  on one side and a  $b$  on the other, but since the edge  $ab$  is on one of the sides the cut set  $j$  in the old ordering accounts for the "new" element with either  $a$  or  $b$ . Thus linear branch-width can not increase with minor operations.

Since the single edge graph is the only graph where linear branch-width is less than path-width the minimal forbidden minors of graphs with linear branch-width less than 1, which consists of the graph  $P_2$ , is the only instance

where the minimal forbidden minors of linear branch-width is not bounded by the minimal forbidden minors for path-width of similar bound.



For graph with linear branch-width less than 2 the minimal forbidden minors is the same as with minimal forbidden minors for the graph of path-width less than 2,  $K_3$  and  $S_{3,2}$ . Both of them has linear branch-width at least 2 since they both have path-width 2. The remaining graph are, again, the caterpillar graphs. The linear branch decomposition of a caterpillar is the same as the one for path decomposition if we interpret the bags of two vertices as an edge, the contiguous nature of path width ensures us that the size of the cut sets can not exceed 2.



We know that the minimal forbidden minors of graphs with path-width less than 3 is a set of 110 graphs [6]. Looking through them we can find out that not all of them are minimal forbidden minors of graphs with linear branch-width less than 3, since some of them has minors with linear branch-width 3. Groups of them even have the same minor with linear branch-width 3. For instance the 3, 4, 5, 6 graph list at the end of the paper all have the  $M_2$  graph as a minor. A very rough count of these groups gives us 68 groups where each has a unique minor with linear branch-width 3 that only has minors of linear branch-width less than 3. In other words a rough estimate on the amount of minimal forbidden minors of graphs with linear branch-width less than 3 is roughly at least 68. It is not clear that all minimal forbidden minors of graphs with linear branch-width at most 3 is a minor of a graph that is a minimal forbidden minors of graphs with path-width less than 3. If this were the case, a quick brute force program could go through the 110 graph and check for graphs with minors with linear branch-width 3 and list the minimal such minors for each graph if not already listed.

	pw	lbw
At most 0		
At most 1		
At most 2	110 graphs [6]	Roughly at least 68 graphs

Figure 23: A table showing the different minimal forbidden minors for given path-width and linear branch-width upper bounds

## 7.4 MFM linear maximum matching-width

We first prove that linear maximum matching-width is closed under minors.

**Theorem 7.1.** *For any  $k$  the set of graphs that has linear maximum matching-width at most  $k$  is minor closed.*

*Proof.* Let  $G$  be a graph with a linear maximum matching decomposition  $\omega$  of width  $k$ . If we delete edges and vertices from  $G$ , then a trimmed  $\omega$  will still be a linear maximum matching decomposition of the resulting graph. The trimming is just removing any vertices from  $\omega$  that is deleted in  $G$ . The only thing that changes is that some of the bipartite graphs from the edge cuts of  $\omega$  lose some of their edges and/or vertices, but the width will not increase.

If we contract edge  $v_i v_j \in E(G)$  then removing  $v_j$  from  $\omega$  will give a linear maximum matching decomposition  $\omega'$  of the resulting graph. Let  $B$  be a bipartite graph from a cut in  $\omega$  and  $B'$  the resulting graph after contracting  $v_i$  and  $v_j$  in  $B$ . Any maximum matching set  $M'$  in  $B'$  has a corresponding maximum matching set  $M$  in  $B$ . The only difference between  $M$  and  $M'$  is whenever  $v_i v_l \in M'$  and  $v_i v_l \notin E(B)$ . If  $v_i$  and  $v_j$  are on the same side in  $B$  it must be the case that  $v_j v_l \in E(B)$ . So we just substitute  $v_i v_l$  with  $v_j v_l$  in  $M'$  to get  $M$ . If  $v_i v_j$  is an edge in  $B$  we just substitute  $v_i v_l$  with  $v_i v_j$  in  $M'$  to get  $M$ . In other words, the maximum matching of  $B$  will never be less than the maximum matching of  $B'$ . Since any cut set in  $\omega'$  will be an edge contracted version of some cut set in  $\omega$  we know that the width of  $\omega'$  can not be more than the width of  $\omega$ .

Since every minor operator never increases linear maximum matching-width it proves the theorem.  $\square$

The only graphs with linear maximum matching-width less than 1 is the empty graph and the trivial graph so the minimal forbidden minors is merely the single edge graph.



The minimal forbidden minors of graphs with linear maximum matching-width less than 2 share some similarities with the two minimal forbidden minors of graphs with path-width less than 2. It includes the  $S_{3,2}$  but instead of the  $C_3$  cycle ( $K_3$ ) it contains the  $C_4$  cycle with the addition of the graph having a cycle of length three and a degree one neighbour for all three vertices in the cycle. Figure 24 depicts these three graphs.

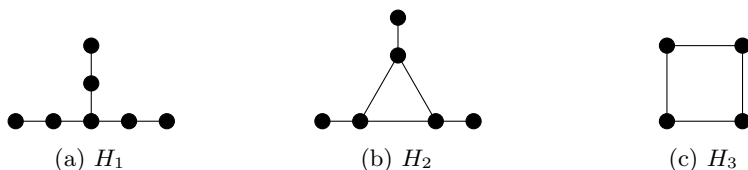


Figure 24: MFM of graphs with linear maximum matching-width less than 2



Before we move onto the proof that these are indeed the minimal forbidden minors we first prove a small lemma regarding minor relations. This will help us assessing whether two graphs does not have a minor relation.

**Lemma 7.1.** *Given a graph  $G$  and  $G'$  where  $G'$  is a minor of  $G$ , then if  $G$  has no cycles then  $G'$  has no cycles. Further, if the longest cycle of  $G$  has length  $l$  then the longest cycle in  $G'$  can not be longer than  $l$ .*

*Proof.* Its trivial to see that this is the case when we use edge and vertex deletion. Let  $G'$  be the graph after contracting a edge  $v_i v_j$  in  $G$  and  $C_l = (v_1, v_2, \dots, v_l)$  a cycle in  $G'$  of length  $l$ . If all the edges in  $C_l$  are in  $G$  we are fine. If not then  $v_i$  is on  $C_l$  and at least one of  $v_i v_{i-1}$  or  $v_i v_{i+1}$  is not an edge in  $G$ . If both are not in  $E(G)$  then replacing  $v_i$  with  $v_j$  in  $C_l$  will yield a cycle in  $G$  of length  $l$ . If only one of them is not part of  $E(G)$  then adding  $v_j$  right before or right after  $v_i$  in  $C_l$  will yield a cycle in  $G$  of length  $l + 1$ .

Since its true for all the minor operators, it will be true for any combination of these.  $\square$

Now for the proof.

**Theorem 7.2.**  *$H_1, H_2$  and  $H_3$  from figure 24 are the minimal forbidden minors of graphs with linear maximum matching-width less than 2.*

*Proof.* Because of their symmetries it's quick to check that the three graphs have linear maximum matching-width 2.

For  $H_1$  we see that if we want an linear maximum matching decomposition  $\omega$  to have width 1 a neighbour pair where one of them is a leaf has to be the two first elements. Then, for position three, the only option is the central vertex. Then any choice for position four will result  $\omega$  having width at least 2. In fact any ordering as described above will have width 2.

In  $H_2$ 's case observe that for any linear maximum matching decomposition  $\omega$  the bipartite graph  $B_{\omega,3}$  will have a maximum matching of size at least 2. An linear maximum matching decomposition  $\omega$  of the vertices of width 2 is first taking two degree one vertices then their two neighbours, then the rest.

For  $H_3$  observe that there are only two unique ways to start a linear maximum matching decomposition  $\omega$ . Either with two neighbours or with two non-neighbours. In both cases the bipartite graph  $B_{\omega,2}$  has maximum matching of size two. For all other bipartite subgraphs of  $H_3$ , the maximum matching is trivially either one or zero.

$H_1$  is not a minor of  $H_2$  or  $H_3$  because it has more vertices than both of them. Same for  $H_2$  not being a minor of  $H_3$ . By lemma 7.1  $H_3$  and  $H_2$  can not be a minor of  $H_1$ . Same for  $H_3$  not being a minor of  $H_2$ .

To prove that all graphs without  $H_3, H_1$  and  $H_2$  as a minor has linear maximum matching-width less than 2 we will describe them generally. Let  $A$  be the family of all graph without  $H_1, H_2$  and  $H_3$  as minors. Let  $G$  be a graph in  $A$  and let  $P = (x_1, x_2, \dots, x_m)$  be a longest path in  $G$ .

Except for the edges in the path  $P$   $G$  can only have two types of edges. The first type are the edges that are incident only to vertices in  $P$ . We will call these

shortcut edges. Let  $x_i x_j$  be such an edge. Without loss of generality let  $i < j$ . If  $j \neq i + 2$ ,  $G$  will have a cycle  $x_i, x_{i+1}, \dots, x_j$  of length at least 4, and  $G$  will have  $H_3$  as a minor. Thus all shortcut edges will be of the form  $x_i x_{i+2}$ . Further  $G$  can not have a shortcut edge  $x_i x_{i+2}$  if  $x_{i+1}$  has a neighbour  $v$  outside  $P$ . If it had such an edge and  $i = 1$  it would mean that  $(v, x_2, x_1, x_3, x_4, \dots, x_m)$  would be longer than  $P$ . An analogous arguments holds when  $i + 2 = m$ . For the case where  $1 < i < m - 2$ ,  $\{x_{i-2}, x_{i-1}, x_i, x_{i+1}, x_{i+2}, v\}$  would induce  $H_2$  in  $G$ , or at least a graph with  $H_2$  as a subgraph. Lastly two shortcut edges can not cross, in the sense that if  $x_i x_{i+2} \in E(G)$  then  $x_{i+1} x_{i+3} \notin E(G)$  since  $(x_i, x_{i+2}, x_{i+3}, x_{i+1})$  would then make a 4 cycle.

The other type of edge are edges that have one end in  $P$  the other outside  $P$ . Let  $x_i v$  be such an edge on  $G$ . Then we already know that  $x_{i-1} x_{i+1}$  cant be an edge in  $G$ . Further,  $1 \neq i \neq m$  since  $P$  is a longest path. Also if  $2 \leq i \leq m - 1$ ,  $v$  can not have a neighbour  $u$  except  $x_i$ . Assume  $v x_j \in E(G)$  for any  $x_j$  in  $P$   $x_j \neq x_i$ . This would contradict  $P$  being a longest path when  $j = i - 1$  or  $j = i + 1$ , and else would cause  $G$  to have a cycle of length at least four. Since  $u$  can not be part of  $P$  assume now  $uv \in E(G)$  with  $u$  not in  $P$ . For  $i = 2$  or  $i = m - 1$  it would violate  $P$  being a longest path, and for  $2 < i < m - 1$   $\{x_{i-2}, x_{i-1}, x_i, v, u, x_{i+1}, x_{i+2}\}$  would induce a graph with  $H_1$  as a subgraph in  $G$ . In short: any vertex of  $G$  not in  $P$  must have degree one, i.e. leaves.

Note that since  $G$  is connected and all vertices not in  $P$  has degree 1,  $G$  can not have any other vertices than the path vertices and the leaf vertices. To summarise, if  $G \in A$  and  $P$  is a longest path in  $G$ , then  $G$  only has three types of edges.

1. The path edges in  $P$ .
2. Non-crossing shortcut edges between vertices with distance two in  $P$ .
3. And leaf edges, but not at the ends of  $P$  and between a shortcut edge.

The linear maximum matching decomposition  $\omega$  of a graph  $G \in A$  is easy to construct. Let  $P = (x_1, x_2, \dots, x_m)$  be a longest path in  $G$ . Start with the ordering  $(x_1, x_2, \dots, x_m)$ . For every vertex  $x_i$  in  $G$  that has leaf vertices  $l_1, l_2, \dots, l_s$  as neighbours, insert these vertices right before  $x_i$ ,  $(x_1, \dots, x_{i-1}, x_i, \dots, x_m) \rightarrow (x_1, \dots, x_{i-1}, l_1, l_2, \dots, l_s, x_i, \dots, x_m)$ .

Let  $\omega = (v_1, v_2, \dots, v_n)$  be the resulting linear maximum matching decomposition of  $G$ . Every bipartite graph  $B_{\omega, i}$  will have maximum matching one. Let  $v_p$  and  $v_q$  be the first path vertices before and after  $v_i$  in  $\omega$  respectively. If  $v_i$  is a leaf vertex the only path edge in  $B_{\omega, i}$  is  $v_p v_q$ . The only leaf edges present in  $B_{\omega, i}$  are the ones incident to  $v_q$  since in  $\omega$  leaf vertices are places right before the vertex they neighbour.  $B_{\omega, i}$  might have a shortcut edge incident to  $v_q$ , but there can not be a shortcut edge from  $v_p$  to some  $v_j$  for  $q < j$ , since  $v_q$  has leaf neighbours. Since all edges in  $B_{\omega, i}$  are incident to  $v_q$  it will have maximum matching of size one.

If  $v_i$  is a path vertex,  $B_{\omega, i}$  will have no leaf edges, since in  $\omega$  leaf vertices are places right before the vertex they neighbour.  $B_{\omega, i}$  will have  $v_i v_p$  as an edge,

unless  $i = n$ , and no other path edges. There are two possible shortcut edges in  $B_{\omega,i}$   $v_p v_q$  and  $v_i v_j$ , where  $v_j$  is the next path vertex after  $v_q$ . These two edges cross,  $B_{\omega,i}$  will only have one of them. If it has  $v_p v_q$  all edges will be incident to  $v_p$ , and if it has  $v_i v_j$  all edges will be incident to  $v_i$ . Thus  $B_{\omega,i}$  will have maximum matching of size 1.

In conclusion: All graphs in A have a linear maximum matching decomposition of width at most 1. □

In figure 25 we see a graph with linear maximum matching-width 1. It is almost a caterpillar graph but it can have edge disjoint cycles of length 3 where at least one of the vertices in each cycle has degree 2.

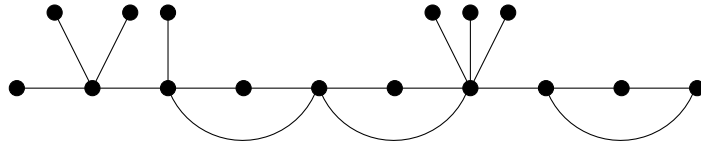


Figure 25: A graph of linear maximum matching-width 1

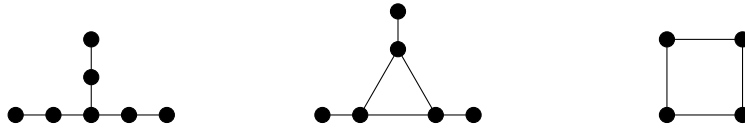


Figure 26

	pw	lbw	lmmw
At most 0			
At most 1			Figure 26
At most 2	110 graphs [6]	Roughly at least 68 graphs	?

Figure 27

## 8 Conclusion

In this thesis we started by carefully presenting the known relationships between the three tree-like parameters, tree-width, branch-width and maximum matching-width, that arise from a common definition using subtree representations. We have solved an open problem related to the fourth parameter definable by the subtree representation by showing that it is the same as tree-width. We also gave the full details of the proof of how maximum matching-width fits into the chordal supergraph framework, as this proof had only been outlined previously. The main contribution of this thesis is the study of the three related path-like parameters, including the well-known path-width and the two new parameters defined in this thesis that we called linear branch-width and linear maximum matching-width. We showed that linear branch-width is either equal to path-width or equal to path-width plus one. We also showed that linear maximum matching-width is less than path-width but never less than half the path-width. Finally, we showed that the classes of graphs of bounded parameter values are closed under minors and investigated the minimal forbidden minors of these path-like parameters. For path-width this set was known for the values 1, 2 and 3. We gave results for minimal forbidden minors of both the class of graphs of linear branch-width bounded by 1, 2 and 3 and also for the class of graphs of linear maximum matching-width bounded by 1 and 2.

One open problem is whether all minimal forbidden minors of graphs with linear branch-width less than  $k$ , is a minor of at least one of the minimal forbidden minors of graph with path-width less than  $k$ . Another is what will we get if we instead of restricting the maximum degree of the decomposition tree from 3 to 2 we loosen the restriction on maximum degree from 3 to 4? Here I think the chordal supergraphs might be the way to attack this problem.

## References

- [1] S. Arnborg, A. Proskurowski, and D. G. Corneil. Forbidden minors characterization of partial 3-trees. *Discrete Mathematics*, 80(1):1–19, 1990.
- [2] H. L. Bodlaender and D. M. Thilikos. Graphs with branchwidth at most three. *Journal of Algorithms*, 32(2):167–194, 1999.
- [3] F. Gavril. The intersection graphs of subtrees in trees are exactly the chordal graphs. *Journal of Combinatorial Theory, Series B*, 16(1):47–56, 1974.
- [4] J. Jeong, S. Ok, and G. Suh. Characterizing graphs of maximum matching width at most 2. *ArXiv e-prints*, jun 2016.
- [5] J. Jeong, S. H. Sæther, and J. A. Telle. Maximum matching width: New characterizations and a fast algorithm for dominating set. In Thore Husfeldt and Iyad Kanj, editors, *10th International Symposium on Parameterized and Exact Computation (IPEC 2015)*, volume 43 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 212–223, Dagstuhl, Germany, 2015. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.
- [6] Nancy G. Kinnarsley and Michael A. Langston. Obstruction set isolation for the gate matrix layout problem. *Discrete Applied Mathematics*, 54(2):169–213, 1994.
- [7] C. Paul and J. A. Telle. New tools and simpler algorithms for branchwidth. In Brodal G. S. and Leonardi S., editors, *Algorithms – ESA 2005: 13th Annual European Symposium, Palma de Mallorca, Spain, October 3-6, 2005. Proceedings*, pages 379–390. Springer Berlin Heidelberg, 2005.
- [8] C. Paul and J. A. Telle. Edge-maximal graphs of branchwidth  $k$ : The  $k$ -branches. *Discrete Mathematics*, 309(6):1467–1475, 2009.
- [9] N. Robertson and P. D. Seymour. Graph minors. x. obstructions to tree-decomposition. *Journal of Combinatorial Theory, Series B*, 52(2):153–190, 1991.
- [10] N. Robertson and P. D. Seymour. Graph minors. xx. wagner’s conjecture. *Journal of Combinatorial Theory, Series B*, 92(2):325–357, 2004.
- [11] N. Robertson and P.D. Seymour. Graph minors. ii. algorithmic aspects of tree-width. *Journal of Algorithms*, 7(3):309–322, 1986.
- [12] M. Vatshelle. *New Width Parameters of Graphs*. PhD thesis, University of Bergen, 2012.