Maximum matching width: new characterizations and a fast algorithm for dominating set

Jisu Jeong*, Sigve Hortemo Sæther, Jan Arne Telle

May 15, 2017

Abstract

A graph of treewidth k has a representation by subtrees of a ternary tree, with subtrees of adjacent vertices sharing a tree node, and any tree node sharing at most $k+1$ subtrees. Likewise for branchwidth, but with a shift to the edges of the tree rather than the nodes. In this paper we show that the mm-width of a graph - maximum matching width - combines aspects of both these representations, targeting tree nodes for adjacency and tree edges for the parameter value. The proof of this new characterization of mm-width is based on a definition of canonical minimum vertex covers of bipartite graphs. We show that these behave in a monotone way along branch decompositions over the vertex set of a graph.

We use these representations to compare mm-width with treewidth and branchwidth, and also to give another new characterization of mm-width, by subgraphs of chordal graphs. We prove that given a graph G and a branch decomposition of maximum matching width k we can solve the Minimum Dominating Set Problem in time $O^*(8^k)$, thereby beating $O^*(3^{\mathrm{tw}(G)})$ whenever $\mathrm{tw}(G) > \log_3 8 \times k \approx 1.893k$. Note that $\mathrm{mmw}(G) \leq \mathrm{tw}(G) + 1 \leq 3\,\mathrm{mmw}(G)$ and these inequalities are tight. Given only the graph G and using the best known algorithms to find decompositions, maximum matching width will be better for Minimum Dominating Set whenever $\mathrm{tw}(G) > 1.549 \times \mathrm{mmw}(G)$.

1 Introduction

The treewidth $\mathrm{tw}(G)$ and branchwidth $\mathrm{bw}(G)$ of a graph G are connectivity parameters of importance in algorithm design. By dynamic programming along the associated tree decomposition or branch decomposition one can solve many graph optimization problems in time linear in the graph size and exponential in the parameter. For every graph G, its treewidth and branchwidth are related by $\mathrm{bw}(G) \leq \mathrm{tw}(G) + 1 \leq \frac{3}{2}\,\mathrm{bw}(G)$ [18]. The two

*Supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No. NRF-2017R1A2B4005020).

parameters are thus equivalent with respect to fixed parameter tractability (FPT), with a problem being FPT parameterized by treewidth if and only if it is FPT parameterized by branchwidth. For some of these problems the best known FPT algorithms are optimal, up to some complexity theoretic assumption. For example, Minimum Dominating Set Problem can be solved in time $O^*(3^{\mathrm{tw}(G)})$ when given a tree decomposition of width $\mathrm{tw}(G)$ [21] but not in time $O^*((3-\varepsilon)^{\mathrm{tw}(G)})$ for every $\varepsilon > 0$ unless the Strong Exponential Time Hypothesis (SETH) fails [13].

Recently, a graph parameter equivalent to treewidth and branchwidth was introduced, the maximum matching width (or mm-width) $\mathrm{mmw}(G)$, defined by a branch decomposition over the vertex set of a graph G, using the symmetric submodular cut function obtained by taking the size of a maximum matching of the bipartite graph crossing the cut (by König's Theorem equivalent to minimum vertex cover) [22]. For a graph G we have $\mathrm{mmw}(G) \leq \mathrm{bw}(G) \leq \mathrm{tw}(G) + 1 \leq 3\,\mathrm{mmw}(G)$ and these inequalities are tight [22], for example any balanced branch decomposition will show that $\mathrm{mmw}(K_n) = \lceil \frac{n}{3} \rceil$ and it is known that $\mathrm{tw}(K_n) = n-1$ and $\mathrm{bw}(G_{n\times n}) = \mathrm{mmw}(G_{n\times n}) = n$ where $G_{n\times n}$ is the $n \times n$-grid graph [18, 10].

In this paper we show that given a branch decomposition over the vertex set of mm-width k we can solve Minimum Dominating Set Problem in time $O^*(8^k)$. This runtime beats the $O^*(3^{\mathrm{tw}(G)})$-time algorithm for treewidth [21] whenever $\mathrm{tw}(G) > \log_3 8 \times k \approx 1.893k$. If we assume only a graph G as input, then since mm-width has a symmetric submodular cut function [20] we can approximate mm-width to within a factor $3\,\mathrm{mmw}(G) + 1$ in $O^*(2^{3\,\mathrm{mmw}(G)})$-time using the generic algorithm of [15], giving a total runtime for solving Minimum Dominating Set Problem of $O^*(2^{9\,\mathrm{mmw}(G)})$. For treewidth we can in $O^*(2^{3.7\,\mathrm{tw}(G)})$-time [1] get an approximation to within a factor $(3+2/3)\,\mathrm{tw}(G)$ giving a total runtime for solving Minimum Dominating Set Problem of $O^*(3^{3.666\,\mathrm{tw}(G)})$.[1] This implies that on input G, using maximum matching width gives better exponential factor whenever $\mathrm{tw}(G) > 1.549\,\mathrm{mmw}(G)$.

Our results are based on a new characterization of graphs of mm-width at most k, as intersection graphs of subtrees of a tree. It can be formulated as follows:

For each $k \geq 2$ a graph G on vertices $v_1, v_2, ..., v_n$ has $\mathrm{mmw}(G) \leq k$ if and only if there exists a tree T of maximum degree at most 3 with nontrivial subtrees $T_1, T_2, ..., T_n$ such that if $v_iv_j \in E(G)$ then subtrees T_i and T_j have at least one node of T in common and for each edge of T there are at most k subtrees using it.

Replacing the three underlined parts in the above characterization by ($\mathrm{tw}(G) \leq k-1$, node, node) we define treewidth, while replacing by ($\mathrm{bw}(G) \leq k$, edge, edge) we define branchwidth [8, 19, 2, 16]. Note that while treewidth has a focus on nodes and branchwidth a focus on edges, mm-width combines aspects of both by a partial focus on nodes and on edges.

In this way the maximum matching width can more easily be compared to the much studied graph parameters treewidth and branchwidth. In our Theorem 3.10 we do this when we show $\mathrm{bw}(G) \leq 2\,\mathrm{mmw}(G)$, improving on the previous bound of $\mathrm{bw}(G) \leq 3\,\mathrm{mmw}(G)$

[1] Note that there is also an $O^*(c^{\mathrm{tw}(G)})$ time 3-approximation of treewidth [4], but the c is so large that the approximation alone has a bigger exponential part than the entire Minimum Dominating Set algorithm when using the 3.666-approximation.

from [22]. Since the proof of $\text{tw}(G) \leq 3\,\text{mmw}(G) - 1$ in [22] was based on a non-monotone cops and robber strategy not known to be efficiently computable, Vatshelle [22] asked whether one can find, in time $O(n^{3.5})$, a tree decomposition of width at most $3k-1$ given a branch decomposition of mm-width k. Using the new characterization, we solve this open problem in Theorem 3.9. We also arrive at the following alternative characterization: a graph G has $\text{mmw}(G) \leq k$ if and only if it is a subgraph of a chordal graph H and for every maximal clique X of H there exist $A, B, C \subseteq X$ with $A \cup B \cup C = X$ and $|A|, |B|, |C| \leq k$ such that each subset of X that is a minimal separator of H is a subset of either A, B or C.

In Section 2 we give definitions. In Section 3 we define canonical minimum vertex covers for all bipartite graphs, show some monotonicity properties of these, and use this properties to give the new characterizations of mm-width. In Section 4 we give the dynamic programming algorithm for Minimum Dominating Set Problem. We end in Section 5 with some discussions.

2 Definitions

For a simple and loopless graph $G = (V, E)$ and its vertex v, let $N(v)$ be the set of all vertices adjacent to v in G, and $N[v] = N(v) \cup \{v\}$. For a subset S of $V(G)$, let $N(S)$ be the set of all vertices that are not in S but are adjacent to some vertex of S in G, and $N[S] = N(S) \cup S$. A subset of vertices $S \subseteq V(G)$ is said to *dominate* the vertices in $N[S]$, and it is a *dominating set* of G if $N[S] = V(G)$. For disjoint $A, B \subseteq V$ we denote by $G[A, B]$ the bipartite subgraph of G containing all edges between a vertex in A and a vertex in B.

A *tree decomposition* of a graph G is a pair $(T, \{X_t\}_{t \in V(T)})$ consisting of a tree T and a family $\{X_t\}_{t \in V(T)}$ of vertex sets $X_t \subseteq V(G)$, called *bags*, satisfying the following three conditions:

(1) each vertex of G is in at least one bag,

(2) for each edge uv of G, there exists a bag that contains both u and v, and

(3) for nodes u, v, w of T, if v is on the path from u to w, then $X_u \cap X_w \subseteq X_v$.

The *width* of a tree decomposition $(T, \{X_t\}_{t \in V(T)})$ is $\max_{t \in V(T)} |X_t| - 1$. The *treewidth* of G, denoted by $\text{tw}(G)$, is the minimum width over all possible tree decompositions of G.

A *branch decomposition* over a finite set X, for some set of elements X, is a pair (T, δ) where T is a tree of maximum degree at most 3, and δ is a bijection from the leaves of T to the elements in X. Each edge ab disconnects T into two subtrees T_a and T_b. Likewise, each edge ab of T partitions the elements of X into two parts A and B, namely the elements mapped by δ from the leaves in T_a, and in T_b, respectively. An edge $ab \in E(T)$ is said to *induce* the partition $\{A, B\}$ of X.

A *rooted branch decomposition* over a finite set X is a branch decomposition (T, δ) over X where we subdivide an edge of T and make the new node the *root* r. In a rooted branch decomposition, for an internal node $x \in V(T)$, we denote by $\delta(x)$ the union of $\delta(\ell)$ for all leaves ℓ having x as its ancestor. By $\overline{\delta(x)}$ we denote its complement, $X \setminus \delta(x)$.

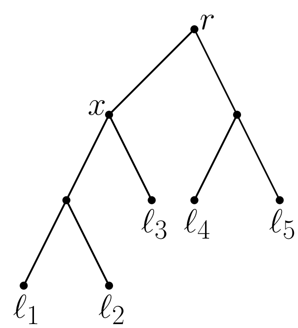

Figure 1: A tree with the root r in a rooted branch decomposition.

For example, $\delta(x) = \{\delta(\ell_1), \delta(\ell_2), \delta(\ell_3)\}$ in Figure 1.

For a finite set X and for all $A, B \subseteq X$, a function $f : 2^X \to \mathbb{R}$ is *symmetric* if $f(A) = f(X \setminus A)$ and *submodular* if $f(A) + f(B) \geq f(A \cup B) + f(A \cap B)$. Given a symmetric submodular function $f : 2^X \to \mathbb{R}$, using branch decompositions over X, we get a nice way of defining width parameters: For a branch decomposition (T, δ) and an edge e in T, we define the *f-value* of the edge e to be the value $f(A) = f(B)$ where A and B are the two parts of the partition induced by e in (T, δ), denoted $f(e)$. We define the *f-width* of a branch decomposition (T, δ) to be the maximum f-value over all edges of T, denoted $f(T, \delta)$: $\max_{e \in T}\{\text{the } f\text{-value of } e\}$. Let the *$f$-width* of a finite set X be the minimum f-width over all branch decompositions over X. If $|X| \leq 1$, then X admits no branch decomposition and we define its f-width to be $f(\emptyset)$.

For a graph G and a subset $S \subseteq E(G)$, the *branchwidth* $\text{bw}(G)$ of G is the f-width of $E(G)$ where $f : 2^{E(G)} \to \mathbb{R}$ is a function such that $f(S)$ is the number of vertices that are incident to both an edge in S and an edge in $E(G) \setminus S$.

The *maximum matching width* of a graph G, mm-width in short, is a width parameter defined on branch decompositions over $V(G)$, using the cardinality of maximum matchings. For a graph G and a subset $S \subseteq V(G)$, we define a function $\text{mm} : 2^{V(G)} \to \mathbb{R}$ such that $\text{mm}(S)$ is the size of a maximum matching in $G[S, V(G) \setminus S]$. The mm-width of a graph G, denoted $\text{mmw}(G)$, is the f-width of $V(G)$ for $f = \text{mm}$.

It is easy to prove that for an integer k the set M_k of graphs of mm-width at most k is minor-closed [10]. Thus, by Robertson-Seymour theorem [17], the set M_k has a finite minor obstruction set for each k. For some small k, the minor obstruction set is completely known.

Proposition 2.1. *A graph G has mm-width at most 1 if and only if G does not contain C_4 as a minor.*

Proof. Suppose G does not contain C_4 as a minor. It is known that this graph is a block graph such that every maximal 2-connected component is either K_2 or K_3. Since K_2 and K_3 have mm-width 1, we can easily construct a branch decomposition of mm-width at most 1.

Now suppose G contains C_4 with vertices v_1, v_2, v_3, v_4 as a minor. Choose an arbitrary branch decomposition (T, δ) of G. There must exist an edge e of T that induces the partition $\{A, B\}$ of $V(G)$ such that A contains two of v_1, v_2, v_3, v_4 and B contains the other two of

v_1, v_2, v_3, v_4. For all cases, the mm-value of e is at least 2. Thus, G has mm-width at least 2. This completes the proof. □

Jeong, Ok, and Suh [10] characterized the set of graphs of mm-width at most 2 using the minor obstruction set. They proved that there are 45 graphs in the minor obstruction set and found all of them. To prove this they used Theorem 3.8, our new characterization of mm-width.

3 Subtrees of a tree representation for mm-width

3.1 König covers.

In this subsection, we will define canonical minimum vertex covers for all bipartite graphs. Our starting point is a well-known result in graph theory.

Theorem 3.1 (König's Theorem [11]). *Given a bipartite graph G, for a maximum matching M and a minimum vertex cover C of G, the number of edges in M is the same as the number of vertices in C; $|M| = |C|$.*

Let (A, B) be the vertex partition of G. This statement can be proved in multiple ways. The harder direction, that a maximum matching is never smaller than a minimum vertex cover, does not hold for general graphs, and is usually proven by taking a maximum matching M and constructing a vertex cover C having size exactly $|M|$, as follows:

> For each edge $ab \in M$ (where $a \in A$ and $b \in B$), if ab is part of an alternating path starting in an unmatched vertex of A, then put b into C, otherwise put a into C.

For a proof that C indeed is a minimum vertex cover of G, see [7]. We will call the vertex cover C constructed by the above procedure an A-*König cover* of G. A B-König cover of G is constructed similarly by changing the roles of A and B (see Figure 2). Lemma 3.2 below

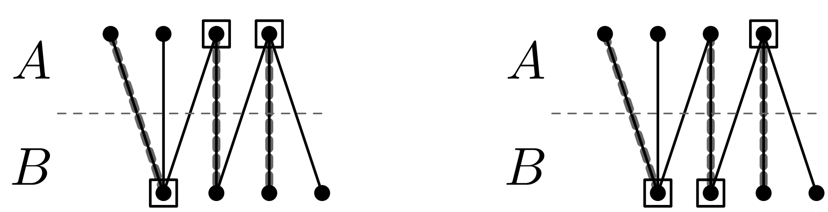

Figure 2: A-König cover and B-König cover.

shows that any A-König cover will, on the A-side consist of the A-vertices in the union over all minimum vertex covers, and on the B-side will consist of the B-vertices in the intersection over all minimum vertex covers.

Lemma 3.2. *Let $G = (A \cup B, E)$ be a bipartite graph and C be a minimum vertex cover of G. The set C is the A-König cover of G if and only if for every minimum vertex cover C' of G we have $A \cap C' \subseteq A \cap C$ and $B \cap C' \supseteq B \cap C$.*

Proof. Let M be a maximum matching of G and C^* be the A-König cover of G constructed from M. Since both C^* and C are minimum vertex covers, by showing that for every minimum vertex cover C' of G we have $A \cap C' \subseteq A \cap C^*$ and $B \cap C' \supseteq B \cap C^*$, as a consequence will also show that $C' = C^*$ if and only if for all minimum vertex covers C' of G we have $A \cap C' \subseteq A \cap C$ and $B \cap C' \supseteq B \cap C$. So this is precisely what we will do.

Let C' be a minimum vertex cover, and b be a vertex in $C^* \cap B$. We will show that $b \in C'$, and from that conclude $B \cap C' \supseteq B \cap C^*$. As $b \in C^*$ there must be some alternating path from b to an unmatched vertex $u \in A$. The vertices b and u are on different sides of the bipartite graph, so the alternating path P between u and b must be of some odd length $2k + 1$. From Theorem 3.1, we deduce that one and only one endpoint of each edge in M must be in C'. As each vertex in $V(P)$ is incident with at most two edges of P, and all edges of P must be covered by C', we need at least $\lceil (2k + 1)/2 \rceil = k + 1$ of the vertices in $V(P)$ to be in C'. However, the vertices of $V(P) - \{b\}$ are incident with only k edges of M. Therefore at most k of the vertices $V(P) - \{b\}$ can be in C'. In order to have at least $k + 1$ vertices from $V(P)$ in C' we thus must have $b \in C'$.

We now show that $C' \cap A \subseteq C^* \cap A$ by showing that $a \in C^*$ if $a \in A \cap C'$. Let E^* and E' be the edges of G not covered by $C^* \cap B$ and $C' \cap B$, respectively. Since $C^* \cap B \subseteq C' \cap B$, the set E^* must contain all the edges of E'. As C' is a minimum vertex cover, and all edges other than E' are covered by $C' \cap B$, a vertex a of A is in C' only if it covers an edge $e \in E'$. As $E' \subseteq E^*$, we have $e \in E^*$, and hence C^* must also cover e by a vertex in A. As G is bipartite, the only vertex from A that covers e is a, and we can conclude that $a \in C^*$. □

Corollary 3.3. *Let $G = (A \cup B, E)$ be a bipartite graph. Let $C_\cup$ and $C_\cap$ be the union and intersection of all minimum vertex covers of G, respectively. Then*

- *the A-König cover of G is equal to $(C_\cup \cap A) \cup (C_\cap \cap B)$,*
- *the B-König cover of G is equal to $(C_\cap \cap A) \cup (C_\cup \cap B)$.*

The following lemma establishes an important monotonicity property for A-König covers. For a set S of vertices and a vertex v, denote $S + v = S \cup \{v\}$.

Lemma 3.4. *Given a graph G and a tripartition (A, B, X) of the vertices $V(G)$, the following two properties hold for the A-König cover C_A of $G[A, B \cup X]$ and every minimum vertex cover C of $G[A \cup X, B]$.*

(1) $A \cap C \subseteq A \cap C_A$.

(2) $B \cap C \supseteq B \cap C_A$.

Proof. To prove this, we will show that it holds for $X = \{x\}$, and then by transitivity of subset inclusion and that a König cover is also a minimum vertex cover, it must hold also when X is an arbitrary subset of $V(G)$.

Let $A' = A + x$ and $B' = B + x$, and let C' be the A-König cover of the graph $G[A, B]$ (notice that this graph has one less vertex than G). We will break the proof into four parts, namely $A \cap C \subseteq A \cap C'$, $A \cap C' \subseteq A \cap C_A$, $B \cap C_A \subseteq B \cap C'$, and $B \cap C' \subseteq B \cap C$. Again, by transitivity of subset inclusion, this will be sufficient for our proof. We now look at each part separately.

$\underline{A \cap C \subseteq A \cap C'}$: Two cases: $|C| = |C'|$ and $|C| > |C'|$. We do the latter first. This means that $C' + x$ must be a minimum vertex cover of $G[A', B]$. Therefore the A'-König cover C^* of $G[A', B]$ must contain $(C' + x) \cap A'$. This means that C^* is a minimum vertex cover of $G[A, B]$, and by C' being the A-König cover of $G[A, B]$, we have from Lemma 3.2 that $C' \cap A \supseteq C^* \cap A$. Since C^* is a A'-König cover of $G[A', B]$ we have $C' \cap A' \supseteq C \cap A'$ and can conclude that $C' \cap A \supseteq A \cap C$. Now assume that the two vertex covers are of equal size. Clearly $x \notin C$, as then $C - \{x\}$ is a smaller vertex cover of $G[A, B]$ than C', so x is not in C. This means that C is a minimum vertex cover of $G[A, B]$, so all vertices in $A \cap C$ must be in C' by Lemma 3.2.

$\underline{A \cap C' \subseteq A \cap C_A}$: Suppose C' is smaller than C_A. This means $C' + x$ is a minimum vertex cover of $G[A, B']$, and hence $(C' + x) \cap A \subseteq C_A \cap A$ by Lemma 3.2. On the other hand, if C' is of the same size as C_A. Then C_A is a minimum vertex cover of $G[A, B]$, and so $x \notin C_A$. This means $C_A \cap N(x) \cap A \subseteq C_A \cap A$. And as C_A is a minimum vertex cover of $G[A, B]$, we know from Lemma 3.2 that $C_A \cap N(x) \cap A \subseteq C'$. In particular, this means C' covers all the edges of $G[A, B']$ not in $G[A, B]$, which means that C' is also a minimum vertex cover of $G[A, B']$. This latter observation means that $C' \cap A \subseteq C_A \cap A$ from Lemma 3.2.

$\underline{B \cap C_A \subseteq B \cap C'}$: Suppose C' is smaller than C_A. This means $C' + x$ is a minimum vertex cover of $G[A, B']$, and thus $B' \cap (C' + x) \supseteq B' \cap C_A$. Which implies that $B \cap C' \supseteq B \cap C_A$. Now assume that C' is of the same size as C_A. This means C_A is a minimum vertex cover of $G[A, B]$ and $x \notin C_A$. Furthermore, this means $N(x) \cap A \subseteq C_A \cap A \subseteq C' \cap A$ by Lemma 3.2 and we conclude that C' is a minimum vertex cover of $G[A, B']$. By Lemma 3.2, this means $B' \cap C_A \subseteq B' \cap C'$ and in particular $B \cap C_A \subseteq B \cap C'$.

$\underline{B \cap C' \supseteq B \cap C}$: Suppose C' is smaller than C. This means $C' + x$ is a minimum vertex cover of $G[A, B']$, and hence by Lemma 3.2 we have $B' \cap (C' + x) \subseteq B' \cap C_2$, which implies $B \cap C' \subseteq B \cap C_2$. Now suppose C' is of the same size as C. This means that C is a minimum vertex cover of $G[A, B]$, and hence we immediately get $C \cap B \supseteq C' \cap B$ by Lemma 3.2.

This completes the proof, as we by transitivity of subset inclusion have that $C_A \cap B \subseteq C \cap B$, and $C \cap A \subseteq C_A \cap A$. $\square$

The following lemma will allow us to prove an important connectedness property of König covers that arise from cuts of a given branch decomposition. Informally, the lemma says that for any tree node v, if we take the union of König covers for nodes in the subtree of v, and König covers of nodes outside the subtree of v, then the intersection of these two sets is in the König cover of v.

Lemma 3.5. *Given a connected graph G and a rooted branch decomposition (T, δ) over $V(G)$, for a node v in T, where $\mathcal{C}$ is the set of descendants of v and C_x means the $\delta(x)$-König cover of $G[\delta(x), \overline{\delta(x)}]$, we have that*

$$\left(\bigcup_{x \in V(T) \setminus \mathcal{C}} C_x \right) \cap \left(\bigcup_{x \in \mathcal{C}} C_x \right) \subseteq C_v.$$

Proof. For all $x \in \mathcal{C}$, since C_x is the $\delta(x)$-König cover and C_v is a minimum vertex cover, from Lemma 3.4, we have that $C_x \cap \overline{\delta(x)} \subseteq C_v \cap \overline{\delta(x)}$. In particular, since $\delta(x) \subseteq \delta(v)$, we have that $C_x \cap \overline{\delta(v)} \subseteq C_v \cap \overline{\delta(v)} \subseteq C_v$. Thus we know the containment relation in the statement of the lemma is true for $\overline{\delta(v)}$.

To prove that it is also true for $\delta(v)$ we show that for all $x \in V(T) \setminus \mathcal{C}$ we have $C_x \cap \delta(v) \subseteq C_v$. For all $x \in V(T) \setminus \mathcal{C}$ we have two cases, either $\delta(v) \subseteq \delta(x)$ (when x is an ancestor of v) or $\delta(v) \subseteq \overline{\delta(x)}$ (when x is neither a descendant of v nor an ancestor of v). In both cases we consider the $\delta(v)$-König cover C_v of $G[\delta(v), \overline{\delta(v)}]$ and the minimum vertex cover C_x of $G[\delta(x), \overline{\delta(x)}]$. Applying these to Lemma 3.4 we see that $C_x \cap \delta(v) \subseteq C_v \cap \delta(v) \subseteq C_v$. $\square$

Informally, the connectedness property following from this is that for any vertex $a \in V(G)$, if we take the set of tree nodes $x \in V(T)$ whose $\delta(x)$-König cover contains a, and consider the edges from these tree nodes to their parent in T then this must induce a connected subtree of T. Formally, define e_x to be the edge from x to its parent in T, then:

Corollary 3.6. *For any $a \in V(G)$ the set $\{e_x : a \in C_x\}$ induces a subtree T_a of T.*

Proof. Consider two vertices u and v such that $a \in C_u \cap C_v$. Let p be the lowest common ancestor of u and v. For every vertex w on the path from p to u and on the path from p to v, except p, we know that exactly one of u, v is a descendant of w. By Lemma 3.5, $(C_u \cap C_v) \subseteq C_w$. It means that if a vertex a of G is in both C_u and C_v then it is also in C_w, which implies that T_a is connected. $\square$

3.2 The new characterizations of graphs of mm-width at most k

Graphs of treewidth and branchwidth at most k can respectively be characterized using intersection graphs of subtrees of a tree as follows.

Theorem 3.7 ([2, 8, 19, 16])**.** *Let $k \geq 2$ be an integer. A graph G has $\mathrm{tw}(G) \leq k-1$ ($\mathrm{bw}(G) \leq k$, respectively) if and only if there exists a tree T of maximum degree at most 3 with nontrivial subtrees T_u for each $u \in V(G)$ such that*

(i) if $uv \in E(G)$ then the subtrees T_u and T_v have at least one node (edge, respectively) of T in common, and

(ii) for every node (edge, respectively) of T, there are at most k subtrees using it.

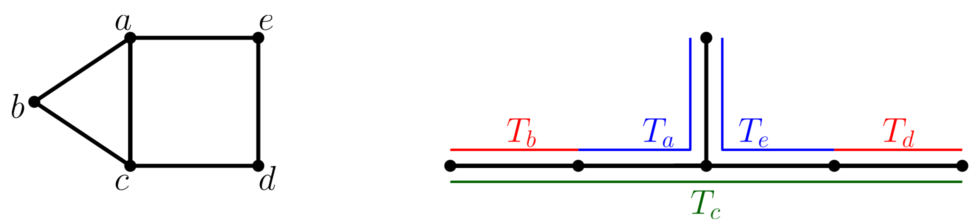

Figure 3: A graph of mm-width at most 2 and its representation as subtrees of a tree.

We give a similar characterization for graphs of mm-width at most k. (See Figure 3.) We say a graph is *nontrivial* if it has an edge.

Theorem 3.8. *A nontrivial graph $G = (V, E)$ has $\mathrm{mmw}(G) \le k$ if and only if there exist a tree T of maximum degree at most 3 and nontrivial subtrees T_u of T for all vertices $u \in V(G)$ such that*

(i) if $uv \in E(G)$ then the subtrees T_u and T_v have at least one node of T in common, and

(ii) for every edge of T there are at most k subtrees using this edge.

Proof. Forward direction: Let (T, δ) be a rooted branch decomposition over $V(G)$ having mm-width at most k, and assume G has no isolated vertices. For each edge $e = uv$ of T, with u a child of v, associate the $\delta(u)$-König cover C_u of $G[\delta(u), \overline{\delta(u)}]$ to the edge uv. For each vertex x of G, define the set of edges of T whose associated König covers contain x and let T_x be the sub-forest of T induced by these edges. Using Corollary 3.6 we have that T_x is a connected forest and thus a subtree of T.

Now, since the branch decomposition has mm-width at most k part (ii) in the statement of the Theorem holds. For an arbitrary edge ab of G, consider any edge e of T on the path from $\delta^{-1}(a)$ to $\delta^{-1}(b)$ and the partition (A, B) induced by e where $a \in A$, $b \in B$. Then the König cover associated to e must contain one of a and b, and thus, (i) holds as well. Finally, T_x is nontrivial because the edge of T incident with a leaf $\delta^{-1}(x)$ is associated with the König cover $\{x\}$. If G has isolated vertices, T_x is not nontrivial for an isolated vertex x. We fix this by setting T_x to consist exactly of the edge incident with $\delta^{-1}(x)$, for each isolated vertex x of G.

Backward direction: For each given subtree $\{T_u\}_{u \in V(G)}$ of T, choose an edge in T_u (it is also in T) and attach a leaf ℓ_u to the edge. We then extend T_u to contain ℓ_u and set $\delta(\ell_u) = u$. Exhaustively remove leaves (from both T and the subtrees) that are not mapped by δ. Call the resulting tree T' and subtrees $\{T'_u\}_{u \in V(G)}$. Note that subtrees $\{T'_u\}_{u \in V(G)}$ and T' still satisfy (i) and also (ii) since the only increase in any subtree T_u was to include the edge to ℓ_u but no other subtree uses this edge. We claim that (T', δ) is a branch decomposition of mm-width at most k. It is clearly a branch decomposition over $V(G)$, and for each edge e of T', if we choose $S \subseteq V(G)$ to be those vertices u with T_u using this edge e, then this will be a vertex cover of the bipartite graph H given by this edge e, and of size at most k because for an edge xy in H, one of T_x and T_y must contain e. $\square$

This new characterization of mm-width, using the subtrees of tree representation, allows for more easy comparison with treewidth and branchwidth. Vatshelle [22] proved $\text{tw}(G) \leq 3\,\text{mmw}(G) - 1$ using non-monotone *cops and robber games*, which are not efficiently computable. Using the new characterization of mm-width in combination with the characterization of treewidth in Theorem 3.7, we get the following theorem solving an open problem of [22]:

Theorem 3.9. *Given a branch decomposition (T, δ) over $V(G)$ of mm-width k, we can in $O(|V(G)|^{3.5})$ time construct a tree decomposition $(T, \{X_t\}_{t\in V(T)})$ of treewidth at most $3k-1$.*

Proof. The proof of Theorem 3.8 was constructive, meaning that given a branch decomposition (T, δ) of mm-width k, we showed how to construct a "subtrees of a tree"-representation conforming to the alternative characterization, based on König covers of the cuts induced by (T, δ). Let $n = |V(G)|$. So, since finding a maximum matching of a bipartite graph can be done in $O(n^{2.5})$ time, by Hopcroft and Karp [9], and transforming this to a König cover only takes an additional linear time suppressed by the O-notation, we can in fact construct the "subtrees of a tree"-representation T with subtrees $\{T_u\}_{u\in V(G)}$ in $O(n^{3.5})$ time (the extra n comes from iterating over all cuts induced by (T, δ)). To construct a tree-decomposition from this, simply have the tree-decomposition $(T, \{X_t\}_{t\in V(T)})$ where $X_t = \{u \in V(G) : \text{ so that } t \in V(T_u)\}$. This is a tree-decomposition of treewidth at most $3k - 1$, because:

(1) each vertex $u \in V(G)$ is in some bag X_t for $t \in V(T)$, in particular in all the bags $\{X_t\}_{t\in V(T)}$.

(2) for any edge $uv \in E(G)$, there must be a bag X_t containing both u and v, since we know T_u and T_v must intersect in at least one node t.

(3) The bags containing any node u of $V(G)$ must form a subtree, since the bags containing u exactly correspond to the vertices of the subtree T_u.

□

Combining the $\text{tw}(G) \leq 3\,\text{mmw}(G)-1$ bound of [22] with the $\text{bw}(G) \leq \text{tw}(G)+1$ bound of [18] it was known that $\text{bw}(G) \leq 3\,\text{mmw}(G)$. We improve this to $\text{bw}(G) \leq 2\,\text{mmw}(G)$ using the new characterization.

Theorem 3.10. *For a nontrivial graph G,* $\text{bw}(G) \leq 2\,\text{mmw}(G)$.

Proof. Suppose that G has maximum matching width k. Then by Theorem 3.8, there exists a tree F of maximum degree at most 3 with nontrivial subtrees F_u of F for all vertices $u \in V(G)$ such that

(i) if $uv \in E(G)$ then the subtrees F_u and F_v have at least one node of F in common, and

(ii) for every edge of F there are at most k subtrees using this edge.

Firstly, subdivide all edges of F and any F_u, and call these T and T_u. Note the conditions above still hold for T and these T_u. For all $u \in V(G)$ we construct a subtree T'_u of T from T_u, such that if T_u and T_v share a node, then T'_u and T'_v will share an edge, to satisfy the branchwidth condition of Theorem 3.7. We first fix an embedding of T in the plane. For an internal node x of T and an edge e incident with x, let e_x be the edge incident with x next to e counterclockwise. The subtree T'_u will contain T_u and q additional edges, where q is the number of leaves ℓ of T_u with ℓ an internal node of T. For each such ℓ, incident to edge e of T_u, we will in T'_u add the edge e_ℓ. Note that if two subtrees T_u and T_v share a node of T but do not share an edge of T, then T'_u and T'_v will share an edge of T, since at least one of those subtrees will have been extended to include an edge of the other subtree. Also, for an edge $e = \{x, y\}$ of T, with y the new vertex arising from subdividing an edge of F, if $e \notin T_u$ but $e \in T'_u$, then T_u must have contained the unique edge e' incident to x such that e is counterclockwise to e'. There were at most k subtrees T_u containing e' and at most k containing e, so there are at most $2k$ subtrees T'_u containing e. Therefore, by Theorem 3.7, G has branchwidth at most $2k$. □

Graphs of treewidth at most k are famously known as being exactly those having a chordal supergraph of maximum clique size $k+1$. Also graphs of branchwidth at most k can be characterized as subgraphs of chordal graphs, as in the following result from [16].

Theorem 3.11 ([16])**.** *A graph has branchwidth at most k if and only if it is a subgraph of a chordal graph H where every maximal clique X of H has three subsets of size at most k each such that any two subsets have union X, with the property that every minimal separator of H contained in X is contained in one of the three subsets.*

As a corollary of Theorem 3.8 we show that also graphs of mm-width at most k can be characterized as subgraphs of chordal graphs.

Corollary 3.12. *A graph has maximum matching width at most k if and only if it is a subgraph of a chordal graph H and for every maximal clique X of H there exists $A, B, C \subseteq X$ with $A \cup B \cup C = X$ and $|A|, |B|, |C| \leq k$ such that any subset of X that is a minimal separator of H is a subset of either A, B or C.*

We only sketch the proof, which is similar to the alternative characterization of branchwidth given in [16]. We say a tree is *subcubic* if all nodes have degree at most 3. Note that a graph is chordal if and only if it is an intersection graph of subtrees of a tree [8]. In the forward direction, take the chordal graph resulting from the subtrees of subcubic tree representation. In the backward direction, we start by taking a clique tree of H, which can be viewed as a tree decomposition, and making it subcubic, as follows. For each bag of degree larger than three, belonging to a maximal clique X, make a bag X with the three neighboring bags A, B, C. For minimal separators $S_1, ..., S_q \subseteq X$ contained in A, make a path extending from bag A of q new bags $A_1, ..., A_q$ each containing A, and then for $1 \leq i \leq q$ attach a new bag containing S_i to A_i. These subtrees, one for each maximal clique, is then connected together in a tree by the structure of the clique tree, adding an edge between bags of identical minimal separators. This results in a subcubic tree decomposition which is easily made into a subtrees of ternary tree representation.

4 Fast Dynamic Programming for Minimum Dominating Set Problem parameterized by mm-width

Given a rooted branch decomposition (T, δ) of G of mm-width k, we will in this section give an $O^*(8^k)$-time algorithm for computing the size of a Minimum Dominating Set of G. This is a dynamic programming algorithm on a rooted tree decomposition $(T', \{X_t\}_{t \in V(T')})$ of G that we compute from (T, δ) as follows.

Given a rooted branch decomposition (T, δ) of G having mm-width k the proof of Theorem 3.8 yields a polynomial-time algorithm to find a family $\{T_u\}_{u \in V(G)}$ of nontrivial subtrees of T (note we can assume T is a rooted tree with the root of degree 2 and all other internal nodes of degree 3) such that (i) if $uv \in E(G)$ then the subtrees T_u and T_v have at least one node of T in common, and (ii) for every edge of T there are at most k subtrees using this edge.

¿From this we construct a rooted tree decomposition $(T', \{X_t\}_{t \in V(T')})$ of G, as follows. See Figure 4. Let T' be a tree with a node set $F \cup H \cup \{r\}$ where F is the set of edges of T, H is the set of non-root nodes (all of degree-3) of T, and r is the root of T and also the root of T'. Two nodes e, v of T' are adjacent if and only if $e \in F$ and $v \in H \cup \{r\}$ are incident in T. For a node $e \in F$, let X_e be the set of vertices in G such that $w \in X_e$ if subtree T_w uses edge e of T. For a node $v \in H$, let X_v be the set of vertices in G such that for the three incident edges e_1, e_2, e_3 of v in T, $X_v = X_{e_1} \cup X_{e_2} \cup X_{e_3}$. Let $X_r = X_{e_1} \cup X_{e_2}$ for e_1 and e_2 incident with r in T. Then $(T', \{X_t\}_{t \in V(T')})$ is a tree decomposition of G with a root r which we will use in the dynamic programming. See Figure 4.

Let us now define the relevant subproblems for the dynamic programming over this tree decomposition. For a node t of the tree we denote by G_t the graph induced by the union of X_u where u is a descendant of t. A coloring of a bag X_t is a mapping $f : X_t \to \{1, 0, *\}$ with the meaning that:

- all vertices with color 1 are contained in the dominating set of this partial solution in G_t,
- all vertices with color 0 are dominated,
- while vertices with color * might be dominated, not dominated, or in the dominating set.

Thus the only restriction is that a vertex with color 1 must be a vertex in the dominating set, which we will call a dominator, and a vertex with color 0 must be dominated. Thus, for every $S \subseteq V(G)$ there is a set $c(S)$ of $3^{|S|}2^{|N(S)|}$ colorings $f : V(G) \to \{1, 0, *\}$ compatible with taking S as a set of dominators, with vertices of S colored 1, 0 or $*$, vertices of $N(S)$ colored 0 or $*$, and the remaining vertices colored $*$.

For a coloring f of a bag X_t, we denote by $\mathcal{T}[t, f]$ (and view this as a 'Table' of values) the minimum $|S|$ over all $S \subseteq V(G_t)$ such that there exists $f' \in c(S)$ with $f'|_{X_t} = f$ and $f'|_{V(G_t) \setminus X_t}$ having everywhere the value 0. In other words, the minimum size of a set S of vertices of G_t that dominates all vertices in $V(G_t) \setminus X_t$, with a coloring f' compatible with taking S as a set of dominators, such that f' restricted to X_t gives f. If no such set S exists, then $\mathcal{T}[t, f] = \infty$. Note that the size of the minimum dominating set of G is the minimum value over all $\mathcal{T}[r, f]$ where $f^{-1}(*) = \emptyset$ at the root r. We initialize the table at a leaf ℓ, with $X_\ell = \{v\}$ as follows. Denote by f_i the coloring from $\{v\}$ to $\{1, 0, *\}$ with $f_i(v) = i$ for $i \in \{1, 0, *\}$. Then for a leaf bag X_ℓ, set $\mathcal{T}[\ell, f_1] := 1$, $\mathcal{T}[\ell, f_0] := \infty$, $\mathcal{T}[\ell, f_*] := 0$.

For internal nodes of the tree, instead of separate 'Join, Introduce and Forget' operations we will give a single update rule with several stages. We will be using an Extend-Table subroutine which takes a partially filled table $\mathcal{T}[t, \cdot]$ and extends it to table $\mathcal{T}'[t, \cdot]$ so the result will adhere to the above definition, ensuring the monotonicity property that $\mathcal{T}'[t, f] \leq \mathcal{T}'[t, f']$ for every f we can get from f' by changing the color of a vertex from 1 to 0 or $*$, or from 0 to $*$. Extend-Table for a bag X_t is implemented as follows:

(a) **Initialize.**

$$\mathcal{T}'[t, f] := \begin{cases} \mathcal{T}[t, f] & \text{if } \mathcal{T}[t, f] \text{ is defined} \\ \infty & \text{otherwise} \end{cases}$$

(b) **Change from 1 to 0.**

$F_1 := F_2 := \ldots := F_{|X_t|} := \emptyset$
  for all f such that $\mathcal{T}'[t, f]$ is defined **do**
    add f to F_q where $q = |\{v : f(v) = 1\}|$
  end for
  for $q = |X_t|$ down to 1 **do**
    for all $f \in F_q$ **do**
      for all choices of $u \in \{v : f(v) = 1\}$ **do**
        define $f_u(u) = 0$ and $f_u(x) = f(x)$ for $x \neq u$
        $\mathcal{T}'[t, f_u] := \min\{\mathcal{T}'[t, f_u], \mathcal{T}'[t, f]\}$
        $F_{q-1} := F_{q-1} \cup \{f_u\}$
      end for
    end for
  end for

(c) **Change from 0 to $*$.** Similarly as in Step (b)

Note the transition from color 1 to $*$ will happen by transitivity. In this way, Extend-Table ensures the monotonicity property that $\mathcal{T}'[t, f] \leq \mathcal{T}'[t, f']$ for every f we can get from f' by changing the color of a vertex from 1 to 0 or $*$, or from 0 to $*$. The time for Extend-Table is proportional to the number of entries in the tables times $|X_t|$.

Assume we have the situation in Figure 4, corresponding to the bags surrounding every degree-3 node x of the tree decomposition. This arises from the branch decomposition (and the subtrees of tree representation) having a node incident to three edges, creating three bags a, b, c containing subsets of vertices A, B, C, respectively, each of size at most k, and

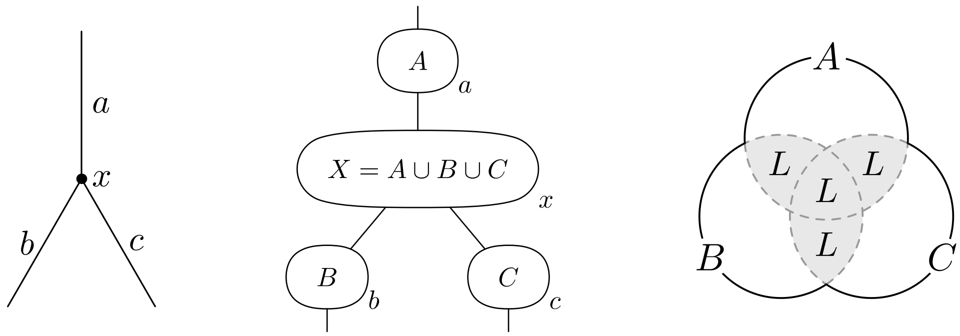

Figure 4: On the left is part of the tree used in the subtree representation of G, with node x having three incident edges a, b, c. In the middle the corresponding part of the tree-decomposition, with four bags, assuming the vertices whose subtrees use edge a are $A \subseteq V(G)$, edge b are $B \subseteq V(G)$, and edge c are $C \subseteq V(G)$. For simplicity we call the bags a, b, c, x. If mm-width is k then the size constraints are $|A|, |B|, |C| \leq k$. On the right an illustration of how the sets A, B, C overlap.

giving rise to the four bags a, b, c, x in Figure 4, with the latter containing subsets of vertices $X = A \cup B \cup C$. Let $L = (A \cap B) \cup (A \cap C) \cup (B \cap C)$. Assume we have already computed $\mathcal{T}[b, f]$ and $\mathcal{T}[c, f]$ for all $3^{|B|}$ and $3^{|C|}$ choices of f, respectively. We want to compute $\mathcal{T}[a, f]$ for all $3^{|A|}$ choices of f, in time $O^*(\max\{3^{|A|}, 3^{|B|}, 3^{|C|}, 3^{|L|}2^{|X \setminus L|}\})$. Note that we will not compute the table $\mathcal{T}[x, \cdot]$, as it would have $3^{|X|}$ entries, which is more than the allowed time bound. Instead, we compute a series of tables:

(1) $\mathcal{T}_b^1[x, \cdot]$ (and $\mathcal{T}_c^1[x, \cdot]$) of size $3^{|B|}$, by extending the coloring f of B for each entry $\mathcal{T}[b, f]$ to a unique coloring f' of X, based on the neighborhood of the dominators in f

(2) $\mathcal{T}_b^2[x, \cdot]$ (and $\mathcal{T}_c^2[x, \cdot]$) of size at most $\max(3^{|B|}, 3^{|B \cap L|}2^{|X \setminus (B \cap L)|})$, by changing each coloring f of X to a coloring f' of X where vertices in $B \setminus L$ having color 1 instead are given color 0 (note these vertices have no neighbors in $V(G) \setminus V(G_x)$)

(3) $\mathcal{T}_b^3[x, \cdot]$ (and $\mathcal{T}_c^3[x, \cdot]$) of size exactly $3^{|B \cap L|}2^{|X \setminus (B \cap L)|}$, with $f^{-1}(1) \subseteq B \cap L$, by running Extend-Table on $\mathcal{T}_b^2[x, \cdot]$

(4) $\mathcal{T}_{sc}^1[x, \cdot]$ of size $3^{|L|}2^{|X \setminus L|}$ by subset convolution over parts of $\mathcal{T}_b^3[x, \cdot]$ and $\mathcal{T}_c^3[x, \cdot]$

(5) $\mathcal{T}_{sc}^2[x, \cdot]$ of size $3^{|L|}2^{|X \setminus L|}$ by running Extend-Table on $\mathcal{T}_{sc}^1[x, \cdot]$

(6) $\mathcal{T}[a, \cdot]$ of size $3^{|A|}$ by going over all $3^{|A|}$ colorings of A and minimizing over appropriate entries of $\mathcal{T}_{sc}^2[x, \cdot]$

Note that in Step (4) we use the following:

Theorem 4.1 (Fast Subset Convolution [3]). *For two functions $g, h : 2^V \to \{-M, \ldots, M\}$, given all the $2^{|V|}$ values of g and h in the input, all $2^{|V|}$ values of the subset convolution of g*

and h over the integer min-sum semiring, that is, $(g * h)(Y) = \min_{Q \cup R = Y \text{ and } Q \cap R = \emptyset} g(Q) + h(R)$*, can be computed in time* $2^{|V|}|V|^{O(1)} \cdot O(M \log M \log\log M)$.

Let us now give the details of the first three steps:

(1) Compute $\mathcal{T}_b^1[x, \cdot]$. In any order, go through all $f : B \to \{1, 0, *\}$ and compute $f' : B \cup A \cup C \to \{1, 0, *\}$ by

$$f'(v) = \begin{cases} f(v) & \text{if } v \in B \\ 0 & \text{if } v \notin B \text{ and there exists } u \in B \text{ such that } f(u) = 1 \text{ and } uv \in E(G) \\ * & \text{otherwise} \end{cases}$$

and set $\mathcal{T}_b^1[x, f'] := \mathcal{T}[b, f]$.

(2) Compute $\mathcal{T}_b^2[x, \cdot]$. This table should only be indexed by f where $f(v) = 1$ implies $v \in B \cap L$. We will update these iteratively, so we first initialize $\mathcal{T}_b^2[x, f] = \infty$ for all $f : B \cup A \cup C \to \{1, 0, *\}$ where $f^{-1}(1) \subseteq B \cap L$. Then, in any order, for any f used to index $\mathcal{T}_b^1[x, f]$ as defined in Step (1), compute $f' : B \cup A \cup C \to \{1, 0, *\}$ by

$$f'(v) = \begin{cases} 0 & \text{if } v \in B \setminus L \text{ and } f(v) = 1 \\ f(v) & \text{otherwise} \end{cases}$$

and set $\mathcal{T}_b^2[x, f'] := \min\{\mathcal{T}_b^2[x, f'], \mathcal{T}_b^1[x, f]\}$. There will be no other entries in $\mathcal{T}_b^2[x, \cdot]$.

(3) Compute $\mathcal{T}_b^3[x, \cdot]$ by Extend-Table on $\mathcal{T}_b^2[x, \cdot]$.

The total time for the above three steps is bounded by $O^*(\max\{3^{|B|}, 3^{|B \cap L|}2^{|X \setminus (B \cap L)|}\})$. Note that $\mathcal{T}_b^3[x, f]$ is defined for all f where vertices in $B \cap L$ take on values $\{1, 0, *\}$ and vertices in $X \setminus (B \cap L)$ take on values $\{0, *\}$. The value of $\mathcal{T}_b^3[x, f]$ will be the minimum $|S|$ over all $S \subseteq V(G_b)$ such that there exists $f' \in c(S)$ with $f'|_X = f$ and $f'|_{V(G_b) \setminus X}$ having everywhere the value 0. Note the slight difference from the standard definition, namely that even though the coloring f is defined on X, the dominators only come from $V(G_b)$, and not from $V(G_x)$. The table $\mathcal{T}_c^3[x, \cdot]$ is computed in a similar way, with the colorings again defined on X but with the dominators now coming from $V(G_c)$.

When computing a Join of these two tables, we want dominators to come from $V(G_b) \cup V(G_c)$. Because of the monotonicity property that holds for these two tables, we can compute their Join $\mathcal{T}_{sc}^1[x, f]$ for all f where vertices in L take on values $\{1, 0, *\}$ and vertices in $X \setminus L$ take on values $\{0, *\}$, by combining colorings as follows:

$$\mathcal{T}_{sc}^1[x, f] = \min_{f_b, f_c}(\mathcal{T}_b^3[x, f_b] + \mathcal{T}_c^3[x, f_c]) - |f^{-1}(1) \cap B \cap C|$$

where f_b, f_c satisfy:

- $f(v) = 0$ if and only if $(f_b(v), f_c(v)) \in \{(0, *), (*, 0)\}$,
- $f(v) = *$ if and only if $f_b(v) = f_c(v) = *$,

- $f(v) = 1$ if and only if $v \in B \cap C$ and $f_b(v) = f_c(v) = 1$, or $v \in B \setminus C$ and $(f_b(v), f_c(v)) = (1, *)$, or $v \in C \setminus B$ and $(f_b(v), f_c(v)) = (*, 1)$.

This means that we can apply subset convolution to compute a table $\mathcal{T}^1_{sc}[x, f]$ on $3^{|L|}2^{|X \setminus L|}$ entries based on $\mathcal{T}^3_b[x, f]$ and $\mathcal{T}^3_c[x, f]$. Note that $(B \cap L) \cup (C \cap L) = L$. For this step we follow the description in [6, Section 11.1.2]. Fix a set $D \subseteq L$ to be the dominating vertices. Let F_D denote the set of $2^{|X \setminus D|}$ functions $f : X \to \{1, 0, *\}$ such that $f^{-1}(1) = D$, that is, with vertices in $X \setminus D$ mapping in all possible ways to $\{0, *\}$. For each $D \subseteq L$ we will by subset convolution compute the values of $\mathcal{T}^1_{sc}[x, f]$ for all $f \in F_D$.

We represent every $f \in F_D$ by the set $S = f^{-1}(0)$ and define $b_S : X \to \{1, 0, *\}$ such that $b_S(x) = 1$ if $x \in D \cap B$, $b_S(x) = 0$ if $x \in S$, $b_S(x) = *$ otherwise. Similarly, define $c_S : X \to \{1, 0, *\}$ such that $c_S(x) = 1$ if $x \in D \cap C$, $c_S(x) = 0$ if $x \in S$, $c_S(x) = *$ otherwise. Then, as explained previously, for every $f \in F_D$ we want to compute

$$\mathcal{T}^1_{sc}[x, f] = \min_{Q \cup R = f^{-1}(0) \text{ and } Q \cap R = \emptyset} (\mathcal{T}^3_b[x, b_Q] + \mathcal{T}^3_c[x, c_R]) - |f^{-1}(1) \cap B \cap C|.$$

Define functions $T_b : 2^{X \setminus D} \to \mathbb{N}$ such that for every $S \subseteq X \setminus D$ we have $T_b(S) = \mathcal{T}^3_b[x, b_S]$. Likewise, define functions $T_c : 2^{X \setminus D} \to \mathbb{N}$ such that for every $S \subseteq X \setminus D$ we have $T_c(S) = \mathcal{T}^3_c[x, c_S]$. Also, define $a_S : X \to \{1, 0, *\}$ such that $a_S(x) = 1$ if $x \in D$, $a_S(x) = 0$ if $x \in S$, $a_S(x) = *$ otherwise. We then compute for every $S \subseteq X \setminus D$,

$$\mathcal{T}^1_{sc}[x, a_S] := (T_b * T_c)(S) - |f^{-1}(1) \cap B \cap C|$$

where the subset convolution is over the mini-sum semiring.

(4) In Step (4), by Fast Subset Convolution, Theorem 4.1, we compute $\mathcal{T}^1_{sc}[x, a_S]$, for all a_S defined by all $f \in F_D$, in $O^*(2^{|X \setminus D|})$ time each. For all such subsets $D \subseteq L$ we get the time

$$\sum_{D \subseteq L} 2^{|X \setminus D|} = \sum_{D \subseteq L} 2^{|X \setminus L|} 2^{|L \setminus D|} = 2^{|X \setminus L|} \sum_{D \subseteq L} 2^{|L \setminus D|} = 2^{|X \setminus L|} 3^{|L|}.$$

(5) In Step (5), we need to run Extend-Table on $\mathcal{T}^1_{sc}[x, \cdot]$ to get the table $\mathcal{T}^2_{sc}[x, \cdot]$. This since the subset convolution was computed for each fixed set of dominators so the monotonicity property of the table may not hold. Note that the value of $\mathcal{T}^2_{sc}[x, f]$ will be the minimum $|S|$ over all $S \subseteq V(G_b) \cup V(G_c)$ such that there exists $f' \in c(S)$ with $f'|_X = f$ and $f'|_{(V(G_b) \cup V(G_c)) \setminus X}$ having everywhere the value 0.

(6) In Step (6), we will compute $f' : B \cup A \cup C \to \{1, 0, *\}$, for each $f : A \to \{1, 0, *\}$, by

$$f'(v) = \begin{cases} 1 & \text{if } v \in A \cap L \text{ and } f(v) = 1 \\ 0 & \text{if } v \in A \text{ and } f(v) = 0 \text{ and } N(v) \cap f^{-1}(1) = \emptyset \\ 0 & \text{if } v \notin A \text{ and } N(v) \cap f^{-1}(1) = \emptyset \\ * & \text{otherwise} \end{cases}$$

and set $\mathcal{T}[a, f] := \mathcal{T}^2_{sc}[x, f'] + |f^{-1}(1) \cap (A \setminus L)|$.

Note that when we iterate over all choices of $f : A \to \{1, 0, *\}$, the vertices colored 0 (in addition to all vertices of $X \setminus A$) must be dominated by either the vertices in $f^{-1}(1)$ or the vertices in $X \setminus V_a$. As we know precisely what vertices of $f^{-1}(0)$ are dominated by $f^{-1}(1)$, we know the rest must be dominated from vertices of $X \setminus V_a$, and therefore we look in $\mathcal{T}_{sc}[x, f']$ at an index f' which colors the rest of $f^{-1}(0)$ by 0. We can also observe that it is not important for us whether or not $f^{-1}(0)$ contains all neighbors of $f^{-1}(1)$, since we are iterating over all choices of f - also those where $f^{-1}(0)$ contains all neighbors of $f^{-1}(1)$.

The total runtime becomes $O^*(\max\{3^{|A|}, 3^{|B|}, 3^{|C|}, 3^{|L|}2^{|(A\cup B\cup C)\setminus L|}\})$, with $L = (A \cap B) \cup (A \cap C) \cup (B \cap C)$ and with constraints $|A|, |B|, |C| \leq k$. This runtime is maximum when $L = \emptyset$, giving a runtime of $O^*(2^{3k})$. We thus have the following theorem.

Theorem 4.2. *Given a graph G and branch decomposition over its vertex set of mm-width k we can solve Dominating Set in time $O^*(8^k)$.*

5 Discussion

We have shown that the graph parameter mm-width will for some graphs be better than treewidth for solving Minimum Dominating Set Problem. The improvement holds whenever $\text{tw}(G) > 1.549 \times \text{mmw}(G)$, if given only the graph as input. In Figure 5 we list some examples of small graphs having treewidth at least twice as big as mm-width. It could be interesting to explore the relation between treewidth and mm-width for various well-known classes of graphs. The given algorithmic technique, using fast subset convolution, should extend to any graph problem expressible as a maximization or minimization over (σ, ρ)-sets, using the techniques introduced for treewidth in [21].

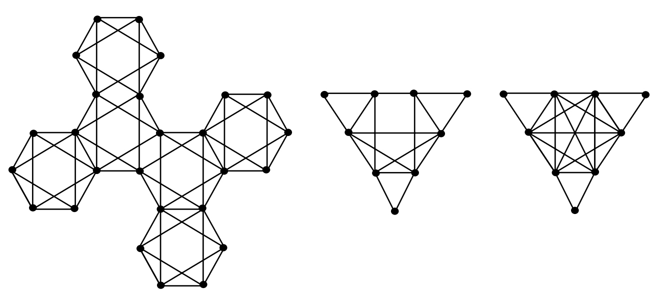

Figure 5: Three graphs of mm-width 2. Left, middle have treewidth 4, and right has treewidth 5.

We may also compare with branchwidth. Let ω be the *exponent of matrix multiplication*, which is less than 2.3728639 [12]. In 2010, Bodlaender, van Leeuwen, van Rooij, and Vatshelle [5] gave an $O^*(3^{\frac{\omega}{2}k})$ time algorithm solving Minimum Dominating Set Problem if an input graph is given with its branch decomposition of width k. This means that given decompositions of $\text{bw}(G)$ and $\text{mmw}(G)$ our algorithm based on mm-width is faster than the algorithm in [5] whenever $\text{bw}(G) > \log_3 8 \cdot \frac{2}{\omega} \cdot \text{mmw}(G) > \frac{2\log_3 8}{2.3728639} \cdot \text{mmw}(G) > 1.6\,\text{mmw}(G)$.

Taking the subtrees of tree representation for treewidth, branchwidth and maximum matching width mentioned in the Introduction as input, our algorithm for dominating set can be seen as a generic one that works for any of treewidth, branchwidth or maximum matching width of the given representation, and in case of both treewidth and mm-width it will give the best runtime known.

We gave an alternative definition of mm-width using subtrees of a tree, similar to alternative definitions of treewidth and branchwidth. We saw that in the subtrees of a tree representation treewidth focuses on nodes, branchwidth focuses on edges, and mm-width combines them both. 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 node (similar to treewidth). However, this fourth parameter is equivalent to treewidth [14].

References

[1] E. Amir. Approximation algorithms for treewidth. *Algorithmica*, 56(4):448–479, 2010.

[2] S. Arnborg and A. Proskurowski. Characterization and recognition of partial k-trees. In *Proceedings of the sixteenth Southeastern international conference on combinatorics, graph theory and computing (Boca Raton, Fla., 1985)*, volume 47, pages 69–75, 1985.

[3] A. Björklund, T. Husfeldt, P. Kaski, and M. Koivisto. Fourier meets Möbius: fast subset convolution. In *STOC'07—Proceedings of the 39th Annual ACM Symposium on Theory of Computing*, pages 67–74. ACM, New York, 2007.

[4] H. L. Bodlaender, P. G. Drange, M. S. Dregi, F. V. Fomin, D. Lokshtanov, and M. Pilipczuk. An $o(c^k n)$ 5-approximation algorithm for treewidth. In *Foundations of Computer Science (FOCS), 2013 IEEE 54th Annual Symposium on*, pages 499–508. IEEE, 2013.

[5] H. L. Bodlaender, E. J. van Leeuwen, J. M. M. van Rooij, and M. Vatshelle. Faster algorithms on branch and clique decompositions. In *Mathematical foundations of computer science 2010*, volume 6281 of *Lecture Notes in Comput. Sci.*, pages 174–185. Springer, Berlin, 2010.

[6] M. Cygan, F. V. Fomin, Ł. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. *Parameterized Algorithms*. Springer International Publishing, New York, 2016.

[7] R. Diestel. *Graph theory*, volume 173 of *Graduate Texts in Mathematics*. Springer, Heidelberg, fourth edition, 2010.

[8] F. Gavril. The intersection graphs of subtrees in trees are exactly the chordal graphs. *J. Combinatorial Theory Ser. B*, 16:47–56, 1974.

[9] J. E. Hopcroft and R. M. Karp. An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. *SIAM Journal on Computing*, 2(4):225–231, 1973.

[10] J. Jeong, S. Ok, and G. Suh. Characterizing graphs of maximum matching width at most 2. Submitted, 2016.

[11] D. König. Gráfok és mátrixok. *Matematikai és Fizikai Lapok*, 38:116–119, 1931.

[12] F. Le Gall. Powers of tensors and fast matrix multiplication. In *ISSAC 2014—Proceedings of the 39th International Symposium on Symbolic and Algebraic Computation*, pages 296–303. ACM, New York, 2014.

[13] D. Lokshtanov, D. Marx, and S. Saurabh. Known algorithms on graphs of bounded treewidth are probably optimal. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 777–789. SIAM, Philadelphia, PA, 2011.

[14] J. Nordstrand. Exploring graph parameters similar to tree-width and path-width. Master thesis, University of Bergen, June 2017.

[15] S. Oum and P. Seymour. Approximating clique-width and branch-width. *J. Combin. Theory Ser. B*, 96(4):514–528, 2006.

[16] C. Paul and J. A. Telle. Edge-maximal graphs of branchwidth k: the k-branches. *Discrete Math.*, 309(6):1467–1475, 2009.

[17] N. Robertson and P. Seymour. Graph minors. XX. Wagner's conjecture. *Journal of Combinatorial Theory, Series B*, 92(2):325–357, 2004. Special Issue Dedicated to Professor W.T. Tutte.

[18] N. Robertson and P. D. Seymour. Graph minors. X. Obstructions to tree-decomposition. *J. Combin. Theory Ser. B*, 52(2):153–190, 1991.

[19] D. J. Rose. On simple characterizations of k-trees. *Discrete Math.*, 7:317–322, 1974.

[20] S. H. Sæther and J. A. Telle. Between treewidth and clique-width. In *Graph-theoretic concepts in computer science*, volume 8747 of *Lecture Notes in Comput. Sci.*, pages 396–407. Springer, Cham, 2014.

[21] J. M. M. van Rooij, H. L. Bodlaender, and P. Rossmanith. Dynamic programming on tree decompositions using generalised fast subset convolution. In *Algorithms—ESA 2009*, volume 5757 of *Lecture Notes in Comput. Sci.*, pages 566–577. Springer, Berlin, 2009.

[22] M. Vatshelle. *New Width Parameters of Graphs*. PhD thesis, University of Bergen, 2012.