Eötvös Loránd University

Faculty of science

Sarolta Oláh

Matroid Parameters

MSc in Applied Mathematics

Supervisor:
Kristóf Bérczi

Department of Operations Research

Budapest, 2024

Acknowledgements

I would like to thank my supervisor, Kristóf Bérczi, for his dedicated time, valuable ideas, encouragement, and thorough answers. These played a crucial role in guiding me through the completion of this thesis.

Moreover, I want to express my heartfelt appreciation to my friends and family for their unwavering emotional support throughout this journey. Additionally, I am grateful to my fiancé for his constant belief in me and continuous encouragement.

Contents

Chapter 1

Introduction

Fixed-parameter tractability provides a framework for addressing computationally hard problems by concentrating on specific parameters. It allows the development of efficient algorithms, with running times that are exponential only in the size of a fixed parameter. This makes certain problems solvable even for large inputs when the parameter is small. Some successful applications are in bioinformatics, network analysis, and combinatorial optimization.

The appearance of FPT algorithms motivated the research of practical parameters in all combinatorial optimization fields. Among the various graph parameters are tree-width and tree-depth. Tree-width measures how close a graph is to being a tree, and many NP-hard problems can be solved efficiently by using dynamic programming on tree decompositions. Some examples include the traveling salesman problem, vertex cover, and coloring problems. The parameter tree-depth measures how far a graph is from being a star. It is commonly defined as the minimum height of a rooted forest whose closure contains the graph. Like tree-width, many difficult problems become solvable in polynomial time when parameterized by tree-depth, for example, the mixed Chinese postman problem.

As with many graph attributes, a natural idea is to try and generalize these types of parameters to the field of matroids. However, this generalization is not straightforward because the above definitions involve vertices, which do not have a direct counterpart in matroids. As a workaround, it is natural to consider using the concept of connectivity functions instead.

For tree-width, Robertson and Seymour [32] defined the parameter branch-width of graphs using the above-mentioned connectivity functions. They showed that branch-width and tree-width are tied for graphs. Geelen, Gerards, and Whittle [16] studied this parameter in the matroid setting. One of the many reasons why branch-width became such an important parameter is because of Hliněnỳ [20]. He showed that we can test any properties defined in the monadic second-order logic of matroids within polynomial time for matroids represented over a fixed finite field with bounded branch-width.

Many researchers defined parameters to generalize tree-depth. DeVos, Kwon, and Oum [9]

introduced the concept of branch-depth as one such generalization. Their definition led to the creation of various parameters, such as the rank-depth of a graph, achieved by substituting different types of connectivity functions.

Other depth parameters include contraction-depth, deletion-depth, and contraction-deletion depth. The first two were initially researched by Robertson and Seymour as C-type and D-type, respectively [31]. Similarly, contraction-deletion depth was examined by Ding, Oporowski, and Oxley under the name type [10]. These concepts represent the minimum number of operations required to reduce a matroid to a trivial form. Consequently, these values are closely related to the complexity of the matroid. Understanding these parameters can aid in developing more effective coding strategies in network coding and demonstrate their resilience. They also have applications in system design and computational biology.

Kardoš, Král', Liebenau, and Mach [23] introduced contraction*-depth, as another analogue of tree-depth in graphs. The last parameter we consider is contraction*-deletion depth, which was first introduced and studied in [1]. The significance of these parameters is highlighted by their connection to preconditioners in combinatorial optimization.

The structure of the thesis is the following. Chapter 2 summarizes basic notation and definitions. Chapter 3 is about the parameter branch-width, its definition, and its computational properties. Chapters 4, 5 discuss depth parameters mentioned above. Finally, Chapter 6 explains the relations between these parameters.

Chapter 2

Preliminaries

The goal of this chapter is to summarize the basic definitions and notation used in this thesis. The first section discusses **fixed-parameter tractable** algorithms, the second covers **matroids**, including their definitions and fundamental properties, and the third focuses on **connectivity functions**.

2.1 Fixed-parameter tractability

To gain a brief understanding of this topic, imagine a small kingdom with several villages connected by roads. Unfortunately, the kingdom faces difficulties, as bandits lurk on the roads, robbing unlucky travelers. The king decides to send guards to protect the villages, but since guards are costly, deploying them to every village is not a financially wise decision. A road is considered protected if at least one of its end villages is guarded. The king's task is to protect all the roads in the kingdom using as few guards as possible. The problem can be rephrased as a vertex cover problem, a basic example for fixed-parameter tractability [8, 12].

Definition 2.1. A *parameterized problem* is a language $L \subset \Sigma^* \times \mathbb{N}$, where Σ is a fixed, finite alphabet and Σ^* denotes the set of all words gained from Σ. For an instance $(x, k) \in \Sigma^* \times \mathbb{N}$, k is called the *parameter*.

Definition 2.2. A parameterized problem $L \subset \Sigma^* \times \mathbb{N}$ is called *fixed-parameter tractable* (FPT) if there exists an algorithm $\mathcal{A}$ (called a fixed-parameter algorithm), a computable function $f : \mathbb{N} \to \mathbb{N}$, and a constant c such that, given $(x, k) \in \Sigma^* \times \mathbb{N}$, the algorithm $\mathcal{A}$ correctly decides whether $(x, k) \in L$ in time bounded by $f(k) \cdot |(x, k)|^c$. The complexity class containing all fixed-parameter tractable problems is called *FPT*.

The vertex cover problem for a graph G parameterized by k is fixed-parameter tractable. The language of this parameterized problem is (G, k), so the question of whether k guards are enough to protect the kingdom can be answered in polynomial time. We can demonstrate

this by creating a rooted tree that includes all potential solutions. The root is marked with an empty set and the graph G. It will have two child nodes, each representing a choice of any edge uv in $E(G)$. One child will be labeled with u, and the other with v, as one of them must be included in a covering set. Both nodes also receive an additional label representing the parts of the graph that still need to be covered. In general, for a node labeled with the set of vertices S, we choose an edge $u'v' \in E(G)$ where neither u' nor v' is in S and create two child nodes labeled, respectively,$S \cup \{u'\}$ and$S \cup \{v'\}$. If there exists a node in the tree with a height of at most k that covers G, there is no need to continue exploring the tree.

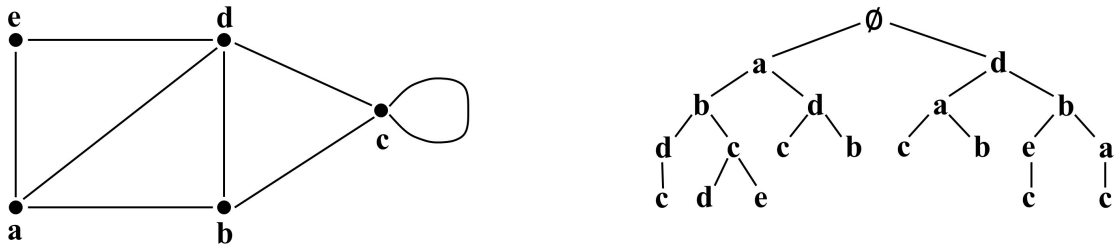


Figure 2.1: The tree on the right is the above-mentioned case separation. It shows that the graph on the left has a vertex cover of size three $\{a, c, d\}$.

2.2 Matroids

This section is devoted to matroids, using definitions from [14] and [29]. The notion of matroid was introduced by Hassler Whitney in 1933, it generalizes the concept of linear independence from vector fields to family of sets. A matroid is given by a pair $(S, \mathcal{F})$, where S represents the set of elements, referred to as the **ground-set**, and $\mathcal{F}$ contains certain subsets of S. These subsets should satisfy the next three axioms.

Definition 2.3. A set-system $M = (S, \mathcal{F})$ is called a **matroid** if it satisfies the following properties, called **independence axioms**.

- $\emptyset \in \mathcal{F}$.
- If $X \subseteq Y \in \mathcal{F}$, then $X \in \mathcal{F}$.
- For every subset $X \subseteq S$, the maximal subsets of X which are in $\mathcal{F}$ have the same cardinality.

The members of $\mathcal{F}$ are called **independent**, and the other subsets of S are called **dependent**. The maximal independent subsets of S, which by the last axiom have the same size, are the **basis**, and the minimal dependent sets are called **circuits**. A co-circuit is the minimal subset of S that intersects every basis. One can get a basis by putting independent elements greedily into

a set. In other words, a matroid is a structure where the greedy algorithm works. The **rank of** $X \subseteq S$, denoted by $r(X)$, is the size of the maximal independent set in X. The function r is the **rank function** and $r(S)$ is the rank of the matroid. An element s with $r(\{s\}) = 0$ is a **loop**, and an element s such that $r(M \setminus \{s\}) = r(M) - 1$ is a **bridge**. Two elements s and s' are **parallel** if they form a circuit, so $r(\{s, s'\}) = 1$ and neither of them is a loop. The **connected components** of a matroid are inclusion-wise maximal sets, that for every two elements of a component, there exists a circle that contains them.

Basis, rank, and circuits have special properties that can be seen as axioms, allowing the matroid to be defined by any of those. In some cases, it may be more convenient to work with matroids using these alternative definitions.

The **dual** of a matroid M, denoted by M^*, is defined on S, and the basis of M^* are the complements of the basis of M. Two matroids are considered **isomorphic** if there exists a bijection between their ground sets so that a subset is independent in the first matroid if and only if its corresponding subset is independent in the second.

The two most important matroid operations are **deletion** and **contraction**. For a set $Z \subseteq S$, $M \setminus Z$ means the **deletion** of Z from the matroid. This operation results in a new matroid with the ground-set $S \setminus Z$, where a set is independent if and only if it is independent in M. On the other hand, M/Z denotes the **contraction** of Z. In this operation, the resulting matroid has the ground-set $S \setminus Z$, and a set Z' is independent if $r(Z \cup Z') = r(Z) + r(Z')$, where Z' is independent in M. A matroid obtained through a series of deletions and contractions is referred to as a **minor**.

Since a matroid is a family of certain subsets of elements, and the number of independent sets is typically exponential, listing or storing them would pose significant challenges. Hence matroids are usually given by **oracles**. An oracle can be thought of as a black box that, given a subset as an input, outputs the rank of the given set, or decides whether the subset is independent. In matroid algorithms, complexity is measured by the number of oracle calls and additional steps.

For demonstration purposes, we provide two examples here, which will also be referenced later. Let S be a finite set of vectors over an arbitrary field $\mathbb{F}$, with $\mathcal{F}$ containing the linearly independent subsets of S. It is evident that $(S, \mathcal{F})$ forms a matroid known as a **vector** or **linear** matroid. A matroid is **representable** over a finite field $\mathbb{F}$ if there exists a vector matroid over $\mathbb{F}$ that is isomorphic to it.

Consider S as the edge set of an undirected graph, with $\mathcal{F}$ comprising subsets of edges that do not form cycles. In other words, the independent sets consist of trees and forests in the graph. This structure is referred to as a **graphic** or **circuit** matroid.

$$\begin{pmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{pmatrix}$$
$$a \quad b \quad c \quad d \quad e \quad f \quad g \quad h$$

Figure 2.2: Linear matroid

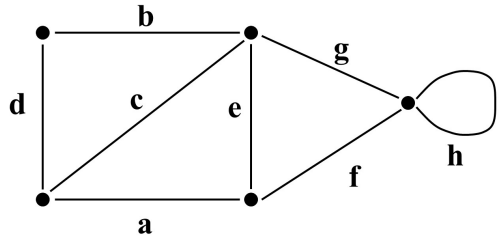


Figure 2.3: Graphic matroid

Figure 2.4: 2.2 and 2.3 are two representations of the same matroid

2.3 Connectivity functions

Let S be a finite set of elements. The next definition from [9] describes general connectivity functions.

Definition 2.4. $\lambda : 2^S \to \mathbb{Z}$ is a **connectivity function** if it satisfies the following three properties.

- $\lambda(\emptyset) = 0$,
- $\lambda(X) = \lambda(S \setminus X)\ \forall X \subseteq S$ (symmetry),
- $\lambda(X) + \lambda(Y) \geq \lambda(X \cap Y) + \lambda(X \cup Y)\ \forall X, Y \subseteq S$ (submodularity).

The following holds for all connectivity functions and some particular subsets of the ground set.

Lemma 2.5. Let λ be a connectivity function on S and let K be a subset of S such that $\lambda(K) = 0$. Then the function $\lambda|_K$ on 2^K defined by $\lambda|_K(X) = \lambda(X)$ is a connectivity function on K and $\lambda(X) = \lambda|_K(X \cap K) + \lambda|_{S \setminus K}(X \setminus K)$ for all $X \subseteq S$.

Generally, in the context of graphs, connectivity means vertex-connectivity or edge-connectivity, which means at least how many vertices or edges should one take out for the graph to fall apart. A connectivity function for a graph is defined in a slightly different manner. For $G = (V, E)$, look at the subsets of its edge set. For $X \subseteq E$, $\lambda_G(X)$ is the number of vertices, that are incident to edges from both X and $E \setminus X$ sets. This function is a connectivity function, meeting the criteria of the first two points in the definition. One approach to confirming the third point is by examining all possible cases of edge placement.

For a matroid $M = (S, \mathcal{F})$, a connectivity function $\lambda_M : 2^S \to \mathbb{Z}$ is defined as $\lambda_M(X) = r_M(X) + r_M(S \setminus X) - r_M(S)$ for $X \subseteq S$, where r denotes the rank function of M. The first two conditions are trivially satisfied, and the third arises from the submodularity of the rank function. From the definition, it can be deduced that the connectivity function of a matroid is the same as that of its dual.

Chapter 3

Branch-width

Branch-width is one of the most researched matroid parameters, initially introduced for graphs by Robertson and Seymour [32]. In their paper, they showed its close connection to tree-width. Tree-width is an important parameter measuring how close a graph is to being a tree, with great results in parameterized algorithms. Nevertheless, the challenge with tree-width lies in its complexity in generalizing to matroids, as its definition heavily relies on graph vertices. However, branch-width has a definition independent of vertices, using only connectivity functions defined in Section 2.3. Therefore, it is easy to generalize it from graphs to matroids, as shown later in Section 3.3.

First let us take a look at exact definitions from [25, 27, 30], we will define this parameter for graphs, some special functions, and matroids.

Definition 3.1. A **branch-decomposition** of H (where H can be an edge set of a graph, hypergraph, the domain of a function or a matroid) is a (T, L) pair, where T is a sub-cubic tree (all nodes have at most 3 neighbours), and L is a bijection between the elements of H, and the leaves of T. If L is solely surjective, (T, L) is a **partial branch-decomposition**.

An edge $e \in T$ splits it into two connected components. This gives a partition (E_1, E_2) in H. From this, we can specify the width of the decomposition, along with the branch-width.

Definition 3.2. For a connectivity function λ the **width of an edge** e is $\lambda(E_1) = \lambda(E_2)$, where (E_1, E_2) is the partition induced by e. For a graph $G = (V, E)$ it is $\lambda_G(E_1) = \lambda_G(E_2)$, where λ_G is the connectivity function of G defined in Section 2.3. For a matroid M, it is $\omega_T(e) = \lambda_M(E_1) + 1 = \lambda_M(E_2) + 1$, where λ_M is the connectivity function of matroids defined in Section 2.3.

Definition 3.3. The **width of T**, if T is a branch-decomposition, is the maximum edge-width for all $e \in T$. **Branch-width** is the minimum width over all branch-decompositions. Its notation for graphs, matroids and connectivity-functions is $\mathrm{bw}(G)$, $\mathrm{bw}(M)$ and $\mathrm{bw}(\lambda)$, respectively.

T is **tight**, if there is no other branch-decomposition with less branch-width.

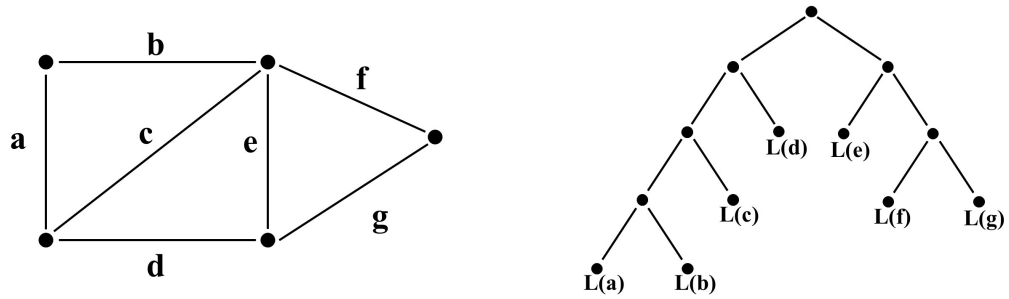


Figure 3.1: Example of a branch-decomposition. The graph on the left is viewed as a graphic matroid, the right picture shows an optimal branch-decomposition with $\mathrm{bw}(M) = 2$.

3.1 Computation

It was shown by Seymour and Thomas [33] that computing the exact number of branch-width is NP-hard even for graphs. Therefore, creating a polynomial-time algorithm for determining the branch-width of a general matroid is unfeasible. Nonetheless, this topic is still intriguing from an algorithmic point of view, as later in this section, there will be some noteworthy algorithms, using fixed parameters.

In [19], Hliněnỳ presented a polynomial-time algorithm that determines whether a matroid has a branch-width of at most 3. For a while, it was uncertain whether a polynomial-time algorithm existed for deciding if a matroid has a branch-width of at most k for any fixed k. For matroids represented over a fixed finite field, Hliněn'y [21] developed an $O(|E(\mathcal{M})|^3)$ algorithm that not only solves the problem mentioned above but also computes the exact branch-width if it is at most k. This demonstrates that branch-width is fixed-parameter tractable. The algorithm is relatively straightforward once its concepts and definitions are understood. The key concepts include parse trees, the monadic second-order logic of matroids, and finite tree automata.

A **parse tree** illustrates the step-by-step construction of a matroid along the tree. Its leaves are single-element 1**-boundaried** and loop 0**- boundaried** matroids. For an $\mathbb{F}$-represented matroid N, $\overline{N} = (N; \delta)$ is a t-boundaried matroid, if $t \geq 0$ integer and $\delta : [1, t] \to E(N)$ injective mapping, where $\delta([1, t])$ independent in N. The other vertices are $\leq t$**-boundaried composition operators**. These operators compose the boundaried matroids together, to get the matroid parsed by the tree. For precise definitions and further details, refer to [21, 22]. The following theorem 3.4 demonstrates the significance of these structures.

Theorem 3.4. *An $\mathbb{F}$-represented matroid M has branch-width at most $t + 1$ if and only if M is parsed by some spanning t-boundaried parse tree.*

In [21] there is a 3-approximation algorithm computing a spanning boundaried parse tree of a matroid, in $O(n^3)$ running time. The second significant definition is the following.

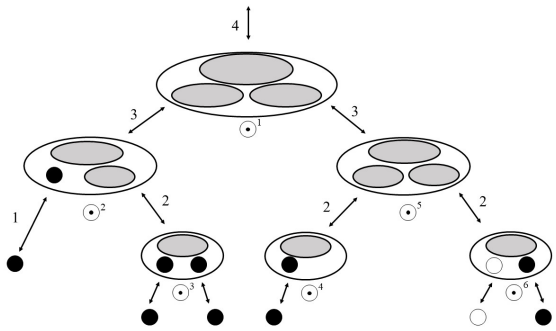


Figure 3.2: Here is an illustration of a parse tree. The numbers along the edges indicate the boundary ranks, while the ovals symbolize the composition operators. The filled circles represent the 1-boundaried matroids, while the empty circles denote the 0-boundaried ones [21].

Definition 3.5. The **monadic second-order logic of matroids (MSOL)** includes variables representing matroid elements and sets of elements, as well as the quantifiers $\forall$ and $\exists$, and logical connectives $\wedge$, $\vee$, and $\neg$. In MSOL, the following three questions can be easily determined:

- equality for elements and their sets,
- whether an element is in a set,
- whether a set is independent in a matroid, hence MSOL uses the independence oracle.

The importance of MSOL comes from graph theory. Courcelle [5, 6] showed, that problems expressible in MSOL can be efficiently solved for graphs with bounded width. Similar to numerous other graph properties, this characteristic can also be extended to matroids. Hliněný [21] also proves that all matroid properties expressible in the monadic second-order logic are uniformly fixed-parameter tractable for represented matroids bounded by branch-width.

A **finite tree automaton** is a machine designed to process tree structures. It can accept or reject a tree based on its purpose and the attributes of the trees it evaluates.

Theorem 3.6. *Let $t \geq 1$ and let $\mathbb{F}$ be a fixed finite field. Assume $\mathcal{M}$ is a set of represented matroids over $\mathbb{F}$ described by a sentence in the monadic second-order logic of matroids. Then there is a finite tree automaton accepting exactly the $\leq (t-1)$-boundaried parse trees of members of $\mathcal{M}$ (of branch-width bounded by t).*

The proof for Theorem 3.6 can be found in [22]. This proof is algorithmic, it includes a method for computing the tree automaton that accepts boundaried parse trees.

Lemma 3.7. For every matroid N there is an MSOL formula ψ_N such that ψ_N is true on matroid M, if and only if N is a minor of M.

This implies that the class of matroids with branch-width at most k is closed under taking minors, a significant property in matroid theory. Consequently, membership can be determined by identifying the excluded minors, which are the minor-minimal matroids that do not belong to the family. For many matroid classes, the number of excluded minors is finite. Specifically, for matroids with bounded branch-width, Geelen, Gerards, and Robertson proved in [15] that the excluded minors have a size of at most $(6^{k+1} - 1)/5$. Using a brute force search, all such minors can be found.

The proof of Lemma 3.7 constructs a formula of this kind, using the elements of the minor as variables. Consequently, Corollary 3.8 naturally follows.

Corollary 3.8. For every $k \geq 1$, there is a computable MSOL formula ϕ_k, such that ϕ_k is true in M if and only if M has branch-width at most k.

Following these theoretical introductions, presented below is the primary theorem of the paper [21].

Theorem 3.9. *Let $\mathbb{F}$ be a finite field, and let $t \geq 1$ be a constant. There is an algorithm that, given a rank-r matrix $\boldsymbol{A} \in \mathbb{F}^{r \times n}$ such that the branch-width of the matroid $M(A)$ is at most $t + 1$, finds the exact branch-width of $M(A)$ in time $O(n^3)$.*

Proof. The proof is the algorithm itself.

- Pre-computation: for $k = 2, \ldots, t+1$ compute the excluded minors of the class of matroids, that has branch-width at most k, and the formula Φ_k from Corollary 3.8. Then compute the finite tree-automaton portrayed in Theorem 3.6, which accepts only the parse trees described by Φ_k.

- Let $M(A)$ be the input n-element matroid represented over a finite field. Algorithm 4.1 from [21] computes an $\leq 3t$-boundaried parse tree T of $M(A)$. From this, find the smallest k_0 that the finite tree-automaton $\mathcal{A}_{k_0}$ accepts T, $k_0 \leq t$. This k_0 is the branch-width. □

Next, another method for computing branch-width is presented. From the definition of branch-width 3.3, it follows that searching for the branch-width of a matroid is equivalent to searching for the branch-width of its connectivity function. The algorithm below, for the branch-width of connectivity functions, is presented by Oum and Seymour in [28]. They approach the search for branch-width from a different perspective by attempting to identify the dual object known as a **tangle**.

Definition 3.10. Let V be a finite set, and let f be a connectivity-function on 2^V. A set Γ of subsets of V is called an f **-tangle** of order $k + 1$ if it satisfies the next three axioms.

- $\forall\, A \subseteq V$, if $f(A) \leq k$, then either $A \in \Gamma$ or $V \setminus A \in \Gamma$,

- if $A, B, C \in \Gamma$, then $A \cup B \cup C \neq V$,
- $\forall\, v \in V$, we have $V \setminus \{v\} \notin \Gamma$.

Robertson and Seymour [32] showed that there is a connection between tangles and branch-width, thus making them each other's dual object.

Theorem 3.11. *Let f be a connectivity function on 2^V. There is no f-tangle of order $k+1$ if and only if the branch-width of f is at most k.*

Oum and Seymour introduced definitions of loose tangles and loose tangle kits, which initially seem weaker but ultimately prove to offer the same benefits as tangles.

Definition 3.12. A set Γ of subsets of V is a **loose f-tangle** of order $k+1$, if it satisfies the next three axioms.

- For a subset $X \subseteq V$, if $|X| \leq 1$ and $f(X) \leq k$, then $X \in \Gamma$,
- if $A, B \in \Gamma$, $C \subseteq A \cup B$, and $f(C) \leq k$, then $C \in \Gamma$,
- $V \notin \Gamma$.

Let $f_{\min}(A, B) = \min_{A \subseteq Z \subseteq V \setminus B} f(Z)$ for an f connectivity function, where $A.B \subseteq V$ disjoint subsets.

Definition 3.13. A pair (P, μ) is a **loose f-tangle kit** of order $k+1$, if $P = \{(A, B) : A, B \subseteq V, A \cap B = \emptyset, \max(|A|, |B|) \leq f_{\min}(A, B) \leq k\}$ and $\mu : P \to 2^V$ is a function satisfying the next three axioms.

- If $|X| \leq 1$ and $f(X) \leq k$, then there exists $(A, B) \in P$ such that $A \subseteq X \subseteq V \setminus B$, $f(X) = f_{\min}(A, B)$, and $X \subseteq \mu(A, B)$,
- if $(A, B), (C, D), (E, F) \in P, E \subseteq X \subseteq (\mu(A, B) \cup \mu(C, D)) \setminus F$, and $f(X) = f_{\min}(E, F)$, then $X \subseteq \mu(E, F)$.
- $\mu(\emptyset, \emptyset) \neq V$ if $(\emptyset, \emptyset) \in P$.

It is stated in [28], that a loose f-tangle kit of order $k+1$ exists if and only if an f-tangle kit of order $k+1$ exists, therefore, the following theorem is also applicable to loose tangle kits.

Theorem 3.14. *Let f be a connectivity function on 2^V. Then, no loose f-tangle of order $k+1$ exists if and only if the branch-width of f is at most k.*

The first algorithm uses loose tangle kits to decide whether the branch-width of a given matroid is at most k. The idea is to try and construct a loose tangle kit of order $k+1$, if it succeeds, then the branch-width is more than k, if it fails, the branch-width is at most k.

Algorithm 1.

1. Create $P = \{(A, B) : A, B \subseteq V, A \cap B = \emptyset, \max(|A|, |B|) \leq f_{\min}(A, B) \leq k\}$.

2. If $(\emptyset, \emptyset) \in P$, let $\mu(\emptyset, \emptyset) = \{v \in V : f(\{v\}) = 0\}$. For every $v \in V, if 0 < f(\{v\}) \leq k$, find a subset $B \subseteq V \setminus \{v\}$ such that $|B| \leq f_{\min}(\{v\}, B) = f(\{v\})$, then let $\mu(\{v\}, B) = \{v\}$. For other $(A, B) \in P$, let $\mu(A, B) = \emptyset$.

3. For the (P, μ) pair, test the third property of loose tangle kits from definition 3.13. If it fails, then there is no loose f-tangle kit of order $k + 1$. STOP. The branch-width is at most k.

4. Test the second property from definition 3.13. If it fails, then we have $(A, B), (C, D), (E, F) \in P$ and X such that $E \subseteq X \subseteq (\mu(A, B) \cup \mu(C, D)) \setminus F$, $f(X) = f_{\min}(E, F)$, and $X \nsubseteq \mu(E, F)$. So let's make $\mu(E, F)$ to be $\mu(E, F) \cup X$, thus increasing $|\mu(E, F)|$ at least by 1. Now go back to step **3.**

5. (P, μ) is a loose f-tangle kit of order $k + 1$. STOP. The branch-width of f is more than k.

The running time of the algorithm is $O(\gamma n^{8k+6} \log n)$, where γ is the computation time of $f(X)$ for any $X \subseteq V$, k is constant and $n = |V|$.

The second algorithm constructs a branch-decomposition f width at most k, if it exists. It uses the the first algorithm as a black box.

Algorithm 2.

1. If $|V| < 1$, then no branch-decomposition exists. For $|V| = 1$, there exists the trivial decomposition. If $|V| = 2$, then there is a unique branch-decomposition, its width is determined by f. If $f(\{v\}) > k$ for a $v \in V$, then branch-width is larger than k. STOP.

2. Find a pair $\{u, v\}$, where $u, v \in V$, such that branch-width of f/uv is at most k with Algorithm 3.1.

3. If there is no such pair, the branch-width of f is larger than k. STOP.

4. Obtain a branch-decomposition (T', L') of f/uv of width at most k by recursively calling this algorithm.

5. Extend (T', L') to a branch-decomposition (T, L) by attaching two leaves u_T and v_T to the leaf $L'(uv)$ of T', then set $L(u) = u_T$ and $L(v) = v_T$.

Algorithm 3.1 has running time $O(n^3\mathcal{A})$, where $\mathcal{A}$ denotes the running time of the first algorithm 3.1.

3.2 Approximation

In [27], Sang-il Oum and Paul Seymour developed an algorithm that approximates the branch-decomposition of certain submodular functions. One application of this algorithm is an $O(n^{3.5})$ time approximation which, for a fixed k and an input n-element matroid, provides a branch-decomposition of at most $3k - 1$ or outputs a witness that the branch-width is greater than k. In [13], Fomin and Korhonen made a 2-approximating FPT algorithm for the same problem; however, their work will not be presented here.

To understand the algorithm, we must first define the interpolation of submodular functions.

Definition 3.15. Let $f : 2^V \to \mathbb{Z}$ be a submodular function such that $f(\emptyset) \leq f(X)$ for all $X \subseteq V$. $f^* : 3^V \to \mathbb{Z}$ is an **interpolation** of f if it satisfies the following:

- $f^*(X, V \setminus X) = f(X)$ for all $X \subseteq V$.
- If $C \cap D = \emptyset$, $A \subseteq C$, and $B \subseteq D$, then $f^*(A, B) \leq f^*(C, D)$.
- $f^*(A, B) + f^*(C, D) \leq f^*(A \cap C, B \cup D) + f^*(A \cup C, B \cap D)$ for all $(A, B), (C, D) \in 3^V$,
- $f^*(\emptyset, \emptyset) = f(\emptyset)$.

The algorithm uses the following properties of interpolations, assuming that $\emptyset$ is the minimizer of f.

Proposition 3.16. *Let $f : 2^V \to \mathbb{Z}$ be a submodular function, and $f(\emptyset) \leq f(X)$ for all $X \subseteq V$, and let $f^* : 3^V \to \mathbb{Z}$ be an interpolation of f. Then:*

- *For all $(X, Y) \in 3^V$, the interpolation $f^*(X, Y) \leq \min_{X \subseteq Z \subseteq V \setminus Y} f(Z)$.*
- *$f^*(\emptyset, Y) = f(\emptyset)$ for all $Y \subseteq V$.*
- *If $f(\{v\}) - f(\emptyset) \leq 1$ for every $v \in V$, then for every fixed $B \subseteq V$, $f^*(X, B) - f(\emptyset)$ is the rank function of a matroid on $V \setminus B$.*

Proposition 3.17. *Let $f : 2^V \to \mathbb{Z}$ be a submodular function such that $f(\emptyset) \leq f(X)$ for all $X \subseteq V$. Then $f_{\min}$ is an interpolation of f, where $f_{\min}(X, Y) = \min_{X \subseteq Z \subseteq V \setminus Y} f(Z)$.*

Generally, there are multiple interpolations for a submodular function. Furthermore, if there is a submodular function a $f^* : 3^V \to \mathbb{Z}$ that satisfies the second point from Proposition 3.16, then there exists a submodular function $f : 2^V \to \mathbb{Z}$ such that $f(\emptyset) \leq f(X)$ and f^* is an interpolation of f.

The main theorems in [27] use the concept of well-linkedness. The witness demonstrating that the branch-width exceeds k will be a well-linked set.

Definition 3.18. Let V be a finite set and let $f : 2^V \to \mathbb{Z}$ be a symmetric submodular function satisfying $f(\emptyset) = 0$. A subset $W \subseteq V$ is **well-linked** with respect to f if for every partition (X, Y) of W and every Z with $X \subseteq Z \subseteq V \setminus Y$, the inequality $f(Z) \geq \min(|X|, |Y|)$ is fulfilled.

Theorem 3.19. *Let V be a finite set with $|V| \geq 2$, and let $f : 2^V \to \mathbb{Z}$ be a symmetric submodular function such that $f(\emptyset) = 0$. If with respect to f there is a well-linked set of size k, then* $\mathrm{bw}(f) \geq k/3$.

Proof. Let (T, L) be a branch-decomposition, and W a well-linked set of size k. By contracting the incident edges, we can assume that T does not have any degree-two vertices. This assumption does not affect the branch-width. For each edge uv, let X_u be the component that contains u after deleting the edge uv in T. The subsets $A_{uv} = L^{-1}(X_u)$ and $B_{uv} = V \setminus A_{uv}$.

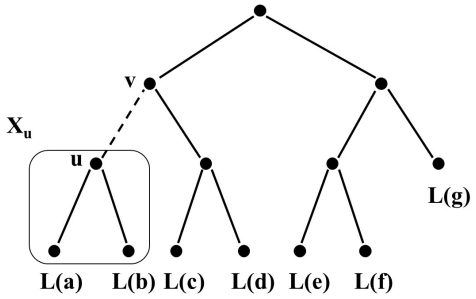


Figure 3.3: Image of the components in a branch-decomposition. Here $A_{uv} = \{a, b\}$.

From $f(\emptyset) = 0$, we may assume that $W \neq \emptyset$. The value $f(\{w\}) \geq 1$, since W is well-linked with respect to f and from a $(\{w\}, W \setminus \{w\})$ partition $f\{w\} \geq \min(|\{w\}|, |W \setminus \{w\}|) = 1$. Thus, the decomposition (T, L) has a width of at least 1. If $k \leq 3$, then $\mathrm{bw}(f) \geq 1 \geq k/3$ satisfies the statement of the theorem. Therefore, we can assume $k > 3$.

To prove the theorem, we need to show that $f(A_{uv}) \geq \min(|A_{uv} \cap W|, |B_{uv} \cap W|) \geq k/3$. Suppose that $\min(|A_{uv} \cap W|, |B_{uv} \cap W|) < k/3$ for all uv edges in T. Construct a directed graph, by orienting all uv edges from u to v, if $|A_{uv} \cap W| < k/3$ and $|B_{uv} \cap W| \geq k/3$.

Now all edges are oriented, because $|W| = k$. Since there are more vertices in the tree than edges, there exists a node $t \in V(T)$ such that every edge incident to it has head t.

If t is a leaf, then let s be its neighbour. Due to the orientation $|B_{st} \cap W| \geq k/3$, but $|B_{st}| = 1$, which is a contradiction.

If t is not a leaf, then it has 3 neighbours (x, y, z) by the assumption from the beginning of the proof. Since t is the head of all its edges, $|A_{xt} \cap W| < k/3$, $|A_{yt} \cap W| < k/3$, and $|A_{zt} \cap W| < k/3$. Therefore $|A_{xt} \cap W| + |A_{yt} \cap W| + |A_{zt} \cap W| < k = |W|$, which is a contradiction.

Thus, there exists an edge uv with $\min(|A_{uv} \cap W|, |B_{uv} \cap W|) \geq k/3$, which implies that $f(A_{uv}) \geq k/3$, and the width of (T, L) is at least $k/3$ in any arbitrary decomposition. □

Before the second theorem, take a look at the following definition.

Definition 3.20. A partial branch-decomposition (T, L) **extends** another (T', L') if T' is obtained by contracting some edges of T and for every $v \in V$ $L'(v)$ in T' corresponds to the vertex $L(v)$ after the contraction.

Theorem 3.21. *Let V be a finite set, let $f : 2^V \to \mathbb{Z}$ be a symmetric submodular function such that $f(\{v\}) \leq 1$ for all $v \in V$ and $f(\emptyset) = 0$, and let $k \geq 0$ be an integer. If there is no well-linked set of size k with respect to f, then* $\mathrm{bw}(f) \leq k$.

Proof. We may assume, that $\mathrm{bw}(f) > 0$, $k > 0$ and $|V| \geq 2$. Consider (T_s, L_s) as a partial branch-decomposition of width at most k. Such a decomposition surely exists because we can take the trivial decomposition 3.4 which has a width at most 1.

Let (T, L) be a partial branch-decomposition, which extends (T_s, L_s). The width of (T, L) is at most k, and the number of leaves is maximal. The proof will show that if there is no well-linked set of size k, then (T, L) is a branch-decomposition, so L is a bijection.

Suppose, that L is not a bijection, so there is a leaf $t \in V(T)$, such that $B = L^{-1}(T)$ has more than one elements. Let $f(B) < k$, $v \in B$ then let's construct another partial decomposition (T', L'), by adding two vertices t_1, t_2, and edge tt_1, tt_2.

Figure 3.4: The figures illustrate the step-by-step process of constructing the partial decomposition to achieve a complete decomposition.

Let $L'(v) = t_1$ and $L'(B \setminus \{v\}) = t_2$. Here (T', L') extends (T, L). From the assumption in the theorem $f(B \setminus \{v\}) + f(\emptyset) \leq f(B) + f(\{v\}) \leq k$. This implies that (T', L') is a partial branch-decomposition, with branch-width at most k and more leaf than (T, L). This leads to a contradiction. Thus, we conclude that $f(B) = k$.

Let f^* be an interpolation of f, the third point of Proporition 3.16 states that $f^*(X, B)$ is the rank function of the matroid on $V \setminus B$. Let X be a base, then $|X| = f^*(V \setminus B, B) = f(V \setminus B) = f(B) = k$, from Definition 3.15 and the condition that f is symmetric. There exists a set $Z \subseteq V$, with $f(Z) < \min(|Z \cap X|, |Z \cap (V \setminus X)|)$ given that X is not well-linked. $Z \cap B$ is not empty, as a result of $f(Z \setminus B) = f^*(Z \setminus B, V \setminus (Z \setminus B)) = f^*(Z \setminus B, B \cup (V \setminus Z)) \geq f^*(Z \cap X, B) = |X \cap Z| > f(Z)$, since f^* is uniform. It can be shown similarly that $B \setminus Z = (V \setminus Z) \cap B = \emptyset$. Add two vertices t_1, t_2, and two edges tt_1, tt_2 to the tree T, to get another subcubic tree T'. Let

$L'(v) = t_1$ if $v \in B$, $L'(v) = t_2$ if $v \in B \setminus Z$. From the attributes of f and f^* we got

$$\begin{aligned}
&|(V \setminus Z) \cap X| + f(B) > f(Z) + f(B) \geq \\
&\geq f(Z \cup B) + f(Z \cap B) = f((V \setminus Z) \setminus B) + f(Z \cap B) = \\
&= f^*((V \setminus Z) \setminus B, B) + f(Z \cap B) \geq f^*((V \setminus Z) \cap X, B) + f(Z \cap B) = \\
&= |(V \setminus Z) \cap X| + f(Z \cap B)
\end{aligned}$$

So $f(Z \cap B) < f(B) \leq k$ and similarly $f(B \setminus Z) < f(B) \leq k$. Hence, (T', L') forms a partial branch-decomposition, which extends (T, L), with a branch-width at most k, and T' has more leaves than T, which contradicts our assumption. □

The proof of Theorem 3.21 offers an algorithm capable of either identifying a well-linked set of size k or constructing a branch-decomposition with a width of at most k. Let $t = k/3$, Theorem 3.19 and Theorem 3.21 combined either concludes, that $\mathrm{bw}(f) > t$, or finds a decomposition with width at most $3t + 1$.

Analyzing the running time involves considering two cases. The first case arises when only the function f is provided as an input, and the interpolation function f^* needs to be computed. In this case let $f_{\min}$ be the interpolation, and use the submodular function minimization algorithm for calculating its value. This computation can be completed in $O(n^5 \gamma \log n)$ time, where γ represents the computation time of $f(X)$ for a subset X. The overall time complexity, with finding a base, checking well-linkedness, etc. is $O(n^7 \gamma \log n)$.

Another case is where both the function f and f^* are provided as input. The overall time complexity for this case is $O(n^6 \delta \log n)$, where δ represents the time required to compute $f^*(X)$ for a subset X.

Applying this algorithm with small modifications for the matroid's connectivity function decreases the general running time. Let $M = (S, r)$ be a matroid, S is the ground-set, and r is the rank function. There is an interpolation of λ_M, which can be computed faster, than $\lambda_{\min}$. This interpolation is $\lambda_B(X, Y) = r(X \cup (B \setminus Y)) + r(Y \cup (B \setminus X)) - |B \setminus X| - |B \setminus Y| + 1$, where B is a base of matroid M.

The original algorithm uses submodular function minimization not only to find an interpolation of λ, but also at another instance. It searches for a set Z for two disjoint sets X, Y, such that $X \subseteq Z \subseteq S \setminus Y$ and $\lambda(Z)$ is minimized. This process involves matroid intersection of $M_1 = M/X \setminus Y$ and $M_2 = M \setminus X/Y$, with rank functions r_1, r_2, respectively. If M is given by the rank oracle, the running time is $O(n^{2.5})$, making the overall running time $O(n^{3.5})$, assuming that oracle calls take unit time.

3.3 Connection to graph parameters

The branch-width of a graph and the branch-width of its cycle matroid is not always the same. The question is, is there a way to rule out those cases where they differ? Mazoit and Thomassé in [25] proved that the branch-width of a bridgeless graph is equal to the branch-width of its cycle matroid. Their theorem will be presented later on in this section, it utilizes hypergraphs instead of graphs. In hypergraphs, the branch-width is defined similarly to that in graphs. Additionally, Hicks and McMurray in [18] showed that if a graph contains a cycle of length at least 2, then the branch-width of the graph and of its cycle matroid is the same.

Figure 3.1 illustrates a graph G where the branch-width matches that of its corresponding graphic matroid. Conversely, the following example illustrates the opposite.

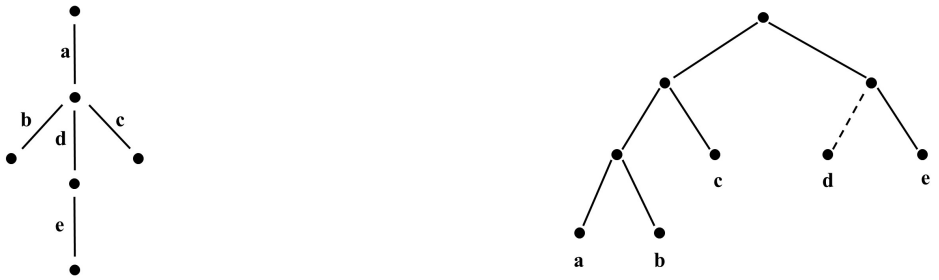


Figure 3.5: Here is an example demonstrating that the branch-width of a graphic matroid does not always match the branch-width of the graph. In the illustration, it is evident that the tree on the right serves as an optimal branch-decomposition for the tree on the left. The dashed edge has the maximum edge-width, indicating that $\mathrm{bw}(G) = 2$. However, if you consider the tree as a graphic matroid, all edges have a width 0, resulting in a branch-width of 1.

The branch-width of a graph is always at least as large as the branch-width of its cycle matroid. To demonstrate this, consider a hypergraph $H = (V, E)$ its cycle matroid M_H. A component of E is the smallest nonempty subset $C \subseteq E$ such that $\lambda_H(C) = \emptyset$, where $\lambda_H(C)$ represents the number of vertices with incident edges from both C and $V \setminus C$. This is the connectivity function for graphs defined in Definition 2.4. For a subset $X \subseteq E$ let $c(X)$ denote the number of components in the subhypergraph spanned by X. A hypergraph is considered connected, if $c(E) = 1$, and bridgeless if $E \setminus e$ is connected for all $e \in E$.

Consider a partition (E_1, E_2), and let n_1, n_2, and n denote the number of vertices spanned in E_1, E_2, and E, respectively. Since in a graphic matroid, the independent subsets are forests and trees, the rank of a subset $X \subseteq E$ corresponds to the size of a maximal forest. Hence:

$$\lambda_{M_H}(E_1) = \lambda_{M_H}(E_2) = r(E_1) + r(E_2) - r(E) + 1 = n_1 - c(E_1) + n_2 - c(E_2) - n + c(E) + 1 =$$
$$= \lambda_H(E_1) + c(E) + 1 - c(E_1) - c(E_2) \leq \lambda_H(E_1).$$

As $\lambda_H(E_1) = \lambda_H(E_2) = n_1 + n_2 - n$. Equality only holds in the above inequality when both E_1 and E_2 are connected.

The main result of [25] demonstrates that if H is connected and bridgeless, then there exists a branch-decomposition with the same width as the branch-width of M_H. Additionally, in every edge-induced partition (E_1, E_2) the sets E_1 and E_2 are connected.

Lemma 3.22. If T is tight, every edge-induced partition (E_1, E_2) is such that either E_1 or E_2 is connected.

Theorem 3.23. *For every branch-decomposition T of a connected hypergraph H, there exists a tighter branch-decomposition T' such that for every edge-induced partition (E_1, E_2) with $c(E_1) > 1$, E_1 consists of components of $H \setminus e$, for some $e \in E_2$. In particular, if H is bridgeless, it has an optimal connected branch-decomposition.*

Chapter 4

Branch-depth

Branch-depth is a concept that generalizes the tree-depth of graphs. While tree-width measures how close a graph is to being a tree, tree-depth measures how close it is to being a star. The challenge in generalizing this parameter from graphs to matroids is similar to the one for branch-width discussed in Chapter 3. The original definitions rely on vertices, which cannot be directly applied to matroids. The solution, as with branch-width, is to use a definition that relies solely on connectivity functions.

Matt DeVos, O-joung Kwon, and Sang-il Oum addressed this problem [9], defining the branch-depth of a connectivity function. The branch-depth of a matroid and a graph is determined by the branch-depth of their connectivity functions as outlined in Section 2.3. The definition of DeVos et al. is the following.

Definition 4.1. Let S be a finite set of elements. A **depth-decomposition** of a connectivity function $\lambda : 2^S \to \mathbb{Z}$ is a (T, L) pair, where T is a tree with at least one internal node, which is a node that has child nodes.

A key difference between this definition and that of a branch-decomposition is that, in this case, the tree does not need to be sub-cubic. However, since an internal node is required, a decomposition consisting of just one node does not qualify as a depth-decomposition. Consequently, if the ground set S has fewer than two elements, no branch-decomposition exists. In such instances, the branch-depth of λ is defined 0.

The **radius** of a (T, L) decomposition is the radius of the tree T. It is the smallest number r, such that there exists a node within a distance of r from every other node.

Definition 4.2. Let (T, L) be a decomposition of a connectivity function λ. For an internal node $v \in V(T)$, the connected components of the graph $T \setminus \{v\}$ give a partition $\mathcal{P}_v$ on E by L. The **width of** v is defined to be $\lambda(\mathcal{P}_v)$, where $\lambda(\mathcal{P}_v) = \max_{\mathcal{P} \subseteq \mathcal{P}_v} \lambda_M \left(\bigcup_{X \in \mathcal{P}} X\right)$. The **width of the decomposition** (T, L) is the maximum width of an internal node of T. The decomposition (T, L) is a (k, r)-decomposition of λ if the width is at most k and the radius is at most r. The **branch-**

depth of λ, denoted by $\mathrm{bd}(\lambda)$, is the minimum k such that there exists a (k, k)-decomposition of λ.

The most notable difference between this definition and the definition of branch-width is that, in this case, we use the vertices of the tree to create the components rather than the edges. As a result, a partition can contain multiple subsets, not just two.

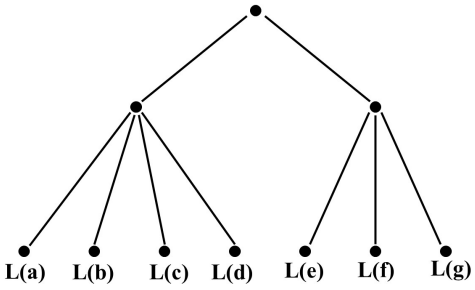


Figure 4.1: An example of an optimal depth-decomposition for the matroid in Figure 3.1. This is a $(2, 2)$ decomposition, as the root has a distance of at most 2 from every node, and $\lambda(\mathcal{P}_v)$ is at most 2 for every node in the tree.

4.1 Computation

Before any computation, Devos, Kwon, and Oum in [9], and Gollin, Hendrey, Mayhew, and Oum in [17], present certain properties of matroids that aid in establishing bounds on branch-depth, or the branch-depth of their minors.

Theorem 4.3. *Let $k \geq 1$. If a matroid M has no circuits of size more than k or no cocircuits of size more than k, then the branch-depth of M is at most $\frac{1}{2}k(k+1)$.*

Theorem 4.3 is a corollary of Theorem 5.11 by using the connection between branch-depth and contraction-deletion depth Theorem 6.1.

Lemma 4.4. Let k be a non-negative integer. Let $M = (S, \mathcal{F})$ be a matroid of branch-depth k and let X, Y be disjoint subsets of S such that $X \cup Y \neq \emptyset$. Then $M \setminus X/Y$ has a component of branch-depth at least $k - |X| - |Y|$.

Let S be a finite set of elements and $\lambda : 2^S \to \mathbb{Z}$ be a connectivity function. The following two lemmas assist in finding decompositions of λ based on partitions.

Lemma 4.5. Let λ be a connectivity function on E and let K be a subset of E such that $\lambda(K) = 0$. Let k, r_1, r_2 be integers such that $\lambda|_K$ has a (k, r_1)-decomposition and $\lambda|_{E\setminus K}$ has a (k, r_2)-decomposition.

- If $r_1 \neq r_2$, then λ has a $(k, \max\{r_1, r_2\})$-decomposition.
- If $r_1 = r_2$, then λ has a $(k, r_1 + 1)$-decomposition.

Proof. Let's assume that $r_1 \geq r_2$, and let (T_1, L_1) be a (k, r_1)-decomposition of $\lambda|_K$. Let v_1 be an internal node of T_1 such that each node of T_1 is within distance radius r_1 from v_1. Similarly, let (T_2, L_2) be a (k, r_2)-decomposition of $\lambda_{E\setminus K}$ and v_2 an internal node of T_2 such that each node of T_2 is within the radius r_2 from v_2. Let T be the tree obtained from the disjoint union of T_1 and T_2 by adding an edge between v_1 and v_2.

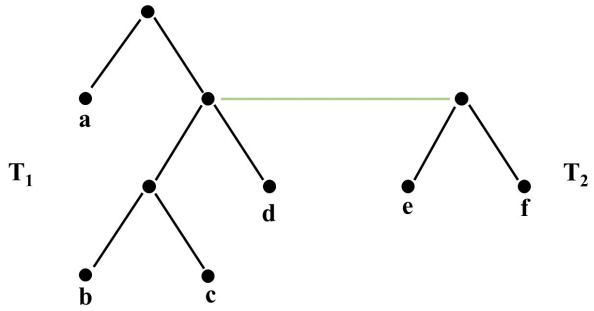


Figure 4.2: Here is an example of the tree resulting from a disjoint union. The new edge is highlighted in green, connecting the two nodes in T_1 and T_2, each of which has a distance of 2 and 1 from every other node, respectively.

Then T with L_1 and L_2 forms a decomposition (T, L) of λ. The function λ is the disjoint union of λ_1 and λ_2, then by using the lemma of connectivity functions 2.5, (T, L) has a width at most k. Look at the radius of T. If $r_2 < r_1$, then it is at most r_1 and consequently (T, L) is a (k, r_1)-decomposition. If $r_2 = r_1$, then the radius of T is at most $r_1 + 1$ so (T, L) is a $(k, r_1 + 1)$-decomposition. □

The second Lemma 4.6 is a generalization of the first Lemma 4.5 for m-partitions.

Lemma 4.6. Let λ be a connectivity function on S. Let $S_1, S_2, ..., S_m$ be a partition of S into non-empty sets such that $\lambda(S_i) = 0$ for all $1 \leq i \leq m$. Let $\lambda_i := \lambda|_{S_i}$ and k_i be the branch-depth of λ_i. Let $k = \max\{k_1, k_2, ..., k_m\}$. Then the branch-depth of λ is k or $k + 1$. In particular, if the branch-depth of λ is $k + 1$, then there exist $i < j$ such that $k_i = k_j = k$ and λ has a $(k, k + 1)$-decomposition.

Proof. We can prove this by induction on m, assuming $m \geq 2$. The branch-depth of λ is at least k, as it's at least the branch-depth of λ_i for every i when we consider subtrees. If $k_i = 0$ for all i, then each $|S_i| = 1$. In this case, the branch-depth of λ is 1 because we can create a $(0, 1)$-decomposition (T, L) where T consists of a root and m leaves. Now, assume that $\max_i k_i$ is positive, and let's say it's k_1. Let (T_1, L_1) be a (k, k)-decomposition of $\lambda|_{S_1}$, and let v_1 be an

internal node of T_1 such that each node of T_1 is within distance k from v_1. If $|S_2| = 1$, we attach a leaf to v_1 corresponding to the element of S_2. This addition gives a (k, k)-decomposition of $\lambda|_{S_1 \cup S_2}$, fulfilling the lemma's statement for $S_1 \cup S_2, S_3, \ldots, S_m$ by induction. Therefore, we may assume that $|S_2|, |S_3|, \ldots, |S_m| \geq 2$, implying $k_2, k_3, \ldots, k_m \geq 1$. We can then apply Lemma 4.5 repeatedly for these cases. □

Lemma 4.7 follows as a corollary from Lemma 4.6.

Lemma 4.7. Let M be a matroid. Let k be the maximum branch-depth of the components of M. Then the branch-depth of M is k or $k + 1$. In particular, if M has at most one component having branch-depth exactly k, then the branch-depth of M is equal to k.

In Lemma 4.5 and Lemma 4.6 we saw how to find a decomposition for matroids with special partitions. The following lemma makes an upper bound of branch-depth based on the matroid's partitions.

Lemma 4.8. Let m and k be non-negative integers, let $M = (S, \mathcal{F})$ be a matroid, and let N_1 and N_2 be minors of M such that $(S(N_1); S(N_2))$ is a partition of S and $\lambda_M(S(N_1)) \leq k$. If all components of both N_1 and N_2 have branch-depth at most m, then M has branch-depth at most $max\{m + k; m + 2\}$.

The computation of branch-depth is closely tied to its relationship with branch-width. If the branch-depth of a matroid is at most k, then in consequence its branch-width is at most k. To determine whether the depth is at most k, we require the following proposition from [9], which highlights a useful parameter trait concerning matroids.

Proposition 4.9. *If N is a minor of a matroid M, then the branch-depth of N is less than or equal to the branch-depth of M.*

Proof. Assume that N has at least two elements, and assume that $|S(M)| > |S(N)|$. Let k be the branch-depth of M, and (T, L) be a (k, k)-decomposition. Let T' be a minimal subtree of T, which contains all $L(v)$ leaves, where $v \in S(N)$. Let L' be the restriction of L on the set of leaves of T'. Since $|S(M)| > |S(N)| \geq 2$, by assumption, T' must have at least one internal node. T' is a subtree of T, hence the radius of it is at most k.

From Lemma 2.5 for all $X \subseteq S(M)$, $\lambda|_N(X \cap S(N)) \leq \lambda_M(X)$, so the width of (T', L') is at most the width of (T, L), k. As a consequence, (T', L') is a (k, k)-decomposition of N. □

In essence, Proposition 4.9 implies that the class of matroids with branch-depth at most k is minor-closed. This enabled Matt DeVos, O-joung Kwon, and Sang-il Oum [9] to straightforwardly prove the following corollary algorithmically.

Corollary 4.10. For each fixed finite field $\mathbb{F}$ and an integer k, we can decide in time $O(n^3)$ whether the input n-element rank-r matroid represented by an $r \times n$ matrix over $\mathbb{F}$ has branch-depth at most k.

Proof. Initially, to discover a branch-decomposition with a branch-width no greater than k, we can use the algorithm proposed by Hliněný [21], detailed earlier in Section 3.1, with a time complexity of $O(n^3)$. Given the connection between branch-depth and branch-width, if no such decomposition exists, it implies that the branch-depth exceeds k. Conversely, if the algorithm does discover such a decomposition, then we can proceed to use the method outlined in [22] to ascertain whether the input matroid has a minor isomorphic to a fixed matroid. □

4.2 Connection to graph parameters

The parameters branch-depth and tree-depth are tied for graphs. For a connected graph, the branch-depth is at most its tree-depth, as will be demonstrated later in this section. However, in general, these two parameters diverge for a graph's cycle matroid, as illustrated by the following figure.

Figure 4.3: P_n serves as a typical example where the matroid branch-depth and tree-depth differ. While its tree-depth is $\lceil \log_2(n+1) \rceil$, as demonstrated in [26], its branch-depth remains 1.

However, certain restrictions can be established to make them tied. When a graph is 3-connected, its tree-depth as a graph and the branch-depth of its cycle matroid become tied to each other.

Lemma 4.11. The branch-depth of a connected graph is less than or equal to its tree-depth.

Proof. Let G be a graph. The branch-depth of a graph with at most one edge is 0, hence we may assume that G has at least two edges.

Let F be a rooted forest with height k, where $k = \mathrm{td}(G)$ denotes the tree-depth of G. This forest is structured such that its closure contains the simplification of G as a subgraph, devoid of loops and parallel edges. The closure of a rooted forest is a simple graph on its vertices with edges between two nodes if there is a directed path between them in the forest. We may assume that $V(F) = V(G)$. Thus F is connected.

For every edge $e = uv \in E(G)$, where v is under u in the forest F, attach a leaf to the node v, thus making a tree T. Let L be the bijection from $E(G)$ to the leaves of T. The claim is that (T, L) is a (k, k)-decomposition.

Figure 4.4: The image on the left is the above-mentioned rooted forest of P_5, its closure trivially contains the path. The image on the right is the tree obtained by attaching leafs to the respective vertices.

Consider an internal node v of T with distance i from the root. This distance is bounded by $k - 1$ due to the height of F. Let P be the partition of $E(G)$ determined by node v. If a vertex in G has edges from multiple parts of this partition, it must lie on the path from the root to v. Therefore, the number of such vertices, which equals the width of v is at most $i + 1 \leq k$.

As the root node has a distance of at most k, the radius of T is also at most k, thereby confirming the claim. □

Furthermore, Theorem 4.12 holds for all graphs.

Theorem 4.12. *Let G be a graph, k be its branch-depth, and t be its tree-depth. Then $k - 1 \leq t \leq \max\{2k^2 - k + 1.2\}$.*

Proposition 4.13. $\mathrm{bd}(M(G)) \leq \mathrm{bd}(G) - 1 \leq t$. *In addition, if G is connected, then* $\mathrm{bd}(M(G)) \leq \mathrm{bd}(G) - 1 \leq t - 1$.

To prove the Proposition 4.13 use the following lemma for connectivity functions from [29].

Lemma 4.14. For a connected graph G, if $\emptyset \neq X \neq S(G)$, then $\lambda_{M(G)}(X) \leq \lambda_G(X) - 1$.

Proof of the Proposition 4.13.

Proof. The inequality from Lemma 4.14, with Lemma 4.11 suggests, that if G is connected, then $\mathrm{bd}(M(G)) \leq \mathrm{bd}(G) - 1 \leq t - 1$.

If G is disconnected, then choose a vertex from each component and contract them into one vertex. That cannot increase the branch-depth of the graph, and the cycle matroid does not change. With Theorem 4.12, the statement is fulfilled. □

Chapter 5

Depth parameters

This section is dedicated to more matroid depth parameters. Let M be a matroid, and let $\mathrm{dd}(M)$, $\mathrm{cd}(M)$ and $\mathrm{cdd}(M)$ denote the **deletion-depth**, **contraction-depth** and **contraction-deletion depth** parameters respectively. These notions were first mentioned in [10], by Ding, Oporowski and Oxley, they referred to contraction-deletion depth as *type*. Similarly, Robertson and Seymour used the names C-type and D-type for contraction-depth and deletion-depth respectively. The parameter names used in this paper are derived from [9] by DeVos, Kwon, and Oum. Their definition is the following 5.1.

Definition 5.1.

- If $E(M) = \emptyset$, then $\mathrm{dd}(M) = \mathrm{cd}(M) = \mathrm{cdd}(M) = 0$.
- If M is not connected, then $\mathrm{dd}(M)$, $\mathrm{cd}(M)$, $\mathrm{cdd}(M)$ is the maximum respective depth of the matroid's components.
- If M is connected, and $E(M) \neq \emptyset$, then:
  - $\mathrm{dd}(M) = 1 + \min_{e \in M}\{\mathrm{dd}(M \setminus e)\}$.
  - $\mathrm{cd}(M) = 1 + \min_{e \in M}\{\mathrm{cd}(M/e)\}$
  - $\mathrm{cdd}(M) = \min\{\min_{e \in M}\{\mathrm{dd}(M \setminus e)\}, \min_{e \in M}\{\mathrm{cd}(M/e)\}\}$

Figure 5.1: Deletion-depth of the matroid in Figure 3.1. By deleting the bottom edge, we got the image on the left containing two components that we can examine recursively. Consequently, the deletion-depth of the matroid is 3.

Figure 5.2: The contraction-depth of the matroid in Figure 3.1. As for Figure 5.1, by contracting the edge c, we get two components that we can examine separately. The contraction-depth of the matroid is 4.

These parameters can also be set, by the height of their decomposition trees. Definition 5.2 is the definition of the deletion-decomposition tree, we get the decomposition trees of the remaining two parameters by changing $M \setminus e$ to M/e or $\min\{\mathrm{dd}(M \setminus e), \mathrm{cd}(M/e)\}$, depending on the parameter.

Definition 5.2. Deletion-decomposition tree:

- If M has a single element, the tree has a single vertex, labelled with the element.
- If M is disconnected the tree is obtained by merging the roots of the decomposition trees of the components of M.
- If M is connected, there exists an element $e \in M$ such that $\mathrm{dd}(M) = \mathrm{dd}(M \setminus e) + 1$. The tree is obtained by attaching the decomposition tree of $M \setminus e$ to a new vertex, label the edge by the deleted element e, and change the root to the newly added vertex. The **deletion-depth** is then the smallest height of the tree.

Figure 5.3: The deletion-decomposition tree and contraction-decomposition tree of the matroid in Figure 3.1.

From Definition 5.1 it follows trivially, that the contraction-deletion depth of a matroid is at most the minimum of its contraction-depth, and deletion-depth.

Another notable observation is the duality between contraction-depth and deletion-depth. For a matroid M, we have $\mathrm{cd}(M) = \mathrm{dd}(M^*)$, where M^* represents the dual matroid. Consequently, for contraction-deletion depth, $\mathrm{cdd}(M) = \mathrm{cdd}(M^*)$.

Two other depth parameters are **contraction*-depth** and **contraction*-deletion depth** denoted by $c^*d(M)$ and $c^*dd(M)$, respectively. The first one was introduced by Kardoš, Král', Liebenau and Mach [23] under the name branch-depth.Around the same period, Devos, Kwon, and Oum introduced a parameter with the same name, presented earlier in Chapter 4, which made the topic slightly confusing. This prompted Král' et al. to rename their parameter. In some papers, it is referred to by the name $KKLM$-depth, derived from the authors' initials.

Let $M = (S, \mathcal{F})$ be a matroid, below are the definitions for each notion.

Definition 5.3. Contraction*-depth decomposition is a pair (T, f), where T is a tree with $r(M)$ edges. The function f maps the elements to the leaves such that for every set of elements $X \subseteq S$, the number of edges in the rooted subtree induced by $f(X)$, denoted by $||T^*(X)||$, is at least $r(X)$. **Contraction*-depth** is the minimum depth of a contraction*-depth decomposition of M.

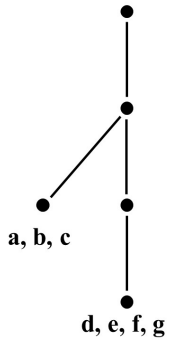


Figure 5.4: The contraction*-decomposition of the matroid in Figure 3.1. The result of this decomposition is that $c^*d(M) = 3$, as it is relatively easy to see that no other decomposition with a lesser depth could satisfy the rank requirements.

For matroids represented over a fixed finite field $\mathbb{F}$, there exists a recursive definition, that does not include the decomposition.

Definition 5.4. Contraction*-depth: For **representable** matroids.

- If M has a single element, then $c^*d(M) = \text{rank}(M)$.
- If M is not connected, then $c^*d(M)$ is the maximum contraction*-depth of a component of M.
- If M is connected, $c^*d(M) = 1+min(M/K)$ factoring M over an arbitrary one-dimensional subspace.

The last point of the definition resembles the definition of contraction-depth 5.1, which explains why the authors ultimately chose this name. The difference here is that the contraction

can be made by any arbitrary one-dimensional subspace; it does not have to be a subspace generated by an element of the matroid. This implies the following inequality: for a matroid M, $\mathrm{c}^*\mathrm{d}(M) \leq \mathrm{cd}(M)$.

The contractions from the definition can be visualized using a contraction*-tree, which is defined similarly to contraction-decomposition trees. In this context, one-vertex trees correspond to matroids with rank 0, and the edges are labeled with one-dimensional subspaces. A **principal contraction*-tree** has all its edge labels generated by elements of the matroid. Naturally, the minimal depth of such a tree serves as an upper bound on the contraction*-depth.

Bria'nski, Král', and Lamaison in [2] proved the following lemma about c^*-depth of a matroid. This is one of their auxiliary results in the course of their research on this parameter.

Lemma 5.5. Let (T, f) be a contraction*-decomposition of a matroid M and let $v_1, \ldots, v_k$ be the maximal descendants of the root that are branching vertices or leaves. Each set $T(v_i), i \in [k]$, is a union of components of the matroid M. In particular, the following holds for every subset X of elements of M:

$$r_M(X) = \sum_{i \in [k]} r_M(X \cap T(v_i))$$

.

The lemma informally states that if the root of the contraction*-decomposition tree has a degree greater than one, then the matroid associated with this decomposition is not connected. The subtrees of the root's children contain the images of each component, meaning all elements of a component are mapped into the same subtree.

The definition of contraction*-deletion depth from [30] applies only to representable matroids. Similar to Definition 5.4, the following definition is recursive.

Definition 5.6.

- If $r(M) = 0$ then $\mathrm{c}^*\mathrm{dd}(M) = 0$.
- If $r(M) = 1$ then $\mathrm{c}^*\mathrm{dd}(M) = 1$.
- If M is disconnected, then $\mathrm{c}^*\mathrm{dd}(M)$ is the maximum contraction*-deletion depth of components of M.
- If M is connected then $\mathrm{c}^*\mathrm{dd}(M)$ is one plus the smaller among the two:
  - the minimum contraction*-deletion depth of $(M \setminus e)$,
  - the minimum contraction*-deletion depth of the matroid (M/K) factoring M over an arbitrary one-dimensional subspace K.

Just like in the case of contraction-deletion depth, it follows trivially from the definition that the contraction*-deletion depth of a matroid is at most its contraction*-depth.

Contraction-depth, deletion-depth, contraction-deletion depth, contraction*-depth, and contraction*-deletion depth can be extended from matroids to matrices. This extension means that given a matrix A, these notions correspond to those of its associated column matroid.

5.1 Computation

In preceding sections 3.1, 4.2, one could read that for branch-width and branch-depth, there exists a polynomial time algorithm, that can decide for a matroid M whether the $\mathrm{bw}(M)$ and the $\mathrm{bd}(M)$ is less than a fixed integer. Moreover, if the answer is yes, it can compute the exact value of these parameters. Unfortunately, there is no such algorithm for the parameters discussed in this chapter. In a recent paper Brianski, Koutecký, Král', Pekárková, and Schröder [1] proved that here these decisions are NP-complete even for represented matroids.

Theorem 5.7. *For every field* **F***, each of the following five decision problems, whose input is an* **F***-represented matroid M and an integer k, is NP-complete:*

- *Is the contraction-depth of M at most k?*
- *Is the contraction***-depth of M at most k?*
- *Is the contraction-deletion depth of M at most k?*
- *Is the contraction***-deletion depth of M at most k?*
- *Is the deletion-depth of M at most k?*

Proof. Let $G = (X, Y; E)$ be an input bipartite graph. G' is obtained from G by connecting all vertices from X with all vertices from Y. Deciding whether there exist k-element subsets $X' \subseteq X$ and $Y' \subseteq Y$ such that $X' \cup Y'$ is independent is NP-complete. The crucial part of the proof lies in demonstrating that the existence of such subsets in G is equivalent to certain statements, which in turn are equivalent to the corresponding questions posed in the theorem.

- The matroid $2M_{\mathbb{F}}(G')$ has contraction-depth at most $|X| + |Y| - k + 1$.
- The matroid $M_{\mathbb{F}}(G')$ has contraction*-depth at most $|X| + |Y| - k$.
- The matroid $(|V(G')| + 1)M_{\mathbb{F}}(G')$ has contraction-deletion depth at most $|X| + |Y| - k + 1$.
- The matroid $(|V(G')| + 1)M_{\mathbb{F}}(G')$ has contraction*-deletion depth at most $|X| + |Y| - k$.

$M_{\mathbb{F}}(G')$ denotes a represented matroid over a field $\mathbb{F}$ obtained from graph G'. This contains $|V(G')| + |E(G')|$ elements, each represented by a vector of dimension $|V(G')|$. For every vertex $v \in V(G')$, its corresponding element is $e_v \in M_{\mathbb{F}}(G')$, where e_v represents a unit vector in

the field. For an edge $uv \in E(G')$, the corresponding element is $e_v - e_u$ or $e_u - e_v$, the order does not change the matroid. For a constant D, $DM_{\mathbb{F}}(G')$ means that each element in $M_{\mathbb{F}}(G')$, represented by a column, is included D times in $DM_{\mathbb{F}}(G')$.

The following lemma concerning bipartite graphs helps establish the equivalence. The proof can be found in [3].

Lemma 5.8. Let G be a bipartite graph with parts X and Y, $\mathbb{F}$ be a field, and k be an integer. Let G' be the graph obtained from G by adding all edges between the vertices of X and between the vertices of Y. The following three statements are row-equivalent.

- The graph G has an independent set containing k elements of X and k elements of Y.
- The contraction*-depth of $M_{\mathbb{F}}(G')$ is at most $|X| + |Y| - k$.
- The contraction-depth of the matroid $2M_{\mathbb{F}}(G')$ is at most $|X| + |Y| - k + 1$.

The equivalence of the first two statements and the existence of an independent set follow directly from the lemma.

The matroid $(|V(G')|+1)M_{\mathbb{F}}(G')$ has rank $|G'|$, indicating that its contraction-deletion depth and contraction*-deletion-depth at most $|G'| + 1$ and at most $|G'|$, respectively. In this matroid each element is parallel to at least $|V(G')|$ others, meaning that the contraction-deletion depth of$M_{\mathbb{F}}(G')$ is the same as its contraction-depth and its contraction*-deletion depth is the same as its contraction*- depth. From Lemma 5.8 follows the equivalence of statements 3 and 4.

The proof easily constructs the listed matroids in polynomial time in the number of input graph's vertices. Hence the NP-completeness is fulfilled.

To prove that deletion-depth is NP-complete, it is important to note that the contraction-depth of a matroid is equal to the deletion-depth of its dual. For a represented matroid it is easy to construct its dual in polynomial time in the number of the elements of the matroid. □

Since it is not possible to create similar algorithms as in the previous chapters, let us examine the upper and lower bounds of the parameters.

For contraction-depth and deletion-depth, DeVos, Kwon, and Oum in [9] proved upper and lower bounds with the size of the matroid's circuit. The proof of their theorem requires the following.

Theorem 5.9. *Seymour (see [10])). If C is a longest circuit in a connected matroid M, then M/C has no circuits of size at least $|C|$.*

Lemma 5.10. Let M be a matroid and $e \in S(M)$. If C is a circuit of a matroid M with $|C| \geq 2$, then M/e has a circuit of size at least $|C|/2$.

Now the theorem by DeVos et. al.

Theorem 5.11. *Let c be the length of a longest circuit in M. (If M has no circuits, then let $c = 1$.)* $\log_2 c \leq \mathrm{cd}(M) \leq c(c+1)/2$.

Proof. First, we prove the upper bound with induction on c. We assume that $c \geq 2$, and M is connected. If $c \leq 1$, the contraction-depth is less than one, as all components have elements at most 1.

Let C be one of the longest circuits in M. According to the definition, $\mathrm{cd}(M) \leq |C| + \mathrm{cd}(M/C)$. Using induction and Theorem 5.9, we can show that $\mathrm{cd}(M/C) \leq c(c-1)/2$. By substituting $\mathrm{cd}(M/C)$ by its upper bound in the first inequality $\mathrm{cd}(M) \leq c(c+1)/2$.

The lower bound is proven by induction on $|S(M)|$. Assume that M is connected, otherwise, we can look at the components of M. Assume $c > 1$, since if $c = 1$, and by connectedness all elements are in the same circuit, the statement would automatically be satisfied. Hence $|S(M)| > 1$.

For all $e \in S(M)$, M/e has a circuit of size at least $c/2$ according to Lemma 5.10. Therefore, by the induction hypothesis $\mathrm{cd}(M/e) \geq \log_2 c - 1$. The statement follows directly from the definition. □

Since contraction-depth and deletion-depth are dual notions, Corollary 5.12 follows from Theorem 5.11.

Corollary 5.12. Let c' be the length of a longest cocircuit in M. (If M has no cocircuits, then let $c' = 1$.) Then $\log_2 c' \leq \mathrm{dd}(M) \leq c'(c'+1)/2$.

These results imply that having a small contraction-depth equals having small circuits, and having a small deletion-depth equals having small cocircuits. More precisely a class of matroids has bounded contraction-depth if and only if all circuits have bounded size, and a class of matroids has bounded deletion depth if and only if all cocircuits have bounded size.

Another implication of these findings is the characterization of a class of matroids that have both bounded contraction-depth and bounded deletion-depth.

Corollary 5.13. A class $\mathcal{M}$ of matroids has bounded deletion-depth and bounded contraction-depth if and only if there exists a constant $m \geq 0$ such that every connected component of a matroid in $\mathcal{M}$ has at most m elements.

Proof. The converse is trivial from the definition of a connected component.

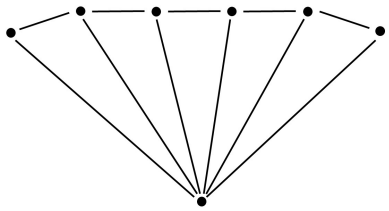
For the forward direction, from Theorem 5.11 and Corollary 5.12 follows, that there exist constants c and c' such that in every connected component of a matroid in $\mathcal{M}$, all circuits have size at most c and all cocircuits have size at most c'.

It was proven by Lemos and Oxley in [24] that if a connected matroid has no circuits of more than c elements and no cocircuits of more than c' elements, then it has at most $cc'/2$ elements. Therefore, each component of a matroid in $\mathcal{M}$ has at most $cc'/2$ elements. □

It was mentioned earlier, that, from the definition contraction-deletion depth is always less than or equal to contraction-depth, and deletion-depth, hence all the upper bounds cited above are upper bounds for cdd.

Easy to see, that another upper bound for contraction-depth is the rank of the matroid. To find a lower bound of contraction-deletion depth on a special matroid, refer to the graph definition below.

Definition 5.14. An **n-fan** denoted by F_n is a graph on $n+1$ vertices, where n vertex creates a path P_n, and the remaining vertex is adjacent to all vertices.



Dittman and Oporowski in [11] proved the theorem below about graphs containing n-fans.

Theorem 5.15. *If a graph G contains F_n as a minor, then the contraction-deletion depth of $M(G)$ is at least $\lfloor \log_2 n \rfloor + 1$.*

For branch-width and branch-depth it was stated, that they are closed under taking a minor. Sadly, it cannot be claimed about contraction-, deletion-, and contraction-deletion depth.

By duality, it is enough to demonstrate that these parameters may increase for minors, as referenced in [9], for one of the first two parameters.

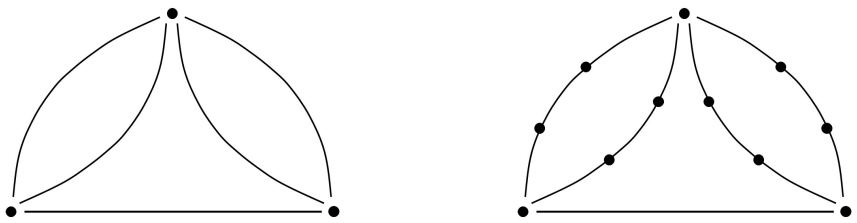


Figure 5.5: The deletion depth of the graph on the left is 3. This is because if we delete the bottom edge, two components remain, both with 2 edges. If we contract the bottom edge, the minor graph has a deletion depth of 4. This relationship also holds for contraction depth by duality. For the graph on the right, the contraction-deletion-depth is 3. If we delete the bottom edge, we get two components, each with a contraction-deletion depth of 3. However, by contracting this edge, the minor will have a circuit of size 6 after an edge deletion, so the contraction-deletion depth is at least 4.

Unlike contraction-depth, the concept of contraction*-depth, as proven by Kardoš, Král', Liebenau, and Mach [23], has this above-mentioned attribution. Specifically, the class of matroids with a c^*-depth of at most k, where k is a constant, is minor-closed. Consequently, the c^*-depth of any minor of a matroid provides a lower bound for this parameter.

Proposition 5.16. *If N is a minor of M, then* $\mathrm{c}^*\mathrm{d}(N) \leq \mathrm{c}^*\mathrm{d}(M)$.

Proof. It is enough to show, that for an element $e \in M$ the contraction*-depth for the minors M/e and $M \setminus e$ are at most $\mathrm{c}^*\mathrm{d}(M)$ since all minors are derived from M through a series of contractions and deletions. Let (T, f) be a depth-decomposition of M with depth at most $\mathrm{c}^*\mathrm{d}(M)$. There are two cases.

The first case is when e is a loop in M. Then $M_0 := M \setminus e = M/e$. Consequently, for every $X \subseteq M_0$ we have $r_{M_0}(X) = r_M(X)$. Therefore, $(T, f|_{M_0})$ is a depth-decomposition of M_0.

If e is not a loop, let $M_1 := M/e$, let u be the leaf $f(e)$, and let v be the parent of u. Let the tree $T_1 = T \setminus e$ and let the function $f_1 : M_1 \to V(T_1)$ be defined as follows:

$$f_1(x) = \begin{cases} v & \text{if } f(x) = u, \text{ and} \\ \mathrm{f(x)} & \text{otherwise} \end{cases}$$

Now we have to show that (T_1, f_1) is a depth-decomposition of M_1. $||T_1|| = r(M_1)$, since $r(M_1) = r(M) - 1$ because of e not being a loop. From this, let $X \subseteq M_1$, the following is satisfied $r_{M_1}(X) = r_M(X \cup \{e\}) - 1$. If $u \in f(X)$, then by the depth-decomposition:

$$||T_1^*(X)|| = ||T^*(X \cup \{e\})|| - 1 \geq r_M(X \cup \{e\}) - 1 = r_{M_1}(X).$$

If $u \notin f(X)$, then

$$||T_1^*(X)|| = ||T^*(X)|| \geq r_M(X) \geq r_{M_1}(X).$$

For the deletion let $M_2 = M \setminus e$. If e is a bridge, then $M/e = M \setminus e$, thus we may assume, that e is not a bridge. The claim is that (T, f_{M_2}) is a depth-decomposition of M_2. Since e is not a bridge $r(M_2) = r(M) = ||T||$, which is the number of edges in the tree. Furthermore, for every $X \subseteq M_2$ holds, that $||T^*(X)|| \geq r_M(X) = r_{M_2}(X)$. □

Similarly to Proposition 5.11 and Corollary 5.12, the following can be established for contraction*-depth [1].

Theorem 5.17. *Let M be a matroid and c the size of its largest circuit. It holds that* $\log_2 c \leq \mathrm{c}^*\mathrm{d}(M) \leq c^2$. *Moreover, there exists a polynomial-time algorithm that for an input oracle-given matroid M outputs a contraction*-decomposition tree of depth at most c^2.*

After reading Chapter 6 the similarity of the theorems will not be surprising. In [1] an example shows, that this quadratic upper bound cannot be lowered, it is optimal up to a constant factor.

Chan, Cooper, Kouteckỳ, Král', and Pekárková [4] presented a dynamic programming algorithm for contraction*-depth decomposition of certain represented matroids. Their theorem is the following. The idea of the proof is to first approximate the parameter with the algorithm from Section 5.2. Then proceed computing along its depth-first-search transversal possible frontiers. A frontier is a tuple, described more in [4].

Theorem 5.18. *For the parameterization by a positive integer d and a prime power q, there exists a fixed parameter algorithm that for a vector matroid M over the q-element field either outputs that $\mathrm{c}^*\mathrm{d}(M)$ is larger than d, or outputs a depth-decomposition of M with depth d.*

5.2 Approximation

František Kardoš, Daniel Král, Anita Liebenau and Lukáš Mach [23] presented an algorithm, which finds a depth-decomposition of a matroid M, with depth at most $4^{\mathrm{c}^*\mathrm{d}(M)}$. Their steps are the following.

Algorithm 3. Input an M connected matroid, a C circuit of this matroid, and a non-loop element $e \in C$.From the definition of connected matroids, it follows that every connected matroid with at least two elements contains a circuit.

If $r(M) = 0$ or $r(M) = 1$, the depth-decomposition is trivial. It consists of either a single vertex (the root) if $r(M) = 0$ or two vertices (the root and one additional vertex where all elements are mapped) if $r(M) = 1$.

Now consider the case, when $r(M) \geq 2$. If the size of the input circle is at most 2, find a different one as follows. Determine a base B with greedy algorithm, then delete all the elements f from B such that $B \cup \{e\} \setminus \{f\}$ is dependent. The resulting set forms a circle of length at least 3.

The next step is to contract e in the matroid and use the matroid intersection algorithm to check if the resulting matroid remains connected. If it is connected, the algorithm calls itself recursively, with input parameters $C \setminus \{e\}$ and an arbitrary $e_1 \in C \setminus \{e\}$. After getting the decomposition of this matroid, we got the decomposition of M by adding a new vertex to be the root.

If it is not connected, the algorithm calls itself recursively for each component. The one component which contains $C \setminus \{e\}$ has input parameters $C \setminus \{e\}$ and an arbitrary $e_1 \in C \setminus \{e\}$. The other components have parameters $C' \setminus \{e\}$ and $e_1 \in C' \setminus \{e\}$, where C' is a different circle that contains e. To get the decomposition of M, add a new vertex which becomes the root of all these small decompositions.

The algorithm is polynomial and returns a valid depth-decomposition, as demonstrated in [23].

Lemma 5.19. The algorithm returns a depth-decomposition of M with depth at most $4^{c^* \mathrm{d}(M)}$.

The following Corollary 5.20 is a result of Lemma 5.19.

Corollary 5.20. The contraction*-depth of a finite matroid M is at most c^2, where c is the size of the largest circuit of M.

5.3 Connection to graph parameters

This section aims to prove that all the parameters defined above are tied to graph tree-depth. However, this is not universally true; a few restrictions need to be applied. The first claim from [9], by DeVos, Kwon, and Oum, is that 2-connectedness for graphs is enough for tree-depth to be tied with contraction-depth.

Proposition 5.21. *Let G be a 2-connected graph of tree-depth t. Then* $1 + \frac{1}{2}\log_2(t-1) \leq \mathrm{cd}(M(G)) \leq 2^{2(t-1)}$

Proof. In [26], Proposition 6.1 states that if a graph contains no path longer than n, then its tree-depth is at most n. G has a path of length $t-1$. and by an old Dirac theorem, which states that if a 2-connected graph has a path of length L, it also has a cycle of length at least $2\sqrt{L}$. Therefore, G has a cycle of length at least $2\sqrt{t-1}$. By Theorem 5.11 the contraction-depth of $M(G)$ is at least $\log_2(2\sqrt{t-1})$.

For the upper bound, let L be the length of a longest cycle of G. Since $t \geq \mathrm{td}(C_L) = 1 + \mathrm{td}(P_L - 1) = 1 + \lceil \log_2 L \rceil$, it follows, that $L \leq 2^{t-1}$. By theorem 5.11 $\mathrm{cd}(M(G)) \leq 2^{t-1}(2^{t-1}+1)/2 \leq 2^{2(t-1)}$. □

While the requirement of being 2-connected is insufficient for establishing a tiedness between tree-depth and other depth parameters, the real criterion is only slightly different. As for branch-depth, DeVos, Kwon, and Oum demonstrated in [9] that if a graph G is 3-connected, then the tree-depth of graphs and the contraction-depth, as well as the contraction-deletion depth of their cycle matroids, are all tied. For completeness, we include branch-depth into the theorem.

Theorem 5.22. *Let G be a 3-connected graph of tree-depth t and let $k = \lfloor \frac{1}{6\sqrt{2}}\sqrt{\log(2(t-1)/5)} \rfloor$. Then* $\frac{\log(2k-1)}{log(1+4\log(2k-1))} \leq \mathrm{bd}(M(G)) \leq \mathrm{cdd}(M(G)) \leq \mathrm{cd}(M(G)) \leq 2^{2(t-1)}$

The upper bound is clear from Proposition 5.21 and from knowing the connection between these parameters from Chapter 6.

The next two propositions from [23] suggest the connection between the contraction*-depth of a graphic matroid and the tree-depth of the corresponding graph.

Proposition 5.23. *For any graph G, the contraction*-depth of the graphic matroid $M(G)$ is at most the tree-depth of the graph G.*

The converse of this Proposition 5.23 does not hold. For example the contraction*-depth of $M(P_n)$ is 1, but the tree-depth of that graph is $\lceil \log_2(n+1) \rceil$. Yet the following can be said.

Proposition 5.24. *For any 2-connected graph G, the contraction*-depth of the graphic matroid $M(G)$ is at least $\frac{1}{2}\log_2(td(G)-1)$.*

Chapter 6

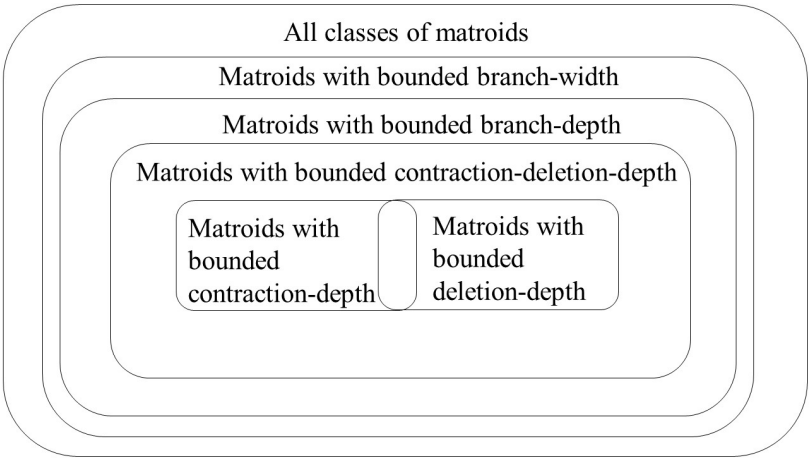
Connections between the parameters

Grasping the connections between these parameters proves quite beneficial as it deepens our understanding of the matroid's structure and potentially enables us to extend known properties to other parameters.

Previously, we mentioned some trivial connections such as $\mathrm{bw}(M) \leq \mathrm{bd}(M)$ and $\mathrm{cdd}(M) \leq \min \mathrm{cd}(M), \mathrm{dd}(M)$, now we will present some additional ones. The first pair to examine are branch-depth and contraction-deletion depth. From definitions 4, 5 the following inequality obtained [9].

Theorem 6.1. *For all matroids M,* $\mathrm{bd}(M) \leq \mathrm{cdd}(M)$.

This establishes a hierarchy among matroids with bounded parameters, as depicted in the figure below.



Moreover, Briański, Král' and Pekárková [3] proved the following theorems, showing that branch-depth can be viewed as a minor closure of contraction-deletion depth.

Theorem 6.2. *Let $\mathcal{M}$ be a class of $\mathbb{F}$-representable matroids. The class has bounded branch-depth if and only if there exists an $\mathbb{F}$-representable class $\mathcal{N}$ of matroids with bounded contraction-deletion depth such that $\mathcal{M}$ is a subclass of the minor closure of $\mathcal{N}$.*

The main theorem of their paper states that if a representable matroid has a rooted (d, r)-decomposition, then it is a minor of a matroid with contraction-deletion depth bounded by a function of d, r. The corollary of this is the subsequent Theorem 6.3, which is also useful for proving Theorem 6.2.

Theorem 6.3. *Every $\mathbb{F}$-representable matroid M is a minor of an $\mathbb{F}$-representable matroid N such that* $\mathrm{cdd}(N) \leq 2\,\mathrm{bd}(M) \cdot (4^{\mathrm{bd}(M)} - 1) + 1$.

Given that branch-depth is minor-closed, Theorem 6.3 implies that a representable matroid has a small branch-depth if and only if it is a minor of a matroid with small contraction-deletion depth.

Figure 6.1: An example illustrating a matroid with small branch-depth and large contraction-deletion depth. The left figure shows a fat cycle, while the right image contains it as a minor. Both have branch-depth 2, but the matroid on the left has a contraction-deletion depth of 6. This can be demonstrated by contracting an edge of each parallel edges; the resulting matroid has a contraction-deletion depth of 1. Conversely, the matroid on the right has a contraction-deletion depth of 3. This example also highlights that contraction-deletion depth is not minor-monotone.

Regarding depth-parameters another easy observation from the definitions, is that $\mathrm{c}^*\mathrm{dd}(M) \leq \mathrm{cdd}(M)$. An interesting relationship exists between contraction-depth and contraction*-depth. Theorem 6.4 proved by Briański et. al [2] demonstrates that a matroid can be augmented in such a manner that the resulting matroid's contraction-depth, minus one, equals the contraction*-depth of the original matroid. A corollary of this statement is that $\mathrm{c}^*\mathrm{d}(M) \leq \mathrm{cd}(M)$.

Theorem 6.4. *Let M be a matroid. The minimum contraction-depth of a matroid M' that contains M as a restriction is equal to the contraction*-depth of M increased by one, i.e., $c^*d(M) + 1 = \min_{M' \sqsubseteq M} cd(M')$ unless every element of M is either a loop or a bridge and M has at least one bridge (when this happens, then $c^*d(M) = cd(M) = 1$).*

The definition of contraction-depth can be altered to exclude exceptions with a single modification: if the matroid consists of one element, then let $\mathrm{cd}'(M)$ be the rank of that element. Furthermore, it has been proved that these two parameters are functionally equivalent [23].

Proposition 6.5. *Contraction-depth and contraction*-depth are functionally equivalent; the following holds for every matroid M,* $\mathrm{c}^*\mathrm{d}(M) \leq \mathrm{cd}(M) \leq 4^{\mathrm{c}^*\mathrm{d}(M)} + 1$.

Chapter 7

Summary

My thesis aimed to familiarize the reader with matroid parameters, highlight their key attributes and connections and demonstrate interesting algorithms for computing them, where feasible. The importance of these parameters was briefly discussed in Chapter 1. Here, I want to emphasize their crucial role in integer programming. If the column matroid of matrix A has a bounded branch-width and A is non-negative, solving an integer program can be achieved in pseudo-polynomial time [7]. Moreover, integer programming becomes fixed-parameter tractable when parameterized by the contraction*-depth of the matroid and the entry complexity of the constraint matrix [4].

Matroid parameters have paved the way for numerous new research avenues. For instance, it has been demonstrated that for graphs with bounded tree-width, determining isomorphism is tractable. This raises the question: can we achieve something similar for matroids with certain bounded parameters? Král' and Pekárková are currently investigating this problem, specifically aiming to design a parameterized algorithm that can determine whether two given represented matroids with bounded branch-width are isomorphic.

Concerning contraction*-depth, an approximation algorithm exists for computing a depth-decomposition. Nonetheless, the issue persists that the approximation factor is not constant. Improving the approximation ratio and devising approximation algorithms for other matroid parameters are open problems.

Bibliography

[1] M. Briański, M. Kouteckỳ, D. Král, K. Pekárková, and F. Schröder. Characterization of matrices with bounded graver bases and depth parameters and applications to integer programming. *Mathematical Programming*, pages 1–35, 2024.

[2] M. Brianski, D. Kral, and A. Lamaison. Closure property of contraction-depth of matroids. *arXiv preprint arXiv:2311.01945*, 2023.

[3] M. Briański, D. Král', and K. Pekárková. Branch-depth is minor closure of contraction-deletion-depth. *arXiv preprint arXiv:2402.16215*, 2024.

[4] T. F. Chan, J. W. Cooper, M. Koutecky, D. Král, and K. Pekarkova. Matrices of optimal tree-depth and a row-invariant parameterized algorithm for integer programming. *SIAM Journal on Computing*, 51(3):664–700, 2022.

[5] B. Courcelle. The monadic second-order logic of graphs. i. recognizable sets of finite graphs. *Information and computation*, 85(1):12–75, 1990.

[6] B. Courcelle. Monadic second-order definable graph transductions: a survey. *Theoretical Computer Science*, 126(1):53–75, 1994.

[7] W. H. Cunningham and J. Geelen. On integer programming and the branch-width of the constraint matrix. In *Integer Programming and Combinatorial Optimization: 12th International IPCO Conference, Ithaca, NY, USA, June 25-27, 2007. Proceedings 12*, pages 158–166. Springer, 2007.

[8] M. Cygan, F. V. Fomin, L. u. Kowalik, D. Lokshtanov, D. Marx, M. Pilipczuk, M. Pilipczuk, and S. Saurabh. *Parameterized algorithms*. Springer, Cham, 2015.

[9] M. DeVos, O.-j. Kwon, and S.-i. Oum. Branch-depth: Generalizing tree-depth of graphs. *European Journal of Combinatorics*, 90:103186, 2020.

[10] G. Ding, B. Oporowski, and J. Oxley. On infinite antichains of matroids. *Journal of Combinatorial Theory, Series B*, 63(1):21–40, 1995.

[11] J. Dittmann and B. Oporowski. Unavoidable minors of graphs of large type. *Discrete mathematics*, 248(1-3):27–67, 2002.

[12] R. G. Downey, M. R. Fellows, et al. *Fundamentals of parameterized complexity*, volume 4. Springer, 2013.

[13] F. V. Fomin and T. Korhonen. Fast fpt-approximation of branchwidth. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 886–899, 2022.

[14] A. Frank. *Connections in combinatorial optimization*, volume 38. Oxford University Press Oxford, 2011.

[15] J. F. Geelen, A. Gerards, N. Robertson, and G. Whittle. On the excluded minors for the matroids of branch-width k. *Journal of Combinatorial Theory, Series B*, 88(2):261–265, 2003.

[16] J. F. Geelen, A. M. Gerards, and G. Whittle. Branch-width and well-quasi-ordering in matroids and graphs. *Journal of Combinatorial Theory, Series B*, 84(2):270–290, 2002.

[17] J. P. Gollin, K. Hendrey, D. Mayhew, and S.-i. Oum. Obstructions for bounded branch-depth in matroids. *arXiv preprint arXiv:2003.13975*, 2020.

[18] I. V. Hicks and N. B. McMurray Jr. The branchwidth of graphs and their cycle matroids. *Journal of Combinatorial Theory, Series B*, 97(5):681–692, 2007.

[19] P. Hliněnỳ. On the excluded minors for matroids of branch-width three. *the electronic journal of combinatorics*, 9(1):R32, 2002.

[20] P. Hliněnỳ. On matroid properties definable in the mso logic. In *Mathematical Foundations of Computer Science 2003: 28th International Symposium, MFCS 2003, Bratislava, Slovakia, August 25-29, 2003. Proceedings 28*, pages 470–479. Springer, 2003.

[21] P. Hlinenỳ. A parametrized algorithm for matroid branch-width. *SIAM Journal on Computing*, 35(2):259–277, 2005.

[22] P. Hliněnỳ. Branch-width, parse trees, and monadic second-order logic for matroids. *Journal of Combinatorial Theory, Series B*, 96(3):325–351, 2006.

[23] F. Kardoš, A. Liebenau, L. Mach, et al. First order convergence of matroids. *European Journal of Combinatorics*, 59:150–168, 2017.

[24] M. Lemos and J. Oxley. A sharp bound on the size of a connected matroid. *Transactions of the American Mathematical Society*, 353(10):4039–4056, 2001.

[25] F. Mazoit and S. Thomassé. Branchwidth of graphic matroids. *Surveys in combinatorics*, 346:275–286, 2007.

[26] J. Nešetřil and P. O. De Mendez. *Sparsity: graphs, structures, and algorithms*, volume 28. Springer Science & Business Media, 2012.

[27] S.-i. Oum and P. Seymour. Approximating clique-width and branch-width. *Journal of Combinatorial Theory, Series B*, 96(4):514–528, 2006.

[28] S.-i. Oum and P. Seymour. Testing branch-width. *Journal of Combinatorial Theory, Series B*, 97(3):385–393, 2007.

[29] J. G. Oxley. *Matroid theory*, volume 3. Oxford University Press, USA, 2006.

[30] K. Pekárková. *Matroid algorithms and their appricationsin optimization*. Phd thesis proposal, Faculty of Informatics, Masaryk University, Brno, Czech Republic, Spring 2023. Available at https://is.muni.cz/th/qsl6r/thesis_proposal.pdf.

[31] N. Robertson and P. D. Seymour. Graph minors—a survey. *Surveys in combinatorics*, 103:153–171, 1985.

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

[33] P. D. Seymour and R. Thomas. Call routing and the ratcatcher. *Combinatorica*, 14:217–241, 1994.