SIAM J. COMPUT.
Vol. 38, No. 3, pp. 1012–1032

FINDING BRANCH-DECOMPOSITIONS AND RANK-DECOMPOSITIONS*

PETR HLINĚNÝ† AND SANG-IL OUM‡

Abstract. We present a new algorithm that can output the rank-decomposition of width at most k of a graph if such exists. For that we use an algorithm that, for an input matroid represented over a fixed finite field, outputs its branch-decomposition of width at most k if such exists. This algorithm works also for partitioned matroids. Both of these algorithms are fixed-parameter tractable, that is, they run in time $O(n^3)$ where n is the number of vertices / elements of the input, for each constant value of k and any fixed finite field. The previous best algorithm for construction of a branch-decomposition or a rank-decomposition of optimal width due to Oum and Seymour [*J. Combin. Theory Ser. B*, 97 (2007), pp. 385–393] is not fixed-parameter tractable.

Key words. rank-width, clique-width, branch-width, fixed-parameter tractable algorithm, graph, matroid

AMS subject classifications. 05C85, 68R10

DOI. 10.1137/070685920

1. Introduction.

Many graph problems are known to be NP-hard in general; however, for practical application we still need to solve them. One method to solve them is to restrict the input graph to have a certain structure. Clique-width, defined by Courcelle and Olariu [4], is very useful for that purpose. Many hard graph problems (in particular all those expressible in monadic second-order logic (MSOL) of adjacency graphs) are solvable in polynomial time as long as the input graph has bounded clique-width and is given in the form of the decomposition for clique-width, called a *k-expression* [3, 24, 6, 15, 10]. A k-expression is an algebraic expression with the following four operations on a vertex-labeled graph with k labels: create a new vertex with label i, take the disjoint union of two labeled graphs, add all edges between vertices of label i and label j, and relabel all vertices with label i to have label j. However, for fixed $k > 3$, it is not known how to find a k-expression of an input graph having clique-width at most k. (If $k \leq 3$, then it has been shown in [2, 1].)

Rank-width is another graph structural invariant introduced by Oum and Seymour [19], aiming at the construction of an $f(k)$-expression of the input graph having clique-width k for some fixed function f in polynomial time. Rank-width is defined (section 7) as the branch-width (see section 2) of the *cut-rank* function of graphs. Rank-width turns out to be very useful for algorithms on graphs of bounded clique-width, since a class of graphs has bounded rank-width if and only if it has bounded clique-width. In fact, if rank-width of a graph is k, then its clique-width lies between

*Received by the editors March 21, 2007; accepted for publication (in revised form) March 3, 2008; published electronically June 25, 2008.
http://www.siam.org/journals/sicomp/38-3/68592.html

†Faculty of Informatics, Masaryk University, Botanická 68a, 602 00 Brno, Czech Republic (hlineny@fi.muni.cz). This author's research was supported by Czech research grant GAČR 201/08/0308 and by Research intent MSM0021622419 of the Czech Ministry of Education.

‡Department of Mathematical Sciences, KAIST, Daejeon, 305-701, Republic of Korea (sangil@kaist.edu). This author's research was done while at the Georgia Institute of Technology and University of Waterloo and partially supported by NSF grant DMS 0354742 and the SRC Program of Korea Science and Engineering Foundation (KOSEF) grant funded by the Korea government (MOST) (R11-2007-035-01002-0).

k and $2^{k+1} - 1$ [19] and an expression can be constructed from a rank-decomposition of width k.

In this paper, we are mainly interested in the following problem.

> Find a fixed-parameter tractable algorithm that outputs a rank-decomposition of width at most k if the rank-width of an input graph (with more than one vertex) is at most k.

The first rank-width algorithm by Oum and Seymour [19] finds only a rank-decomposition of width at most $3k + 1$ for n-vertex graphs of rank-width at most k in time $O(n^9 \log n)$. This algorithm has been improved by Oum [18] to output a rank-decomposition of width at most $3k$ in time $O(n^3)$. Using this approximation algorithm and finiteness of excluded vertex-minors [17], Courcelle and Oum [5] have constructed an $O(n^3)$-time algorithm to decide whether a graph has rank-width at most k. However, this is only a decision algorithm; if the rank-width is at most k, then this algorithm verifies that the input graph contains none of the excluded graphs for rank-width at most k as a vertex-minor. It does *not* output a rank-decomposition showing that the graph indeed has rank-width at most k.

In another paper, Oum and Seymour [20] have constructed a polynomial-time algorithm that can output a rank-decomposition of width at most k for graphs of rank-width at most k. However, it is not fixed-parameter tractable; its running time is $O(n^{8k+12} \log n)$. Obviously, it is very desirable to have a fixed-parameter tractable algorithm to output such an "optimal" rank-decomposition, because most algorithms on graphs of bounded clique-width require a k-expression on their input. So far, the only known efficient way of constructing an expression with bounded number of labels for a given graph of bounded clique-width uses rank-decompositions.

In this paper, we present an affirmative answer to the above problem (Theorem 7.3). An amusing aspect of our solution is that we deeply use submodular functions and matroids to solve the rank-decomposition problem, which shows (somehow unexpectedly) a "truly geometrical" nature of this graph-theoretical problem. In fact we solve the following related problem on matroids, too (Theorem 6.7).

> Find a fixed-parameter tractable algorithm that, given a matroid represented by a matrix over a fixed finite field, outputs a branch-decomposition of width at most k if the branch-width of the input matroid (with more than one element) is at most k.

So to give the final solution of our first problem, Theorem 7.3, we are going to bring together two previously separate lines of research: We will combine Oum and Seymour's above sketched work on rank-width and on branch-width of submodular functions with Hliněný's recent works [13, 14] on parametrized algorithms for matroids over finite fields.

Namely, Hliněný [13] has given a parametrized algorithm which in time $O(n^3)$ either outputs a branch-decomposition of width $\leq 3k + 1$ of an input matroid M represented over a fixed finite field or confirms that the branch-width of M is more than $k + 1$ (Algorithm 6.1). Using the ideas of [14] and minor-monotonicity of the branch-width parameter, he has concluded with an $O(n^3)$ fixed-parameter tractable algorithm [13] for the (exact value of) branch-width k of an input matroid M represented over a fixed finite field (Theorem 5.1). Similarly as above, this algorithm is only a decision algorithm and does not output a branch-decomposition of width k.

We last remark that the following (indeed widely expected) hardness result has been given only recently by Fellows et al. [7]: It is NP-hard to find graph clique-width. To argue that it is NP-hard to find rank-width, we combine some known results: Hicks and McMurray Jr. [11] and Mazoit and Thomassé [16] independently proved that the branch-width of the cycle matroid of a graph is equal to the branch-width of the graph if it is 2-connected. Hence, we can reduce (section 7) the problem of finding branch-width of a graph to finding rank-width of a certain bipartite graph, and finding graph branch-width is NP-hard as shown by Seymour and Thomas [23]. Still, the main advantage of rank-width over clique-width is the fact that we currently have a fixed-parameter tractable algorithm for rank-width but not for clique-width.

Our paper is structured as follows: The next section briefly introduces definitions of branch-width, partitions, matroids, and the amalgam operation on matroids. In section 3, we explain the notion of so-called titanic partitions, which we further use in section 4 to "model" partitioned matroids in ordinary matroids. At this point it is worth noting that partitioned matroids present the key tool that allows us to shift from a branch-width-testing algorithm [13] to a construction of an "optimal" branch-decomposition (see Theorem 5.7) and of a rank-decomposition. In section 5, we will discuss a simple but slow algorithm for matroid branch-decompositions. In section 6, we will present a faster algorithm. As the main application, we then use our result to give an algorithm for constructing a rank-decomposition of optimal width of a graph in section 7.

2. Definitions. *Branch-width.* Let $\mathbb{Z}$ be the set of integers. For a finite set V, a function $f : 2^V \to \mathbb{Z}$ is called *symmetric* if $f(X) = f(V \setminus X)$ for all $X \subseteq V$, and is called *submodular* if $f(X) + f(Y) \geq f(X \cap Y) + f(X \cup Y)$ for all subsets X, Y of V. A tree is *subcubic* if all vertices have degree 1 or 3. For a symmetric submodular function $f : 2^V \to \mathbb{Z}$ on a finite set V, the branch-width is defined as follows (see Figure 1).

A *branch-decomposition* of the symmetric submodular function f is a pair (T, μ) of a subcubic tree T and a bijective function $\mu : V \to \{t : t \text{ is a leaf of } T\}$. (If $|V| \leq 1$, then f admits no branch-decomposition.) For an edge e of T, the connected components of $T \setminus e$ induce a partition (X, Y) of the set of leaves of T. (In such a case, we say that $\mu^{-1}(X)$ (or $\mu^{-1}(Y)$) is *displayed* by e in the branch-decomposition (T, μ). We also say that V and $\emptyset$ are displayed by the branch-decomposition.) The *width* of an edge e of a branch-decomposition (T, μ) is $f(\mu^{-1}(X))$. The *width* of (T, μ) is the maximum width of all edges of T. The *branch-width* of f, denoted by $\mathrm{bw}(f)$, is the minimum of the width of all branch-decompositions of f. (If $|V| \leq 1$, we define $\mathrm{bw}(f) = f(\emptyset)$.)

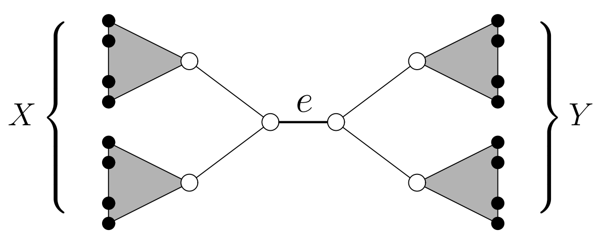

FIG. 1. *An illustration of the definition of a branch-decomposition (T, μ) of f: An edge e of the tree T displays the sets $\mu^{-1}(X)$ and $\mu^{-1}(Y)$, and the width of e is $f(\mu^{-1}(X))$.*

A natural application of this definition is the branch-width of a graph, as introduced by Robertson and Seymour [22] along with better known tree-width, and its direct matroidal counterpart below in this section. We also refer to further formal definition of rank-width in section 7.

Partitions. A *partition* $\mathcal{P}$ of V is a collection of nonempty pairwise disjoint subsets of V whose union is equal to V. Each element of $\mathcal{P}$ is called a *part*. For a symmetric submodular function f on 2^V and a partition $\mathcal{P}$ of V, let $f^{\mathcal{P}}$ be a function on $2^{\mathcal{P}}$ (also symmetric and submodular) such that $f^{\mathcal{P}}(X) = f(\cup_{Y\in X} Y)$. The *width* of a partition $\mathcal{P}$ is $f(\mathcal{P}) = \max\{f(Y) : Y \in \mathcal{P}\}$.

We will often denote a partition by a function as follows. For a function $\pi : V \to W$, let $\pi_y = \{x : \pi(x) = \pi(y)\}$ for $y \in V$, and let $[\pi] = \{\pi_x : x \in V\}$ be the partition of V induced by π.

Matroids. We refer to Oxley [21] in our matroid terminology. A *matroid* is a pair $M = (E, \mathcal{B})$, where $E = E(M)$ is the ground set of M (elements of M), and $\mathcal{B} \subseteq 2^E$ is a nonempty collection of *bases* of M, no two of which are in an inclusion. Moreover, matroid bases satisfy the "exchange axiom": if $B_1, B_2 \in \mathcal{B}$ and $x \in B_1 \setminus B_2$, then there is $y \in B_2 \setminus B_1$ such that $(B_1 \setminus \{x\}) \cup \{y\} \in \mathcal{B}$. We consider only finite matroids. A typical example of a matroid is given by a set of vectors (forming the columns of a matrix $\boldsymbol{A}$) with usual linear independence. The matrix $\boldsymbol{A}$ is then called a *representation* of the matroid.

All matroid bases have the same cardinality called the *rank* $r(M)$ of the matroid. Subsets of bases are called *independent*, and sets that are not independent are *dependent*. A matroid M is *uniform* if all subsets of $E(M)$ of certain size are the bases, and M is *free* if $E(M)$ is a basis. The *rank function* $r_M(X)$ in M is the maximum cardinality of an independent subset of a set $X \subseteq E(M)$. The *dual matroid* M^* is defined on the same ground set with the bases as set-complements of the bases of M. For a subset X of E, the *deletion* $M \setminus X$ of X from M, or the *restriction* $M \restriction (E \setminus X)$ of M to $E \setminus X$, is the matroid on $E \setminus X$ in which $Y \subseteq E \setminus X$ is independent in $M \setminus X$ if and only if Y is an independent set of M. The *contraction* M/X of X in M is the matroid $(M^* \setminus X)^*$. Matroids of the form $M/X \setminus Y$ are called *minors* of M.

To define the branch-width of a matroid, we consider its (symmetric and submodular) connectivity function

$$\lambda_M(X) = r_M(X) + r_M(E \setminus X) - r_M(E) + 1$$

defined for all subsets $X \subseteq E = E(M)$. A "*geometric*" meaning is that the subspaces spanned by X and $E \setminus X$ intersect in a subspace of dimension $\lambda_M(X) - 1$. *Branch-width* $\mathrm{bw}(M)$ and *branch-decompositions* of a matroid M are defined as the branch-width and branch-decompositions of λ_M. Notice that $\lambda_{M^*} \equiv \lambda_M$.

Partitioned matroids. A pair $(M, \mathcal{P})$ is called a *partitioned matroid* if M is a matroid and $\mathcal{P}$ is a partition of $E(M)$. A partitioned matroid $(M, \mathcal{P})$ is *representable* if M is representable. A connectivity function of a partitioned matroid $(M, \mathcal{P})$ is defined as $\lambda_M^{\mathcal{P}}$. We note that $\lambda_M^{\mathcal{P}}$ is *symmetric* and *submodular*; in other words,

$$\lambda_M^{\mathcal{P}}(X) = \lambda_M^{\mathcal{P}}(\mathcal{P} \setminus X),$$
$$\lambda_M^{\mathcal{P}}(X) + \lambda_M^{\mathcal{P}}(Y) \geq \lambda_M^{\mathcal{P}}(X \cap Y) + \lambda_M^{\mathcal{P}}(X \cup Y).$$

Branch-width $\mathrm{bw}(M, \mathcal{P})$ and *branch-decompositions* of a partitioned matroid $(M, \mathcal{P})$ are defined as branch-width, branch-decompositions of $\lambda_M^{\mathcal{P}}$.

Amalgams of matroids. Let M_1, M_2 be matroids on E_1, E_2, respectively, and $T = E_1 \cap E_2$. Moreover, let us assume that $M_1 \restriction T = M_2 \restriction T$. If M is a matroid on $E_1 \cup E_2$ such that $M \restriction E_1 = M_1$ and $M \restriction E_2 = M_2$, then M is called an *amalgam* of M_1 and M_2 (see Figure 2).

It is known that an amalgam of two matroids need neither exist nor be unique. However, there are certain interesting cases when an amalgam is known to exist.

We show one such example here, and we use another one in Proposition 5.4 with representable matroids. Let r_1, r_2 be the rank function of M_1, M_2, respectively. Let r be the rank function of $M_1 \restriction T$. Let

$$\eta(X) = r_1(X \cap E_1) + r_2(X \cap E_2) - r(X \cap T)$$

and

$$\zeta(X) = \min\{\eta(Y) : Y \supseteq X\}.$$

Proposition 2.1 (see [21, Proposition 12.4.2]). *If ζ is submodular, then ζ is the rank function of a matroid that is an amalgam of M_1 and M_2.*

If ζ is submodular, then the matroid on $E_1 \cup E_2$ having ζ as its rank function is called the *proper amalgam* of M_1 and M_2.

Lemma 2.2. *If $M_1 \restriction T$ is free, then ζ is submodular and therefore the proper amalgam of M_1 and M_2 exists.*

Proof. Since $M_1 \restriction T$ is a free matroid, we have $r(X \cap T) = |X \cap T|$ and therefore η is submodular. We will show that this implies that ζ is submodular, that is to show that $\zeta(X_1) + \zeta(X_2) \geq \zeta(X_1 \cap X_2) + \zeta(X_1 \cup X_2)$. For $i \in \{1, 2\}$, let Y_i be a set such that $Y_i \supseteq X_i$ and $\zeta(X_i) = \eta(Y_i)$. Then $\zeta(X_1) + \zeta(X_2) = \eta(Y_1) + \eta(Y_2) \geq \eta(Y_1 \cap Y_2) + \eta(Y_1 \cup Y_2) \geq \zeta(X_1 \cap X_2) + \zeta(X_1 \cup X_2)$. By Proposition 2.1, the proper amalgam of M_1 and M_2 exists. ∎

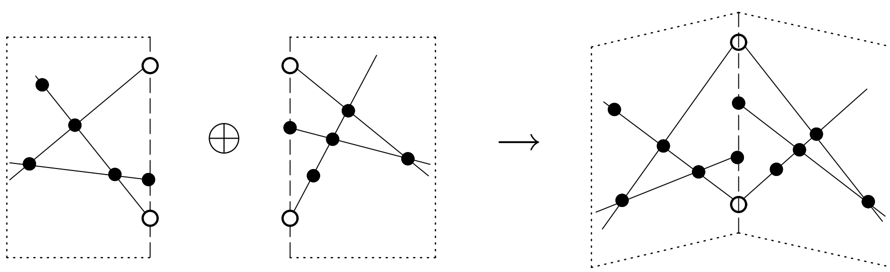

Fig. 2. *A "geometrical" illustration of an amalgam of two matroids, in which hollow points are the shared elements T.*

3. Titanic partitions. This technical section is about general symmetric submodular functions. Let V be a finite set and f be a symmetric submodular function on 2^V.

A set $\mathcal{T} \subset 2^V$ of subsets of V is called an *f-tangle* of order $k+1$ if it satisfies the following three axioms.

(T1) For all $A \subseteq V$, if $f(A) \leq k$, then either $A \in \mathcal{T}$ or $V \setminus A \in \mathcal{T}$.

(T2) If $A, B, C \in \mathcal{T}$, then $A \cup B \cup C \neq V$.

(T3) For all $v \in V$, we have $V \setminus \{v\} \notin \mathcal{T}$. Robertson and Seymour [22] showed that tangles are related to branch-width.

Theorem 3.1 (Robertson and Seymour [22]). *There is no f-tangle of order $k+1$ if and only if the branch-width of f is at most k.*

A subset X of V is called *titanic* with respect to f if whenever A_1, A_2, A_3 are pairwise disjoint subsets of X such that $A_1 \cup A_2 \cup A_3 = X$, there is $i \in \{1, 2, 3\}$ such that $f(A_i) \geq f(X)$.

Lemma 3.2. *Let V be a finite set and f be a symmetric submodular function on 2^V. Let X be a titanic set with respect to f. If $X_1 \cup X_2 \cup X_3 = X$, then there exists*

$i \in \{1,2,3\}$ *such that* $f(X_i) \geq f(X)$. *(Note that* X_1, X_2, X_3 *are not required to be pairwise disjoint.)*

Proof. Suppose not. We pick a counterexample with minimum $|X_1|+|X_2|+|X_3|$. If X_1, X_2, X_3 are pairwise disjoint, then by definition the lemma is true.

We may assume that $X_1 \cap X_2 \neq \emptyset$. Let Y_1 be a set minimizing $f(Y_1)$ subject to the condition $X_1 \setminus X_2 \subseteq Y_1 \subseteq X_1$. Then $f(X_1) \geq f(Y_1)$ and $f(X_1 \setminus X_2) \geq f(Y_1)$. Let $Y_2 = X_2 \setminus Y_1$. By the submodular inequality,

$$f(Y_1) + f(X_2) \geq f(X_1 \setminus X_2) + f(Y_2) \geq f(Y_1) + f(Y_2),$$

and therefore $f(X_2) \geq f(Y_2)$. Since $Y_1 \cup Y_2 \cup X_3 = X$, $|Y_1|+|Y_2|+|X_3| < |X_1|+|X_2|+|X_3|$, and $f(X_3) < f(X)$, we conclude that either $f(Y_1) \geq f(X)$ or $f(Y_2) \geq f(X)$. But both cases lead to the conclusion that either $f(X_1) \geq f(X)$ or $f(X_2) \geq f(X)$, a contradiction. $\square$

A partition $\mathcal{P}$ of V is called *titanic* with respect to f if every part of $\mathcal{P}$ is titanic with respect to f. The following lemmas are equivalent to a lemma by Geelen, Gerards, and Whittle [9, Proposition 4.4], which generalizes a result of Robertson and Seymour [22, Proposition (8.3)].

LEMMA 3.3. *Let* V *be a finite set and* f *be a symmetric submodular function on* 2^V *of branch-width* k. *Let* Q *be a nonempty titanic set with respect to* f. *Let* $y \in Q$. *Let* $\pi(x) = x$ *if* $x \notin Q$ *and* $\pi(x) = y$ *if* $x \in Q$. *If* $f(Q) \leq k$, *then the branch-width of* $f^{[\pi]}$ *is at most* k.

Proof. Suppose that the branch-width of $f^{[\pi]}$ is larger than k. Then by Theorem 3.1, there is an $f^{[\pi]}$-tangle $\mathcal{T}^{[\pi]}$ of order $k+1$. Let $\mathcal{T}' = \{\cup_{Z \in Y} Z : Y \in \mathcal{T}^{[\pi]}\}$.

We would like to construct an f-tangle $\mathcal{T}$ of order $k+1$ as follows:

$$\mathcal{T} = \left\{X \subseteq V : f(X) \leq k \text{ and either } X \cup Q \in \mathcal{T}' \text{ or } X \setminus Q \in \mathcal{T}'\right\}.$$

To show that $\mathcal{T}$ is an f-tangle of order $k+1$, it is enough to verify the three axioms (T1)–(T3).

For (T1), suppose that $f(X) \leq k$ and $X, V \setminus X \notin \mathcal{T}$. Since Q is titanic, either $f(X \cap Q) \geq f(Q)$ or $f(Q \setminus X) \geq f(Q)$. We may assume that $f(X \cap Q) \geq f(Q)$ by replacing X with $V \setminus X$ if necessary. By the submodular inequality,

$$f(X) + f(Q) \geq f(X \cup Q) + f(X \cap Q) \geq f(X \cup Q) + f(Q),$$

and therefore $f(X \cup Q) \leq f(X) \leq k$. Since $\mathcal{T}^{[\pi]}$ is an $f^{[\pi]}$-tangle, we know that either $X \cup Q \in \mathcal{T}'$ or $V \setminus (X \cup Q) \in \mathcal{T}'$. If $X \cup Q \in \mathcal{T}'$, then $X \in \mathcal{T}$. If $V \setminus (X \cup Q) = (V \setminus X) \setminus Q \in \mathcal{T}'$, then $V \setminus X \in \mathcal{T}$. So, (T1) is proved.

To show (T2), suppose that there are $X_1, X_2, X_3 \in \mathcal{T}$ such that $X_1 \cup X_2 \cup X_3 = V$. By Lemma 3.2, there exists $i \in \{1,2,3\}$ such that $f(X_i \cap Q) \geq f(Q)$. We may assume that $i = 1$. By the submodular inequality, we deduce that $f(X_1 \cup Q) \leq f(X_1) \leq k$. Since $\mathcal{T}'$ has no three sets whose union is V, we have $X_1 \cup Q \notin \mathcal{T}'$. Therefore, $X_1 \setminus Q \in \mathcal{T}'$ and $V \setminus (X_1 \cup Q) \in \mathcal{T}'$. By (T3) of $\mathcal{T}^{[\pi]}$, we have $V \setminus Q \notin \mathcal{T}'$. Since $f(Q) \leq k$, we have $Q \in \mathcal{T}'$ by (T1) of $\mathcal{T}^{[\pi]}$. However, $(V \setminus (X_1 \cup Q)) \cup (X_1 \setminus Q) \cup Q = V$, contradictory to the fact that $\mathcal{T}^{[\pi]}$ is an $f^{[\pi]}$-tangle.

To show (T3), suppose that $X = V \setminus \{v\} \in \mathcal{T}$ for some $v \in V$. Since $V, V \setminus Q \notin \mathcal{T}'$ and $V \setminus \{v\} \notin \mathcal{T}'$ when $v \notin Q$, we deduce that $v \notin Q$ and $V \setminus \{v\} \setminus Q \in \mathcal{T}'$. We know that $\{v\}, Q \in \mathcal{T}'$. Then the union of three sets $\{v\}, Q, V \setminus \{v\} \setminus Q$ is equal to V, a contradiction.

Now we conclude that $\mathcal{T}$ is an f-tangle of order $k+1$. However, this is a contradiction to Theorem 3.1, because we assumed that the branch-width of f is k. $\square$

LEMMA 3.4. *Let V be a finite set and f be a symmetric submodular function on 2^V of branch-width at most k. If $\mathcal{P}$ is a titanic partition of width at most k with respect to f, then the branch-width of $f^{\mathcal{P}}$ is at most k.*

Proof. Suppose that there is a counterexample. We pick a counterexample with a minimum number of parts having at least two elements. If all parts have exactly one element, then it is trivial.

Choose one member from each part of $\mathcal{P}$ and consider a function $\pi : V \to V$ that maps each element x of V to a representative of the part containing x. Then $[\pi] = \mathcal{P}$.

Let y be an element of V such that $|\pi_y| \geq 2$. Then let $\pi' : V \to V$ be a function such that

$$\pi'(x) = \begin{cases} \pi(x) & \text{if } x \notin \pi_y, \\ x & \text{if } x \in \pi_y. \end{cases}$$

By definition, $[\pi'] = \{\pi_x : x \notin \pi_y\} \cup \{\{x\} : x \in \pi_y\}$. Since the number of parts in $[\pi']$ having at least two elements is strictly smaller than that of $[\pi] = \mathcal{P}$ and $[\pi']$ is a titanic partition of width at most k, we know that the branch-width of $f^{[\pi']}$ is at most k.

Then $Q = \{\{x\} : x \in \pi_y\}$ is titanic with respect to $f^{[\pi']}$, because $\mathcal{P}$ is a titanic partition and $\pi_y \in \mathcal{P}$. In addition, $f^{[\pi']}(Q) = f(\pi_y) \leq k$. Therefore, by Lemma 3.3, the branch-width of $f^{[\pi']}$ is at most k. This contradicts the assumption that $\mathcal{P}$ is chosen as a counterexample with a minimum number of parts with more than one element. ∎

4. Replacing each part by a gadget. The purpose of this section is to show how a partitioned matroid may be "modeled" by an ordinary matroid having the same branch-width.

LEMMA 4.1. *Let M be a matroid and T be a subset of $E(M)$. If $|T|+1 > \lambda_M(T)$, then there is $e \in T$ such that one of the following is true:*

1. *$\lambda_{M/e}(X) = \lambda_M(X)$ for all $X \subseteq E(M) \setminus T$, or*
2. *$\lambda_{M\setminus e}(X) = \lambda_M(X)$ for all $X \subseteq E(M) \setminus T$.*

Proof. Let X be a subset of $E(M) \setminus T$. If there is an element $e \in T$ that is not spanned by $E(M) \setminus T$, then $\mathrm{r}_{M/e}(X) = \mathrm{r}_M(X)$. Therefore, $\lambda_{M/e}(X) = \mathrm{r}_{M/e}(X) + \mathrm{r}_{M/e}\big(E(M)\setminus(\{e\}\cup X)\big) - \mathrm{r}(M/e) + 1 = \mathrm{r}_M(X) + \mathrm{r}_M\big(E(M)\setminus X\big) - \mathrm{r}_M(\{e\}) - \big(\mathrm{r}(M) - \mathrm{r}_M(\{e\})\big) + 1 = \lambda_M(X)$.

So, we may assume that $E(M)\setminus T$ spans T. Since $|T|+1 > \lambda_M(T) = \mathrm{r}_M(T)+1$, T is dependent in M, and hence in the dual matroid M^* the set T is not spanned by $E(M^*) \setminus T$. We apply the previous argument to M^*. (Note that $(M \setminus e)^* = M^*/e$ and $\lambda_{M^*} \equiv \lambda_M$.) ∎

COROLLARY 4.2. *Let M be a matroid, and let T be a subset of $E(M)$. Then there exist disjoint subsets C, D of T such that $\lambda_{M/C\setminus D}(T \setminus (C \cup D)) = |T \setminus (C \cup D)| + 1$, and $\lambda_{M/C\setminus D}(X) = \lambda_M(X)$ for all $X \subseteq E(M) \setminus T$.*

We aim to transform a partitioned matroid $(M, \mathcal{P})$ to another partitioned matroid $(M^{\#}, \mathcal{P}^{\#})$, such that they have the same branch-width and $\mathcal{P}^{\#}$ is a titanic partition with respect to $\lambda_{M^{\#}}$. To do so, we use an amalgam operation on matroids, described in section 2.

Let $(M, \mathcal{P})$ be a partitioned matroid. We may assume each part T of $\mathcal{P}$ satisfies $\lambda_M(T) = |T| + 1$ if $|T| > 1$, because otherwise we can contract or delete elements in T while preserving $\mathrm{bw}(M, \mathcal{P})$ by Corollary 4.2. This means that $M \restriction T$ is a free matroid. For each part T of $(M, \mathcal{P})$, if $|T| > 1$, then we define a matroid U_T (a *titanic* *gadget* of T) as a rank-$|T|$ uniform matroid on the ground set $E_T = E(U_T)$ such that $|E_T| = 3|T| - 2$, $E(M) \cap E_T = T$, and $E_T \cap E_{T'} = \emptyset$ if $T' \neq T$ is a part of $\mathcal{P}$ and $|T'| > 1$. Since $M \restriction T = U_T \restriction T$ is a free matroid, an amalgam of M and U_T exists by Lemma 2.2.

LEMMA 4.3. *Let M be a matroid and T be a subset of $E(M)$ such that $\lambda_M(T) = |T| + 1$. Let M' be an amalgam of M and U_T. Then the following are true:*

1. *If $T \subseteq X \subseteq E(M')$, then $\mathrm{r}_M(X \cap E(M)) = \mathrm{r}_{M'}(X)$.*
2. *$\lambda_M(X) = \lambda_{M'}(X)$ for all $X \subseteq E(M) \setminus T$.*
3. *The set $E(U_T)$ is titanic in the matroid M'.*

Proof. 1. Because $M' \restriction E(M) = M$, we have $\mathrm{r}_M(X \cap E(M)) = \mathrm{r}_{M'}(X \cap E(M)) \leq \mathrm{r}_{M'}(X)$.

Since T is independent in U_T, we can pick a maximally independent subset I of X in M' such that $T \subseteq I$. Since $M' \restriction E(U_T) = U_T$, the set $I \cap E(U_T)$ is independent in U_T, and therefore $I \cap E(U_T) = T$. So, $I \subseteq E(M)$. Therefore, $\mathrm{r}_M(X \cap E(M)) \geq |I| = \mathrm{r}_{M'}(X)$.

2. Let $Y = E(M') \setminus X$. We note that $E(U_T)$ is a subset of Y. By definition,

$$\lambda_M(X) = \mathrm{r}_M(X) + \mathrm{r}_M(Y \cap E(M)) - \mathrm{r}(M) + 1,$$
$$\lambda_{M'}(X) = \mathrm{r}_{M'}(X) + \mathrm{r}_{M'}(Y) - \mathrm{r}(M') + 1.$$

Since $M' \restriction E(M) = M$, we have $\mathrm{r}_M(X) = \mathrm{r}_{M'}(X)$. By 1, $\mathrm{r}_M(Y \cap E(M)) = \mathrm{r}_{M'}(Y)$ and $\mathrm{r}(M') = \mathrm{r}(M)$. Thus $\lambda_M(X) = \lambda_{M'}(X)$.

3. We claim that if X is a subset of $E(U_T)$ and $|X| \geq |T|$, then $\lambda_{M'}(X) \geq \lambda_{M'}(E(U_T))$. Since U_T is a uniform matroid of rank $|T|$, we have

$$\mathrm{r}_{M'}(X) = |T| = \mathrm{r}_{M'}(E(U_T)),$$
$$\mathrm{r}_{M'}(E(M') \setminus X) \geq \mathrm{r}_{M'}(E(M') \setminus E(U_T)).$$

Therefore, $\lambda_{M'}(X) \geq \lambda_{M'}(E(U_T))$.

Now suppose that X_1, X_2, X_3 are pairwise disjoint subsets of $E(U_T)$. Then there is $i \in \{1, 2, 3\}$ such that $|X_i| \geq \lceil |E(U_T)|/3 \rceil = |T|$ and therefore $\lambda_{M'}(X_i) \geq \lambda_{M'}(E(U_T))$. Therefore, $E(U_T)$ is titanic in M'. □

Using Corollary 4.2, we first obtain a minor M_0 of M such that $\lambda_{M_0}(T \cap E(M_0)) = |T \cap E(M_0)| + 1$ for all parts $T \in \mathcal{P}$, and if a subset X of $E(M)$ satisfies that $X \cap T \in \{\emptyset, T\}$ for all parts $T \in \mathcal{P}$, then $\lambda_{M_0}(X \cap E(M_0)) = \lambda_M(X)$. Let $\mathcal{P}_0$ be the partition of $E(M_0)$ induced by $\mathcal{P}$. Then we deduce from Corollary 4.2 that the branch-width of $(M, \mathcal{P})$ is equal to the branch-width of $(M_0, \mathcal{P}_0)$. However, the branch-width of the matroid M_0 may still be different from the branch-width of the partitioned matroid $(M_0, \mathcal{P}_0)$.

In the following theorem, we will extend $(M_0, \mathcal{P}_0)$ by amalgamating uniform matroids in the fashion of Lemma 4.3 so that the obtained partitioned matroid $(M^\#, \mathcal{P}^\#)$ has the same branch-width as the matroid $M^\#$ itself.

THEOREM 4.4. *Let $(M_0, \mathcal{P}_0)$ be a partitioned matroid, and let $T_1, T_2, \ldots, T_m$ be the parts of $\mathcal{P}_0$ having at least two elements. Assume that $\lambda_{M_0}(T_i) = |T_i| + 1$ for every $i \in \{1, 2, \ldots, m\}$. For all $i = 1, 2, \ldots, m$, let M_i be an amalgam of M_{i-1} and U_{T_i}. Then the branch-width of M_m is equal to the branch-width of the partitioned matroid $(M_0, \mathcal{P}_0)$.*

We call the resulting $M^\# = M_m$ the *normalized matroid* of the partitioned matroid $(M_0, \mathcal{P}_0)$.

Proof. Let $\mathcal{P}_i = (\mathcal{P}_{i-1} \setminus \{T_i\}) \cup \{E(U_{T_i})\}$. By 2. of Lemma 4.3, the branch-width of $(M_i, \mathcal{P}_i)$ is equal to that of $(M_{i-1}, \mathcal{P}_{i-1})$, and therefore the branch-width of $(M_m, \mathcal{P}_m)$ is equal to the branch-width of $(M_0, \mathcal{P}_0)$. By 3. of Lemma 4.3, $\mathcal{P}_m$ is a titanic partition. Let k be the branch-width of M_m. It is easy to see that the branch-width of the uniform matroid U_{T_i} is $|T_i| + 1 = \lambda_{M_m}(E(U_{T_i}))$. Since U_{T_i} is a minor of M_m, the branch-width of M_m is at least $|T_i| + 1$ for all i, and therefore the width of $\mathcal{P}_m$ is at most k. We conclude that the branch-width of $(M_m, \mathcal{P}_m)$ is at most k by Lemma 3.4.

To finish the proof, we need to show that the branch-width of $(M_m, \mathcal{P}_m)$ is at least k. Let (T, μ) be the branch-decomposition of $(M_m, \mathcal{P}_m)$ of width at most w. From (T, μ), we would like to obtain a branch-decomposition (T', μ') of M_m whose width is at most w as follows. Let v_i be the leaf of T corresponding to $E(U_{T_i})$. We prepare a rooted binary tree with a bijection from its leaves to $E(U_{T_i})$ and then identify the root with v_i. Let T' be the new tree obtained by the above process for all i. A bijection μ' from $E(M_m)$ to leaves of T' is easily obtained from the above process. Since $\lambda_{M_m}(X) = |X|+1 \leq \lambda_{M_m}(E(U_{T_i})) \leq w$ for all $X \subseteq E(U_{T_i})$, the width of (T', μ') is at most w. ∎

5. Branch-decompositions of represented partitioned matroids. We now specialize the above ideas to the case of representable matroids. We aim to provide an efficient algorithm for testing small branch-width on such partitioned matroids. For the rest of our paper, a *represented matroid* is the vector matroid of a (given) matrix over a *fixed* finite field. We also write an $\mathbb{F}$-*represented* matroid to explicitly refer to the field $\mathbb{F}$. In other words, an $\mathbb{F}$-represented matroid is a set of points (a *point configuration*) in a (finite) projective geometry over $\mathbb{F}$. To begin, we restate a previous result of Hliněný.

THEOREM 5.1 (see [13, Theorems 4.14 and 5.5]). *Let $k > 1$ be a constant, and let $\mathbb{F}$ be a fixed finite field. There is a parametrized algorithm that, for a given matroid M represented by an $r \times n$ matrix over $\mathbb{F}$ for some $r \leq n$, tests whether the branch-width of M is at most k in time $O(n^3)$.*

We remark that the algorithm for Theorem 5.1 in [13] is purely of theoretical importance. First, it uses the result of Geelen et al. [8] stating that minor-minimal matroids of branch-width larger than k have size at most $(6^k - 1)/5$. Second, it tests whether the input matroid contains such a minor-minimal matroid as a minor by encoding the question in a monadic second-order logic formula on matroids and using a generic algorithm to solve an MSOL formula for $\mathbb{F}$-represented matroids of branch-width at most k. Since our algorithm will use Theorem 5.1, it will be purely theoretical and difficult to implement.

Not all matroids are representable over $\mathbb{F}$. Particularly, in the construction of the normalized matroid (Theorem 4.4) we apply amalgams with uniform matroids which need not be $\mathbb{F}$-representable. To achieve their representability, we extend the field $\mathbb{F}$ in a standard algebraic way.

Remark 5.2. Let $\mathbb{F}$ be a fixed finite field with q elements and d be a fixed positive integer. We assume that one can perform arithmetic operations over $\mathbb{F}$ in time depending only on q. Then, one can construct by brute force an extension (finite) field $\mathbb{F}' = \mathbb{F}(\alpha)$ with q^d elements by searching for a polynomial root α of degree d over $\mathbb{F}$. This can be done by searching through all polynomials in $\mathbb{F}[x]$ of degree d for the irreducible ones.

LEMMA 5.3. *The n-element rank-r uniform matroid $U_{r,n}$ is representable over a (finite) field $\mathbb{F}$ if $|\mathbb{F}| \geq n - 1$.*

Proof. Let $|\mathbb{F}| = q$. It is trivial to represent $U_{0,n}, U_{1,n}, U_{n-1,n}$, or $U_{n,n}$ over every field. Furthermore, standard arguments of projective geometry show that a so-called normal rational curve in a projective geometry over $\mathbb{F}$ is a representation of the uniform matroid $U_{r,\,q+1}$, for every $1 < r < q$; see, for instance, [12, section 3]. Although it is not useful in our context, it is worth noting that the size bound $q+1$ is almost optimal in most cases. Finally, if $q+1 > n$, then we delete arbitrary $q+1-n$ points from the representation to get $U_{r,n}$. $\square$

Recall the notion of a matroid amalgam from section 2 from the perspective of represented matroids. We shall use the following proposition.

PROPOSITION 5.4. *Let M_1, M_2 be two matroids such that $E(M_1) \cap E(M_2) = T$ and $M_1 \restriction T = M_2 \restriction T$. If both M_1, M_2 are $\mathbb{F}$-represented, and the matroid $M_1 \restriction T$ has a unique $\mathbb{F}$-representation up to linear transformations, then there exists an amalgam of M_1 and M_2 which is also $\mathbb{F}$-represented.*

Proof. We denote by $[\boldsymbol{A}_1 \,|\, \boldsymbol{A}_T]$ the matrix over $\mathbb{F}$ representing M_1, where the columns of $\boldsymbol{A}_T$ represent the set T. Analogously we denote by $[\boldsymbol{A}_2 \,|\, \boldsymbol{A}'_T]$ the matrix representing M_2. Since $M_1 \restriction T = M_2 \restriction T$ has a unique $\mathbb{F}$-representation, there is a linear transformation carrying $\boldsymbol{A}'_T$ onto $\boldsymbol{A}_T$. This transformation takes a whole $[\boldsymbol{A}_2 \,|\, \boldsymbol{A}'_T]$ to an equivalent matrix $[\boldsymbol{A}'_2 \,|\, \boldsymbol{A}_T]$ representing the same matroid M_2. It is now trivial to verify that the matroid M which is $\mathbb{F}$-represented by the composed matrix $[\boldsymbol{A}_1 \,|\, \boldsymbol{A}_T \,|\, \boldsymbol{A}'_2]$ is an amalgam of M_1 and M_2. $\square$

We remark that representable matroids typically do have inequivalent vector representations, but we are using Proposition 5.4 only in the case when $M_1 \restriction T$ is a free matroid which clearly has a unique $\mathbb{F}$-representation for every field $\mathbb{F}$.

We now outline a simple fixed-parameter tractable algorithm for testing branch-width $\leq k$ on $\mathbb{F}$-represented partitioned matroids.

ALGORITHM 5.5. Testing branch-width of a represented partitioned matroid.

Parameters: A finite field $\mathbb{F}$, and a positive integer k.

Input: A rank-r matrix $\boldsymbol{A} \in \mathbb{F}^{r \times n}$ and a partition $\mathcal{P}$ of the columns of $\boldsymbol{A}$. (Assume $n \geq 2$.)

Output: For the vector matroid $M = M(\boldsymbol{A})$ on the columns of $\boldsymbol{A}$ partitioned by $\mathcal{P}$, a correct answer whether the branch-width of $(M, \mathcal{P})$ is at most k.

1. First, we extend $\mathbb{F}$ to a nearest field $\mathbb{F}'$ such that $|\mathbb{F}'| \geq 3k-6$ (Remark 5.2).
2. If the width of the partition $\mathcal{P}$ in given $(M, \mathcal{P})$ is more than k, then we answer NO.
3. Otherwise, we directly construct the normalized matroid $M^{\#}$ (Theorem 4.4), together with its vector representation over $\mathbb{F}'$ (Lemma 5.3 and Proposition 5.4).
4. Finally, we use Theorem 5.1 to test whether the branch-width of $M^{\#}$ is at most k.

Hence, we can conclude with the following theorem.

THEOREM 5.6. *Let $k > 1$ be a constant, and let $\mathbb{F}$ be a fixed finite field. There is a parametrized algorithm that, for a partitioned matroid $(M, \mathcal{P})$ represented over $\mathbb{F}$, tests in time $O(|E(M)|^3)$ whether the branch-width of $(M, \mathcal{P})$ is at most k.*

Proof. We refer to Algorithm 5.5. Denote $n = |E(M)|$. In the first step we find the extension field $\mathbb{F}'$ which takes only constant time by Remark 5.2. Since $\mathbb{F}' \supseteq \mathbb{F}$, we do not need to touch the input vector representation of M. Step 2. of Algorithm 5.5 is trivial.

The technical part comes in step 3. of Algorithm 5.5. For each part $X \in \mathcal{P}$ of more than one element, we compute the intersection of the spans of X and of $E(M) \setminus X$, called the *guts* of the separation $(X, E(M) \setminus X)$, in the representation of M. The reader should understand that we deal with represented matroids, that means we

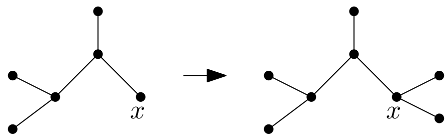

Fig. 3. *Splitting x.*

compute with actual vectors and subspaces in a projective geometry over $\mathbb{F}'$. If the dimension $\lambda_M(X)-1$ of these guts was at least k, then each branch-decomposition of $(M,\mathcal{P})$ should have width at least $\lambda_M(X)>k$, and we have answered NO in step 2. of Algorithm 5.5 in such a case. Therefore, the dimension of the guts of $(X, E(M)\setminus X)$ is bounded by a constant k, and a set T of its independent generator vectors can be easily computed in time $O(n^2)$ (see, e.g., [13, Algorithm 4.4]), per each part of $\mathcal{P}$.

In order to get the situation anticipated in Theorem 4.4—each part representing a free matroid—we replace the vectors of each nonsingleton part $X\in\mathcal{P}$ by the respective vectors T computed in the previous paragraph. Let the resulting represented partitioned matroid be denoted by $(M_0,\mathcal{P}_0)$, and note that $\mathrm{bw}(M_0,\mathcal{P}_0)=\mathrm{bw}(M,\mathcal{P})$. For each $T\in\mathcal{P}_0$ we construct an $\mathbb{F}'$-representation of the titanic gadget (uniform matroid) U_T from section 4 using Lemma 5.3, and then construct an amalgam of M_0 with U_T according to Proposition 5.4. Since U_T is of bounded size, this last step can be computed in time proportional to the vector length $O(n)$, per each part of $\mathcal{P}_0$.

After processing all $O(n)$ nonsingleton parts in $\mathcal{P}_0$ by the previous procedure, we get a vector $\mathbb{F}'$-representation of the normalized matroid $M^{\#}$ for $(M_0,\mathcal{P}_0)$. By Theorem 4.4, $\mathrm{bw}(M^{\#})=\mathrm{bw}(M_0,\mathcal{P}_0)=\mathrm{bw}(M,\mathcal{P})$. So, we call the algorithm of Theorem 5.1 to determine whether the branch-width of $M^{\#}$ (as well as the branch-width of $(M,\mathcal{P})$) is at most k. This takes only $O(n^3)$ time since both k, $\mathbb{F}'$ are of bounded size here. $\square$

We are now able to test branch-width of partitioned matroids. We show how this result can be extended to finding an appropriate branch-decomposition.

Theorem 5.7. *Let $\mathcal{K}$ be a class of matroids, and let k be an integer. If there is an $f(|E(M)|,k)$-time algorithm to decide whether a partitioned matroid $(M,\mathcal{P})$ has branch-width at most k for every pair of a matroid $M\in\mathcal{K}$ and a partition $\mathcal{P}$ of $E(M)$, then a branch-decomposition of the partitioned matroid $(M,\mathcal{P})$ of width at most k, if it exists, can be found in time $O\big(|\mathcal{P}|^3\cdot f(|E(M)|,k)\big)$.*

The idea of the proof is due to Jim Geelen, published by Oum and Seymour in [20]. Details of the algorithm follow. Clearly, we may assume that the branch-width of the partitioned matroid $(M,\mathcal{P})$ in question is at most k, since otherwise there is nothing to compute. A *splitting* of a leaf x of a subcubic tree is an operation that creates two new leaves and makes them adjacent to x (see Figure 3).

Algorithm 5.8. Computing a branch-decomposition of a partitioned matroid.
Oracle: A subroutine for testing the branch-width of a partitioned matroid $(M,\mathcal{P})$, where M belongs to a given class $\mathcal{K}$, and $\mathcal{P}$ is a partition of $E(M)$.
Input: A partitioned matroid $(M,\mathcal{P})$ of branch-width at most k, where $M\in\mathcal{K}$.
Output: A branch-decomposition of $(M,\mathcal{P})$ of width at most k.

1. If $|\mathcal{P}|\le 2$, then it is trivial to output a branch-decomposition.
2. We find a pair X,Y of disjoint parts of $\mathcal{P}$ such that a partitioned matroid $(M,(\mathcal{P}\setminus\{X,Y\})\cup\{X\cup Y\})$ has branch-width at most k. Let $\mathcal{P}'=(\mathcal{P}\setminus\{X,Y\})\cup\{X\cup Y\}$.
3. Let (T',μ') be the branch-decomposition of $(M,\mathcal{P}')$ of width at most k ob-

tained by calling this algorithm recursively.

4. Let T be a tree obtained from T' by splitting the leaf $\mu'(X \cup Y)$ into two leaves which we denote by $\mu(X)$ and $\mu(Y)$. Let $\mu(Z) = \mu'(Z)$ for all $Z \in \mathcal{P} \setminus \{X, Y\}$. We output (T, μ) as the resulting branch-decomposition of (M, P) of width at most k.

Proof of Theorem 5.7. We start by estimating the running time of the algorithm. At each level of recursion, we call our oracle (the decision algorithm) at most $\binom{|\mathcal{P}|}{2} = O(|\mathcal{P}|^2)$ times. The depth of recursion is $|\mathcal{P}| - 1$, and therefore the number of calls to the decision algorithm is at most $O(|\mathcal{P}|^3)$. Thus, the running time of the algorithm is $O\big(|\mathcal{P}|^3 \cdot f(|E(M)|, k)\big)$.

It remains to show correctness of the algorithm. It is obvious that in every subcubic tree with at least three leaves, there are two leaves that have a common neighbor. Suppose that (T, μ) is a branch-decomposition of $(M, \mathcal{P})$ of width at most k. Then there are two leaves $\mu(X)$ and $\mu(Y)$ having a common neighbor z in T. It is easy to see that if we remove $\mu(X)$ and $\mu(Y)$ from T and map $X \cup Y$ to z by μ, then (T, μ) is a branch-decomposition of $(M, \mathcal{P}')$ of width at most k. Therefore, the branch-width of $(M, \mathcal{P}')$ is at most k.

Conversely, suppose that (T', μ') is the branch-decomposition of $(M, \mathcal{P}')$. Since $(M, \mathcal{P})$ has a branch-width of at most k, we know that $\lambda_M(X) \le k$ and $\lambda_M(Y) \le k$. Thus (T, μ) is a branch-decomposition of $(M, \mathcal{P})$ of width at most k. $\square$

COROLLARY 5.9. *For a constant k and a fixed finite field $\mathbb{F}$, we can find a branch-decomposition of a given $\mathbb{F}$-represented matroid M of branch-width at most k, if it exists, in time $O(|E(M)|^6)$.*

Proof. Let $\mathcal{P} = \{\{x\} : x \in E(M)\}$. Then the branch-decomposition of M is one-to-one correspondent to the branch-decomposition of a partitioned matroid $(M, \mathcal{P})$. By Theorem 5.6, the problem of deciding whether branch-width is at most k can be done in time $O(|E(M)|^3)$, and therefore we can construct the branch-decomposition of width at most k in time $O(|\mathcal{P}|^3 \cdot |E(M)|^3) = O(|E(M)|^6)$ by Theorem 5.7. $\square$

Remark 5.10. One can actually improve the bound in Theorem 5.7 to $O\big(|\mathcal{P}|^2 \cdot f(|E(M)|, k)\big)$ time. The basic idea is the following: At the first level of recursion we find not only one pair of parts but a maximal set of disjoint pairs of parts from $\mathcal{P}$ that can be joined (pairwise) while keeping the branch-width at most k. This again requires $O(|\mathcal{P}|^2)$ calls to the decision algorithm. At the deeper levels of recursion we then use the same approach but process only such pairs of parts that contain one joined at the previous level. An amortized complexity analysis shows that only additional $O(|\mathcal{P}|^2)$ calls to the decision algorithm are necessary at all subsequent levels of recursion together. Detailed arguments of this approach can be found further in Theorem 6.7, part (III) of the proof.

6. Faster algorithm for branch-decompositions. Even with Remark 5.10 in account, the approach of section 5 results in an $O(n^5)$ parametrized algorithm for constructing a branch-decomposition of an n-element matroid represented over a fixed finite field. That is still far from the cubic running time of the decision algorithm in Theorem 5.1. Although not straightforwardly, we are able to improve the running time of our constructive algorithm to asymptotically match cubic time of that and [5].

It is the purpose of this section to present a detailed analysis of such a faster implementation of Algorithm 5.8 using Remark 5.10. For that we have to dive into fine details of the ideas and algorithms in [13]. To be consistent, we also adopt the writing style of [13] for this section and recall a few necessary technical definitions here. More technical details are given along with a formal setting of Algorithm 6.6. We first give a brief informal outline of our faster algorithm, which seems necessary since Algorithm 6.6 itself is quite long and complex. We also collect formal statements of useful subroutines from [13], and then we conclude with the main algorithm (Algorithm 6.6) and a proof of its correctness.

One key point in the approach of [13, 14] is that a matroid M, which is $\mathbb{F}$-represented by a matrix $\boldsymbol{A}$, is equivalent to a projective point configuration over $\mathbb{F}$, and therefore we can speak about M in schematic geometric terms. Briefly speaking, a *parse tree* [14] of an $\mathbb{F}$-represented matroid M is a rooted tree $\mathcal{T}$, with at most two children per node, such that the leaves of $\mathcal{T}$ hold nonloop elements of M represented by points of a projective geometry over $\mathbb{F}$, or loops of M represented by the empty set. The internal nodes of $\mathcal{T}$, on the other hand, hold suitable "composition operators over $\mathbb{F}$."

Such a *composition operator* $\odot$ is a configuration in the projective geometry over $\mathbb{F}$ such that $\odot$ has three subspaces (possibly empty) distinguished as its *boundaries*; two of which are used to "glue" the matroid elements represented in the left and right subtrees, respectively, together. The third one, *upper boundary*, is then used to "glue" this node further up in the parse tree $\mathcal{T}$. Our "glue" operation, called the boundary sum by Hliněný [14], is analogous to the amalgam of matroids in Proposition 5.4. The ranks of adjacent boundaries of two composition operators in $\mathcal{T}$ must be equal for "gluing." A parse tree $\mathcal{T}$ is $\le t$*-boundaried* if all composition operators in $\mathcal{T}$ have boundaries of rank at most t. Such a parse tree actually gives a branch-decomposition of width at most $t+1$ and vice versa, by [14, Theorem 3.8]. See [14] for additional details.

We will use the following algorithm shown by Hliněný [13].

Algorithm 6.1 (see [13, Algorithm 4.1]). Computing a parse tree of a represented matroid.

Parameters: A finite field $\mathbb{F}$, and a positive integer k.

Input: A rank-r matrix $\boldsymbol{A} \in \mathbb{F}^{r\times n}$ representing a matroid M over $\mathbb{F}$. (Assume $n \ge 2$.)

Output: Computed in time $O(n^3)$; either a $\le 3k$-boundaried parse tree $\mathcal{T}$ of the matroid M, or a proof that the branch-width of M is more than $k+1$.

The basic idea of Algorithm 6.1 is as follows: We start with a basis $\boldsymbol{I}_r$ of the input matrix $\boldsymbol{A} = [\boldsymbol{I}_r \mid \boldsymbol{A}'] \in \mathbb{F}^{r\times n}$, and assign an arbitrary parse tree $\mathcal{T}$ to $\boldsymbol{I}_r$. Then we are adding, one by one, the remaining elements of $\boldsymbol{A}'$ arbitrarily to $\mathcal{T}$. Whenever the largest boundary rank (the *width*) of $\mathcal{T}$ exceeds certain constant threshold, say $10k$, we "compress" $\mathcal{T}$ into a new parse tree $\mathcal{T}'$ of width at most $3k$ again. However, if the compression step fails, then we have a certificate that the branch-width of $M(\boldsymbol{A})$ is more than $k+1$. The compression routine, [13, Algorithm 4.1, step 3] and [13, Lemma 4.13], is crucial also in our new algorithm, and thus we restate it explicitly here.

Algorithm 6.2 (see [13, Algorithm 4.1]). "Compressing" a parse tree of bounded width.

Parameters: A finite field $\mathbb{F}$, and a positive integer k.

Input: A $\le ck$-boundaried parse tree $\mathcal{T}$ (of an n-element matroid M represented over $\mathbb{F}$), where $c > 3$ is a fixed constant, say $c = 10$.

Output: Computed in time $O(n^2)$; either a $\le 3k$-boundaried parse tree $\mathcal{T}'$ of the same matroid M, or a proof that the branch-width of M is more than $k+1$.

Outline of the faster algorithm. Before giving full details of our new Algorithm 6.6 for computing a branch-decomposition of a represented partitioned matroid, we sketch its main ideas with respect to the previous Algorithms 6.2 and 6.1.

Parameters: A finite field $\mathbb{F}$, and a positive integer k.

Input: A rank-r matrix $\boldsymbol{A} \in \mathbb{F}^{r\times n}$ and a partition $\mathcal{P}$ of the columns of $\boldsymbol{A}$. (Assume $n \geq 2$.)

Initial phase. Let $M = M(\boldsymbol{A})$ be the vector matroid on the columns of $\boldsymbol{A}$. We run Algorithm 5.5 to obtain the represented normalized matroid $M^{\#}$ for our M and $\mathcal{P}$, and to decide whether $\mathrm{bw}(M^{\#}) \leq k$. In the positive case, we also call Algorithm 6.1 to obtain a $\leq 3(k-1)$-boundaried parse tree $\mathcal{T}$ for the matroid $M^{\#}$.

Construction phase. We construct a branch-decomposition of $(M, \mathcal{P})$ as a rooted forest D which is initialized to the set of disconnected nodes $\mathcal{P}_1 := \mathcal{P}$. A *rooted forest* is a forest in which every connected component has a specified vertex called a *root*.

In the first iteration, we find an inclusion-wise maximal collection of pairwise disjoint pairs $\{X_i, Y_i\}$, $i = 1, 2, \ldots, c$, of parts of $\mathcal{P}_1$ such that the branch-width of $(M, \mathcal{P}'_1)$ is at most k, where $\mathcal{P}'_1$ is obtained from $\mathcal{P}_1$ via replacing parts X_i, Y_i with $X_i \cup Y_i$ for $i = 1, 2, \ldots, c$. The meaning is that these pairs $\{X_i, Y_i\}$ simultaneously correspond to pairs of leaves of distance two in some branch-decomposition of width $\leq k$. We let $\mathcal{Q}_1 = \{X_i \cup Y_i : i = 1, 2, \ldots, c\}$, and add the new nodes from $\mathcal{Q}_1$ to our forest D connected to the appropriate X_i, Y_i's. Then we set $\mathcal{P}_1 := \mathcal{P}'_1$.

In each of the subsequent iterations, we again find an inclusion-wise maximal collection of pairwise disjoint pairs $\{X_i, Y_i\}$, $i = 1, 2, \ldots, c$, of parts of $\mathcal{P}_1$ such that the branch-width of $(M, \mathcal{P}'_1)$ is at most k, but now we *restrict* $Y_i \in \mathcal{Q}_1$ (whereas $X_i \in \mathcal{P}_1$). Then we continue analogously to the first iteration, until D becomes a tree.

Output: Either a branch-decomposition D of $(M, \mathcal{P})$ of width at most k, or the answer NO if $\mathrm{bw}(M, \mathcal{P}) > k$.

There are two important points to notice in the above outline, which make the whole algorithm run in time $O(n^3)$. First, we only consider altogether $O(n^2)$ pairs $\{X_i, Y_i\}$ of parts for possible merging, throughout all iterations of the algorithm. A formal proof of this fact in included as part (III) of the proof of Theorem 6.7. Second, to be able to run a quick test whether the branch-width of $(M, \mathcal{P}'_1)$ exceeds k or not, we need to maintain a certain "working" parse tree $\mathcal{T}_1$ of bounded width. Then, as noted already after Theorem 5.1, such a test can be done by looking for the excluded minors for branch-width at most k because each excluded minor has size at most $(6^k - 1)/5$, shown by Geelen et al. [8].

Theorem 6.3 (see [14, Theorem 6.1] and [13, Corollary 5.4]). *Let $t > 1$ be a constant, and let $\mathbb{F}$ be a fixed finite field. There is a parametrized algorithm that, for every $k \leq t$ and for a given $\leq t$-boundaried parse tree $\mathcal{T}$ of an n-element matroid M, decides whether the branch-width of M is at most k in time $O(n)$.*

We have skipped, for simplicity, an explicit reference to the "working" parse tree $\mathcal{T}_1$ in the above outline; however, one can roughly say that $\mathcal{T}_1$ is maintained as a parse tree of the normalization of the current partitioned matroid $(M, \mathcal{P}_1)$. This will be precise in Algorithm 6.6. It is essential that we keep the width of $\mathcal{T}_1$ bounded throughout the computation, for which we use to call Algorithm 6.2 after each of the $O(n)$ major updates to $\mathcal{T}_1$.

Therefore, to quickly test whether merging a pair of parts $X_i, Y_i \in \mathcal{P}_1$ increases the branch-width of $(M, \mathcal{P}_1)$ above k or not, we temporarily modify the parse tree $\mathcal{T}_1$

each time by replacing $W = X_i \cup Y_i$ with the titanic gadget (amalgamated according to section 4). As this (Algorithm 6.4) does not increase the width of $\mathcal{T}_1$ much, we then solve the task in time $O(n)$ using Theorem 6.3.

To make a precise statement of this procedure, we introduce an additional technical definition inspired by section 4: Let M be a matroid and $X \subseteq E(M)$. Let $F = E(M) \setminus X$ and Y be disjoint from $E(M)$. Assume M' is a matroid on $E(M) \cup Y$ such that $M' \restriction F = M \restriction F$, $\mathrm{r}_{M'}(X \cup Y) = \mathrm{r}_M(X)$ (i.e., Y is spanned by X), and $\lambda_{M' \setminus X}(Y) = |Y| + 1 = \lambda_M(X) > 1$. If a matroid N is an amalgam of $M' \setminus X$ and the titanic gadget U_Y, then we say that N is obtained from M by a (titanic) *normalization* of the set X. If, on the other hand, $\lambda_M(X) = 1$, then a normalization of the set X in M results in $M \setminus X$. The point is that, by Lemmas 3.3 and 4.3, part 3., the branch-width of N equals the branch-width of $(M, \mathcal{P}_X)$, where $\mathcal{P}_X = \big\{\{X\}\big\} \cup \big\{\{y\} : y \in E(M) \setminus X\big\}$.

ALGORITHM 6.4. Computing a titanic normalization of a point set on the parse tree.

Parameters: A finite field $\mathbb{F}$, and an integer $k \geq 1$. (We may assume that $|\mathbb{F}| \geq 3k-6$ as in Remark 5.2.)

Input: A $\leq (3k-1)$-boundaried parse tree $\mathcal{T}_1$ representing a matroid M_1 with n elements, and a set $W \subseteq E(M_1)$ such that $\lambda_{M_1}(W) = \ell \leq k$.

Output: A $\leq (3k+\ell-2)$-boundaried parse tree $\mathcal{T}_2$ of an $\mathbb{F}$-represented matroid M_2 such that M_2 can be obtained from M_1 by the normalization of W.

Algorithm 6.4 is an immediate extension of [13, Algorithm 4.9] for computing $\lambda_{M_1}(W)$. We describe it in terms of a projective geometry and the point configuration representing a matroid M_1 via the parse tree $\mathcal{T}_1$. If $\ell = 1$, then we return $\mathcal{T}_1$ without W, immediately.

At the beginning we make $\mathcal{T}_2$ a copy of $\mathcal{T}_1$. The idea is to "enlarge" all of the composition operators in $\mathcal{T}_2$ to fully contain the guts Γ (a projective subspace of rank $\ell - 1$ with a basis Y) of the separation $\big(W, E(M_1) \setminus W\big)$, and then to "glue" or amalgamate a decomposition of the titanic gadget $U_Y \simeq U_{\ell-1,3\ell-5}$ to the root of $\mathcal{T}_2$ so that it matches Y in Γ. For that we apply leaf-to-root dynamic programming on $\mathcal{T}_2$ with constant-time operations at each node.

At a node $x \in V(\mathcal{T}_2)$, we compute the subspace Σ_x of Γ spanned by the elements of W held in the leaves below x. Knowing $\Sigma_{x'}$ and $\Sigma_{x''}$ for the children x', x'' of x in $\mathcal{T}_2$, it is a constant-time manipulation to determine Σ_x using the composition operator $\odot$ at x. Notice that as our algorithm is set up, Σ_x is spanned by $\odot$. If the upper boundary of $\odot$ does not fully contain Σ_x, we enlarge it accordingly and also freely extend the matching boundary at the parent node of x. Note that $\Sigma_r = \Gamma$ will become the upper boundary of the root node r.

After finishing that computation, we take an arbitrary parse tree $\mathcal{T}_3$ of the titanic gadget (i.e., uniform matroid) $U_Y \simeq U_{\ell-1,3\ell-5}$, and add to $\mathcal{T}_2$ a new root node r' adjacent to the former root r of $\mathcal{T}_2$ and to the root of $\mathcal{T}_3$. The composition operator at r' "glues" U_Y directly to Σ_r. Finally, we strip from $\mathcal{T}_2$ all leaves holding the points of W. This is trivial since our definition of a parse tree allows nodes with only one descendant.

Since we use only constant-time operations at each node of $\mathcal{T}_2$, we conclude with the following lemma.

LEMMA 6.5. *Algorithm* 6.4 *computes correctly in time* $O(n)$.

We are now ready to restate the above algorithmic outline in a formal setting. Our notation of variables in Algorithm 6.6 essentially follows the outline, but we need

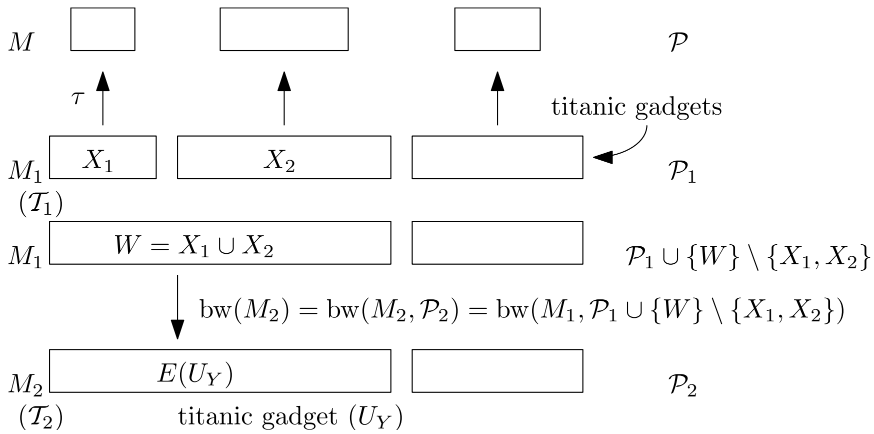

FIG. 4. *An illustration of Algorithm* 6.6.

a few more of them. For instance, $\mathcal{Q}_2$ at each round holds the set of all pairs of parts among which we are looking for the admissible ones. See also an informal hint in Figure 4.

ALGORITHM 6.6. Computing a branch-decomposition of a represented partitioned matroid.

Parameters: A finite field $\mathbb{F}$, and a positive integer k.

Input: A rank-r matrix $\boldsymbol{A} \in \mathbb{F}^{r\times n}$ and a partition $\mathcal{P}$ of the columns of $\boldsymbol{A}$. (Assume $n \geq 2$.)

Output: For the vector matroid $M = M(\boldsymbol{A})$ on the columns of $\boldsymbol{A}$, either a branch-decomposition of the partitioned matroid $(M, \mathcal{P})$ of width at most k, or the answer NO if $\mathrm{bw}(M, \mathcal{P}) > k$.

1. Using brute force, we extend the field $\mathbb{F}$ to a (nearest) finite field $\mathbb{F}'$ such that $|\mathbb{F}'| \geq 3k - 6$ (Remark 5.2 and Lemma 5.3).
2. We check whether $\mathrm{bw}(M, \mathcal{P}) \leq k$, using Algorithm 5.5. If $\mathrm{bw}(M, \mathcal{P}) > k$, then we answer NO. Otherwise we keep the *normalized matroid* $M^\#$ and its $\mathbb{F}'$-representation $\boldsymbol{A}^\#$ obtained at this step. We denote by $\mathcal{P}_1$ the (titanic) partition of $E(M^\#)$ corresponding to $\mathcal{P}$, and by $\tau(X) \in \mathcal{P}$ for $X \in \mathcal{P}_1$ the corresponding parts.
3. Calling Algorithm 6.1, we compute a $\leq 3(k-1)$-boundaried parse tree $\mathcal{T}$ for the matroid $M^\#$ which is $\mathbb{F}'$-represented by $\boldsymbol{A}^\#$ (regardless of $\mathcal{P}_1$).
4. We initially set $\mathcal{T}_1 := \mathcal{T}$, $\mathcal{Q}_1 := \emptyset$, $\mathcal{Q}_2 := \big\{\{X_1, X_2\} : X_1 \neq X_2, X_1, X_2 \in \mathcal{P}_1\big\}$, and create a new rooted forest D consisting so far of the set of disconnected nodes $\mathcal{P}_1$.
   Let M_1 (M_2) denote the matroid represented by $\mathcal{T}_1$ ($\mathcal{T}_2$, respectively) at each step. Then we repeat the following steps (a), (b), until $\mathcal{P}_1$ contains at most two parts:
   (a) While there is $\{X_1, X_2\} \in \mathcal{Q}_2$ such that $X_1, X_2 \in \mathcal{P}_1$, we perform the following steps:
      i. Let $\mathcal{Q}_2 := \mathcal{Q}_2 \setminus \big\{\{X_1, X_2\}\big\}$. Calling [13, Algorithm 4.9] in linear time, we compute connectivity value $\ell = \lambda_{M_1}(X_1 \cup X_2)$ over the parse tree $\mathcal{T}_1$. If $\ell > k$, then we continue this cycle again from (a).
      ii. We call Algorithm 6.4 on $\mathcal{T}_1$ and $W = X_1 \cup X_2$ to compute a $\leq (3k + \ell - 2)$-boundaried parse tree $\mathcal{T}_2$ of a matroid M_2 which is obtained

by a titanic normalization of the part W.

By Lemmas 3.4 and 4.3 we have $\mathrm{bw}\big(M_1,\ \mathcal{P}_1 \cup \{W\} \setminus \{X_1, X_2\}\big) = \mathrm{bw}(M_2)$.

iii. We check whether branch-width $\mathrm{bw}(M_2) \le k$ by applying Theorem 6.3. If $\mathrm{bw}(M_2) > k$, then we continue this cycle again from (a).

iv. If $\mathrm{bw}\big(M_1,\ \mathcal{P}_1 \cup \{W\} \setminus \{X_1, X_2\}\big) = \mathrm{bw}(M_2) \le k$, then we add a *new node* $Z = E(U_Y)$ (U_Y given by the normalization of W in Algorithm 6.4) adjacent to X_1 and X_2 in the rooted forest D, and make Z the root for its component. We update $\mathcal{P}_1 := \mathcal{P}_2 = \mathcal{P}_1 \cup \{Z\} \setminus \{X_1, X_2\}$, and $\mathcal{Q}_1 := \mathcal{Q}_1 \cup \{Z\}$.

v. Last, by calling Algorithm 6.2 on $\mathcal{T}_2$, we compute in a *new* $\le 3(k-1)$-*boundaried parse tree* $\mathcal{T}_3$ for the matroid M_2, and set $\mathcal{T}_1 := \mathcal{T}_3$.

(b) When the "while" cycle (4.a) is finished, we set $\mathcal{Q}_2 := \big\{\{X_1, X_2\} : X_1 \neq X_2,\ X_1 \in \mathcal{P}_1,\ X_2 \in \mathcal{Q}_1\big\}$ and $\mathcal{Q}_1 := \emptyset$, and continue from (4.a).

5. Finally, if $|\mathcal{P}_1| = 2$, then we connect by an edge in D the two nodes $X_1, X_2 \in \mathcal{P}_1$. We output (D, τ) as the branch-decomposition of $(M, \mathcal{P})$.

THEOREM 6.7. *Let k be a fixed integer and $\mathbb{F}$ be a fixed finite field. We assume that a vector matroid $M = M(\boldsymbol{A})$ is given as an input together with a partition $\mathcal{P}$ of $E(M)$, where $n = |E(M)|$ and $|\mathcal{P}| \ge 2$. Algorithm* 6.6 *outputs in time $O(n^3)$ (parametrized by k and $\mathbb{F}$) a branch-decomposition of the partitioned matroid $(M, \mathcal{P})$ of width at most k, or confirms that* $\mathrm{bw}(M, \mathcal{P}) > k$.

Proof. We refer to the above outline. Our proof of the theorem constitutes the following three claims holding true if $\mathrm{bw}(M, \mathcal{P}) \le k$.

(I) The computation of Algorithm 6.6 maintains invariants, with respect to the actual matroid M_2 of $\mathcal{T}_2$, the rooted forest D, and the current value $\mathcal{P}_2$ of the partition variable $\mathcal{P}_1$ after each call to step (4.a.iv), such that
- $\mathcal{P}_2$ is the set of roots of D, and a titanic partition of M_2 such that $\mathrm{bw}(M_2, \mathcal{P}_2) = \mathrm{bw}(M_2) \le k$,
- $\lambda_M\big(\tau^D(\mathcal{S})\big) = \lambda_{M_2}^{\mathcal{P}_2}(\mathcal{S})$ for each $\mathcal{S} \subseteq \mathcal{P}_2$, where $\tau^D(\mathcal{S})$ is a shortcut for the union of $\tau(X)$ with X running over all leaves of the connected components of D whose root is in $\mathcal{S}$ (see Algorithm 6.6, step 2. for τ).

(II) Each iteration of the main cycle in Algorithm 6.6 (4.) succeeds to step (4.a.iv) at least once.

(III) The main cycle in Algorithm 6.6 step 4. is repeated $O(n)$ times. Moreover, the total number of calls to the steps in (4.a) is $O(n^2)$ for steps i, ii, iii, and $O(n)$ for steps iv, v.

Having all of these facts at hand, it is now easy to finish the proof. It is immediate from (I) that the resulting (D, τ) is a branch-decomposition of width at most k of $(M, \mathcal{P})$. Note that all parse trees involved in the algorithm have constant width less than $4k$ (see in steps (4.a.ii,v)). The starting steps (1.), (2.), (3.) of the algorithm are already known to run in time $O(n^3)$ (Theorem 5.6 and Algorithm 6.1), and the particular steps in (4.a) need time (III) $O(n^2) \cdot O(n) + O(n) \cdot O(n^2) = O(n^3)$ by Lemma 6.5 and Algorithm 6.2. The size of the matroid M_1 clearly stays linear in n after $O(n)$ constant-size updates. Hence, our Algorithm 6.6 runs correctly in parametrized time $O(n^3)$, provided that (I)–(III) hold true.

The proof of (I) essentially extends the arguments of Theorem 5.7. Initially, with M_1 and $\mathcal{P}_1$ in place of M_2, $\mathcal{P}_2$, all the claims of (I) obviously hold true, analogously to Theorem 5.6. Each call to step (4.a.iv) then adds a new titanic set $E(U_Y)$ to $\mathcal{P}_2$ (see Lemma 4.3 (3)), and hence the partition $\mathcal{P}_2$ remains titanic for M_2 and, subsequently,

$\mathrm{bw}(M_2, \mathcal{P}_2) = \mathrm{bw}\left(M_1, \mathcal{P}_1 \cup \{W\} \setminus \{X_1, X_2\}\right) = \mathrm{bw}(M_2) \le k$ follows from Lemma 3.4. The most complex claim of (I) is the last assertion, that $\lambda_M(\tau^D(\mathcal{S})) = \lambda_{M_2}^{\mathcal{P}_2}(\mathcal{S})$ for each $\mathcal{S} \subseteq \mathcal{P}_2$. By induction, we may assume that $\lambda_M(\tau^D(\mathcal{S}_1)) = \lambda_{M_1}^{\mathcal{P}_1}(\mathcal{S}_1)$ holds for all $\mathcal{S}_1 \subseteq \mathcal{P}_1$ just before this call to (4.a.iv). Now, by Algorithm 6.4, the titanic gadget $E(U_Y)$ in the representation spans exactly the same subspace because it is the guts of the separation $(X_1 \cup X_2,\ E(M_1) \setminus (X_1 \cup X_2))$ in M_1. Therefore, for all $\mathcal{S}_1 \subseteq \mathcal{P}_1$ such that $|\mathcal{S}_1 \cap \{X_1, X_2\}| \ne 1$, the corresponding $\mathcal{S} \subseteq \mathcal{P}_2$ satisfies $\lambda_{M_2}^{\mathcal{P}_2}(\mathcal{S}) = \lambda_{M_1}^{\mathcal{P}_1}(\mathcal{S}_1)$. This proves the assertion.

To prove (II), we use that $\mathrm{bw}(M_1, \mathcal{P}_1) \le k$ at each iteration of the main cycle (4.), which directly follows from above $\mathrm{bw}(M_2, \mathcal{P}_2) \le k$. Then, by the same arguments as in Theorem 5.7, there is a pair $\{X_1, X_2\} \subset \mathcal{P}_1$ for which (4.a) would succeed up to step (4.a.iv), which happens if $\mathrm{bw}\left(M_1, \mathcal{P}_1 \cup \{X_1 \cup X_2\} \setminus \{X_1, X_2\}\right) \le k$. We call such a pair X_1, X_2 *admissible*. It remains to argue that all admissible pairs $\{X_1, X_2\} \subset \mathcal{P}_1$ belong also to $\mathcal{Q}_2$, which is trivial only during the first round of (4.). For a contradiction, assume that $\{X_1, X_2\} \notin \mathcal{Q}_2$ at the least round $i > 1$. Consider now the values of our variables $\mathcal{P}_1, \mathcal{Q}_1, \mathcal{Q}_2$ at the previous round $i-1$: It was $\{X_1, X_2\} \cap \mathcal{Q}_1 = \emptyset$ by the assignment to $\mathcal{Q}_2$ in (4.b), and so $\{X_1, X_2\} \subset \mathcal{P}_1$ is already there. That means the pair X_1, X_2 has been admissible since round $i-1$ started, but it has not been processed only due to $\{X_1, X_2\} \notin \mathcal{Q}_2$ at round $i-1$, which contradicts our least choice of i.

Concerning (III), each iteration of (4.) adds at least one new node to the decomposition D by (II), and hence no more than $O(n)$ iterations occur. The same argument also bounds the total number of calls to the crucial steps (4.a.iv–v). The situation with steps i, ii, iii is more versatile, and we bound the total number of calls to them from above by the total number of iterations of the cycle in (4.a): During the initial round of the main cycle (4.), there are clearly at most $|\mathcal{Q}_2| = O(n^2)$ iterations of (4.a). For each subsequent round $i > 1$, the number of iterations is at most $|\mathcal{Q}_2| \le q_i \cdot |\mathcal{P}_1|$, where $q_i = |\mathcal{Q}_1|$ at the end of the previous run $i-1$. Hence, the total number of iterations of the cycle in (4.a) is at most $O(n^2) + O(n) \cdot \sum_{i=2}^{r} q_i$ since $|\mathcal{P}_1| = O(n)$ always. It remains to argue that $\sum_{i=2}^{r} q_i = O(n)$, which follows from the fact that each element ever assigned to $\mathcal{Q}_1$ in step (4.a.iv) appears as an internal node of the decomposition D, and $|V(D)| = O(n)$.

This also finishes the whole proof of Theorem 6.7. $\square$

7. Finding a rank-decomposition of a graph. In this last section, we present a fixed-parameter tractable algorithm to find a rank-decomposition of width at most k or confirm that the input graph has rank-width larger than k. It is a direct translation of the algorithm of Theorem 6.7. Let us first review necessary definitions from [19] and [17]. We assume that all graphs in this section have no loops and no parallel edges.

We have seen in section 2 that every symmetric submodular function can be used to define branch-width. We define a symmetric submodular function on a graph, called the *cut-rank* function of a graph. For an $X \times Y$ matrix $\boldsymbol{R}$ and $A \subseteq X$, $B \subseteq Y$, let $\boldsymbol{R}[A, B]$ be the $A \times B$ submatrix of $\boldsymbol{R}$. For a graph $G = (V, E)$, let $\boldsymbol{A}(G)$ be the adjacency matrix of G, that is a $V \times V$ matrix over the binary field $\mathrm{GF}(2)$ such that an entry is 1 if and only if vertices corresponding to the column and the row are adjacent in G. The cut-rank function $\rho_G(X)$ of a graph $G = (V, E)$ is defined as the rank of the matrix $\boldsymbol{A}(G)[X, V \setminus X]$ for each subset X of V. Then ρ_G is symmetric and submodular; see [19]. *Rank-decomposition* and *rank-width* of a graph G is branch-decomposition

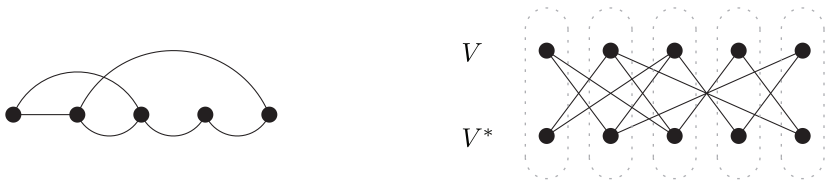

FIG. 5. *Graph G and the associated bipartite graph* $\mathrm{bip}(G)$ *with its canonical partition.*

and branch-width of the cut-rank function ρ_G of the graph G, respectively. So, if the graph has at least two vertices, then the rank-width is at most k if and only if there is a rank-decomposition of width at most k.

Now let us recall why bipartite graphs are essentially binary matroids. Oum [17] showed that the connectivity function of a binary matroid is exactly one more than the cut-rank function of its *fundamental graph*. The fundamental graph of a binary matroid M on $E = E(M)$, with respect to a basis B, is a bipartite graph on E such that two vertices in E are adjacent if and only if one vertex v is in B, another vertex w is not in B, and $(B \setminus \{v\}) \cup \{w\}$ is independent in M. Given a bipartite graph G, we can easily construct a binary matroid having G as a fundamental graph; if (C, D) is a bipartition of $V(G)$, then take the matrix

$$\begin{array}{cc} & \begin{array}{cc} \quad C \quad & \quad D \end{array} \\ C & \left(\begin{array}{ccc|c} 1 & & 0 & \boldsymbol{A}(G)[C,D] \\ & \ddots & & C \times D \text{ submatrix of} \\ 0 & & 1 & \text{the adjacency matrix} \end{array}\right) \end{array}$$

as the representation of a binary matroid. Thus, the column indices are elements of the binary matroid, and a set of columns is independent in the matroid if and only if its vectors are linearly independent. After all, finding the rank-decomposition of a bipartite graph is equivalent to finding the branch-decomposition of the associated binary matroid, that is essentially Theorem 6.7.

To find a rank-decomposition of nonbipartite graphs, we transform the graph into a canonical bipartite graph. For a finite set V, let V^* be a disjoint copy of V, that is, formally speaking, $V^* = \{v^* : v \in V\}$ such that $v^* \neq w$ for all $w \in V$ and $v^* \neq w^*$ for all $w \in V \setminus \{v\}$. For a subset X of V, let $X^* = \{v^* : v \in X\}$. For a graph $G = (V, E)$, let $\mathrm{bip}(G)$ be the bipartite graph on $V \cup V^*$ such that vw^* are adjacent in $\mathrm{bip}(G)$ if and only if v and w are adjacent in G (see Figure 5). Let $P_v = \{v, v^*\}$ for each $v \in V$. Then $\Pi(G) = \{P_v : v \in V\}$ is a canonical partition of $V(\mathrm{bip}(G))$.

LEMMA 7.1. *For every subset X of $V(G)$, $2\rho_G(X) = \rho_{\mathrm{bip}(G)}(X \cup X^*)$.*

Proof. This is clear from the construction of $\mathrm{bip}(G)$. Let $Y = V(G) \setminus X$. Let $N = \boldsymbol{A}(G)[X, Y]$. Since

$$\rho_{\mathrm{bip}(G)}(X \cup X^*) = \mathrm{rank} \begin{array}{cc} & \begin{array}{cc} Y & Y^* \end{array} \\ \begin{array}{c} X \\ X^* \end{array} & \begin{pmatrix} 0 & N \\ N^t & 0 \end{pmatrix} \end{array},$$

we conclude that $\rho_{\mathrm{bip}(G)}(X \cup X^*) = 2 \,\mathrm{rank}\, N = 2\rho_G(X)$. ☐

COROLLARY 7.2. *Let $p : V(G) \to \Pi(G)$ be the bijective function such that $p(x) = P_x$. If (T, μ) is a branch-decomposition of $\rho_{\mathrm{bip}(G)}^{\Pi(G)}$ of width k, then $(T, \mu \circ p)$*

is a branch-decomposition of ρ_G of width $k/2$. Conversely, if (T,μ') is a branch-decomposition of ρ_G of width k, then $(T,\mu'\circ p^{-1})$ is a branch-decomposition of $\rho^{\Pi(G)}_{\mathrm{bip}(G)}$ of width $2k$. Therefore, the branch-width of ρ_G is equal to half of the branch-width of $\rho^{\Pi(G)}_{\mathrm{bip}(G)}$.

Let $M=\mathrm{mat}(G)$ be the binary matroid on $V\cup V^*$ represented by the matrix

$$\begin{array}{cc} & \begin{array}{cc} V & V^* \end{array} \\ V & \left(\begin{array}{c|c} \text{Identity matrix} & \boldsymbol{A}(G) \end{array}\right). \end{array}$$

Since the bipartite graph $\mathrm{bip}(G)$ is a fundamental graph of M, we have $\lambda_M(X)=\rho_{\mathrm{bip}(G)}(X)+1$ for all $X\subseteq V\cup V^*$ (see Oum [17]) and therefore (T,μ) is a branch-decomposition of a partitioned matroid $(M,\Pi(G))$ of width $k+1$ if and only if it is a branch-decomposition of $\rho^{\Pi(G)}_{\mathrm{bip}(G)}$ of width k. Corollary 7.2 implies that a branch-decomposition of $\rho^{\Pi(G)}_{\mathrm{bip}(G)}$ of width k is equivalent to that of ρ_G of width $k/2$. So, we can deduce the following theorem from Theorem 6.7.

THEOREM 7.3. *Let k be a constant. Let $n\geq 2$. For an n-vertex graph G, we can output the rank-decomposition of width at most k or confirm that the rank-width of G is larger than k in time $O(n^3)$.*

Proof. We apply Theorem 6.7 to find a branch-decomposition of a partitioned matroid $(\mathrm{mat}(G),\Pi(G))$ of width at most $2k+1$. If such a branch-decomposition is found, then one can canonically transform it into a rank-decomposition of G of width at most k by Corollary 7.2. If there is no such branch-decomposition, then the rank-width of G is larger than k. ∎

Acknowledgments. The authors are very grateful to Jim Geelen for his comments on the possible approach to the problem. We would also like to thank the anonymous referees for helpful comments.

REFERENCES

[1] D. G. Corneil, M. Habib, J.-M. Lanlignel, B. A. Reed, and U. Rotics, *Polynomial time recognition of clique-width ≤ 3 graphs* (extended abstract), in LATIN 2000: Theoretical informatics, G. H. Gonnet, D. Panario, and A. Viola, eds., Lecture Notes in Comput. Sci. 1776, Springer, Berlin, 2000, pp. 126–134.

[2] D. G. Corneil, Y. Perl, and L. K. Stewart, *A linear recognition algorithm for cographs*, SIAM J. Comput., 14 (1985), pp. 926–934.

[3] B. Courcelle, J. A. Makowsky, and U. Rotics, *Linear time solvable optimization problems on graphs of bounded clique-width*, Theory Comput. Syst., 33 (2000), pp. 125–150.

[4] B. Courcelle and S. Olariu, *Upper bounds to the clique width of graphs*, Discrete Appl. Math., 101 (2000), pp. 77–114.

[5] B. Courcelle and S. Oum, *Vertex-minors, monadic second-order logic, and a conjecture by Seese*, J. Combin. Theory Ser. B, 97 (2007), pp. 91–126.

[6] W. Espelage, F. Gurski, and E. Wanke, *How to solve NP-hard graph problems on clique-width bounded graphs in polynomial time*, in Graph-Theoretic Concepts in Computer Science (Boltenhagen, 2001), Lecture Notes in Comput. Sci. 2204, Springer, Berlin, 2001, pp. 117–128.

[7] M. R. Fellows, F. A. Rosamond, U. Rotics, and S. Szeider, *Clique-width minimization is NP-hard*, in Proceedings of the 38th Annual ACM Symposium on Theory of Computing, ACM Press, New York, 2006, pp. 354–362.

[8] J. F. Geelen, A. M. H. Gerards, N. Robertson, and G. P. Whittle, *On the excluded minors for the matroids of branch-width k*, J. Combin. Theory Ser. B, 88 (2003), pp. 261–265.

[9] J. F. Geelen, A. M. H. Gerards, and G. Whittle, *Tangles, Tree-Decompositions, and Grids in Matroids*, Research report 04-5, School of Mathematical and Computing Sciences, Victoria University of Wellington, Wellington, New Zealand, 2004.
[10] M. U. Gerber and D. Kobler, *Algorithms for vertex-partitioning problems on graphs with fixed clique-width*, Theoret. Comput. Sci., 299 (2003), pp. 719–734.
[11] I. V. Hicks and N. B. McMurray, Jr., *The branchwidth of graphs and their cycle matroids*, J. Combin. Theory Ser. B, 97 (2007), pp. 681–692.
[12] J. W. P. Hirschfeld and L. Storme, *The packing problem in statistics, coding theory and finite projective spaces: Update* 2001, in Finite Geometries, Dev. Math. 3, Kluwer Academic Publishers, Dordrecht, 2001, pp. 201–246.
[13] P. Hliněný, *A parametrized algorithm for matroid branch-width*, SIAM J. Comput., 35 (2005), pp. 259–277.
[14] P. Hliněný, *Branch-width, parse trees, and monadic second-order logic for matroids*, J. Combin. Theory Ser. B, 96 (2006), pp. 325–351.
[15] D. Kobler and U. Rotics, *Edge dominating set and colorings on graphs with fixed clique-width*, Discrete Appl. Math., 126 (2003), pp. 197–221.
[16] F. Mazoit and S. Thomassé, *Branchwidth of graphic matroids*, in Surveys in Combinatorics, London Math. Soc. Lecture Note Ser. 346, Cambridge University Press, Cambridge, 2007, pp. 275–286.
[17] S. Oum, *Rank-width and vertex-minors*, J. Combin. Theory Ser. B, 95 (2005), pp. 79–100.
[18] S. Oum, *Approximating rank-width and clique-width quickly*, submitted, 2006.
[19] S. Oum and P. Seymour, *Approximating clique-width and branch-width*, J. Combin. Theory Ser. B, 96 (2006), pp. 514–528.
[20] S. Oum and P. Seymour, *Testing branch-width*, J. Combin. Theory Ser. B, 97 (2007), pp. 385–393.
[21] J. G. Oxley, *Matroid Theory*, Oxford University Press, New York, 1992.
[22] N. Robertson and P. Seymour, *Graph minors.* X. *Obstructions to tree-decomposition*, J. Combin. Theory Ser. B, 52 (1991), pp. 153–190.
[23] P. Seymour and R. Thomas, *Call routing and the ratcatcher*, Combinatorica, 14 (1994), pp. 217–241.
[24] E. Wanke, *k-NLC graphs and polynomial algorithms. Efficient algorithms and partial k-trees*, Discrete Appl. Math., 54 (1994), pp. 251–266.