APPROXIMATING CLIQUE-WIDTH AND BRANCH-WIDTH

SANG-IL OUM AND PAUL SEYMOUR

Abstract. We construct a polynomial-time algorithm to approximate the branch-width of certain symmetric submodular functions, and give two applications.

The first is to graph "clique-width". Clique-width is a measure of the difficulty of decomposing a graph in a kind of tree-structure, and if a graph has clique-width at most k then the corresponding decomposition of the graph is called a "k-expression". We find (for fixed k) an $O(n^9 \log n)$-time algorithm that, with input an n-vertex graph, outputs either a $(2^{3k+2}-1)$-expression for the graph, or a true statement that the graph has clique-width at least $k+1$. (The best earlier algorithm algorithm, by Johansson [13], constructed a $k \log n$-expression for graphs of clique-width at most k.) It was already known that several graph problems, NP-hard on general graphs, are solvable in polynomial time if the input graph comes equipped with a k-expression (for fixed k). As a consequence of our algorithm, the same conclusion follows under the weaker hypothesis that the input graph has clique-width at most k (thus, we no longer need to be provided with an explicit k-expression).

Another application is to the area of matroid branch-width. For fixed k, we find an $O(n^4)$-time algorithm that, with input an n-element matroid in terms of its rank oracle, either outputs a branch-decomposition of width at most $3k-1$ or a true statement that the matroid has branch-width at least $k+1$. The previous algorithm by Hliněný [11] was only for representable matroids.

1. Introduction

Some algorithmic problems, NP-hard on general graphs, are known to be solvable in polynomial time when the input graph admits a decomposition into trivial pieces by means of a tree-structure of cutsets of bounded order. However, it makes a difference whether the input graph is presented together with the corresponding tree-structure of cutsets or not. We have in mind two kinds of decompositions, "tree-width" and "clique-width" decompositions. These are similar graph invariants, and while the results of this paper concern clique-width, we begin with tree-width for purposes of comparison.

Having bounded clique-width is more general than having bounded tree-width, in the following sense. Every graph G of tree-width at most k has clique-width at most $O(2^k)$ [5, 7], and for such graphs (for k fixed) the clique-width of G can be determined in linear time [9]. No bound in the reverse direction holds, for there are graphs of arbitrary large tree-width with clique-width at most k. (But, for fixed t, if G does not contain $K_{t,t}$ as a subgraph, then the tree-width is at most $3k(t-1)-1$ [10].)

The algorithmic situation with tree-width is as follows:

- Numerous problems have been shown to be solvable in polynomial time when the input graph is presented together with a decomposition of bounded tree-width. Indeed, every

Supported by ONR grant N00014-01-1-0608, and NSF grant DMS-0070912.

graph property expressible in monadic second order logic with quantifications over vertices, vertex sets, edges, and edge sets (MSO_2-logic) can be solved in polynomial time (see [6]).
- For fixed k there is a polynomial time algorithm [18] that either decides that an input graph has tree-width at least $k+1$, or outputs a decomposition of tree-width at most $4k$.
- Consequently, by combining these algorithms, it follows that the same class of problems mentioned above can be solved on inputs of bounded tree-width; the input does not need to come equipped with the corresponding decomposition.
- In particular, one of these problems is the problem of deciding whether a graph has tree-width at most k. Consequently, for fixed k there is a polynomial (indeed, linear) time algorithm [1] to test whether an input graph has tree-width at most k, and if so to output the corresponding decomposition.

For inputs of bounded clique-width, less progress has so far been made. (We will define clique-width properly later.)

- Some problems have been shown to be solvable in polynomial time when the input graph is presented together with a decomposition of bounded clique-width. This class of problems is smaller than the corresponding set for tree-width, but still of interest. For instance, deciding whether the graph is Hamiltonian [22], finding the chromatic number [14], and various partition problems [8] are solvable in polynomial time; and so is any problem that can be expressed in monadic second order logic with quantifications over vertices and vertex sets (MSO_1-logic; see [3, 6]).
- For fixed (general) k there was so far no known polynomial time algorithm that either decides that an input graph has clique-width at least $k+1$, or outputs a decomposition of clique-width bounded by any function of k. The best hitherto was an algorithm of Johansson [13], that with input an n-vertex graph G, either decides that G has clique-width at least $k+1$ or outputs a decomposition of clique-width at most $k \log n$. Our main result fills this gap.
- Consequently, it follows that the same class of problems mentioned above can be solved on inputs of bounded clique-width; the input does not need to come equipped with the corresponding decomposition.
- However, the problem of deciding whether a graph has clique-width at most k is not known to belong to this class. There is still no polynomial time algorithm to test whether G has clique-width at most k, for fixed general k.

We shall prove the following.

Theorem 1.1. *For fixed k, there is an algorithm that with input an n-vertex graph G, either decides that G has clique-width at least $k+1$, or outputs a decomposition of G with clique-width at most $2^{3k+2}-1$. Its running time is $O(n^9 \log n)$.*

The main tool for this algorithm is branch-width, which is closely related to tree-width, and was introduced in [19]. We develop a general algorithm to approximate the branch-width of certain symmetric submodular functions. Then we define the “rank-width” of a graph to be the branch-width of a symmetric submodular function determined by a graph; and since our algorithm applies to this submodular function, we can approximate the rank-width of a graph in polynomial time. But we also prove that if clique-width is bounded, then rank-width is bounded, and vice versa; and consequently we can approximate clique-width in polynomial time.

We also apply this algorithm to matroids, and obtain an algorithm to approximate the branch-width of matroids, which was known before only for representable matroids by Hliněný [11]. We prove:

Theorem 1.2. *For fixed k there is an algorithm which, with input an n-element matroid $\mathcal{M}$ in terms of its rank oracle, either decides that $\mathcal{M}$ has branch-width at least $k+1$, or outputs a branch-decomposition for $\mathcal{M}$ of width at most $3k-1$. Its running time and number of oracle calls is at most $O(n^4)$.*

2. Branch-width

Let V be a finite set and $f : 2^V \to \mathbb{Z}$ be a function. If

$$f(X) + f(Y) \geq f(X \cap Y) + f(X \cup Y)$$

for all $X, Y \subseteq V$, then f is said to be *submodular*. If f satisfies $f(X) = f(V \setminus X)$ for all $X \subseteq V$, then f is said to be *symmetric*.

A *subcubic tree* is a tree with at least two vertices such that every vertex is incident with at most three edges. A *leaf* of a tree is a vertex incident with exactly one edge. We call (T, L) a *partial branch-decomposition* of a symmetric submodular function f if T is a subcubic tree and $L : V \to \{v : v \text{ is a leaf of } T\}$ is a surjective function. (If $|V| \leq 1$ then f admits no partial branch-decomposition.) If in addition L is bijective, we call (T, L) a *branch-decomposition* of f. If $L(v) = t$, then we say t is *labeled by* v and v is *a label of* t.

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. The *width* of an edge e of a partial branch-decomposition (T, L) is $f(L^{-1}(X))$. The *width* of (T, L) is the maximum width of all edges of T. The *branch-width* $\mathrm{bw}(f)$ of f is the minimum width of a branch-decomposition of f. (If $|V| \leq 1$, we define $\mathrm{bw}(f) = f(\emptyset)$.)

For the application to matroids, we assume that the reader is familiar with the basic notions of matroid theory (see [16]). Let us review matroid theory briefly for the purpose of this paper.

A matroid $\mathcal{M} = (E, r)$ is a pair formed by a finite set E of *elements* and a *rank* function $r : 2^E \to \mathbb{Z}$ satisfying the following axioms:

i) $0 \leq r(X) \leq |X|$ for all $X \subseteq E$.
ii) If $X \subseteq Y \subseteq E$, then $r(X) \leq r(Y)$.
iii) r is submodular.

We write $E(\mathcal{M}) = E$. For $Y \subseteq E(\mathcal{M})$, $\mathcal{M} \setminus Y$ is the matroid $(E(\mathcal{M}) \setminus Y, r')$ where $r'(X) = r(X)$. For $X \subseteq E(\mathcal{M})$, $\mathcal{M}/Y$ is the matroid $(E(\mathcal{M}) \setminus Y, r')$ where $r'(X) = r(X \cup Y) - r(Y)$. If $Y = \{e\}$, we denote $\mathcal{M} \setminus e = \mathcal{M} \setminus \{e\}$ and $\mathcal{M}/e = \mathcal{M}/\{e\}$. It is routine to prove that $\mathcal{M} \setminus Y$ and $\mathcal{M}/Y$ are matroids.

For $X \subseteq E$, $\lambda(X) = r(X) + r(E(\mathcal{M}) \setminus X) - r(\mathcal{M}) + 1$ is the *connectivity* function of $\mathcal{M}$. A *branch-decomposition* and the *branch-width* of a matroid $\mathcal{M}$ are defined as a branch-decomposition and the branch-width of λ.

3. Clique-width

The notion of clique-width was first introduced by Courcelle and Olariu [7]. Let k be a positive integer. We call (G, lab) a *k-graph* if G is a graph and lab is a mapping from its vertex set to $\{1, 2, \dots, k\}$. (In this paper, all graphs are finite and have no loops or parallel edges.) We call $lab(v)$ the *label* of a vertex v.

We need the following definitions and operations on k-graphs.

(1) For $i \in \{1, \dots, k\}$, let $\cdot_i$ denote an isolated vertex labeled by i.
(2) For $i, j \in \{1, 2, \dots, k\}$ with $i \neq j$, we define a unary operator $\eta_{i,j}$ such that
$$\eta_{i,j}(G, lab) = (G', lab)$$
where $V(G') = V(G)$, and $E(G') = E(G) \cup \{vw : v, w \in V, lab(v) = i, lab(w) = j\}$. This adds edges between vertices of label i and vertices of label j.
(3) We let $\rho_{i \to j}$ be the unary operator such that
$$\rho_{i\to j}(G, lab) = (G, lab')$$
where
$$lab'(v) = \begin{cases} j & \text{if } lab(v) = i, \\ lab(v) & \text{otherwise.} \end{cases}$$
This mapping relabels every vertex labeled by i into j.
(4) Finally, $\oplus$ is a binary operation that makes the disjoint union. Note that $G \oplus G \neq G$.

A well-formed expression t in these symbols is called a *k-expression*. The k-graph produced by performing these operations in order therefore has vertex set the set of occurrences of the constant symbols in t; and this k-graph (and any k-graph isomorphic to it) is called the *value* $val(t)$ of t. If a k-expression t has value (G, lab), we say that t is a *k-expression of* G. The *clique-width* of a graph G, denoted by $\operatorname{cwd}(G)$, is the minimum k such that there is a k-expression of G.

For instance, K_4 (the complete graph with four vertices) can be constructed by
$$\rho_{2\to1}(\eta_{1,2}(\rho_{2\to1}(\eta_{1,2}(\rho_{2\to1}(\eta_{1,2}(\cdot_1 \oplus \cdot_2)) \oplus \cdot_2)) \oplus \cdot_2)).$$
Therefore, K_4 has a 2-expression, and $\operatorname{cwd}(K_4) \le 2$. It is easy to see that $\operatorname{cwd}(K_4) > 1$, and therefore $\operatorname{cwd}(K_4) = 2$.

Some other examples: cographs, which are graphs with no induced path of length 3, are exactly the graphs of clique-width at most 2; the complete graph K_n $(n > 1)$ has clique-width 2; and trees have clique-width at most 3 [7].

For some classes of graphs, it is known that clique-width is bounded and algorithms to construct a k-expression have been found. For example, cographs [4], graphs of clique-width at most 3 [2], and P_4-sparse graphs (every five vertices have at most one induced subgraph isomorphic to a path of length 3) [3] have such algorithms.

4. Extension of a submodular function

In this section, we define an "interpolation" of an integer-valued submodular function. Later we will use it to prove the main theorem.

For a finite set V, we define (with a slight abuse of terminology) 3^V to be $\{(X,Y) : X, Y \subseteq V, X \cap Y = \emptyset\}$.

Definition 4.1. Let $f : 2^V \to \mathbb{Z}$ be an integer-valued submodular function such that $f(\emptyset) \leq f(X)$ for all $X \subseteq V$. We call $f^* : 3^V \to \mathbb{Z}$ an *interpolation* of f if

i) $f^*(X, V \setminus X) = f(X)$ for all $X \subseteq V$,
ii) (uniform) if $C \cap D = \emptyset$, $A \subseteq C$, and $B \subseteq D$, then $f^*(A,B) \leq f^*(C,D)$,
iii) (submodular) $f^*(A,B) + f^*(C,D) \geq f^*(A\cap C, B \cup D) + f^*(A \cup C, B \cap D)$ for all $(A,B),(C,D) \in 3^V$.
iv) $f^*(\emptyset,\emptyset) = f(\emptyset)$.

Assuming that $\emptyset$ is a minimizer of f is not a serious restriction, because first of all it is true for all symmetric submodular functions, and secondly if we let

$$g(X) = \begin{cases} f(X) & \text{if } X \neq \emptyset \\ \min_Z f(Z) & \text{otherwise,} \end{cases}$$

then g is also submodular.

Proposition 4.1. *Let $f : 2^V \to \mathbb{Z}$ be a submodular function such that $f(\emptyset) \leq f(X)$ for all $X \subseteq V$, and let $f^* : 3^V \to \mathbb{Z}$ be an interpolation of f. Then:*

(1) for all $(X,Y) \in 3^V \to$, $f^*(X,Y) \leq \min_{X\subseteq Z \subseteq V\setminus Y} f(Z)$.
(2) $f^*(\emptyset, Y) = f(\emptyset)$ for all $Y \subseteq V$.
(3) 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 $V \setminus B$.

Proof.

(1) If $X \subseteq Z \subseteq V \setminus Y$, then $f^*(X,Y) \leq f^*(Z, V\setminus Z) = f(Z)$.
(2) $f(\emptyset) = f^*(\emptyset,\emptyset) \leq f^*(\emptyset, Y) \leq f^*(\emptyset, V) = f(\emptyset)$.
(3) Let $r(X) = f^*(X,B) - f(\emptyset)$. It is trivial that r is submodular and nondecreasing. Moreover,
$$0 \leq r(X) = f^*(X,B) - f(\emptyset) \leq f(X) - f(\emptyset) \leq |X|,$$
and therefore r is the rank function of a matroid on $V \setminus B$. □

We define $f_{\min}(X,Y) = \min f(Z)$, the minimum being taken over all Z satisfying $X \subseteq Z \subseteq V \setminus Y$.

Proposition 4.2. *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.*

Proof. The first, second, and last conditions are trivial. Let us prove submodularity. Let X, Y be subsets of V such that $A \subseteq X \subseteq V \setminus B$, $C \subseteq Y \subseteq V \setminus D$, $f_{\min}(A,B) = f(X)$, and $f_{\min}(C,D) = f(Y)$. Then

$$\begin{aligned} f(X) + f(Y) &\geq f(X\cap Y) + f(X \cup Y) \\ &\geq f_{\min}(A \cap C, B \cup D) + f_{\min}(A \cup C, B \cap D). \end{aligned}$$

Thus, $f_{\min}$ is an interpolation. □

In general $f_{\min}$ is not the only interpolation of a function f, and sometimes it is better for us to work with other interpolations that can be evaluated more quickly.

We remark that if $f^* : 3^V \to \mathbb{Z}$ is a uniform submodular function satisfying $f^*(\emptyset, \emptyset) = f^*(\emptyset, V)$, then there is a submodular function $f : 2^V \to \mathbb{Z}$ such that $f(\emptyset) \le f(X)$ for all $X \subseteq V$ and f^* is an interpolation of f.

5. Branch-Width and Well-Linkedness

Definition 5.1. Let V be a finite set and let $f : 2^V \to \mathbb{Z}$ be a symmetric submodular function satisfying $f(\emptyset) = 0$. We say that $W \subseteq V$ is *well-linked* with respect to f if for every partition (X, Y) of W and every Z with $X \subseteq Z \subseteq V \setminus Y$, we have

$$f(Z) \ge \min(|X|, |Y|).$$

This notion is analogous to the notion of well-linkedness [17] related to tree-width of graphs.

Theorem 5.1. *Let V be a finite set with $|V| \ge 2$, 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* $\mathrm{bw}(f) \ge k/3$.

Proof. Let W be a well-linked set of size k, and suppose that (T, L) is a branch decomposition of f. We will show that (T, L) has width at least $k/3$. We may assume that T does not have a vertex of degree 2, by suppressing any such vertices. For each edge $e = uv$ of T, let A_{uv} be the set of elements of V that are mapped by L into the connected component of $T \setminus e$ containing u, and let $B_{uv} = V \setminus A_{uv}$.

We may assume that $W \ne \emptyset$; choose $w \in W$. Since W is well-linked with respect to f, $f(\{w\}) \ge 1$, and therefore the width of (T, L) is at least 1. Consequently we may assume that assume $k > 3$.

Suppose first that $\min(|A_{uv} \cap W|, |B_{uv} \cap W|) < k/3$ for every edge uv of T. Direct every edge uv from u to v if $|A_{uv} \cap W| < k/3$ and $|B_{uv} \cap W| \ge k/3$. By the assumption, each edge is given a unique direction. Since the number of vertices is more than the number of edges in T, there is a vertex $t \in V(T)$ such that every edge incident with t has head t.

If t is a leaf of T, let s be the neighbour of t. Since ts has head t, it follows that $|B_{st} \cap W| \ge k/3$. But $|B_{st}| = 1 < k/3$, a contradiction.

So, t has three neighbours x, y, z in T such that $|A_{xt} \cap W| < k/3$, $|A_{yt} \cap W| < k/3$, and $|A_{zt} \cap W| < k/3$. But $|W| = |A_{xt} \cap W| + |A_{yt} \cap W| + |A_{zt} \cap W| < k = |W|$, a contradiction.

We deduce that there exists $uv \in E(T)$ such that $|A_{uv} \cap W| \ge k/3$ and $|B_{uv} \cap W| \ge k/3$. Hence $f(A_{uv}) \ge \min(|A_{uv} \cap W|, |B_{uv} \cap W|) \ge k/3$, and the width of (T, L) is at least $k/3$. $\square$

Theorem 5.2. *Let V be a finite set, let $f : 2^V \to \mathbb{Z}$ be a symmetric submodular function such that $f(\{v\}) \le 1$ for all $v \in V$ and $f(\emptyset) = 0$, and let $k \ge 0$ be an integer. If with respect to f, there is no well-linked set of size k, then* $\mathrm{bw}(f) \le k$.

Proof. We may assume that $\mathrm{bw}(f) > 0$, and so $|V| \ge 2$. We may assume that $k > 0$. For two partial branch-decompositions (T, L) and (T', L') of f, we say that (T, L) *extends* (T', L') if T' is obtained by contracting some edges of T and for every $v \in V$, $L'(v)$ is the vertex of T' that corresponds to $L(v)$ under the contraction.

We will prove that, if there is no well-linked set of size k with respect to f, then for every partial branch-decomposition (T_s, L_s) of f with width at most k, there is a branch-decomposition of f of width at most k extending (T_s, L_s). Since $k \geq 1$ and f trivially admits a partial branch-decomposition of width 1 (using the two-vertex tree with vertices u, v, and mapping all vertices of V except one to u, and the last to v), this implies the statement of the theorem.

Pick a partial branch-decomposition (T, L) of f extending (T_s, L_s) such that the width of (T, L) is at most k and the number of leaves of T is maximum.

We claim that (T, L) is a branch-decomposition of f, that is, L is a bijection. Suppose therefore that there is a leaf t of T such that $B = L^{-1}(\{t\})$ has more than one element.

(1) $f(B) = k$.

Suppose that $f(B) < k$. Let $v \in B$. Construct a subcubic tree T' by adding two vertices t_1 and t_2 and edges t_1t, t_2t to T. Let $L'(v) = t_1$ and $L'(w) = t_2$ for all $w \in B \setminus \{v\}$ and $L'(x) = L(x)$ for all $x \in V \setminus B$. Then (T', L') is a partial branch-decomposition extending (T, L). Moreover $f(\{v\}) \leq 1 \leq k$ and $f(B \setminus \{v\}) \leq f(B) + f(\{v\}) \leq k$, and so the width of (T', L') is at most k. But the number of leaves of T' is greater than that of T, a contradiction. □

Let f^* be an interpolation of f. By Proposition 4.1, $f^*(X, B)$ is the rank function of a matroid on $V \setminus B$. Let X be a base of this matroid. Then $|X| = f^*(V \setminus B, B) = f(B) = k$.

Since X is not well-linked, there exists $Z \subseteq V$ such that

$$f(Z) < \min(|Z \cap X|, |(V \setminus Z) \cap X|).$$

Since $f(Z \setminus B) = f^*(Z \setminus B, B \cup (V \setminus Z)) \geq f^*(Z \cap X, B) = |Z \cap X| > f(Z)$, it follows that $Z \cap B \neq \emptyset$. Similarly $B \setminus Z = (V \setminus Z) \cap B \neq \emptyset$.

Construct a subcubic tree T' by adding two vertices t_1 and t_2 and edges t_1t, t_2t to T. Let $L'(x) = t_1$ if $x \in B \cap Z$, $L'(x) = t_2$ if $x \in B \setminus Z$ and $L'(x) = L(x)$ otherwise.

By submodularity,

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

and so $f(Z \cap B) < f(B) \leq k$ and similarly $f(B \setminus Z) < f(B) \leq k$. Therefore (T', L') is a partial branch-decomposition extending (T, L) of width at most k. But the number of leaves of T' is greater than that of T, a contradiction. □

Corollary 5.3. *For all $k \geq 0$, there is a polynomial-time algorithm that, with input a set V with $|V| \geq 2$ and a symmetric submodular function $f : 2^V \to \mathbb{Z}$ with $f(\{v\}) \leq 1$ for all $v \in V$ and $f(\emptyset) = 0$, outputs either a well-linked set of size k or a branch-decomposition of width at most k.*

The proof of Theorem 5.2 shows an algorithm that either finds a well-linked set of size k, or constructs a branch-decomposition of f of width at most k. By combining with Theorem 5.1, we get an algorithm that either concludes that $\mathrm{bw}(f) > k$ or finds a branch-decomposition of width at most $3k+1$.

Let us analyze the running time of the algorithm of Theorem 5.2. To do so, we must be more precise about how the input function f and f^* are accessed. We consider two different situations, as follows:

- In the first case, we assume that only f is given as input, and in the sense that we can compute $f(X)$ for a set X; and we need to compute values of f^* from this input.
- In the second case, we assume that an interpolation f^* of f is given as input (in the same sense, that for any pair (X, Y) we can compute $f^*(X, Y)$), and we need to compute f from f^*.

For the first analysis, let γ be the time to compute $f(X)$ for any set X. In this case we shall use $f^* = f_{\min}$. To calculate $f_{\min}$, we use the submodular function minimization algorithm [12], whose running time is $O(n^5\gamma \log M)$ where M is the maximum value of f and $n = |V|$. Thus, we can calculate $f_{\min}$ in $O(n^5\gamma \log n)$ time. Finding a base X can be done by calculating f^* at most $O(n)$ times, and therefore takes time $O(n^6\gamma \log n)$. To check whether X is well-linked, we try all partitions of X; 2^{k-1} tries (a constant). And finding the set Z for a given partition of X can be done in time $O(n^5\gamma \log n)$ by submodular function minimization algorithms. Since the process is cycled through at most $O(n)$ times (because if (T, L) is a partital branch-decomposition then $|V(T)| \leq 2n-2$), it follows that in this case the time complexity is $O(n^7\gamma \log n)$.

For the second analysis, let δ be the time to compute $f^*(X)$ for any set X. Finding a base X can be done in time $O(n\delta)$. Finding Z to show that X is not well-linked can be done in time $O(n^5\delta \log n)$. Thus, the time complexity in this case is $O(n^6\delta \log n)$.

In summary, then, we have shown the following two statements.

Corollary 5.4. *For given k, there is an algorithm as follows. It takes as input a finite set V with $|V| \geq 2$ and a symmetric submodular function $f : 2^V \to \mathbb{Z}$, such that $f(\{v\}) \leq 1$ for all $v \in V$ and $f(\emptyset) = 0$. It either concludes that* $\mathrm{bw}(f) > k$ *or outputs a branch-decomposition of f of width at most* $3k+1$*; and its running time (excluding evaluating f) and number of evaluations of f are both* $O(|V|^7 \log |V|)$*.*

Corollary 5.5. *For given k, there is an algorithm as follows. It takes as input a finite set V with $|V| \geq 2$ and a function f^* which is an interpolation of some symmetric submodular function $f : 2^V \to \mathbb{Z}$, such that $f(\{v\}) \leq 1$ for all $v \in V$ and $f(\emptyset) = 0$. It either concludes that* $\mathrm{bw}(f) > k$ *or outputs a branch-decomposition of f of width at most* $3k+1$*; and its running time is* $O(|V|^6\delta \log |V|)$*, where δ is the time for each evaluation of f^*.*

6. Application to Clique-width

Definition 6.1. Let G be a graph and let A, $B \subseteq V(G)$ be disjoint. Let $M_A^B(G)$ be the matrix $(m_{ij} : i \in A, j \in B)$ over the 2-element field GF(2), where $m_{ij} = 1$ if i, j are adjacent in G, and $m_{ij} = 0$ otherwise. We define $\mathrm{cutrk}^*_G(A, B) = \mathrm{rk}(M_A^B(G))$ where rk is the matrix rank function; and we define the *cut-rank* function cutrk_G of G by $\mathrm{cutrk}_G(X) = \mathrm{cutrk}^*_G(X, V(G) \setminus X)$ for

$X \subseteq V(G)$. We will show that cutrk_G is symmetric submodular and cutrk^*_G is an interpolation of cutrk_G.

Proposition 6.1. *Let $M = (m_{ij} : i \in C, j \in R)$ be a matrix over a field F. For $X \subseteq R$ and $Y \subseteq C$, let $M[X,Y]$ denote the submatrix $(m_{ij}; i \in X, j \in Y)$. Then for all $X_1, X_2 \subseteq R$ and $Y_1, Y_2 \subseteq C$, we have*

$$\mathrm{rk}(M[X_1, Y_1]) + \mathrm{rk}(M[X_2, Y_2]) \geq \mathrm{rk}(M[X_1 \cup X_2, Y_1 \cap Y_2]) + \mathrm{rk}(M[X_1 \cap X_2, Y_1 \cup Y_2]).$$

Proof. See [15, Proposition 2.1.9], [21, Lemma 2.3.11], or [20]. □

Corollary 6.2. *Let G be a graph. If $(X_1, Y_1), (X_2, Y_2) \in 3^{V(G)}$ then*

$$\mathrm{cutrk}^*_G(X_1, Y_1) + \mathrm{cutrk}^*_G(X_2, Y_2) \geq \mathrm{cutrk}^*_G(X_1 \cap X_2, Y_1 \cup Y_2) + \mathrm{cutrk}^*_G(X_1 \cup X_2, Y_1 \cap Y_2).$$

Moreover, if $X_1, X_2 \subseteq V(G)$, then

$$\mathrm{cutrk}_G(X_1) + \mathrm{cutrk}_G(X_2) \geq \mathrm{cutrk}_G(X_1 \cap X_2) + \mathrm{cutrk}_G(X_1 \cup X_2).$$

Proof. Let M be the $V(G) \times V(G)$ adjacency matrix of G over GF(2). The first statement follows from 6.1 applied to M. The second follows from the first by setting $Y_i = V(G) \setminus X_i$ $(i = 1, 2)$. □

A *rank-decomposition* of G is a branch-decomposition of cutrk_G, and the *rank-width* $\mathrm{rwd}(G)$ of G is the branch-width of cutrk_G.

The following proposition shows a relation between clique-width and rank-width.

Proposition 6.3. *For any graph G,* $\mathrm{rwd}(G) \leq \mathrm{cwd}(G) \leq 2^{\mathrm{rwd}(G)+1} - 1$.

Proof. We may assume that $|V(G)| \geq 2$, because if $|V(G)| \leq 1$, then $\mathrm{rwd}(G) = 0$ and $\mathrm{cwd}(G) \leq 1$.

A *rooted binary tree* is a subcubic tree with a specified vertex, called the *root*, such that every non-root vertex has one, two or three incident edges and the root has at most two incident edges. A vertex u of a rooted binary tree is called a *descendant* of a vertex v if v belongs to the path from the root to u; and u is called a *child* of v if u, v are adjacent in T and u is a descendant of v.

First we show that $\mathrm{rwd}(G) \leq \mathrm{cwd}(G)$. Let $k = \mathrm{cwd}(G)$. Let t be a k-expression with value (G, lab) for some choice of lab. We recall that a k-expression is a well-formed expression with four types of symbols; the constants, two unary operators, and the binary operator forming disjoint union. The parentheses of the expression form a tree structure. Thus there is a rooted binary tree T, each vertex v of which corresponds to a k-expression say $N(v)$; and letting V_0, V_1, V_2 denote the sets of vertices in T with zero, one and two children respectively, we have for each vertex $v \in V(T)$:

- if $v \in V_0$ then $N(v)$ is a 1-term expression consisting just of a constant term
- if $v \in V_1$ with child u, then $N(v)$ is obtained from $N(u)$ by applying one of the two unary operators
- if $v \in V_2$ with children u_1, u_2, then $N(v)$ is obtained from $N(u_1), N(u_2)$ by applying $\oplus$
- if v is the root then $N(v) = (G, lab)$.

In particular, each vertex $v \in V_0$ gives rise to a unique vertex of G; let us call this $L(v)$. Then L is a bijection between $V(G)$ and the set of leaves of T. Consequently (T, L) is a branch-decomposition of $cutrk_G$. Let us study its width. Let $u, v \in V(T)$, where u is a child of v, and let T_1, T_2 be the components of $T \setminus e$, where e is the edge uv and $u \in V(T_1)$. Let $X_i = \{L(t) : t \in V_0 \cap V(T_i)\}$ for $i = 1, 2$. Thus (X_1, X_2) is a partition of $V(G)$, and we need to investigate $cutrk_G(X_1)$. Let $N(u) = (G_1, lab_1)$. Thus $V(G_1) = X_1$. If $x, y \in X_1$, and $lab_1(x) = lab_1(y)$, then x, y are adjacent in G to the same members of X_2, from the properties of the iterative construction of (G, lab); and since the function lab_1 has at most k different values, it follows that X_1 can be partitioned into k subsets so that the members of each subset have the same neighbours in X_2. Consequently $cutrk_G(X_1) \le k$. Since this applies for every edge of T, we deduce that (T, L) is a branch-decomposition of $cutrk_G$ with width at most k. Hence $\mathrm{rwd}(G) \le k = \mathrm{cwd}(G)$.

Now we show the second statement of the theorem, that $\mathrm{cwd}(G) \le 2^{\mathrm{rwd}(G)+1} - 1$. Let $k = \mathrm{rwd}(G)$ and (T, L) be a rank-decomposition of G of width k. By subdividng one edge of T, and suppressing all other vertices of T with degree 2, we may assume that T is a rooted binary tree; its root has degree 2, and all other vertices have degree 1 or 3.

For $v \in V(T)$, let $D_v = \{x \in V(G) : L(x)$ is a descendant of v in $T\}$, and let G_v denote the subgraph of G induced on D_v. We claim that for every $v \in V(T)$, there is a map lab_v and a $(2^{k+1} - 1)$-expression t_v with value (G_v, lab_v), such that

(i) if $lab_v(x) = 1$ then $x \in D_v$ is nonadjacent to every vertex of $G \setminus D_v$,
(ii) if $x, y \in D_v$ and there exists $z \in V(G) \setminus D_v$ such that x is adjacent to z but y is not, then $lab_v(x) \ne lab_v(y)$,
(iii) for each $x \in D_v$, $lab_v(x) \in \{1, 2, \dots, 2^k\}$.

We prove this by induction on the number of vertices of T that are descendants of v. If v is a leaf, let $t_v = \cdot_1$. Then t_v satisfies the above conditions. Thus we may assume that v has exactly two children v_1, v_2.

By the inductive hypothesis, there are $(2^{k+1} - 1)$-expressions t_1, t_2 with values (G_{v_i}, lab_{v_i}) for $i = 1, 2$, satisfying the statements above. Let F be the set of pairs (i, j) with $i, j \in \{1, 2, \dots, 2^k\}$, such that there is an edge xy of G, with $x \in D_{v_1}$, $lab_{v_1}(x) = i$, $y \in D_{v_2}$ and $lab_{v_2}(y) = j$. It follows from the second condition above that if $(i, j) \in F$ then every vertex $x \in D_{v_1}$ with $lab_{v_1}(x) = i$ is adjacent in G to every vertex $y \in D_{v_2}$ with $lab_{v_2}(y) = j$. Let

$$t^* = \left(\mathop{\circ}_{(i,j)\in F} \eta_{i,j+2^k-1} \right) \left(t_{v_1} \oplus \left(\mathop{\circ}_{i=2}^{2^k} \rho_{i\to i+2^k-1} \right) (t_{v_2}) \right).$$

Then t^* is a $(2^{k+1} - 1)$-expression with value (G_v, lab^*) say, and it satisfies the first two displayed conditions above. However, it need not yet satisfy the third. Let us choose a $(2^{k+1} - 1)$-expression t_v with value (G_v, lab_v) say, satisfying the first two conditions above, and satisfying the following:

- $\{lab_v(x) : x \in D_v\}$ is minimal
- subject to this condition, $\max(lab_v(x) : x \in D_v)$ $(= r$ say) is as small as possible.

(We call these the "first and second optimizations".) For $i = 1, \dots, r$ let $X_i = \{x \in D_v : lab_v(x) = i\}$. The definition of r implies that $X_r \ne \emptyset$. If there exists i with $2 \le i < r$ such that $X_i = \emptyset$, then applying the function $\rho_{r\to i}$ to t_v produces a k-expression contradicting the second optimization. Thus, $X_2, \dots, X_r$ are all nonempty. For $1 \le i \le r$ let Y_i be the set of vertices of $V(G) \setminus D_v$ with a neighbour in X_i. From the first condition above, $Y_1 = \emptyset$. From the second condition above, every vertex in X_i is adjacent to every member of Y_i for all i with $1 \le i \le r$. If there exist i, j with $1 \le i < j \le r$ such that $Y_i = Y_j$, then applying $\rho_{j \to i}$ to t_v produces a k-expression contradicting the first optimization. Thus $Y_1, \dots, Y_r$ are all distinct.

Let M be the matrix $(m_{ij} : i \in D_v, j \in V(G) \setminus D_v)$, where $m_{ij} = 1$ if i, j are adjacent and 0 otherwise. Then M has $r-1$ distinct nonzero rows. Since (T, L) has width k, it follows that M has rank at most k, and therefore M has at most $2^k - 1$ distinct nonzero rows (this is an easy fact about any matrix over GF(2)). We deduce that $r \le 2^k$, and therefore t_v satisfies the third condition above.

This completes the proof that the k-expressions t_v exist as described above. In particular, if v is the root of T then $G_v = G$, and so t_v is a $2^{k+1} - 1$-expression of G. We deduce that $cwd(G) \le 2^{k+1} - 1$.

□

The above proof gives an algorithm that converts a rank-decomposition of order k into a $(2^{k+1} - 1)$-expression. Let $n = |V(G)|$, and let (T, L) be the input rank-decomposition. At each non-leaf vertex v of T, we first construct F, in $O((2^k)^2) = O(1)$ time. Then merging sets with the same neighbours outside D_v will take time $O\left(2^{2k} n\right) = O(n)$. The number of non-leaf vertices v of T is $O(n)$. Therefore, the time complexity is $O(n^2)$. Note that we may assume that checking the adjacency of two vertices can be done in constant time, because we preprocess the input to construct an adjacency matrix in time $O(n^2)$.

Corollary 6.4. *For given k, there is an algorithm that, with input an n-vertex graph G, either concludes that $\mathrm{rwd}(G) > k$ or outputs a rank-decomposition of width at most $3k + 1$. Its running time is $O(n^9 \log n)$.*

Proof. cutrk^*_G can be calculated in time $O(n^3)$, so the claim follows from 5.5. □

Corollary 6.5. *For given k, there is an algorithm that, with input a graph G, either concludes that $\mathrm{cwd}(G) > k$ or outputs a $(2^{3k+2} - 1)$-expression of G. Its running time is $O(n^9 \log n)$.*

Proof. This is immediate from 6.4 and 6.3. □

7. Application to the Branch-width of a Matroid

In this section, we will show an interpolation of the connectivity function λ of a matroid that can be evaluated faster than $\lambda_{\min}$, and a method to apply the matroid intersection theorem to avoid the general submodular function minimization algorithms. So, approximating the branch-width of matroids can be done much faster than that of general symmetric submodular functions.

The following proposition is due to Jim Geelen (private communication).

Proposition 7.1. *Let $\mathcal{M}$ be a matroid with rank function r. Let $\lambda(X) = r(X) + r(E(\mathcal{M}) \setminus X) - r(\mathcal{M}) + 1$ be the connectivity function of $\mathcal{M}$. Let B be a base of $\mathcal{M}$. Then*

$$\lambda_B(X, Y) = r(X \cup (B \setminus Y)) + r(Y \cup (B \setminus X)) - |B \setminus X| - |B \setminus Y| + 1$$

is an interpolation of λ.

Proof. We verify the three conditions of the definition of an interpolation.

1) If $Y = E(\mathcal{M}) \setminus X$, then

$$\lambda_B(X,Y) = r(X) + r(Y) - r(B\cap X) - r(B\cap Y) + 1 = r(X) + r(Y) - r(\mathcal{M}) + 1 = \lambda(X).$$

2) Let $X_1 \subseteq X_2$ and $Y_1 \subseteq Y_2$. Then

$$r(X_2 \cup (B\setminus Y_2)) \geq r(X_1 \cup (B\setminus Y_2)) \geq r(X_1 \cup (B \setminus Y_1)) - (|B\setminus Y_1| - |B\setminus Y_2|).$$

Therefore,

$$r(X_2 \cup (B\setminus Y_2)) - |B\setminus Y_2| \geq r(X_1 \cup (B\setminus Y_1)) - |B\setminus Y_1|.$$

Similarly,

$$r(Y_2 \cup (B\setminus X_2)) - |B\setminus X_2| \geq r(Y_1 \cup (B\setminus X_1)) - |B\setminus X_1|.$$

By adding both inequalities, we deduce that $\lambda_B(X_2,Y_2) \geq \lambda_B(X_1,Y_1)$.

3) Let $X_1\cap Y_1 = \emptyset$ and $X_2 \cap Y_2 = \emptyset$. It is easy to show that

$$(P\cap R)\cup(Q\cap S) \subseteq (P\cup Q)\cap(R\cup S)$$

for any choice of sets P, Q, R, S. Since r is submodular and increasing,

$$\begin{aligned} & r(X_1\cup(B\setminus Y_1)) + r(X_2\cup(B\setminus Y_2)) \\ \geq\ & r((X_1\cup(B\setminus Y_1))\cup(X_2\cup(B\setminus Y_2))) + r((X_1\cup(B\setminus Y_1))\cap(X_2\cup(B\setminus Y_2))) \\ \geq\ & r((X_1\cup X_2)\cup(B\setminus(Y_1\cap Y_2))) + r((X_1\cap X_2)\cup(B\setminus(Y_1\cup Y_2))). \end{aligned}$$

Similarly

$$r(Y_1\cup(B\setminus X_1)) + r(Y_2\cup(B\setminus X_2)) \geq r((Y_1\cup Y_2)\cup(B\setminus(X_1\cap X_2))) + r((Y_1\cap Y_2)\cup(B\setminus(X_1\cup X_2))).$$

But also

$$|B\setminus X_1| + |B\setminus X_2| = |B\setminus(X_1\cap X_2)| + |B\setminus(X_1\cup X_2)|.$$

Adding, we deduce that

$$\lambda_B(X_1,Y_1) + \lambda_B(X_2,Y_2) \geq \lambda_B(X_1\cap X_2, Y_1\cup Y_2) + \lambda(X_1\cup X_2, Y_1\cap Y_2).$$

$\square$

To apply 5.5 to matroid branch-width, we needed a submodular function minimization algorithm that, given a matroid $\mathcal{M}$ and two disjoint subsets X and Y, will output $Z \subseteq E(\mathcal{M})$ such that $X \subseteq Z \subseteq E(\mathcal{M})\setminus Y$ and $\lambda(Z)$ is minimum. We would like to show that this can be done by the matroid intersection algorithm. Let $\mathcal{M}_1 = \mathcal{M}/X\setminus Y$, $\mathcal{M}_2 = \mathcal{M}\setminus X/Y$. Let r_1, r_2 be the rank function of $\mathcal{M}_1$, $\mathcal{M}_2$, respectively. Then by the matroid intersection algorithm, we can find $U \subseteq E(\mathcal{M})\setminus X\setminus Y$ minimizing $r_1(U) + r_2(E(\mathcal{M})\setminus X\setminus Y\setminus U)$. Using the fact $r_1(U) = r(U\cup X) - r(X)$, $r_2(U) = r(U\cup Y) - r(Y)$, we get Z, $X \subseteq Z \subseteq E(\mathcal{M})\setminus Y$ minimizing $\lambda(Z)$. And this can be done in $O(n^3)$ time (in terms of the rank oracle), where $n = |E(\mathcal{M})|$.

We deduce:

Corollary 7.2. *For given k, there is an algorithm that, with input an n-element matroid $\mathcal{M}$, given by its rank oracle, either concludes that $\mathrm{bw}(\mathcal{M}) > k$ or outputs a branch-decomposition of $\mathcal{M}$ of width at most $3k-1$. Its running time and number of oracle calls is at most $O(n^4)$.*

Proof. Pick a base B of $\mathcal{M}$ arbitrarily. We use λ_B as an interpolation of λ. For a given partition (A, B), finding a base X can be done in time $O(n)$. Finding Z to prove that X is not well-linked can be done in $O(2^{3k-2}n^3)$. Therefore, the time complexity is $O(n + n(n + 2^{3k-2}n^3)) = O(8^k n^4)$. □

Acknowledgment. We would like to thank Jim Geelen for suggesting the branch-width of symmetric submodular functions and Proposition 7.1, and Frank Gurski for discussions which led to a slight improvement of Proposition 6.3.

References

[1] Hans L. Bodlaender, "A linear-time algorithm for finding tree-decompositions of small treewidth", *SIAM J. Comput.* **25** (1996), no. 6, 1305–1317.

[2] Derek G. Corneil, Michel Habib, Jean-Marc Lanlignel, Bruce Reed, and Udi Rotics, "Polynomial time recognition of clique-width ≤ 3 graphs (extended abstract)", Gonnet, Gastón H. (ed.) et al., *LATIN 2000: Theoretical informatics.* 4th Latin American symposium, Punta del Este, Uruguay, April 10-14, 2000. Proceedings. Berlin: *Springer. Lect. Notes Comput. Sci.* 1776, 126-134 , 2000.

[3] Bruno Courcelle, J. A. Makowsky, and Udi Rotics, "Linear time solvable optimization problems on graphs of bounded clique-width", *Theory Comput. Syst.* **33** (2000), no. 2, 125–150.

[4] Derek G. Corneil, Y. Perl, and L. K. Stewart, "A linear recognition algorithm for cographs", *SIAM J. Comput.* **14** (1985), no. 4, 926–934.

[5] Derek G. Corneil and Udi Rotics, "On the relationship between clique-width and treewidth (extended abstract)", *Graph-theoretic concepts in computer science* (Boltenhagen, 2001), *Lecture Notes in Comput. Sci.*, vol. 2204, Springer, Berlin, 2001, pp. 78–90.

[6] Bruno Courcelle, "The expression of graph properties and graph transformations in monadic second-order logic", *Handbook of graph grammars and computing by graph transformation*, Vol. 1, World Sci. Publishing, River Edge, NJ, 1997, pp. 313–400.

[7] Bruno Courcelle and Stephan Olariu, "Upper bounds to the clique width of graphs", *Discrete Appl. Math.* **101** (2000), no. 1-3, 77–114.

[8] Wolfgang Espelage, Frank Gurski, and Egon Wanke, "How to solve NP-hard graph problems on clique-width bounded graphs in polynomial time", *Graph-theoretic concepts in computer science* (Boltenhagen, 2001), *Lecture Notes in Comput. Sci.*, vol. 2204, Springer, Berlin, 2001, pp. 117–128.

[9] Wolfgang Espelage, Frank Gurski, and Egon Wanke, "Deciding clique-width for graphs of bounded tree-width", *Journal of Graph Algorithms and Applications* **7** (2003), no. 2, 141–180.

[10] Frank Gurski and Egon Wanke, "The tree-width of clique-width bounded graphs without $K_{n,n}$", *Graph-theoretic concepts in computer science* (Konstanz, 2000), *Lecture Notes in Comput. Sci.*, vol. 1928, Springer, Berlin, 2000, pp. 196–205.

[11]

[12] Satoru Iwata, Lisa Fleischer, and Satoru Fujishige, "A combinatorial strongly polynomial algorithm for minimizing submodular functions", *Journal of the ACM* **48** (2001), no. 4, 761–777.

[13] Öjvind Johansson, "$\log n$-approximative NLC_k-decomposition in $O(n^{2k+1})$ time (extended abstract)", *Graph-theoretic concepts in computer science* (Boltenhagen, 2001), *Lecture Notes in Comput. Sci.*, vol. 2204, Springer, Berlin, 2001, pp. 229–240.

[14] Daniel Kobler and Udi Rotics, "Edge dominating set and colorings on graphs with fixed clique-widthi", *Discrete Appl. Math.* **126** (2003), no. 2-3, 197–221.

[15]

[16] James G. Oxley, *Matroid theory*, Oxford Science Publications, The Clarendon Press Oxford University Press, New York, 1992.
[17] Bruce A. Reed, "Tree width and tangles: a new connectivity measure and some applications", *Surveys in Combinatorics*, 1997 (London), *London Math. Soc. Lecture Note Ser.*, vol. 241, Cambridge Univ. Press, Cambridge, 1997, pp. 87–162.
[18] Neil Robertson and P. D. Seymour, "Graph minors. V. Excluding a planar graph", *J. Combin. Theory Ser. B* **41** (1986), no. 1, 92–114.
[19] Neil Robertson and P. D. Seymour, "Graph minors. X. Obstructions to tree-decomposition", *J. Combin. Theory Ser. B* **52** (1991), no. 2, 153–190.
[20]
[21] K. Truemper, *Matroid decomposition*, Academic Press Inc., Boston, MA, 1992.
[22] Egon Wanke, "k-NLC graphs and polynomial algorithms", *Discrete Appl. Math.* **54** (1994), no. 2-3, 251–266.

E-mail address: sangil@princeton.edu

E-mail address: pds@math.princeton.edu

Program in Applied and Computational Mathematics, Princeton University, USA