FASTER ALGORITHMS FOR VERTEX PARTITIONING PROBLEMS PARAMETERIZED BY CLIQUE-WIDTH

SANG-IL OUM, SIGVE HORTEMO SÆTHER, AND MARTIN VATSHELLE

Abstract. Many NP-hard problems, such as Dominating Set, are FPT parameterized by clique-width. For graphs of clique-width k given with a k-expression, Dominating Set can be solved in $4^k n^{\mathcal{O}(1)}$ time. However, no FPT algorithm is known for computing an optimal k-expression. For a graph of clique-width k, if we rely on known algorithms to compute a $(2^{3k}-1)$-expression via rank-width and then solving Dominating Set using the $(2^{3k}-1)$-expression, the above algorithm will only give a runtime of $4^{2^{3k}} n^{\mathcal{O}(1)}$. There have been results which overcome this exponential jump; the best known algorithm can solve Dominating Set in time $2^{\mathcal{O}(k^2)} n^{\mathcal{O}(1)}$ by avoiding constructing a k-expression [Bui-Xuan, Telle, and Vatshelle. Fast dynamic programming for locally checkable vertex subset and vertex partitioning problems. *Theoret. Comput. Sci.*, 2013. doi: 10.1016/j.tcs.2013.01.009]. We improve this to $2^{\mathcal{O}(k \log k)} n^{\mathcal{O}(1)}$. Indeed, we show that for a graph of clique-width k, a large class of domination and partitioning problems (LC-VSP), including Dominating Set, can be solved in $2^{\mathcal{O}(k \log k)} n^{\mathcal{O}(1)}$. Our main tool is a variant of rank-width using the rank of a 0-1 matrix over the rational field instead of the binary field.

1. Introduction

Parameterized complexity is a field of study dedicated to solving NP-hard problems efficiently on restricted inputs, and has grown to become a well known field over the last 20 years. Especially the subfields of Fixed Parameter Tractable (FPT) algorithms and kernelizations have attracted the interest of many researchers. Parameterized algorithms measure the runtime in two parameters; the input size n and a secondary measure k (called a parameter, either given as part of the input or being computable from the input). An algorithm is FPT if it has runtime $f(k)n^{\mathcal{O}(1)}$. Since we study NP-hard problems, we must expect that $f(k)$ is exponentially larger than n for some instances. However, a good parameter is one where $f(k)$ is polynomial in n for a large class of inputs. For a survey on parameterized complexity and FPT, we refer the reader to [13, 12, 22].

The *clique-width* of a graph G, introduced by Courcelle and Olariu [11], is the minimum k such that G can be expressed by a k-expression, where a k-expression is an algebraic expression using the following four operations:

(Oum) Department of Mathematical Sciences, KAIST, Daejeon, South Korea

(Sæther and Vatshelle) Department of Informatics, University of Bergen, Norway

E-mail addresses: sangil@kaist.edu, sigve.sether@ii.uib.no, vatshelle@ii.uib.no.

Key words and phrases. clique-width, parameterized complexity, dynamic programming, generalized domination, rank-width.

The first author is supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (2011-0011653). The second and third authors are supported by the Research Council of Norway.

- $i(v)$: construct a graph consisting of a single vertex with label $i \in \{1, 2, \ldots, k\}$.
- $G_1 \oplus G_2$: take the disjoint union of labelled graphs G_1 and G_2.
- $\eta_{i,j}$ for distinct $i, j \in \{1, 2, \ldots, k\}$: add an edge between each vertex of label i and each vertex of label j.
- $\rho_{i \to j}$ for $i, j \in \{1, 2, \ldots, k\}$: relabel each vertex of label i to j.

Clique-width is a well-studied parameter in parameterized complexity theory. It is therefore interesting to be able to expand our knowledge on the parameter and to improve on the preciseness of problem complexity when parameterizing by clique-width.

Courcelle, Makowsky, and Rotics [10] showed that, for an input graph of clique-width at most k, every problem expressible in $MSOL_1$ (monadic second-order logic of the first kind) can be solved in FPT time parameterized by k if a k-expression for the graph is given together with the input graph. Later, Oum and Seymour [25] gave an algorithm to find a $(2^{3k+2} - 1)$-expression of a graph having clique-width at most k in time $2^{3k} n^{\mathcal{O}(1)}$.[1] By combining these results, we deduce that for an input graph of clique-width at most k, every $MSOL_1$ problem is in FPT, even if a k-expression is not given as an input. However the dependency in k is huge and can not be considered of practical interest. In order to increase the practicality of FPT algorithms, it is very important to control the runtime as a function of k.

If we rely on finding an approximate k-expression first and then doing dynamic programming on the obtained k-expression, we have two ways to make improvements; either we improve the algorithm that uses the k-expression, or we find a better approximation for clique-width. Given a k-expression, Independent Set and Dominating Set can be solved in time $2^k n^{\mathcal{O}(1)}$ [16] and $4^k n^{\mathcal{O}(1)}$ [3], respectively. Lokshtanov, Marx and Saurabh [19] show that unless the Strong ETH fails[2], Dominating Set can not be solved in $(3 - \epsilon)^k n^{\mathcal{O}(1)}$ time even if a k-expression is given[3]. Hence, there is not much room for improvement in the existing algorithms when a k-expression is given.

There are no known FPT algorithms for computing optimal k-expressions, and the best known FPT algorithm for approximating an optimal k-expression via rank-width has an approximation ratio which is exponential in the optimal clique-width [23]. Therefore, even for the simple NP-hard problems such as Independent Set and Dominating Set, all known algorithms following this procedure has a runtime where the dependency is double exponential in the clique-width. The question of finding a better approximation algorithm for clique-width is an important and challenging open problem.

However, there is a way around this by avoiding a k-expression: Bui-Xuan, Telle and Vatshelle [5] showed that by doing dynamic programming directly on a rank decomposition, Dominating Set can be solved in $2^{k^2} n^{\mathcal{O}(1)}$ for graphs of clique-width k. Their algorithm with a runtime of $2^{\mathcal{O}(k^2)} n^{\mathcal{O}(1)}$ is not only for Independent Set and Dominating Set but also for a wide range of problems, called

[1] Later, Oum [23] obtained an improved algorithm to find a $(2^{3k} - 1)$-expression of a graph having clique-width at most k in time $2^{3k} n^{\mathcal{O}(1)}$.

[2] The Strong Exponential Time Hypothesis (Strong ETH) states that SAT can not be solved in $\mathcal{O}((2 - \epsilon)^n)$ time for any constant $\epsilon > 0$. Here n denotes the number of variables.

[3] Their proof uses pathwidth, but the statement holds since clique-width is at most 1 higher than pathwidth.

$d(\pi)$	Standard name
d	d-Dominating set
$d+1$	Induced d-Regular Subgraph
d	Subgraph of Min Degree $\geq d$
$d+1$	Induced Subgraph of Max Degree $\leq d$
2	Strong Stable set or 2-Packing
2	Perfect Code or Efficient Dominating set
2	Total Nearly Perfect set
2	Weakly Perfect Dominating set
2	Total Perfect Dominating set
2	Induced Matching
2	Dominating Induced Matching
2	Perfect Dominating set
1	Independent set
1	Dominating set
1	Independent Dominating set
1	Total Dominating set

Table 1. A table of some vertex subset properties whose optimization problems belong to LC-VSP. The meaning of the problem specific constant $d(\pi)$ is discussed in subsection 2.3.

$d(\pi)$	Standard name
1	H-coloring or H-homomorphism
1	H-role assignment or H-locally surjective homomorphism
2	H-covering or H-locally bijective homomorphism
2	H-partial covering or H-locally injective homomorphism

Table 2. A table of some homomorphism problems in LC-VSP for fixed simple graph H. These are expressible with a degree constraint matrix D_q where $q(\pi) = |V(H)|$. The meaning of D_q, $d(\pi)$ and $q(\pi)$ is explained in subsection 2.3.

the *locally checkable vertex subset and partitioning problems* (LC-VSP problems). Tables 1 and 2 list some well known problems in LC-VSP.

In this paper we improve on these results by using a slightly modified definition of rank-width, called $\mathbb{Q}$-*rank-width*, based on the rank function over the rational field instead of the binary field. The idea of using fields other than the binary field for rank-width was investigated earlier in [18], but our work is the first to use $\mathbb{Q}$-rank-width to speed up an algorithm.

We will show the following:

- For any graph, its $\mathbb{Q}$-rank-width is no more than its clique-width.
- There is an algorithm to find a decomposition confirming that $\mathbb{Q}$-rank-width is at most $3k+1$ for graphs of $\mathbb{Q}$-rank-width at most k in time $2^{3k}n^{\mathcal{O}(1)}$.
- If a graph has $\mathbb{Q}$-rank-width at most k, then every fixed LC-VSP problem can be solved in $2^{\mathcal{O}(k \log k)}n^{\mathcal{O}(1)}$-time.

This allows us to construct an algorithm that runs in time $2^{\mathcal{O}(k \log k)} n^{\mathcal{O}(1)}$ for graphs of clique-width at most k and solve every fixed LC-VSP problem, improving the previous runtime $2^{\mathcal{O}(k^2)} n^{\mathcal{O}(1)}$ of the algorithm by Bui-Xuan et al. [7].

We also relate the parameter $\mathbb{Q}$-rank-width to other existing parameters. There are several factors affecting the quality of a parameter, such as: Can we compute or approximate the parameter? Which problems can we solve in FPT time? Can we reduce the exponential dependency in the parameter for specific problems? And, how large and natural is the class of graphs having a bounded parameter value?

This paper is organized as follows: In Section 2 we introduce the main parts of the framework used by Bui-Xuan et al. [7], including the general algorithm they give for LC-VSP problems. Section 3 revolves around $\mathbb{Q}$-rank-width and is where the results of this paper reside. We show how $\mathbb{Q}$-rank-width relates to clique-width, and reveal why we have a good FPT algorithm for approximating a decomposition. In Section 4, we give our main result, which is an improved upper bound on solving LC-VSP problems parameterized by clique-width when we are not given a decomposition. We end the paper with Section 5 containing some concluding remarks and open problems.

2. Framework

We write $V(G)$ and $E(G)$ to denote the set of vertices and edges, respectively, of a graph G. For $A \subseteq V(G)$, let $\overline{A} = V(G) \setminus A$. For a vertex $v \in V(G)$, let $N_G(v)$ be the set of all neighbours of v in G. We omit the subscript if it is clear from the context. For a set $S \subseteq V(G)$ we define $N(S) = \bigcup_{v \in S} N(v) \setminus S$.

2.1. Branch Decompositions.

The algorithm of Bui-Xuan et al. [7] needs a branch decomposition as input. A *branch decomposition* (T, δ) of a graph G consists of a subcubic tree T (a tree of maximum degree 3) and a bijective function δ from the leaves of T to the vertices of G. (Note that this definition differs from that of [27] by δ mapping to the vertices of G instead of the edges.)

Every edge in a tree splits the tree into two connected components. In a branch decomposition (T, δ) for a graph G, we say that each edge e of T induces a *cut* in G. This induced cut is a bipartition $(A, \overline{A})$ of the vertices of $V(G)$ so that A is the set of vertices mapped by δ from vertices of one component of $T - e$, and $\overline{A}$ is the set of vertices mapped by δ from the other component of $T - e$ (see Figure 1).

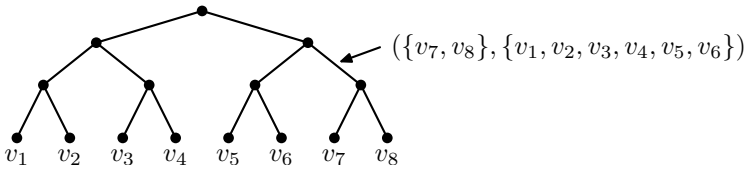

FIGURE 1. A branch decomposition (T, δ) of a graph with 8 vertices. The leaves of T is mapped by δ to each of the vertices. Each edge in T induce a cut.

A function $f : 2^{V(G)} \to \mathbb{R}$ is called a *cut-function* if it is symmetric, that is $f(A) = f(V(G) \setminus A)$ for all $A \subseteq V(G)$. As there is a bijection from each subset $A \subseteq V(G)$ to each cut $(A, \overline{A})$, we may abuse notation slightly and say that f is also defined for cuts of $V(G)$, by regarding $f(A, \overline{A})$ as $f(A)$.

Given a cut-function f and a branch decomposition (T, δ) of a graph G,

- the *f-width* of (T, δ) is the maximum value of f over all the cuts of (T, δ), and
- the *f-width* of G is the minimum f-width over all possible branch decompositions of G.

If $|V(G)| \leq 1$, then G admits no branch decomposition and we define its f-width to be $f(\emptyset)$.

Many width parameters of graphs can be defined in terms of f-width for some cut-function f. For example, in a graph G, if we define $f(A)$ to be the number of maximal independent sets of the subgraph of G induced by edges having one end in A and the other end in $\overline{A}$, then the f-width is exactly the boolean-width [6].

When we speak of the f-width of a graph, we address it as a *width parameter* of the graph.

2.2. **Neighbourhood Equivalence.** Two sets of vertices S_1, S_2 are *neighbourhood equivalent* if they have the same set of neighbours, in other words, $N(S_1) = N(S_2)$. We are particularly interested in neighbourhood equivalence in bipartite graphs, or more specifically, cuts defined by a branch decomposition. This concept was generalized with respect to cuts in [7]. We define the *d-neighbour equivalence* relation $\equiv_A^d$, and use this to define the parameter nec_d.

For a cut $(A, \overline{A})$ of a graph G, and a positive integer d, two subsets $X, Y \subseteq A$ are *d-neighbour equivalent*, $X \equiv_A^d Y$, over $(A, \overline{A})$ if:

$$\text{for each vertex } v \in \overline{A}, \quad \min\{d,\ |N(v) \cap X|\} = \min\{d,\ |N(v) \cap Y|\}.$$

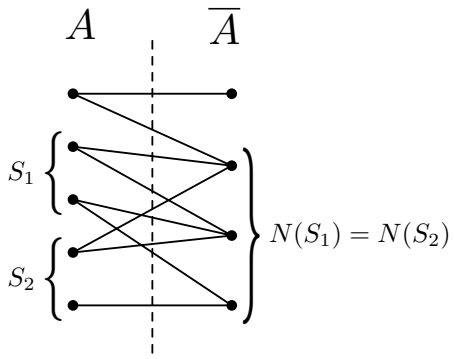

FIGURE 2. The sets S_1 and S_2 are neighbourhood equivalent over $(A, \overline{A})$. That is, $S_1 \equiv_A^1 S_2$. However, in this example it is not the case that $S_1 \equiv_A^2 S_2$.

The *number of d-neighbour equivalence classes*, $nec_d(A)$, is the number of equivalence classes of $\equiv_A^d$ over $(A, \overline{A})$.

In other words, $X \equiv_A^d Y$ over the cut $(A, \overline{A})$ if each vertex in $\overline{A}$ is either adjacent to at least d vertices in both X and Y, or is adjacent to exactly the same number of vertices in X as in Y. The algorithm in [7] uses this relation to limit the number of partial solutions to try. Therefore, the runtime is dependent on the number of d-neighbour equivalence classes.

2.3. **Locally Checkable Vertex Subset and Vertex Partitioning Problems.** Telle and Proskurowski [29] introduced the *Locally Checkable Vertex Subset and Vertex Partitioning problems* (LC-VSP), also called $[\sigma,\rho]$-problems and D_q-partition problems. This is a framework to describe many well-known graph problems, see [29, 7]. Tables 1 and 2 list some of them. For completeness, we give the definitions of the problem class LC-VSP, however, they are not used directly in this paper and can be skipped by the reader.

For finite or co-finite sets σ and ρ of non-negative integers, a set S of vertices of a graph G is a $[\sigma,\rho]$-*set* of G if for each vertex v of G,

$$|N(v)\cap S| \in \begin{cases}\sigma & \text{if } v\in S,\\ \rho & \text{if } v\in V(G)\setminus S.\end{cases}$$

The *Locally Checkable Vertex Subset problems* (LC-VS), or $[\sigma,\rho]$-*problems*, are those problems that consist of finding a minimum or maximum $[\sigma,\rho]$-set of the input graph.

The *LC-VSP problems*, or D_q-*partition problems*, is a generalization of the LC-VS problems. A *degree constraint matrix* D_q is a $q\times q$ matrix such that each cell is a finite or co-finite set of non-negative integers. We say that a partition $V_1, V_2, \ldots, V_q$ of $V(G)$ satisfies D_q if for $1\le i,j\le q$, the number of neighbours in V_j of a vertex of V_i is in the set $D_q[i,j]$. In other words,

$$|N(v)\cap V_j| \in D_q[i,j] \text{ for all } 1\le i,j\le q \text{ and } v\in V_i.$$

For a given degree constraint matrix D_q, the LC-VSP problem is to decide whether the vertex set of a graph admits a partition satisfying D_q.

For each LC-VSP-problem π, there are two problem-specific constants $d(\pi)$ and $q(\pi)$. The number $q(\pi)$ equals the number of parts in a partition that the problem requests, or equivalently, the row/column size of the constraint matrix (i.e., for problem π with degree constraint matrix D_q we have $q=q(\pi)$. The number $d(\pi)$ is defined to be one more than the largest number in all the finite sets and in all the complements of the co-finite sets of the degree constraint matrix used for expressing π. If all the finite sets and complements of co-finite sets are empty, $d(\pi)$ is zero.

For example, Dominating Set can be described by a degree constraint matrix D_2 where $D_2[1,1]=D_2[1,2]=D_2[2,2]=\mathbb{N}$ and $D_2[2,1]=\mathbb{N}\setminus\{0\}$, and we ask to minimize $|V_1|$. If we alter $D_2[1,1]$ to $\{0\}$, the problem is changed to Independent Dominating Set, as no vertex in V_1 can be adjacent to another vertex in V_1. For both problems we have a 2×2-matrix, and so $q(\pi)=2$ and $d(\pi)=0+1=1$.

The algorithm of Bui-Xuan et al. [7] solves each of the LC-VSP problems with a runtime dependent on $nec_{d(\pi)}$-width and $q(\pi)$ by using the d-neighbour equivalence relation $\equiv_A^{d(\pi)}$.

Theorem 2.1 (Bui-Xuan et al. [7, Theorem 2])**.** *Let π be a problem in LC-VSP. For a graph G given with its branch decomposition of $nec_{d(\pi)}$-width k, the problem π can be solved in time $\mathcal{O}\big(|V(G)|^4\cdot q(\pi)\cdot k^{3q(\pi)}\big)$.*

3. $\mathbb{Q}$-rank-width of a Graph

The $\mathbb{Q}$-*cut-rank* function of a graph G is a function on the subsets of $V(G)$ that maps $X\subseteq V(G)$ to the rank of an $|X|\times\big|\overline{X}\big|$-matrix $A=(a_{ij})_{i\in X,j\in\overline{X}}$ over the rational field such that $a_{ij}=1$ if i and j are adjacent in G and $a_{ij}=0$ otherwise.

We let $\text{cutrk}^{\mathbb{Q}}(X)$ denote the $\mathbb{Q}$-cut-rank of X. For a subset $X \subseteq V(G)$, the matrix A associated with $\text{cutrk}^{\mathbb{Q}}(X)$ is the *adjacency matrix* of the cut $(X, \overline{X})$. Note that if the underlying field of the matrix A is the binary field $GF(2)$, then we obtain the definition of the usual cut-rank function [25]. By $\mathbb{Q}$-*rank-width* of a graph, we mean its $\mathbb{Q}$-cut-rank-width (see subsection 2.1). We may denote the $\mathbb{Q}$-rank-width simply as $\text{rw}_{\mathbb{Q}}$.

$$A \begin{array}{c} \\ \end{array} \begin{array}{c} \overline{A} \\ \begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

FIGURE 3. The adjacency matrix of the cut depicted in Figure 2. The rank over the rational field is 4, so the $\mathbb{Q}$-cut-rank of this cut is 4.

Since the $\mathbb{Q}$-cut-rank function is symmetric submodular and is computable in polynomial time, by applying the result of Oum and Seymour [25], we get the following theorem.

Theorem 3.1 (Oum and Seymour [25])**.** *There is a $2^{3k}n^{\mathcal{O}(1)}$-time algorithm for which, given a graph G as input and a parameter k, either outputs a branch decomposition for G of $\mathbb{Q}$-rank-width at most $3k+1$ or confirms that $\mathbb{Q}$-rank-width of G is more than k.*

3.1. **$\mathbb{Q}$-rank-width versus clique-width/rank-width.** The question of how useful the $\mathbb{Q}$-rank-width is as a width parameter is hard to answer. To better understand this question, it would be interesting to know the relation to other well-known width parameters such as treewidth, rank-width and clique-width.

The following relates $\mathbb{Q}$-rank-width to the closely related parameter rank-width, yet we see that rank-width can be substantially lower than $\mathbb{Q}$-rank-width.

Lemma 3.2. *For any graph G we have* $\text{rw}(G) \leq \text{rw}_{\mathbb{Q}}(G) \leq \text{cw}(G) \leq 2^{\text{rw}(G)+1} - 1$.

Proof. The first inequality is from the fact that a set of 0-1 vectors linearly dependent over $\mathbb{Q}$ must also be linearly dependent over $GF(2)$.

The second and third inequalities follow from [25, Proposition 6.3] since their proof is not dependent on the type of field rank-width uses. They show that a k-expression can be translated to a branch decomposition where for every cut $(A, \overline{A})$ in the decomposition, either the number of distinct rows or the number of distinct columns in the adjacency matrix M of its induced bipartite graph, is bounded by k. Since this means the rank of M over $\mathbb{Q}$ is at most k, we have $\text{rw}_{\mathbb{Q}}(G) \leq \text{cw}(G)$. The idea of showing $\text{cw}(G) \leq 2^{\text{rw}(G)+1} - 1$, is that a branch decomposition where the adjacency matrix of each cut has its number of distinct columns/rows (approximately) bounded by some k, can be translated to a k-expression. As the number of distinct columns/rows for any 0-1 matrix of rank rw is at most 2^{rw}, we get our inequality. The last two inequalities are also proved in [18]. □

We believe Lemma 3.2 is tight. There are existing results showing that it is almost tight. A $n \times n$ grid has rank-width $n-1$ [17] and clique-width $n+1$ [15], hence the first two inequalities are almost tight. There exist graphs with treewidth k and hence $\mathbb{Q}$-rank-width at most k and clique-width at least $2^{\lfloor k/2 \rfloor - 1}$ [9].

3.2. **$\mathbb{Q}$-rank-width versus treewidth/branch-width.** Oum [24] proved that the rank-width of a graph is less than or equal to its tree-width plus 1. We prove a similar result for $\mathbb{Q}$-rank-width.

In order to show this, we use the notion of *tangles* and *branch-width* of symmetric submodular functions, see [14, 26]. For a symmetric submodular function f on a finite set V, an *f-tangle* $\mathcal{T}$ of order $k+1$ is a set of subsets of V satisfying the following:

(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 $V \setminus \{v\} \notin \mathcal{T}$.

Theorem 3.3 (Robertson and Seymour [27, (3.5)]; Geelen, Gerards, Robertson, and Whittle [14, Theorem 3.2])**.** *There is no f-tangle of order $k+1$ if and only if the branch-width of f is at most k.*

For a set X of edges, let T_X be the set of vertices incident with at least one of the edges in X. For a set X of edges, let $\eta(X) = |T_X \cap T_{E(G)\setminus X}|$, that is the number of vertices incident to both edges in X and edges in $E(G) \setminus X$. Then the *branch-width* of a graph G is the branch-width of the function η on $E(G)$ [27].

Lemma 3.4. *For $X \subseteq E(G)$, we have* $\mathrm{cutrk}^{\mathbb{Q}}(T_X) \leq \eta(X)$.

Proof. Suppose $k = \eta(X)$. Then T_X has at most k vertices having neighbors in $V(G) \setminus T_X$ by the definition of η. Thus $\mathrm{cutrk}^{\mathbb{Q}}(T_X) \leq k$ as the rank of a matrix with at most k non-zero rows is at most k. □

Lemma 3.5. *Let $k \geq 2$. If G has $\mathbb{Q}$-rank-width at least $k+1$, then G has branch-width at least $k+1$.*

Proof. We may assume that G is connected without loss of generality. Let ρ be the $\mathbb{Q}$-cut-rank function of G. Since the $\mathbb{Q}$-rank-width of G is larger than k, there exists a ρ-tangle $\mathcal{T}$ of order $k+1$.

We aim to construct the tangle $\mathcal{U}$ of order $k+1$ as follows. Let

$$\mathcal{U} = \{X \subseteq E(G) : \eta(X) \leq k,\ T_X \in \mathcal{T}\}.$$

We claim that $\mathcal{U}$ is an η-tangle of order $k+1$.

(1) Suppose that $\eta(X) \leq k$ for a set X of edges. We need to show that either $X \in \mathcal{U}$ or $E(G) \setminus X \in \mathcal{U}$. Suppose that $X \notin \mathcal{U}$ and $E(G) \setminus X \notin \mathcal{U}$. Then, $T_X \notin \mathcal{T}$. Since $\rho(T_X) \leq \eta(X) \leq k$ and T is a ρ-tangle, we know that $V(G) \setminus T_X \in \mathcal{T}$. Similarly we deduce that $V(G) \setminus T_{E(G)\setminus X} \in \mathcal{T}$. Moreover since $\eta(X) \leq k$, $T_X \cap T_{E(G)\setminus X} \in \mathcal{T}$ (easy to show by induction—any set of at most k vertices belongs to a ρ-tangle of order $k+1$). This leads a contradiction because $(V(G) \setminus T_X) \cup (T_X \cap T_{E(G)\setminus X}) \cup (V(G) \setminus T_{E(G)\setminus X}) = V(G)$ and $\mathcal{T}$ is a ρ-tangle.

(2) Suppose that $X \cup Y \cup Z = E(G)$ for three sets $X, Y, Z \in \mathcal{U}$. If $v \notin T_X \cup T_Y \cup T_Z$, then v is an isolated vertex. Since G is connected, there is no such v. Thus, $T_X \cup T_Y \cup T_Z = V(G)$ and $T_X, T_Y, T_Z \in \mathcal{T}$. A contradiction.

(3) For each edge e, $\eta(\{e\}) \leq 2$ and therefore if $k \geq 2$, then $T_{\{e\}} \in \mathcal{T}$. So $\{e\} \in \mathcal{U}$.

By (1)–(3), we checked all axioms for η-tangles. □

Theorem 3.6. $\mathrm{rw}_{\mathbb{Q}}(G) \leq \max(\textit{branch-width}(G), 1) \leq \textit{treewidth}(G) + 1$.

Proof. If the branch-width of G is larger than 1, then by Lemma 3.5, we know that the rank-width is at most the branch-width of G. If the branch-width of G is 1, then G is a forest and therefore the rank-width is at most 1. (But G may have edges, even if branch-width of G is 0 and in this case, the rank-width of G is 1.) Robertson and Seymour [27] showed that branch-width is at most tree-width plus 1. □

We remark that an identical proof can be used to show an analogous result for variations of rank-width on different fields.

Figure 4 shows a comparison diagram of graph parameters. The idea of such a diagram is that parameterized complexity results will propagate up and down in this diagram. Positive results propagate upward; for instance, since Dominating Set is solvable in $2^{\mathcal{O}(tw)}n^{\mathcal{O}(1)}$ for a graph of treewidth tw [28], we see that Dominating Set is solvable in $2^{\mathcal{O}(pw)}n^{\mathcal{O}(1)}$ for a graph of pathwidth pw. Negative results propagate downward; for example, since unless ETH fails, Dominating Set can not be solved in $2^{o(pw)}n^{\mathcal{O}(1)}$ where pw is the pathwidth of the input graph [20], so is the case for treewidth, clique-width, $\mathbb{Q}$-rank-width, rank-width and boolean-width. From this table, we can deduce that the entire class LC-VSP cannot be in FPT parameterized by OCT, D2Chordal or D2Perfect unless P = NP, since Dominating Set is NP-hard for both bipartite [2] and chordal graphs [4]. Furthermore, LC-VSP parameterized by either of the remaining parameters is in fact in FPT, since we have FPT algorithms solving all problems in LC-VSP parameterized by rank-width, boolean-width, and D2Interval[4].

4. Bounding nec_d-width by $\mathbb{Q}$-rank-width and its Algorithmic Consequences

Now we know how to find a branch decomposition with a low $\mathbb{Q}$-rank-width. We are going to discuss its nec_d-width to apply Theorem 2.1. Theorem 2.1 provides the runtime of the algorithm in terms of the nec_d-width of the given decomposition. So, if we manage to give a bound on the nec_d-width of a decomposition in terms of the $\mathbb{Q}$-rank-width, we will also get a bound on the runtime of the algorithm in terms of $\mathbb{Q}$-rank-width. We will prove such a bound shortly, but in order to do this we first need the following lemma, based on a proof of Belmonte and Vatshelle [1, Lemma 1].

[4]Given a graph of D2Interval$(G) = k$, we have a fixed-parameter tractable algorithm to construct a branch deceomposition of nec_d-width at most $2^k n^d$ and thus by the algorithm of [7], we have a fixed-parameter tractable algorithm to solve the problem. To do this, we first find a vertex set S of size k so that $G - S$ is an interval graph, and then find a branch decomposition of nec_d-width at most n^d. Arbitrarily adding the vertices of S anywhere in the branch decomposition cannot increase the nec_d-width by more than $2^{|S|}$, and thus the resulting branch decomposition has nec_d-width at most $2^k n^d$. We have a fixed-parameter tractable algorithm to find S shown in [8] and constructing a branch decomposition for $G - S$ of nec_d-width at most n^d can be done in polynomial time by [1].

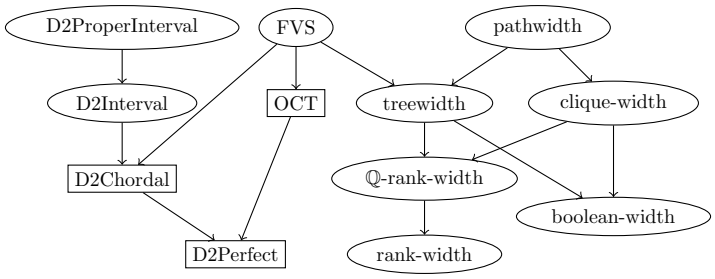

FIGURE 4. A comparison diagram of some graph parameters. A parameter κ_1 is drawn below a parameter κ_2 if there is a constant c such that $\kappa_1(G) \leq c \cdot \kappa_2(G)$ for all graphs G. The abbreviations are: FVS = Feedback Vertex Set number, OCT = Odd Cycle Transversal number, D2Π = Vertex Deletion distance to a member of Π. For the circled parameters all the LC-VSP problems are in FPT, and unless P = NP for each of the remaining parameters at least one of the LC-VSP problems is not in FPT.

Lemma 4.1. *Given a positive integer d and a cut $(A, \overline{A})$ of $\mathbb{Q}$-cut-rank k, for every subset $S \subseteq A$, there exists a subset $R \subseteq S$ so that $|R| \leq dk$ and $R \equiv_A^d S$ over the cut.*

Proof. We proceed by induction on d. If $d = 1$, then let S' be a minimal subset of S so that $S' \equiv_A^1 S$. Since S' is minimal, removing any vertex of S' will decrease $|N(S')|$. Therefore, every vertex of S' is adjacent to at least one vertex that none of the other vertices in S' are adjacent to. In the adjacency matrix M of $(A, \overline{A})$, this means that each of the corresponding rows of S' has a 1 in a column where all the other rows of S' has a 0. Hence, the rows of S' are linearly independent and so $|S'| \leq \text{cutrk}^{\mathbb{Q}}(A) = k$.

So we may assume that $d > 1$. By the above, there exists a subset $S_1 \subseteq S$ such that $|S_1| \leq k$ and $S_1 \equiv_A^1 S$. By the induction hypothesis, there exists a set $S_2 \subseteq (S \setminus S_1)$ so that $S_2 \equiv_A^{d-1} (S \setminus S_1)$ and $|S_2| \leq (d-1)k$.

We claim that $S_1 \cup S_2 \equiv_A^d S$. Let $v \in \overline{A}$. We may assume that v has at most $d-1$ neighbours in $S_1 \cup S_2$.

If v has a neighbour in $S \setminus (S_1 \cup S_2)$, then $|N(v) \cap (S \setminus S_1)| > |N(v) \cap S_2|$ and therefore v has at least $d-1$ neighbours in S_2 and so v has no neighbors in S_1. This contradicts our assumption that $S_1 \equiv_A^1 S$.

Thus v has no neighbour in $S \setminus (S_1 \cup S_2)$. This proves the claim. Since $|S_1 \cup S_2| \leq dk$, this completes the proof of the lemma. □

Lemma 4.1 implies that to count distinct d-neighbour equivalence classes for a cut of a branch decomposition of $\mathbb{Q}$-rank-width k, it is enough to search subsets of size at most dk. The same result is true, even if we replace $\mathbb{Q}$-rank-width with rank-width or boolean-width ([31], [7, Lemma 5]).

Then what is the contribution of $\mathbb{Q}$-rank-width instead of rank-width or boolean-width? Here comes the crucial difference. For both rank-width k or boolean-width k, the number of vertices with distinct neighbourhoods over the cut is no more than 2^k [31, 7]. Putting this together gives a trivial bound of $nec_d \leq 2^{dk^2}$. We

can improve this bound if k is $\mathbb{Q}$-rank-width, thanks to the fact that the row space of some matrix over $\mathbb{Q}$ not only contains all the rows of the matrix, but also all the different sums of the rows in the matrix. So, we can bound $nec_d(A)$ by using a more direct connection between $\mathbb{Q}$-rank-width and the number of distinct d-neighbourhoods than that of the trivial bound.

Theorem 4.2. *If the $\mathbb{Q}$-rank-width of a branch decomposition is k, then the nec_d-width of the same decomposition is no more than $(dk+1)^k = 2^{k \log_2 (dk+1)}$.*

Proof. It is enough to prove that if a cut $(A, \overline{A})$ has $\mathbb{Q}$-cut-rank k, then $nec_d(A) \leq (dk+1)^k$. Let M be the $A \times \overline{A}$ adjacency matrix of the cut $(A, \overline{A})$ over $\mathbb{Q}$.

For a subset S of A, let $\sigma(S)$ be the sum of the row vectors of M corresponding to S. If $\sigma(S) = \sigma(S')$ then $S \equiv_A^d S'$ for all d, because the entries of $\sigma(S)$ represent the number of neighbours in S for each vertex in $\overline{A}$.

By Lemma 4.1, each equivalence class of $\equiv_A^d$ can be represented by a subset S of A having at most dk vertices. Notice that for such S, each entry of $\sigma(S)$ is in $\{0, 1, 2, \ldots, dk\}$.

Let B be a set of k linearly independent columns of M. Since M has rank k, every linear combination of row vectors of M is completely determined by its entries in B. Thus the number of possible values of $\sigma(S)$ is at most $(dk+1)^k$ (see Figure 5). This proves the theorem. □

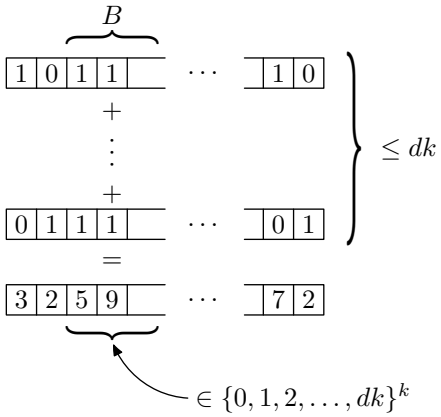

FIGURE 5. As described by Theorem 4.2, we can determine the sum of the vectors by looking at the values in the columns B. As we sum over at most dk rows, and each row either increases the value of a column by exactly one or exactly zero, the number of unique sums possible is at most $|\{0, 1, \ldots\}|^{|B|} = (1+dk)^k$.

This result, combined with Theorems 2.1 and 3.1, shows that all the LC-VSP problems can be solved in time $2^{\mathcal{O}(k \log k)} n^{\mathcal{O}(1)}$. Expressing the runtime in terms of clique-width, we get the following corollary.

Corollary 4.2.1. *Every LC-VSP problem π on n-vertex graphs of clique-width* cw *can be solved in $2^{\mathcal{O}(\mathrm{cw} \log (\mathrm{cw} \cdot d(\pi)) q(\pi))} n^{\mathcal{O}(1)}$-time.*

Proof. Let k be the $\mathbb{Q}$-rank-width of G. By Theorem 3.1 we can find a branch decomposition of $\mathbb{Q}$-rank-width at most $3k+1$ in time $2^{3k} n^{\mathcal{O}(1)}$. By Theorems 2.1

and 4.2, the LC-VSP problem π can be solved in time $2^{9k\log(3k\cdot d(\pi)+1)q(\pi)}n^{\mathcal{O}(1)}$. This completes the proof because $k \leq \text{cw}$ by Lemma 3.2. □

5. Conclusion

If we are given a k-expression as input, the best known FPT algorithm parameterized by k solving the Dominating Set is by Bodlaender et al. [3] and runs in time $4^k n^{\mathcal{O}(1)}$. However, it is currently open whether we can construct a $\mathcal{O}(k)$-expression of an input graph of clique-width at most k in polynomial time. We have shown the existence of algorithms with runtime $2^{\mathcal{O}(\text{cw}\log\text{cw})}n^{\mathcal{O}(1)}$ for all LC-VSP problems, without assuming that a k-expression is given as an input. This still leaves the natural open question:

Open Problem 1. *Can* Independent Set *or* Dominating Set *be solved in* $2^{\mathcal{O}(\text{cw})}n^{\mathcal{O}(1)}$ *time, where* cw *is the clique-width of the graph?*

We know that for a graph of treewidth tw, Independent Set can be solved in time $2^{\mathcal{O}(tw)}n^{\mathcal{O}(1)}$ time. This leads us to an interesting question of what parameters give a single exponential runtime for Independent Set. Two such parameters are the Vertex Deletion Distance to Proper Interval graphs (D2ProperInterval) and the Odd Cycle Transversal number (OCT number):

(1) For a graph G, the D2ProperInterval of a graph is the minimum number of vertices needed to be removed in order to make G into a proper interval graph. For a graph G with D2ProperInterval equal k, Villanger and van 't Hof [30] gave a $6^k n^{\mathcal{O}(1)}$-time algorithm for finding such a set S to be removed. To solve Independent Set on a graph $G = (V, E)$, we guess the intersection S' of S and an optimal solution, and then combining it with the optimal solution of Independent Set on the proper interval graph $G-(S\cup N(S'))$. As Independent Set is solvable in $n^{\mathcal{O}(1)}$ time on proper interval graphs, this yields a $2^{\mathcal{O}(k)}n^{\mathcal{O}(1)}$ time algorithm.

(2) The OCT number of a graph G is the minimum number of vertices needed to remove from G in order to make it bipartite. For a graph G with OCT number equal k, Lokshtanov, Saurabh and Sikdar [21] gave a $3^k n^{\mathcal{O}(1)}$-time algorihm for finding the minimum set S of vertices to remove from G to make it bipartite. As with the algorithm above, we can solve Independent Set by guessing the intersection S' of S and the optimal solution and then combine it with the optimal solution of the bipartite graph $G-(S\cup N(S'))$. As Independent Set is trivially solvable in $n^{\mathcal{O}(1)}$ time on bipartite graphs, this yields a $2^{\mathcal{O}(k)}n^{\mathcal{O}(1)}$ time algorithm.

Note, however, that these parameters are not bounded by treewidth (and thus also not bounded by clique-width), see Figure 4.

References

[1] R. Belmonte and M. Vatshelle. Graph classes with structured neighborhoods and algorithmic applications. *Theoret. Comput. Sci.*, 511:54–65, 2013.

[2] A. A. Bertossi. Dominating sets for split and bipartite graphs. *Inform. Process. Lett.*, 19(1):37–40, 1984.

[3] H. Bodlaender, E. van Leeuwen, J. van Rooij, and M. Vatshelle. Faster algorithms on clique and branch decompositions. In *Proceedings of MFCS*, volume 6281 of *LNCS*, pages 174–185. Springer, 2010.

[4] K. S. Booth and J. H. Johnson. Dominating sets in chordal graphs. *SIAM J. Comput.*, 11(1):191–199, 1982.
[5] B.-M. Bui-Xuan, J. A. Telle, and M. Vatshelle. H-join decomposable graphs and algorithms with runtime single exponential in rankwidth. *Discrete Appl. Math.*, 158(7):809–819, 2010.
[6] B.-M. Bui-Xuan, J. A. Telle, and M. Vatshelle. Boolean-width of graphs. *Theor. Comput. Sci.*, 412(39):5187–5204, 2011.
[7] B.-M. Bui-Xuan, J. A. Telle, and M. Vatshelle. Fast dynamic programming for locally checkable vertex subset and vertex partitioning problems. *Theoret. Comput. Sci.*, 511:66–76, 2013.
[8] Y. Cao and D. Marx. Interval deletion is fixed-parameter tractable. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 122–141, 2014.
[9] D. G. Corneil and U. Rotics. On the relationship between clique-width and treewidth. *SIAM J. Comput.*, 34(4):825–847 (electronic), 2005.
[10] B. Courcelle, J. A. Makowsky, and U. Rotics. Linear time solvable optimization problems on graphs of bounded clique-width. *Theory Comput. Syst.*, 33(2):125–150, 2000.
[11] B. Courcelle and S. Olariu. Upper bounds to the clique width of graphs. *Discrete Appl. Math.*, 101(1–3):77–114, 2000.
[12] R. G. Downey and M. R. Fellows. *Parameterized complexity*. Monographs in Computer Science. Springer-Verlag, New York, 1999.
[13] J. Flum and M. Grohe. *Parameterized complexity theory*. Texts in Theoretical Computer Science. An EATCS Series. Springer-Verlag, Berlin, 2006.
[14] J. F. Geelen, B. Gerards, N. Robertson, and G. Whittle. Obstructions to branch-decomposition of matroids. *J. Combin. Theory Ser. B*, 96(4):560–570, 2006.
[15] M. C. Golumbic and U. Rotics. On the clique-width of some perfect graph classes. *Internat. J. Found. Comput. Sci.*, 11(3):423–443, 2000.
[16] F. Gurski. A comparison of two approaches for polynomial time algorithms computing basic graph parameters. *CoRR*, abs/0806.4073, 2008.
[17] V. Jelínek. The rank-width of the square grid. *Discrete Appl. Math.*, 158(7):841–850, 2010.
[18] M. M. Kanté and M. Rao. The rank-width of edge-coloured graphs. *Theory Comput. Syst.*, 52(4):599–644, 2013.
[19] D. Lokshtanov, D. Marx, and S. Saurabh. Known algorithms on graphs of bounded treewidth are probably optimal. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 777–789, Philadelphia, PA, 2011. SIAM.
[20] D. Lokshtanov, D. Marx, and S. Saurabh. Lower bounds based on the exponential time hypothesis. *Bulletin of the EATCS*, 105:41–72, 2011.
[21] D. Lokshtanov, S. Saurabh, and S. Sikdar. Simpler parameterized algorithm for OCT. In *Combinatorial algorithms*, volume 5874 of *Lecture Notes in Comput. Sci.*, pages 380–384. Springer, Berlin, 2009.
[22] R. Niedermeier. *Invitation to fixed-parameter algorithms*, volume 31 of *Oxford Lecture Series in Mathematics and its Applications*. Oxford University Press, Oxford, 2006.
[23] S. Oum. Approximating rank-width and clique-width quickly. *ACM Trans. Algorithms*, 5(1):Art. 10, 20, 2008.
[24] S. Oum. Rank-width is less than or equal to branch-width. *J. Graph Theory*, 57(3):239–244, 2008.
[25] S. Oum and P. Seymour. Approximating clique-width and branch-width. *J. Combin. Theory Ser. B*, 96(4):514–528, 2006.
[26] S. Oum and P. Seymour. Testing branch-width. *J. Combin. Theory Ser. B*, 97(3):385–393, 2007.
[27] N. Robertson and P. Seymour. Graph minors. X. Obstructions to tree-decomposition. *J. Combin. Theory Ser. B*, 52(2):153–190, 1991.
[28] J. A. Telle and A. Proskurowski. Practical algorithms on partial k-trees with an application to domination-like problems. In *Proceedings of the Third Workshop on Algorithms and Data Structures*, pages 610–621. Springer-Verlag, 1993.
[29] J. A. Telle and A. Proskurowski. Algorithms for vertex partitioning problems on partial k-trees. *SIAM J. Discrete Math.*, 10(4):529–550, 1997.
[30] P. van 't Hof and Y. Villanger. Proper interval vertex deletion. *Algorithmica*, 65(4):845–867, 2013.
[31] M. Vatshelle. *New width parameters of graphs*. PhD thesis, University of Bergen, May 2012. `http://hdl.handle.net/1956/6166`.