On the Boolean-Width of a Graph: Structure and Applications*

Isolde Adler[1], Binh-Minh Bui-Xuan[1], Yuri Rabinovich[2], Gabriel Renault[1], Jan Arne Telle[1], and Martin Vatshelle[1]

[1] Department of Informatics, University of Bergen, Norway
[2] Department of Computer Science, Haifa University, Israel

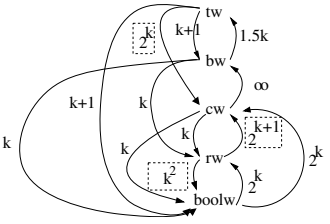
Abstract. Boolean-width is a recently introduced graph invariant. Similar to tree-width, it measures the structural complexity of graphs. Given any graph G and a decomposition of G of boolean-width k, we give algorithms solving a large class of vertex subset and vertex partitioning problems in time $O^*(2^{O(k^2)})$. We relate the boolean-width of a graph to its branch-width and to the boolean-width of its incidence graph. For this we use a constructive proof method that also allows much simpler proofs of similar results on rank-width in [S. Oum. Rank-width is less than or equal to branch-width. *Journal of Graph Theory* 57(3):239–244, 2008]. For an n-vertex random graph, with a uniform edge distribution, we show that almost surely its boolean-width is $\Theta(\log^2 n)$ – setting boolean-width apart from other graph invariants – and it is easy to find a decomposition witnessing this. Combining our results gives algorithms that on input a random graph on n vertices will solve a large class of vertex subset and vertex partitioning problems in quasi-polynomial time $O^*(2^{O(\log^4 n)})$.

1 Introduction

Width parameters of graphs, like tree-width, branch-width, clique-width and rank-width, have many applications in the field of graph algorithms and especially in Fixed Parameter Tractable (FPT) algorithmics, see *e.g.* Downey and Fellows [7], Flum and Grohe [8], and Hliněný et al. [11]. When comparing width-parameters, we should consider the values of the parameters on various graph classes, the runtime of algorithms for finding the corresponding optimal decomposition, the classes of problems that can be solved by dynamic programming along such a decomposition, and the runtime of these algorithms. Recently, Bui-Xuan et al. [2] introduced a new width parameter of graphs called boolean-width. While rank-width is based on the number of GF(2)-sums ($1 + 1 = 0$) of rows of adjacency matrices, boolean-width is based on the number of Boolean sums ($1 + 1 = 1$) of these rows. Although is it open whether computing boolean-width is FPT, the number of Boolean sums of rows for a matrix is easy to compute in FPT time by an incremental approach, and surprisingly is the same for the matrix and its transpose.

* Supported by the Norwegian Research Council, projects PARALGO and Graph Searching.

D.M. Thilikos (Ed.): WG 2010, LNCS 6410, pp. 159–170, 2010.



	tree-width	branch-width	clique-width	rank-width	boolean-width
MDS	$O^*(2^{1.58tw})$[22]	$O^*(2^{2bw})$ [6]	$O^*(2^{4cw})$ [17]	$O^*(2^{0.76rw^2})$ [3,9]	$O^*(2^{3boolw})$ [2]

Fig. 1. Upper bounds tying parameters tw=tree-width, bw=branch-width, cw=clique-width, rw=rank-width and $boolw$=boolean-width, and runtimes achievable for Minimum Dominating Set using various parameters. In the upper part of the figure, an arrow from P to Q labelled $f(k)$ means that any class of graphs having parameter P at most k will have parameter Q at most $f(k)$, and ∞ means that no such upper bound can be shown. Except for the labels in boxes the bounds are known to be tight, meaning that there is a class of graphs for which the bound is $\Omega(f(k))$. For the two boxes containing labels 2^k and 2^{k+1}, a $\Omega(2^{k/2})$ bound is known [4]. For the box containing label k^2 a $\Omega(k)$ bound is known [2]. The arrows $bw \to boolw$ and $tw \to boolw$ are proven in Section 4 of this paper.

This paper gives new algorithmic applications of boolean-width, and new structural properties of graphs of bounded boolean-width. It is well-known that for any class of graphs their tree-width is bounded by a constant if and only if their branch-width is bounded by a constant: we say the two parameters are equivalent. Likewise, clique-width, rank-width, and boolean-width are equivalent. For any graph class we have only three possibilities: either all five parameters are bounded (*e.g.* for trees) or none of them are bounded (*e.g.* for grids) or only clique-width, rank-width and boolean-width are bounded (*e.g.* for cliques). Capturing known results and new insights from Section 4, we show in Figure 1 information allowing for a finer comparison. Let us say that parameter P is polylog on a graph class C if the value of P for any graph G in C is polylogarithmic in the size of G. Then if P is polylog on C any algorithm with runtime[1] $O^*(2^{poly(P)})$ single exponential in parameter P runs in quasi-polynomial time on input a graph in C. From Figure 1 we see that if any of tree-width, branch-width, clique-width or rank-width is polylog on a class of graphs then so is boolean-width, while in Section 5 we show that the random graphs give an example of a class where boolean-width is polylog but none of the other parameters are. A finer comparison can be made by looking at the bounds between the parameters in combination with the runtimes achievable for a particular problem, as done for Minimum Dominating Set (MDS) in Figure 1. In this way, for MDS, and in fact all problems addressed in Section 3, boolean-width compares well to the other parameters. The paper is organized as follows.

[1] We use O^* notation that hides polynomial factors.

In Section 2 we define branch-width, rank-width, and boolean-width in a common framework. In Section 3 we depict algorithms that given a decomposition tree of boolean-width k of a graph, solve a large class of NP-hard vertex subset and vertex partitioning problems, namely (σ, ρ)-problems and D_q-problems [23], in time $O^*(2^{O(k^2)})$. These are monadic second order logic expressible problems related to domination, independence and homomorphism, including Max or Min Perfect Code, Max or Min Independent Dominating Set, Min k-Dominating Set, Max Induced k-Regular Subgraph, Max Induced k-Bounded Degree Subgraph, H-Coloring, H-Homomorphism, H-Covering, H-Partial Covering. From Courcelle's theorem [5] they belong to FPT when parameterized by either the tree-width, branch-width, clique-width, rank-width or boolean-width of the graph, when an appropriate decomposition is given. Although the runtime given in Courcelle's theorem contains a highly exponential factor (tower of powers), the problems behave very well for tree-width and branch-width: given a decomposition tree of tree-width tw, they can be solved in $O^*(2^{O(tw)})$ time [23]. In particular, (σ, ρ)-problems can be solved in $O^*((d(\sigma) + d(\rho) + 2)^{tw})$ time [22] for some problem specific constants $d(\sigma)$ and $d(\rho)$ (see Section 3). This is not the same situation for clique-width, where until now the best runtime contains a $O^*(2^{2^{poly(cw)}})$ double exponential factor [10]. Having small boolean-width is witnessed by a decomposition of the graph into cuts with few different unions of neighborhoods across the cut. This makes the decomposition natural to guide dynamic programming algorithms to solve problems, like Max Independent Set, where vertex sets having the same neighborhoods can be treated as equivalent [2]. Surprisingly, in this paper we extend such an observation to the much larger class of vertex subset and vertex partitioning problems. Several new techniques are introduced in order to achieve this and the runtime of these algorithms is $O^*(2^{O(boolw^2)})$, which then can also be interpreted as $O^*(2^{O(cw^2)})$ and $O^*(2^{O(rw^4)})$ by using the relationships in Figure 1, improving the $O^*(2^{2^{poly(cw)}})$ runtime in [10].

In Section 4 we relate boolean-width to branch-width. We prove for every graph G with $\mathbf{bw}(G) \neq 0$ that $\mathbf{boolw}(G) \leq \mathbf{bw}(G)$. For the proof we develop a general method of constructive manipulations of the decompositions that gives a good understanding of the connections between the graph parameters. In [20], Oum studies the relation of rank-width and branch-width using deep results from matroid theory. Our framework also allows to address this relation in a simpler and direct way. Independently, Kanté [14] gave a constructive proof showing that the rank-width of a graph is at most 4 times its tree-width plus 2. We show constructively that (except for some trivial cases) rank-width is at most branch-width, and also constructively that rank-width is at most tree-width plus one, simplifying Oum's proof and improving Kanté's construction.

In Section 5 we show for a random graph on n vertices where the edges are drawn with respect to a uniform distribution that almost surely[2] its boolean-width is $\Theta(\log^2 n)$, and it is easy to find a decomposition tree witnessing the upper bound. This contrasts sharply with a series of negative results establishing that almost

[2] We use term "almost surely" to denote events whose asymptotic probability is 1.

surely a random graph on n vertices has tree-width and branch-width [16], clique-width [13] and rank-width [18] all in $\Theta(n)$. The importance of this result is possibly not in the random graphs themselves, but in the indication that boolean-width is quite often much smaller than all the other parameters, and therefore potentially very useful. Our result also implies the following: any problem solvable by dynamic programming in time $O^*(2^{poly(k)})$ *given* a decomposition of boolean-width k, can be solved in quasi-polynomial time on input a random graph (where we do not need a decomposition as part of input). Such problems include Minimum Dominating Set and Maximum Independent Set which can be solved in time $O(n(n+2^{3k}k))$ [2]. Moreover, combining our results from Sections 3 and 5 we get an algorithm that given a random graph on n vertices, solves (σ,ρ)-problems and D_q-problems in quasi-polynomial time $O^*(2^{O(\log^4 n)})$.

2 Framework

We address loopless simple undirected graphs. Let G be a graph with vertex set $V(G)$ and edge set $E(G)$. For a vertex $v \in V(G)$ let $N(v)$ be the set of all neighbours of v in G. We extend this to subsets $X \subseteq V(G)$ by letting $N(X) := \bigcup_{v\in X} N(v)$. For a tree T we denote the set of leaves by $L(T)$. A tree is *subcubic* if every vertex has degree either 1 or 3.

Let A be a finite set. For a subset $X \subseteq A$ let $\overline{X} := A \setminus X$. Let $f\colon 2^A \to \mathbb{R}$ be a *symmetric* set function, i.e. f satisfies $f(X) = f(\overline{X})$ for all $X \subseteq A$. A *decomposition tree* of f (on A) is a pair (T,δ), where T is a subcubic tree and $\delta\colon L(T) \to A$ is a bijection. Each edge $e \in E(T)$ yields a partition P_e of A, induced by the leaf labels of the two trees we get by removing e from T: if T_1 and T_2 denote the two components of $T-e$, then $P_e := \big(\delta(L(T_1)\cap L(T)), \delta(L(T_2)\cap L(T))\big)$. We extend the domain of f to edges e of T by letting $f(e) := f(X)$ for $P_e = (X,\overline{X})$. This is well-defined because f is symmetric. The *f-width* of a decomposition tree (T,δ) is $f\text{-w}(T,\delta) := \max\{f(e) \mid e \in E(T)\}$. The *width* of f is $\mathbf{width}(f) := \min\{f\text{-w}(T,\delta) \mid (T,\delta)$ decomposition tree of $f\}$. If $|A| \le 1$, then f has no decomposition tree and we let $\mathbf{width}(f) := f(A)$.

We now define branch-width of a graph G. For any subset $X \subseteq E(G)$ let

$$\partial(X) := \{v \in V(G) \mid v \text{ incident to both an edge from } X \text{ and from } \overline{X}\}$$

denote the *border* of X. We define $\mathbf{cut\text{-}bw}_G := \mathbf{cut\text{-}bw}\colon 2^{E(G)} \to \mathbb{N}$ as $\mathbf{cut\text{-}bw}(X) := |\partial(X)|$. Clearly, **cut-bw** is symmetric. The *branch-width* of G is defined as $\mathbf{bw}(G) := \mathbf{width}(\mathbf{cut\text{-}bw})$.

For subsets $X, Y \subseteq V(G)$ let $M_{(X,Y)}$ denote the $X \times Y$-submatrix of the adjacency matrix of G. Let Δ denote the symmetric difference of sets: $A \Delta B = (A \setminus B) \cup (B \setminus A)$. We define $\mathbf{cut\text{-}rk}_G := \mathbf{cut\text{-}rk}\colon 2^{V(G)} \to \mathbb{N}$ as

$$\mathbf{cut\text{-}rk}(X) := \log_2 \left| \{B \subseteq \overline{X} \mid \exists A \subseteq X,\ B = \bigtriangleup_{v\in A} N(v) \cap \overline{X}\} \right| = \mathrm{rk}\big(M_{(X,\overline{X})}\big),$$

where $\mathrm{rk}\big(M_{(X,\overline{X})}\big)$ denotes the GF(2)-rank of $M_{(X,\overline{X})}$. Then the *rank-width* of G is $\mathbf{rw}(G) := \mathbf{width}(\mathbf{cut\text{-}rk})$.

For boolean-width we define $\mathbf{cut\text{-}bool}_G := \mathbf{cut\text{-}bool}\colon 2^{V(G)} \to \mathbb{R}$ as

$$\mathbf{cut\text{-}bool}(X) := \log_2 \left|\{B \subseteq \overline{X} \mid \exists A \subseteq X \text{ with } B = N(A) \cap \overline{X}\}\right|.$$

Surprisingly, the function **cut-bool** is symmetric [15, Theorem 1.2.3]. The *boolean-width* of a graph G is $\mathbf{boolw}(G) := \mathbf{width}(\mathbf{cut\text{-}bool})$. Let us give an alternative view on boolean-width. Let $R(M_{(X,Y)})$ denote the set of all vectors spanned by the rows of $M_{(X,Y)}$ by taking Boolean sums, i.e. $1 + 1 = 1$. It is easy to see that $\mathbf{cut\text{-}bool}(X) = \log_2 \left|R(M_{(X,\overline{X})})\right|$.

3 Vertex Subset and Vertex Partitioning Problems

Given a graph G together with a decomposition tree of **cut-bool** of width $boolw$, we depict algorithms with runtime $O^*(2^{O(boolw^2)})$ solving a large class of problems, the so-called (σ, ρ) vertex subset and D_q vertex partitioning problems as defined in [23].

Definition 1. Let σ and ρ be finite or co-finite subsets of natural numbers. A subset X of vertices of a graph G is a *sigma-rho set*, or simply (σ, ρ)*-set*, of G if

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

The *vertex subset problems* consist of finding the size of a minimum or maximum (σ,ρ)-set in G. Several NP-hard problems are expressible in this framework, *e.g.*, Max Independent Set($\{0\}$, $\mathbb{N}$), Min Dominating Set($\mathbb{N}$, $\mathbb{N} \setminus \{0\}$), Max Strong Stable Set($\{0\}$, $\{0, 1\}$), Max or Min Perfect Code($\{0\}$, $\{1\}$). Also if we let $M_k = \{0, 1, 2, \ldots k\}$ then Min k-Dominating Set($\mathbb{N}$, $\mathbb{N} \setminus M_k$), Max Induced k-Regular Subgraph($\{k\}$, $\mathbb{N}$) (see [23] for further details and a more complete list). This framework is extendible to problems asking for a partition of $V(G)$ into q classes, with each class satisfying a certain (σ, ρ)-property:

Definition 2. A *degree constraint* matrix D_q is a q by q matrix with entries being finite or co-finite subsets of natural numbers. A D_q*-partition* in a graph G is a partition $\{V_1, V_2, ..., V_q\}$ of $V(G)$ such that for $1 \leq i, j \leq q$ we have $\forall v \in V_i : |N(v) \cap V_j| \in D_q[i, j]$.

The *vertex partitioning problems* for which we give algorithms in this paper consist of deciding if G has a D_q partition, the so-called $\exists D_q$ problem. NP-hard problems fitting into this framework include *e.g.* for any fixed graph H the problems known as H-Coloring or H-Homomorphism (with q-Coloring being K_q-Coloring), H-Covering, H-Partial Covering, and in general the question of deciding if a graph has a partition into q (σ, ρ)-sets [23].

We focus on algorithms for the vertex subset problems. Let a graph G and a decomposition tree (T, δ) of **cut-bool** be given as input. Our algorithm will

follow a bottom-up dynamic programming approach: subdivide an arbitrary edge of T to obtain a root r, and denote by T_r the resulting rooted tree. With each node w of T_r we associate a table data structure Tab_w, that will store optimal solutions to subproblems related to V_w, the set of vertices of G mapped to the leaves of the subtree of T_r rooted at w. Each index of the table will be associated with a certain class of equivalent subproblems that we need to define depending on the problem on which we are focusing.

Let $d(\mathbb{N}) = 0$ and let $d(\emptyset) = 0$. For every finite or co-finite set $\mu \subseteq \mathbb{N}$, let $d(\mu) = 1 + min\{max_{x\in\mathbb{N}} x : x \in \mu, max_{x\in\mathbb{N}} x : x \notin \mu\}$. We denote by $d(\sigma, \rho)$, or simply by d when it appears clearly in the context that σ and ρ are involved, the value $d = d(\sigma, \rho) = max\{d(\sigma), d(\rho)\}$. Note that when checking if a subset A of vertices is a (σ, ρ)-set, as in Definition 1, it suffices to count the number of neighbors up to d that a vertex has in A. This is the key to getting fast algorithms and motivates the following equivalence relation.

Definition 3 (d-neighbor equivalence). Let G be a graph and $A \subseteq V(G)$. Two vertex subsets $X \subseteq A$ and $X' \subseteq A$ are *d-neighbor equivalent* w.r.t. A, denoted by $X \equiv_A^d X'$, if

$$\forall v \in \overline{A},\ \ (|N(v) \cap X| = |N(v) \cap X'|) \vee (|N(v) \cap X| \geq d \wedge |N(v) \cap X'| \geq d)\,.$$

We now depict the entries of the table data structure Tab_w. Roughly, we aim at solving the vertex subset problems using one d-neighbor equivalence class per entry in Tab_w. For this, we first define a canonical representative for every d-neighbor equivalence class.

Lemma 1. *Let G be a graph and $A \subseteq V(G)$. Then, for every $X \subseteq A$, there is $R \subseteq A$ such that $R \equiv_A^d X$ and $|R| \leq d \cdot \mathbf{cut\text{-}bool}(A)$. Moreover, the number of equivalence classes of $\equiv_A^d$ is at most $2^{d\cdot \mathbf{cut\text{-}bool}(A)^2}$.*

We now define the canonical representative $can_{V_w}^d(X)$ of every subset $X \subseteq V_w$, and the canonical representative $can_{\overline{V_w}}^d(Y)$ of every subset $Y \subseteq \overline{V_w}$. For simplicity we define this for V_w only, but the definition can be used for $\overline{V_w}$ as well, since everything we say about $X \subseteq V_w$, $can_{V_w}^d(X)$ and $\equiv_{V_w}^d$ will hold also for $can_{\overline{V_w}}^d(Y)$, $Y \subseteq \overline{V_w}$ and $\equiv_{\overline{V_w}}^d$. Canonical representatives are to be used for indexing the table Tab_w at node w of the tree T_r. Three properties will be required. Firstly, if $X \equiv_{V_w}^d X'$, then we must have $can_{V_w}^d(X) = can_{V_w}^d(X')$. Secondly, given (X, Y), we should have a fast routine that outputs a pointer to the entry $Tab_w[can_{V_w}^d(X)][can_{\overline{V_w}}^d(Y)]$. Thirdly, we should have a list whose elements can be used as entries of the table, *i.e.* a list containing all canonical representatives w.r.t. $\equiv_{V_w}^d$. The following definition trivially fulfills the first requirement.

Definition 4. We assume that a total ordering of the vertices of $V(G)$ is given. For every $X \subseteq V_w$, the canonical representative $can_{V_w}^d(X)$ is defined as the lexicographically smallest set $R \subseteq V_w$ such that: $|R|$ is minimized and $R \equiv_{V_w}^d X$.

Definition 5. Let G be a graph, $A \subseteq V(G)$, and $\mu \subseteq \mathbb{N}$. For $X \subseteq V(G)$, we say that X *μ-dominates* A if $\forall v \in A : |N(v) \cap X| \in \mu$. For $X \subseteq A$, $Y \subseteq \overline{A}$,

we say that (X, Y) σ, ρ-*dominates* A if $(X \cup Y)$ σ-dominates X and $(X \cup Y)$ ρ-dominates $A \setminus X$.

Definition 6. Let *opt* stand for either function *max* or function *min*, depending on whether we are looking for a maximum or minimum (σ, ρ)-set, respectively. For every node w of T_r, for $X \subseteq V_w$ and $Y \subseteq \overline{V_w}$, let $R_X = can^d_{V_w}(X)$ and $R_Y = can^d_{\overline{V_w}}(Y)$. We define the contents of $Tab_w[R_X][R_Y]$ as:

$$Tab_w[R_X][R_Y] \stackrel{\text{def}}{=} \begin{cases} opt_{S \subseteq V_w}\{|S| : S \equiv^d_{V_w} X \text{ and } (S, Y) \ \sigma, \rho\text{-dominates } V_w\}, \\ -\infty \text{ if no such set } S \text{ exists and } opt = max, \\ +\infty \text{ if no such set } S \text{ exists and } opt = min. \end{cases}$$

Lemma 2. *For any node w of T_r with $k =$* **cut-bool**(V_w)*, we can compute a list containing all canonical representatives w.r.t. $\equiv^d_{V_w}$ in time $O(m + d \cdot k \cdot 2^{2d \cdot k^2 + k})$. For any subset $X \subseteq V_w$, a pointer to $can^d_{V_w}(X)$ can be found in time $O(|X| \cdot 2^k)$.*

Note that at the root r of T_r the value of $Tab_r[X][\emptyset]$ (for all $X \subseteq V(G)$) would be exactly equal to the size of a maximum, resp. minimum, (σ, ρ)-set of G (*cf.* $\equiv^d_{V_r}$ has only one equivalence class). For initialization, the value of every entry of Tab_w will be set to $+\infty$ or $-\infty$ depending on whether we are solving a minimization or maximization problem, respectively. For a leaf l of T_r, we perform a brute-force update: let $A = \{l\}$ and $B = \overline{A}$, for every canonical representative R w.r.t. $\equiv^d_B$, we set:

- If $|N(l) \cap R| \in \sigma$ then $Tab_l[A][R] = 1$.
- If $|N(l) \cap R| \in \rho$ then $Tab_l[\emptyset][R] = 0$.

For a node w of T_r with children a and b, the algorithm proceeds as follows. For every canonical representative $R_{\overline{w}}$ w.r.t. $\equiv^d_{\overline{V_w}}$, for every canonical representative R_a w.r.t. $\equiv^d_{V_a}$, and for every canonical representative R_b w.r.t. $\equiv^d_{V_b}$, do:

- Compute $R_w = can^d_{V_w}(R_a \cup R_b)$, $R_{\overline{a}} = can^d_{\overline{V_a}}(R_b \cup R_{\overline{w}})$ and $R_{\overline{b}} = can^d_{\overline{V_b}}(R_a \cup R_{\overline{w}})$
- Update $Tab_w[R_w][R_{\overline{w}}] = opt(Tab_w[R_w][R_{\overline{w}}], Tab_a[R_a][R_{\overline{a}}] + Tab_b[R_b][R_{\overline{b}}])$.

Lemma 3. *The table at node w is updated correctly, i.e. for any canonical representatives R_w and $R_{\overline{w}}$ w.r.t. $\equiv^d_{V_w}$ and $\equiv^d_{\overline{V_w}}$, if $Tab_w[R_w][R_{\overline{w}}]$ is not $\pm\infty$ then*

$$Tab_w[R_w][R_{\overline{w}}] = opt_{S \subseteq V_w}\{|S| : S \equiv^d_{V_w} R_w \wedge (S, R_{\overline{w}}) \ \sigma, \rho\text{-dominates } V_w\}.$$

If the value of the table is $\pm\infty$ then there is no such above set S.

Theorem 1. *For every n-vertex, m-edge graph G given along with a decomposition tree (T, δ) for* **cut-bool***, any (σ, ρ)-vertex subset problem on G with $d = d(\sigma, \rho)$ can be solved in time*
$O(n(m + d \cdot \textbf{cut-bool-w}(T, \delta) 2^{3d \cdot \textbf{cut-bool-w}(T,\delta)^2 + \textbf{cut-bool-w}(T,\delta)}))$.

Proof. Correctness follows directly from what has been said in this section. For complexity analysis, for every node w of T_r, we basically call the first computation of Lemma 2 once, then loop through every triplet $R_{\overline{w}}$, R_a, R_b of equivalence classes, call the second computation of Lemma 2 three times, and perform the table update. □

The algorithms for vertex partitioning problems are similar but require some graph-theoretic observations and several technical details. For space reasons this has all been omitted.

Theorem 2. *For every n-vertex, m-edge graph G given along with a decomposition tree (T, δ) of* **cut-bool***, any D_q-problem on G, with $d = \max_{i,j} d(D_q[i, j])$, can be solved in time*
$O(n(m + qd \cdot \mathbf{cut\text{-}bool\text{-}w}(T, \delta)2^{3qd \cdot \mathbf{cut\text{-}bool\text{-}w}(T,\delta)^2 + \mathbf{cut\text{-}bool\text{-}w}(T,\delta)}))$.

4 Boolean-Width is Less than or Equal to Branch-Width

We relate boolean-width to branch-width, and show the following

Theorem 3. *Any graph G satisfies* $\mathbf{boolw}(G) \leq \mathbf{bw}(G)$ *(unless $E(G) \neq \emptyset$ and no two edges of G are adjacent).*

In order to clarify how the decomposition trees relate to each other, we divide our result into two steps, addressing the intermediary notion of an incidence graph (see Lemmata 4 and 5). However, we will also show how to easily derive from our method a direct proof without incidence graphs. Our framework not only applies for boolean-width, but also captures other settings including rank-width. The *incidence graph* $I(G)$ of a graph G is the graph with vertex set $V(G) \dot\cup E(G)$, where x and y are adjacent in $I(G)$ if one of x, y is a vertex of G, the other is an edge of G and x and y are incident in G.

Lemma 4. *For any graph G,* $\mathbf{boolw}(I(G)) \leq \mathbf{bw}(G)$ *and* $\mathbf{rw}(I(G)) \leq \mathbf{bw}(G)$*, unless $E(G) \neq \emptyset$ and no two edges of G are adjacent. In this case,* $\mathbf{bw}(G) = 0$ *and* $\mathbf{rw}(I(G)) = \mathbf{boolw}(I(G)) = 1$.

The proof is omitted, but let us sketch the idea. Starting with a decomposition tree (T, δ) of $\mathbf{cut\text{-}bw}_G$ of width k, we modify the decomposition tree in two steps. In the first step, we replace every leaf ℓ of T by a subcubic tree with three leaves, and we label one of the three leaves with the edge $\delta(\ell)$ and we label the other two leaves with the two vertices incident with $\delta(\ell)$. In a second step, for each $v \in V(G)$ we choose one leaf with label v, we keep this leaf, and delete all other leaves that are labelled by v. In this way we obtain a decomposition tree of $\mathbf{cut\text{-}bool}_{I(G)}$ (and of $\mathbf{cut\text{-}rk}_{I(G)}$) of boolean-width and rank-width both at most k.

Lemma 5. *For any graph G,*

$$\max\{\mathbf{boolw}(G), \mathbf{rw}(G)\} \leq \min\{\mathbf{boolw}(I(G)), \mathbf{rw}(I(G))\}.$$

Theorem 3 now follows immediately from Lemmata 4 and 5, as well as the fact that rank-width is at most branch-width (except for the trivial cases). It is also easy to give a direct proof using the proof idea of Lemma 4. The only difference is in the first modification step. Instead of taking a subcubic tree with three leaves, we take the subcubic tree with two leaves (since we do not need to assign leaves to graph edges). Note that there is no bound in the converse direction: the class of all complete graphs has unbounded branch-width and the boolean-width is at most 1. Nevertheless, moving to incidence graphs we prove a weak converse.

Lemma 6. *For any graph G,* $\mathbf{bw}(G) \leq 2 \cdot \min\{\mathbf{boolw}(I(G)), \mathbf{rw}(I(G))\}$.

Corollary 1. *Any graph G satisfies* $\mathbf{boolw}(G) \leq \mathbf{bw}(G) \leq 2 \cdot \mathbf{boolw}(I(G))$ *(unless $E(G) \neq \emptyset$ and no two edges of G are adjacent).*

Corollary 2. *For any graph G,*

1. $\mathbf{boolw}(I(G)) \leq \mathbf{bw}(I(G)) \leq 2 \cdot \mathbf{boolw}(I(G))$ *and*
2. $\mathbf{boolw}(I(G)) \leq \mathbf{rw}(I(G)) + 1 \leq 2 \cdot \mathbf{boolw}(I(G)) + 1$.

Proof. Note that $\mathbf{bw}(G) = \mathbf{bw}(I(G))$, unless $E(G) \neq \emptyset$ and no two edges of G are adjacent. In this case, $\mathbf{bw}(G) = 0$ and $\mathbf{bw}(I(G)) = 1$. Then, the first statement follows from Lemmata 4 and 6. The second statement follows from the first by using a theorem from [20] stating that $\mathbf{rw}(I(G)) \in \{\mathbf{bw}(G), \mathbf{bw}(G) - 1\}$. □

5 Random Graphs

Let G_p be a random graph on n vertices where each edge is chosen randomly and independently with probability p (independent of n). There has been a series of negative results [13,16,18] establishing that almost surely G_p has rank-width, tree-width, branch-width and clique-width $\Theta(n)$. In contrast we show in this section the following.

Theorem 4. *Almost surely,* $\mathbf{boolw}(G_p) = \Theta\left(\frac{\ln^2 n}{p}\right)$.

We start with the upper bound and first prove the following lemma.

Lemma 7. *Let G_p be a graph as above, and let $k_p = \lfloor \frac{2 \ln n}{p} \rfloor$. Then, almost surely, for* all *subsets of vertices $S \subset V(G)$ with $|S| = k_p$ it holds that $|N(S) \backslash S| \geq |\overline{S}| - k_p$.*

Proof. In what follows, we write simply G and k. Fix a particular S with $|S| = k$. For every $v \in \overline{S}$, let X_v be 1 if $v \notin N(S)$, and 0 otherwise. Clearly, $X_v = 1$ with probability $(1-p)^k$, and $\sum_{v \in \overline{S}} X_v = |\overline{S} \setminus N(S)|$. Observe that $E[\sum_{v \notin S} X_v] = (1-p)^k(n-k) < (1-p)^k n$. Call this expectation μ. By Chernoff's Bound (see *e.g.* [19], p.68),

$$\Pr\left[\sum_{v \in \overline{S}} X_v \geq k\right] < \left(\frac{e\mu}{k}\right)^k < \left((1-p)^k n\right)^k = \left((1-p)^{2 \ln n/p} n\right)^k < n^{-k},$$

the last inequality due to the fact that for $p \in (0,1)$, $(1-p)^{\frac{1}{p}} \leq e^{-1}$.

Applying the union bound, we conclude that the probability that there exists S of size k such that $|N(S) \setminus S| < |\overline{S}| - k$ is at most $\binom{n}{k} \cdot n^{-k} < (k!)^{-1} = o(1)$ and the statement follows. □

Corollary 3. *For $G = G_p$ and $k = k_p$ as before, for* all *cuts $\{A, \overline{A}\}$ in G it holds almost surely that* **cut-bool**$(A) = O\left(\frac{\ln^2 n}{p}\right)$.

Proof. The number of distinct sets $N(S) \cap \overline{A}$ contributed by the sets $S \subseteq A$ with $|S| \leq k$ is at most $\sum_{i=0}^{k} \binom{n}{i}$. By the previous lemma, for all sets $S \subseteq A$ with $|S| \geq k$, it holds almost surely that $|N(S) \cap \overline{A}| \geq |\overline{A}| - k$. Therefore, almost surely, the sets $S \subseteq A$ with $|S| \geq k$, also contribute at most $\sum_{i=0}^{k} \binom{n}{i}$ distinct sets $N(S) \cap \overline{A}$. Thus, almost surely there are at most $2\sum_{i=0}^{k} \binom{n}{i}$ distinct sets $N(S) \cap \overline{A}$ altogether. Taking the logarithm allows to conclude. □

The upper bound of Theorem 4 now follows easily: for *any* decomposition tree of **cut-bool**, all the cuts it defines will almost surely have boolean-width at most $O\left(\frac{\ln^2 n}{p}\right)$. Next, we move to the lower bound of Theorem 4. For simplicity of exposition, we restrict the discussion to the case $p = 0.5$. The lower bound for that case follows from:

Lemma 8. *Let $\{A, \overline{A}\}$ be a cut where $|A| = |\overline{A}| = m$, and the edges are chosen independently at random with probability* 0.5*. Then,* $\Pr[$ **cut-bool**$(A) = \Omega(\log^2 m)\ ] \geq 1 - 2^{-\Omega(m^{1.3})}$*. More concretely, the probability that among the neighborhoods of the subsets of A of size $k = 0.25 \cdot \log_2 m$, there are less than $2^{c \log^2 m}$ different ones (for a suitable constant c), is at most $2^{-\Omega(m^{1.3})}$.*

To prove this lemma we need some notation and preliminary results first. Let the (random) set $S_i \subseteq \overline{A}$, $i = 1, 2, \ldots, m$ be the neighborhood of the vertex $i \in A$, and let $S_I = \cup_{i \in I} S_i$. We shall only be interested in the I's of size k as above. Call such I *bad* if $m - |S_I| < m^{0.5}$. Call a set I of size k *thick* if there are at least $m^{0.9}$ indices $i \in \{1, 2, \ldots, m\} - I$ such that $S_i \subseteq S_I$. Lemma 8 can be proved using below Corollary 4.

Claim 1. $\Pr\left[\frac{\textit{the number of bad } I\textit{'s}}{\binom{m}{k}} \geq 0.5\right] < e^{-\Omega(m^{1.74})}$.

Claim 2. *For a fixed set I of size k, the probability that I is thick conditioned on its being good (that is, not bad), is at most $e^{-\Omega(m^{1.3})}$.*

Corollary 4. $\Pr\left[\textit{ the number of thick } I\textit{'s } > 0.5 \cdot \binom{m}{k}\right] < e^{-\Omega(m^{1.3})}$.

Proof of Theorem 4: The upper bound has already been proved. For the lower bound we restrict for simplicity of exposition to the case $p = 0.5$. Consider a $(\frac{1}{3}, \frac{2}{3})$-balanced cut in G, that is a cut $(X, \overline{X})$ with $\frac{n}{3} \leq |X| \leq \frac{2n}{3}$. Due to the monotonicity of the **cut-bool** with respect to taking induced subcuts, Lemma 8 applies in this case with $m = n/3$. Therefore, the probability that **cut-bool** of this cut is $\Omega(\log^2 n)$ is $1 - e^{-\Omega(n^{1.3})}$. Since there at most 2^n cuts in G, we conclude

that with probability $1 - e^{-\Omega(n^{1.3})}$ *all* balanced cuts have such **cut-bool**. Since any decomposition tree of G must contain a $(\frac{1}{3}, \frac{2}{3})$-balanced cut, the statement follows. □

6 Further Research

In this paper we have seen that for random graphs boolean-width is the right parameter to consider: any decomposition tree will have boolean-width polylogarithmic in n. This also hints at the existence of large classes of graphs where boolean-width is polylogarithmic in the value of the other parameters, and raises the question of identifying these. One such class of graphs is defined by the so-called Hsu-grids [2], where boolean-width is $\Theta(\log n)$ and rank-width, branch-width, tree-width and clique-width are $\Theta(\sqrt{n})$. In contrast, we know that the boolean-width of regular graphs is $\Theta(n)$ [21], thus such an above mentioned class should exclude regular graphs.

We believe that boolean-width should be useful for *practical* applications. We have initiated research to find fast and good heuristics computing decompositions of low boolean-width [12], similar to what is done for treewidth in the TreewidthLIB project [1].

A big open question is to decide if the boolean-width of a graph can be computed in FPT time. The relationship between rank-width and boolean-width is still not completely clear. Could it be that the boolean-width of any graph is linear in its rank-width? Currently the best bound is $boolw(G) \leq \frac{1}{4}rw(G)^2 + \frac{5}{4}rw(G) + \log rw(G)$ [2].

The runtime of the algorithms given here for (σ, ρ)-problems and D_q-problems have the square of the boolean-width as a factor in the exponent. For problems where $d = 1$ we can in fact improve this to a factor linear in the exponent [2], but that requires a special focus on these cases. In fact, we believe that also for the other problems (with any constant value of d) we could get runtimes with an exponential factor linear in boolean-width. We must then improve the bound in Lemma 1, by showing that the number of d-neighborhood equivalence classes is no more than the number of 1-neighborhood equivalence classes raised to some function of d. This question can be formulated as a purely algebraic one as follows: First generalize the concept of Boolean sums ($1+1=1$) to d-Boolean sums ($i + j = \min(i + j, d)$). For a Boolean matrix A let $R_d(A)$ be the set of vectors over $\{0, 1, ..., d\}$ that arise from all possible d-Boolean sums of rows of A. Is there a function f such that $|R_d(A)| \leq |R_1(A)|^{f(d)\log\log|R_1(A)|}$?

References

1. Bodlaender, H., Koster, A.: Treewidth Computations I Upper Bounds. Technical Report UU-CS-2008-032, Department of Information and Computing Sciences, Utrecht University (2008)
2. Bui-Xuan, B.-M., Telle, J.A., Vatshelle, M.: Boolean-width of graphs. In: 4th International Workshop on Parameterized and Exact Computation (IWPEC 2009). LNCS, vol. 5917, pp. 61–74. Springer, Heidelberg (2009)

3. Bui-Xuan, B.-M., Telle, J.A., Vatshelle, M.: H-join decomposable graphs and algorithms with runtime single exponential in rankwidth. Discrete Applied Mathematics 158(7), 809–819 (2010)
4. Corneil, D., Rotics, U.: On the relationship between clique-width and treewidth. SIAM Journal on Computing 34(4), 825–847 (2005)
5. Courcelle, B.: Graph rewriting: An algebraic and logic approach. In: Handbook of Theoretical Computer Science, Volume B: Formal Models and Sematics (B), pp. 193–242 (1990)
6. Dorn, F.: Dynamic programming and fast matrix multiplication. In: Azar, Y., Erlebach, T. (eds.) ESA 2006. LNCS, vol. 4168, pp. 280–291. Springer, Heidelberg (2006)
7. Downey, R., Fellows, M.: Parameterized Complexity. Springer, Heidelberg (1999)
8. Flum, J., Grohe, M.: Parameterized Complexity Theory. Springer, Heidelberg (2006)
9. Ganian, R., Hliněný, P.: On Parse Trees and Myhill-Nerode-type Tools for handling Graphs of Bounded Rank-width. Discrete Applied Mathematics 158(7), 851–867 (2010)
10. Gerber, M., Kobler, D.: Algorithms for vertex-partitioning problems on graphs with fixed clique-width. Theoretical Computer Science 299(1-3), 719–734 (2003)
11. Hliněný, P., Oum, S., Seese, D., Gottlob, G.: Width parameters beyond tree-width and their applications. The Computer Journal 51(3), 326–362 (2008)
12. Hvidevold, E.: Implementation of heuristics for computing boolean-width, Master thesis, University of Bergen (September 2010) (to appear)
13. Johansson, Ö.: Clique-decomposition, NLC-decomposition and modular decomposition – Relatiohships and results for random graphs. Congressus Numerantium 132, 39–60 (1998)
14. Kanté, M.: Vertex-minor reductions can simulate edge contractions. Discrete Applied Mathematics 155(17), 2328–2340 (2007)
15. Kim, K.H.: Boolean matrix theory and applications. Marcel Dekker, New York (1982)
16. Kloks, T., Bodlaender, H.: Only few graphs have bounded treewidth. Technical Report UU-CS-92-35, Department of Information and Computing Sciences, Utrecht University (1992)
17. Kobler, D., Rotics, U.: Edge dominating set and colorings on graphs with fixed clique-width. Discrete Applied Mathematics 126(2-3), 197–221 (2003); Abstract at SODA 2001
18. Lee, C., Lee, J., Oum, S.: Rank-width of Random Graphs, `http://arxiv.org/pdf/1001.0461`
19. Motwani, R., Raghavan, P.: Randomized Algorithms. Cambridge University Press, Cambridge (1995)
20. Oum, S.: Rank-width is less than or equal to branch-width. Journal of Graph Theory 57(3), 239–244 (2008)
21. Rabinovich, Y., Telle, J.A.: On the boolean-width of a graph: structure and applications, `http://arxiv.org/pdf/0908.2765`
22. Rooij, J., Bodlaender, H., Rossmanith, P.: Dynamic programming on tree decompositions using generalised fast subset convolution. In: Fiat, A., Sanders, P. (eds.) ESA 2009. LNCS, vol. 5757, pp. 566–577. Springer, Heidelberg (2009)
23. Telle, J.A., Proskurowski, A.: Algorithms for vertex partitioning problems on partial k-trees. SIAM Journal on Discrete Mathematics 10(4), 529–550 (1997)