Parameterized algorithms for thinness via the cluster module number[⋆]

Flavia Bonomo[1,2][0000−0002−9872−7528], Eric Brandwein[1][0009−0003−2559−7173][⋆⋆], and Ignasi Sau[3][0000−0002−8981−9287]

[1] Universidad de Buenos Aires, Facultad de Ciencias Exactas y Naturales, Departamento de Computación, Buenos Aires, Argentina.
{fbonomo,ebrandwein}@dc.uba.ar

[2] CONICET-Universidad de Buenos Aires. Instituto de Investigación en Ciencias de la Computación (ICC), Buenos Aires, Argentina.

[3] LIRMM, Université de Montpellier, CNRS, Montpellier, France.
ignasi.sau@lirmm.fr

Abstract. In 2007, Mannino et al. defined k-thin graphs as a generalization of interval graphs, and defined the thinness of a graph to be the minimum k such that the graph is k-thin. When given a k-thin representation of a graph, several NP-complete problems can be solved in XP time parameterized by k. Thus, the problem of computing the thinness of a graph, as well as the corresponding representation, has various algorithmic applications. In this work we define a new graph parameter that we call the *cluster module number* of a graph, which generalizes twin-cover and neighborhood diversity, and can be computed in linear time. We then present a linear kernel for the problem of calculating the thinness on graphs with bounded cluster module number. As a corollary, this results in a linear kernel for THINNESS when the input graph has bounded neighborhood diversity, and exponential kernels when the input graph has bounded twin-cover or vertex cover. On the negative side, we observe that THINNESS parameterized by treewidth, pathwidth, bandwidth, (linear) mim-width, clique-width, modular-width, or the thinness itself, has no polynomial kernel assuming $\mathsf{NP} \not\subseteq \mathsf{coNP/poly}$.

Keywords: Thinness · Vertex cover · Cluster module number · Parameterized complexity · Graph parameters · Polynomial kernels.

1 Introduction

A graph G is *k-thin* if there exists a strict total order $\prec$ on $V(G)$ and a partition S of $V(G)$ into k classes such that, for each $u, v, w \in V(G)$ with $u \prec v \prec w$, if u and v belong to the same class and $(u, w) \in E(G)$, then $(v, w) \in E(G)$. The three

[⋆] Partially supported by ANPCyT (PICT-2021-I-A-00755), CONICET (PIP 11220200100084CO), UBACyT (20020220300079BA and 20020190100126BA), IRP SINFIN, and French project ELIT (ANR-20-CE48-0008-01).

[⋆⋆] PhD Student at Universidad de Buenos Aires.

vertices u, v, w satisfying those conditions are said to form a *consistent triple*. An order and a partition satisfying those properties are said to be *consistent* with G. We call the tuple $(\prec, S)$ a *consistent solution* for G. The minimum k such that G is k-thin is called the *thinness* of G, and denoted by $\mathsf{thin}(G)$. A consistent solution for G that uses $\mathsf{thin}(G)$ classes is said to be *optimal.*

The notion of thinness was introduced by Mannino, Oriolo, Ricci and Chandran [24] as a generalization of interval graphs, which are exactly the 1-thin graphs [26, 27]. There exist graphs of arbitrarily large thinness, and a relationship diagram of thinness and other graph width parameters can be found in [7]. We combine in Figure 1 the figure in [7] with figures in [19] and [1], and further include the parameter defined in this work.

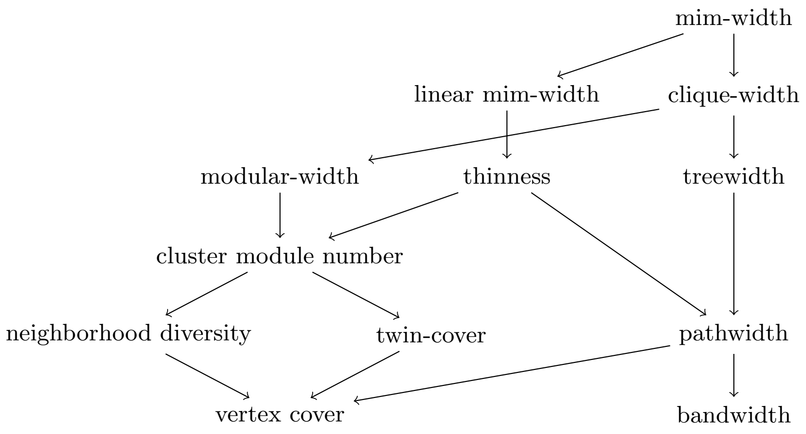

Fig. 1. The relationships between the different width parameters that we consider in this work, and some classical ones. Most of the them were known and appear in diagrams of [1, 7, 19]. The remaining ones are proved in Theorem 2, Lemmas 5 and 6, and the last paragraph of Section 4. To explain the incomparability between thinness and modular-width, observe that cographs have modular-width 2 and unbounded thinness [4], while paths have thinness 1 and unbounded modular-width. Cographs have bounded linear mim-width, but inspired by the ideas in [22], we can also build graph families of bounded modular-width and unbounded linear mim-width. We omit the proof due to space limitations.

Regarding algorithmic applications of thinness, in the seminal paper [24], Maximum Independent Set is solved in XP time parameterized by k, given a k-thin representation. Later, in [5], Capacitated Coloring (for each color, there is an upper bound on the number of vertices that can get that color) is solved in XP time parameterized by k plus the number of colors, and in [10], Chaplick, Töpfer, Voborník and Zeman solve List Coloring in XP time parameterized by k plus the total number of colors. In both cases, a k-thin representation should be given. A framework of problems that can be solved in XP time parameterized by k and some problem parameters, given a k-thin representation, is presented in [4]. These problems generalize the three ones mentioned above, and other classical graph problems like CLIQUE, and can be described shortly as optimization versions of list matrix partition problems [18] with the possibility of adding some cardinality constraints. One of the parameters involved in the complexity is the matrix size, which in the case of representing coloring problems as matrix partition problems equals the number of colors. Indeed, it was recently proved that CHROMATIC NUMBER is NP-complete for 2-thin graphs [6].

Given the algorithmic applications and the need of consistent solutions of bounded size for them, the problem of calculating or approximating thinness becomes relevant. At the moment, nothing is known about the computational complexity of approximating thinness. In [4], a polynomial-time algorithm for finding a smallest vertex partition that is consistent with a given vertex ordering is provided. However, in [28], it is announced that the THINNESS decision problem is NP-complete. This prompts the investigation of algorithms for THINNESS on restricted classes of inputs. THINNESS was solved in polynomial time for cographs in [4] and for trees in [9]. Some classes of graphs with bounded thinness, where a bounded-size consistent solution can be obtained in polynomial time, are identified in [7].

The field of *parameterized complexity* seeks to find efficient algorithms for problems when the input is accompanied by a number that bounds a *parameter* of the input. Algorithms that take polynomial time when that parameter is fixed are of special interest. Formally, a *parameterized problem* is a language $Q \subseteq \Sigma^* \times N$, and such a problem is *fixed-parameter tractable* (FPT) if there is an algorithm that decides membership of an instance (x, k) in time $f(k)|x|^{\mathcal{O}(1)}$ for some computable function f. A *kernelization* (or *kernel*) for Q is a polynomial-time algorithm that transforms each input (x, k) into an *equivalent* instance (x', k') such that $|x'|, k' \leq g(k)$ for some computable function g, which is the *size* of the kernel. Finding a kernel for a problem is equivalent to proving that it is FPT [16, Lemma 2]. Kernels of polynomial size are of particular interest due to their practical applications.

The existing results suggest that computing the thinness of a graph can be quite hard from an algorithmic point of view. Our main contribution is to identify a new graph parameter that allows to present positive algorithmic results for computing thinness: the *cluster module number* of a graph, which we define below. It generalizes the graph parameters neighborhood diversity, twin-cover, and vertex cover, and is less general than modular-width.

This work presents linear kernels for the THINNESS problem when parameterized by cluster module number and neighborhood diversity, and exponential kernels when parameterized by twin-cover and vertex cover.

The paper is structured as follows. In Section 2 we state some necessary definitions. In Section 3 we prove some lemmas about the thinness of a graph after removing vertices that belong to the same module. In Section 4 we define the cluster module number of a graph, and show a linear algorithm to compute it. In

Section 5 we use the results from Section 3 to compute kernels for the Thinness problem parameterized by the cluster module number, neighborhood diversity, twin-cover, and vertex cover. In Section 6 we observe that polynomial kernels are unlikely to exist for the Thinness problem parameterized by some common graph parameters, including treewidth, modular-width, and the thinness itself, and in Section 7 we present some possible future avenues of research.

2 Preliminaries

All graphs in this work are finite, undirected, and have no loops or multiple edges. Let G be a graph. We denote by $V(G)$ and $E(G)$ its vertex and edge set. We denote by $N(v)$ and $N[v]$, respectively, the neighborhood and closed neighborhood of a vertex $v \in V(G)$. The complement of G is denoted $\overline{G}$. Let $X \subseteq V(G)$. We denote by $G[X]$ the subgraph of G induced by X, and by $G \setminus W$ the graph $G[V(G) \setminus W]$. If $W = v$, we simply write $G \setminus v$.

A subset X of $V(G)$ is *complete* (or a *clique*) if every pair of vertices in X are adjacent, and *independent* if no pair of vertices in X are adjacent.

A *module* of a graph G is a subset X of its vertices such that every vertex not in X is either adjacent to all vertices in X, or to none of them. We can *contract X into a vertex*, obtaining the graph $G|_X$, by removing all vertices in X from G and adding a new vertex x adjacent to all neighbors of a vertex in X.

A module is *proper* if it is not $V(G)$. A proper module X is *maximal* if there is no strict superset of X that is a proper module. The maximal modules of a graph G form a partitive family [11], and thus a decomposition tree can be defined over a graph G such that each node is a maximal module that does not overlap with other maximal modules. The *modular decomposition tree* of G is a tree T defined recursively as follows:

- The root of the tree is the entire graph G.
- If G contains only one vertex, then the root is called a *Leaf* node.
- If G is disconnected, then the children of the root are the modular decomposition trees of the connected components of G, and the root is called a *Parallel* node.
- If the complement of G is disconnected, then the children of the root are the modular decomposition trees of the connected components of $\overline{G}$, and the root is a *Series* node.
- Otherwise, the root is a *Prime* node, and its children are the modular decomposition trees of the maximal modules of G.

The *modular-width* $\mathsf{mw}(G)$ of G [13] is the maximum number of children of a prime node in that modular decomposition tree, or 2, if no prime node exists.

3 Reducing modules

We begin by showing that if a sufficiently large group of vertices have the same neighborhood, some of them can be removed from a graph without affecting its thinness. We state here a result of [8] on the thinness of a graph after contracting a complete module.

Lemma 1 ([8, Theorem 35]). *Let H be a module of a graph G, and $G|_H$ be the graph obtained by contracting H into a vertex. If H is complete, then* $\mathsf{thin}(G) = \mathsf{thin}(G|_H)$*.*

A similar lemma holds for modules that are independent sets.

Lemma 2. *Let G be a graph. Let $\{u, v, w\} \subseteq V(G)$. If $N(u) = N(v) = N(w)$, then* $\mathsf{thin}(G) = \mathsf{thin}(G \setminus w)$*.*

Proof. Let $G' = G \setminus w$. As G' is an induced subgraph of G, $\mathsf{thin}(G) \geq \mathsf{thin}(G')$. Now, let $(\prec', S')$ be a consistent solution with $\mathsf{thin}(G')$ classes of the vertices of G'. Without loss of generality, let $u \prec' v$. Let $u \in S'_i$ for S'_i a class in S'. We construct the partition S of the vertices of G, starting from S' and adding vertex w to class S'_i, and construct the order $\prec$ from $\prec'$ by adding w right after u. Clearly, S has $\mathsf{thin}(G')$ classes, as S has the same number of classes as S'. Let us see that $(\prec, S)$ is a consistent solution for the vertices of G. To see this, we will see that every triple of ordered vertices in $\prec$ is consistent. It is enough to consider triples involving w, since the order and partition restricted to G' are consistent. Let $x, y \in V(G) \setminus \{u, w\}$ if they exist. We distinguish several cases:

– $x \prec u \prec w$: As (x, u, v) is a consistent triple in $(\prec', S')$, and $(u, v) \notin E(G')$, if x belongs to the same class as u, then x is not adjacent to v. As $N(w) = N(v)$, x is not adjacent to w either, so these triples are always consistent.
– $u \prec w \prec x$: If $(u, x) \in E(G)$, also $(w, x) \in E(G)$, as u and w share neighborhoods, so these triples are always consistent.
– $w \prec x \prec y$, $x \prec w \prec y$, $x \prec y \prec w$: As the order of w relative to the vertices in $V(G) \setminus \{u, w\}$ is the same as the order of u, both vertices belong to the same class, and they have the same neighbors in $V(G) \setminus \{u, w\}$, replacing w for u will not change the consistency of these triples. As $(\prec', S')$ is a consistent solution, all the triples including u must be consistent, and so are the original ones, that use w instead of u.

This proves that $(\prec, S)$ is a consistent solution with $\mathsf{thin}(G')$ classes for the vertices in $V(G)$, and thus $\mathsf{thin}(G) \leq \mathsf{thin}(G')$. Combining this result with $\mathsf{thin}(G) \geq \mathsf{thin}(G')$ we have that $\mathsf{thin}(G) = \mathsf{thin}(G')$. □

Applying Lemma 2 exhaustively leads to the following.

Lemma 3. *Let an independent set H of size at least two be a module of a graph G. Define G' to be the graph obtained by removing all but two vertices of H from G. Then* $\mathsf{thin}(G) = \mathsf{thin}(G')$*.*

We can combine Lemma 1 and Lemma 3 to see that, if a module H is a disjoint union of complete graphs, almost all vertices of H can be removed without altering the thinness of the graph. Thus, if the vertices of a graph can be partitioned into a small number of modules that are disjoint unions of complete graphs, the number of vertices remaining after reducing the graph will also be small. We define a parameter in the next section to measure the minimum size of such a partition.

4 Cluster module number

We begin this section by defining a cluster module partition of a graph.

Definition 1 (Cluster module). *Let G be a graph. A set $H \subseteq V(G)$ is a* cluster module *if it is a module and also an induced disjoint union of cliques in G.*

Definition 2 (Cluster module partition). *Let G be a graph. A* cluster module partition *is a partition of the vertices of G into cluster modules. The* cluster module number $\mathsf{cm}(G)$ *of G is the minimum size of a cluster module partition of G.*

The cluster module number of a graph, and a corresponding partition, can be quickly obtained for any graph by first obtaining its modular decomposition.

Theorem 1. *An optimal cluster module partition of a graph G can be computed in linear time.*

Proof. First, we compute the modular decomposition tree T of G, which can be done in linear time [15, 25, 29]. Then, for each node $G_t \in T$ (which is an induced subgraph of G), we compute an optimal cluster module partition S_t of the vertices of G_t depending on the type of the node G_t. Let $G_{t_1}, \ldots, G_{t_h}$ be the children of G_t, if there are any.

- **Leaf**: The only class in S_t is formed by the only vertex in G_t. This vertex is trivially a cluster module, and the partition is unique, thus optimal.
- **Parallel or Series**: We define $S_t = \{S_t^1\} \cup S'_t$, where:

$$S_t^1 = \bigcup \{S_{t_i}^1 \mid i \in \mathbb{N}, 1 \le i \le h, G_{t_i} \text{ is complete}\}$$
$$S'_t = \bigcup \{S_{t_i} \mid i \in \mathbb{N}, 1 \le i \le h, G_{t_i} \text{ is not complete}\}.$$

  In other words, we take the partitions of the child nodes t_i for which G_{t_i} is a complete graph, and join them into one class to build the first class of S_t. Then, we add the classes of all the other partitions as they are.
  Let $F = \{G_{t_1}, \ldots, G_{t_h}\}$. As shown by Gallai [20], the only modules present in G_t are the possible unions of elements of F, and the submodules of elements of F. That is to say, G_t has no modules that contain some but not all vertices of some G_{t_i}, and some but not all vertices of some G_{t_j}, with $i \neq j$.
  Let us see first that this is a valid cluster module partition. By the previous remark, S_t^1 and every element of S'_t are modules. If G_t is a parallel node, S_t^1 is a union of complete graphs, meaning, a cluster. If G_t is a series node, S_t^1 is itself a complete graph, and so it is also a cluster. The elements of S'_t are clusters, as they are part of the cluster module partitions of the children of G_t. Moreover, all vertices of G_t are present in some set of the partition, and thus S_t is a valid cluster module partition.
  Next, let us see that it is optimal. Suppose there is an optimal partition Q_t of the vertices of G_t that uses fewer classes than S_t. As noted before, the

classes of Q_t must each be either a union of elements of F, or submodules of elements of F.

Suppose that some class Q_t^i of Q_t is a union of more than one element of F, and that one of these elements G_{t_j} is not complete. We separate in two cases:

- *G_t is a parallel node*: Every element of F is a connected graph. As G_{t_j} is not complete, it cannot be part of a cluster.
- *G_t is a series node*: Let G_{t_l} be another element of F present in Q_t^i. Every vertex of G_{t_j} is adjacent to every vertex of G_{t_l}. As G_{t_j} is not complete, there are two nonadjacent vertices v and w in $V(G_{t_j})$ that are each adjacent to a vertex u in G_{t_l}. Thus, v, w and u cannot be part of the same cluster.

These two cases show that the previous scenario cannot happen, and so every class of Q_t is either a union of elements of F that are complete graphs, or submodules of elements of F. Moreover, there cannot be more than one class of Q_t that is a union of elements of F, because otherwise we could join those classes into only one, reducing the size of Q_t. Also, there cannot be a proper submodule of an element G_{t_i} of F that is a complete graph as a class in Q_t, as otherwise we could join that submodule with the rest of G_{t_i} to reduce the size of Q_t. Thus, the union of the elements of F that are complete graphs represents a class in Q_t. Finally, as each other class is a submodule of an element of F, the classes that have some vertex of a given graph G_{t_i} must form a partition of the vertices of G_{t_i}, meaning, they should be a cluster module partition of G_{t_i}. Partition Q_t then contains an optimal cluster partition of each G_{t_i}, and necessarily has the same amount of classes as S_t.

– **Prime**: We define S_t to be the union of S_{t_i} for every $1 \leq i \leq h$.

As shown by Gallai [20], the proper modules of G_t are exactly the submodules of the associated graphs of children of G_t. Thus, an optimal cluster module partition of G_t must contain an optimal cluster partition of every child G_{t_i}, and so S_t is optimal.

To achieve linear time on this algorithm, the partitions S_t are not actually computed. Instead, we set a flag on each child t_i of a series or parallel node t such that G_{t_i} belongs to S_t^1. Recognizing these children can be done with a simple bottom-up traversal of the tree that marks each node whose associated graph is complete (i.e., a parallel node whose children are leaves). Finally, a top-down traversal of the tree allows us to propagate to which class each of the leaf nodes belongs. These operations can be performed in time proportional to the amount of nodes in the tree, and thus this algorithm runs in linear time. □

This algorithm shows also that the modular-width generalizes the cluster module number. They are not equivalent, since cographs have modular-width 2 [12] but unbounded cluster module number (for instance, $\mathsf{cm}(\overline{tK_2}) = t$).

5 Parameterizations for computing thinness

We are now ready to present the kernels for THINNESS parameterized by the cluster module number, neighborhood diversity, twin-cover, and vertex cover.

5.1 Cluster module number

Theorem 2. *For every graph G,* $\mathsf{thin}(G) \leq 2\mathsf{cm}(G)$*. Also,* THINNESS *admits a linear kernel when parameterized by the cluster module number of the input graph.*

Proof. We reduce an instance (G, k) of THINNESS PARAMETERIZED BY CLUSTER MODULE PARTITION to an instance (G'', k'') in time $n^{\mathcal{O}(1)}$ such that $k'' \leq k$ and $V(G'') \leq 2k$.

We first compute an optimal cluster module partition P of G in linear time by Theorem 1.

Let $P_1, \dots, P_k$ be the parts of P. For each $P_i \in P$, let c_i be the number of connected components of P_i, and let $P_i^1, \dots, P_i^{c_i}$ be the connected components of P_i, which are cliques. We contract these cliques each into a different vertex, obtaining $G|_{P_i^1} \dots |_{P_i^{c_i}}$. We repeat this procedure on the resulting graph for each P_i to obtain G'. As each clique is itself a module of G, by Lemma 1, $\mathsf{thin}(G) = \mathsf{thin}(G')$.

Let H_i be the set of vertices of G' corresponding to the contracted cliques of P_i. Each H_i is an independent set which is a module of G'. To obtain G'', we remove from G' all vertices except two of each H_i s.t. $|H_i| \geq 2$. Now, by Lemma 3, $\mathsf{thin}(G') = \mathsf{thin}(G'')$.

We now have a graph G'' such that $|V(G'')| \leq 2k$ and $\mathsf{thin}(G'') = \mathsf{thin}(G)$ which serves as our kernel, defining $k'' = k$. Note that all operations we performed on the graph can be done in time $n^{\mathcal{O}(1)}$, and so the reduction is polynomial. This also shows that $\mathsf{thin}(G) \leq 2\mathsf{cm}(G)$, as desired. □

We will see that the cluster module number of a graph generalizes the following graph parameters, and so we will apply Theorem 2 to find kernels for these other parameters.

5.2 Neighborhood diversity

The definition of neighborhood diversity is based on the notion of neighborhood types.

Definition 3 (Neighborhood type [23, Definition 1]). *We say that two vertices v and v' of a graph G have the same* neighborhood type *if and only if $N(v) \setminus \{v'\} = N(v') \setminus \{v\}$.*

In other words, v and v' have the same neighborhood type if and only if they induce a module in G.

Definition 4 (Neighborhood diversity [23, Definition 2]). *A graph G has* neighborhood diversity *at most w if there exists a partition of $V(G)$ into at most w sets such that all the vertices in each set have the same neighborhood type. We will call this partition a* neighborhood partition *of the vertices of G. The neighborhood diversity of G, denoted $\mathsf{nd}(G)$, is the smallest such w.*

The following lemmas show that the neighborhood partition of a graph is also a cluster module partition.

Lemma 4 ([23, Theorem 7.1]). *Let G be a graph with neighborhood diversity k, and let $V_1, \ldots, V_k$ be a neighborhood partition of width k of the vertices of G. Then each V_i induces either a clique or an independent set.*

Lemma 5. *For every graph G, $\mathsf{cm}(G) \leq \mathsf{nd}(G)$. Moreover, a neighborhood partition of the vertices of G is also a cluster module partition of G.*

Proof. Let $V_1, \ldots, V_k$ be a neighborhood partition of width k of the vertices of G. By Lemma 4, each V_i induces either a clique or an independent set. In either case, V_i is a cluster of G. Also, each V_i is a module, since all the vertices in V_i have the same neighborhood type, and so must all have the same neighbors outside V_i. Therefore, $V_1, \ldots, V_k$ is a valid cluster module partition of width k of G, and so $\mathsf{cm}(G) \leq k$. □

On the other hand, the neighborhood diversity of a graph can be arbitrarily larger than the cluster module number, as the graph cK_2 has cluster module number 1 but neighborhood diversity c.

Theorem 3. Thinness *is* FPT *when parameterized by the neighborhood diversity of the input graph. Moreover, a linear kernel can be found in polynomial time.*

Proof. Let G be a graph with neighborhood diversity k. By Lemma 5, $\mathsf{cm}(G) \leq k$. We can thus use the algorithm from Theorem 2 to obtain a kernel of linear size on k in polynomial time. □

5.3 Twin-cover

We follow the same procedure we used for neighborhood diversity. We first give the definition of twin-cover.

Definition 5 (Twin-cover [21, Definition 3.1]). *$X \subseteq V(G)$ is a* twin-cover *of G if for every edge $(a, b) \in E(G)$ either $a \in X$, or $b \in X$, or a and b are* twins, *i.e. $N[a] = N[b]$. The* twin-cover number *of G, $\mathsf{tc}(G)$, is the minimum size of a twin-cover of G.*

The following lemma shows that the cluster module number of a graph generalizes the twin-cover number.

Lemma 6. *For every graph G with twin-cover number k, $\mathsf{cm}(G) \leq 2^k + k$.*

Proof. Let X be a twin-cover of G of size k. We construct a partition P with parts $P_1, \ldots, P_{2^k+k}$ of the vertices of G and prove that it is a cluster module partition.

We first put each vertex $x_i \in X$ in the set P_{2^k+i}. Then, we partition the vertices not in X by the set of vertices in X that they are adjacent to. As there are k vertices in X, there are 2^k possible subsets of X. We assign these sets to $P_1, \ldots, P_{2^k}$.

Sets $P_{2^k+1}, \ldots, P_{2^k+k}$ have only one vertex, and so are cluster modules. It remains to show that sets $P_1, \ldots, P_{2^k}$ are cluster modules.

We first prove that they are modules. By definition, every vertex in X is either adjacent to every vertex in P_i or to none.

We will now show that if $v \in P_i$ and $w \in P_j$ with $1 \leq i < j \leq 2^k$, there v and w are not adjacent. By definition of the first 2^k sets, the neighbors of v and w in X are not the same, as they belong to different sets. This means, in particular, that v and w are not twins, and then, as none of them belongs to the twin-cover X, they cannot be adjacent.

Now, we prove that sets $P_1, \ldots, P_{2^k}$ are clusters. To see that, it is sufficient to show that if $a, b \in P_i$ are adjacent, then $N[a] = N[b]$. As neither a nor b are in X, they are twins, and so $N[a] = N[b]$. □

The twin-cover number, similarly to the neighborhood diversity, can be arbitrarily larger than the cluster module number. The class of complete bipartite graphs $K_{n,n}$ acts as an example here, as those graphs have cluster module number 2 and twin-cover number n.

Theorem 4. THINNESS *is* FPT *when parameterized by the size of a minimum twin-cover of the input graph.*

Proof. Let G be a graph with twin-cover number k. By Lemma 6, $\mathsf{cm}(G) \leq 2^k+k$. We thus use the algorithm from Theorem 2 to obtain a kernel of size $\mathcal{O}(2^k)$ in polynomial time. □

Note that the kernel presented above is not polynomial, as the bound on the cluster module number given a twin-cover is $\mathcal{O}(2^k)$.

5.4 Vertex cover

Definition 6. *Let G be a graph. A subset X of $V(G)$ is a vertex cover of G if for every edge (u, v) of $E(G)$ at least one of $\{u, v\}$ belongs to X.*

Theorem 5. THINNESS *is* FPT *when parameterized by the size of a minimum vertex cover of the input graph.*

Proof. As every vertex cover is a twin-cover, we can use the algorithm described in Theorem 4 to obtain a kernel in polynomial time. □

6 Kernelization lower bounds

We now present some parameterizations of THINNESS for which there are no polynomial kernels under some standard complexity assumptions. For this, we will use the cross-composition framework introduced by Bodlaender, Jansen and Kratsch [3]. In their paper, they define the concept of AND-cross-composition. First, they define the following relation.

In this section, Σ will denote a finite set of symbols, and Σ^* will denote the set of finite strings of symbols in Σ.

Definition 7 (Polynomial equivalence relation [3]). *An equivalence relation $\mathcal{R}$ on Σ^* is called a* polynomial equivalence relation *if the following two conditions hold:*

1. *There is an algorithm that given two strings $x, y \in \Sigma^*$ decides whether x and y belong to the same equivalence class in time polynomial in $|x| + |y|$.*
2. *For any finite set $S \subseteq \Sigma^*$ the equivalence relation $\mathcal{R}$ partitions the elements of S into a number of classes that is polynomially bounded in the size of the largest element of S.*

Definition 8 (AND-cross-composition [3]). *Let $L \subseteq \Sigma^*$ be a language, let $\mathcal{R}$ be a polynomial equivalence relation on Σ^*, and let $\mathcal{Q} \subseteq \Sigma^* \times \mathbb{N}$ be a parameterized problem. An* AND-cross-composition of L into $\mathcal{Q}$ *(with respect to $\mathcal{R}$) is an algorithm that, given t instances $x_1, x_2, \ldots, x_t \in \Sigma^*$ of L belonging to the same equivalence class of $\mathcal{R}$, takes time polynomial in $\sum_{i=1}^{t}|x_i|$ and outputs an instance $(y, k) \in \Sigma^* \times \mathbb{N}$ such that k is polynomially bounded in $\max_i|x_i| + \log t$, and the instance (y, k) belongs to $\mathcal{Q}$ if and only if all instances x_i belong to L.*

The cross-composition framework is a generalization of earlier work by Bodlaender, Downey, Fellows, and Hermelin [2]. A conjecture presented there was proven later by Drucker [17], which allows to state the following theorem.

Theorem 6 ([3]). *Let $L \subseteq \Sigma^*$ be an NP-hard language and let $\mathcal{Q} \subseteq \Sigma^* \times \mathbb{N}$ be a parameterized problem. If L AND-cross-composes into $\mathcal{Q}$, then $\mathcal{Q}$ has no polynomial compression, assuming NP $\not\subseteq$ coNP/poly.*

The main objective of this section is to prove the following theorem.

Theorem 7. *Let p be a graph parameter such that for every graph G, we have $\mathsf{p}(G) \leq |V(G)|$, and such that if H is the disjoint union of two graphs G_1 and G_2, we have $\mathsf{p}(H) \leq \max\{\mathsf{p}(G_1), \mathsf{p}(G_2)\}$. Then* THINNESS *parameterized by p has no polynomial kernel assuming NP $\not\subseteq$ coNP/poly.*

Examples of graph parameters that meet these criteria are treewidth, pathwidth, bandwidth, clique-width, mim-width, linear mim-width, modular-width, and even thinness, as the following folklore lemma shows (see e.g. [4]).

Lemma 7. *For graphs G_1 and G_2, $\mathsf{thin}(G_1 \cup G_2) = \max\{\mathsf{thin}(G_1), \mathsf{thin}(G_2)\}$.*

We now prove Theorem 7. The proof is quite simple but we include it here for completeness.

Proof (of Theorem 7). We define an AND-cross-composition of Thinness into Thinness(p). As Thinness is NP-hard, we can use Theorem 6 to prove that Thinness(p) has no polynomial kernel, unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$.

First, we define the relation $\mathcal{R}$ on instances of Thinness such that two instances (G_1, k_1) and (G_2, k_2) are related if and only if $k_1 = k_2$. This relation fits in Definition 7, as checking if $k_1 = k_2$ takes time polynomial in $|(G_1, k_1)| + |(G_2, k_2)|$, and every subset S of instances of Thinness is partitioned into a number of classes not greater than the maximum number of vertices in the graph of an instance (as $\mathsf{thin}(G) \leq |V(G)|$, we can consider only the instances with $k \leq |V(G)|$).

Let $(G_1, k), \ldots, (G_t, k)$ be t instances of Thinness in the same equivalence class of $\mathcal{R}$. We construct an instance $((G', k), p)$ of Thinness(p) where $G' = \bigcup_{i=1}^{t} G_i$ and $p = \max_i |V(G_i)|$. This can be done in time polynomial in $\sum_{i=1}^{t} |(G_i, k)|$, and p is polynomially bounded by $\max_i |(G_i, k)| + \log t$. Also, p is greater than or equal to $\mathsf{p}(G')$, as we required $\mathsf{p}(G')$ to be not greater than the maximum of $\mathsf{p}(G_i)$ for each G_i. Finally, by Lemma 7, graph G' is k-thin if and only if all graphs G_i are k-thin, so $((G', k), p)$ belongs to Thinness(p) if and only if all instances $(G_1, k), \ldots, (G_t, k)$ belong to Thinness.

We described an AND-cross-composition of Thinness into Thinness(p), and thus Thinness(p) has no polynomial kernel unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$. □

7 Open problems

Possible avenues of research that this work leaves open include:

- Is Thinness FPT (or even XP) when parameterized by the thinness of the input graph? Note that both a positive or a negative answer are compatible with the non-existence of a polynomial kernel proved in Theorem 7. Combining the results of [4] and [28], we can reduce an instance of the Non-Betweenness problem to an instance of Thinness. The thinness of the constructed graph depends in part on the size of the Non-Betweenness instance, and thus leaves open the possibility of a polynomial-time algorithm for Thinness when the thinness of the input graph is bounded by a constant.
- Is Thinness FPT (or even XP) when parameterized by other parameters, like for example the treewidth of the input graph? The total order involved in the definition of thinness makes it unclear to be amenable to a CMSO logic formulation in order to apply Courcelle's theorem [14]. On the other hand, note that Theorem 7 implies that Thinness is unlikely to admit polynomial kernels parameterized by treewidth.
- Does Thinness admit polynomial kernels when parameterized by the vertex cover or the twin-cover number of the input graph?

References

1. Belmonte, R., Hanaka, T., Lampis, M., Ono, H., Otachi, Y.: Independent set reconfiguration parameterized by modular-width. Algorithmica **82**(9), 2586–2605 (2020)
2. Bodlaender, H.L., Downey, R.G., Fellows, M.R., Hermelin, D.: On problems without polynomial kernels. J. Comput. Syst. Sci. **75**(8), 423–434 (2009)
3. Bodlaender, H.L., Jansen, B.M., Kratsch, S.: Kernelization lower bounds by cross-composition. SIAM J. Discrete Math. **28**(1), 277–305 (2014)
4. Bonomo, F., De Estrada, D.: On the thinness and proper thinness of a graph. Discrete Appl. Math. **261**, 78–92 (2019)
5. Bonomo, F., Mattia, S., Oriolo, G.: Bounded coloring of co-comparability graphs and the pickup and delivery tour combination problem. Theor. Comput. Sci. **412**(45), 6261–6268 (2011)
6. Bonomo-Braberman, F., Brandwein, E., S. Oliveira, F., Sampaio, M.S., Sansone, A., Szwarcfiter, J.L.: Thinness and its variations on some graph families and coloring graphs of bounded thinness. RAIRO – OR (2024), to appear
7. Bonomo-Braberman, F., Brettell, N., Munaro, A., Paulusma, D.: Solving problems on generalized convex graphs via mim-width. J. Comput. Syst. Sci. **140**, 103493 (2024)
8. Bonomo-Braberman, F., Gonzalez, C.L., S. Oliveira, F., Sampaio, M.S., Szwarcfiter, J.L.: Thinness of product graphs. Discrete Appl. Math. **312**, 52–71 (2022)
9. Bonomo-Braberman, F., Brandwein, E., Gonzalez, C.L., Sansone, A.: On the thinness of trees. In: Ljubić, I., Barahona, F., Dey, S., Mahjoub, A.R. (eds.) Proc. ISCO 2022. LNCS, vol. 13526, pp. 189–200 (2022)
10. Chaplick, S., Töpfer, M., Voborník, J., Zeman, P.: On H-topological intersection graphs. Algorithmica **83**(11), 3281–3318 (2021)
11. Chein, M., Habib, M., Maurer, M.C.: Partitive hypergraphs. Discrete Math. **37**(1), 35–50 (1981)
12. Corneil, D., Lerchs, H., Stewart Burlingham, L.: Complement reducible graphs. Discrete Appl. Math. **3**(3), 163–174 (1981)
13. Courcelle, B., Makowsky, J.A., Rotics, U.: Linear time solvable optimization problems on graphs of bounded clique-width. Theory Comput. Syst. **33**(2), 125–150 (2000)
14. Courcelle, B.: The Monadic Second-Order Logic of graphs. I. Recognizable sets of finite graphs. Inform. Comput. **85**(1), 12–75 (1990)
15. Cournier, A., Habib, M.: A new linear algorithm for modular decomposition. In: Tison, S. (ed.) Proc. CAAP'94. LNCS, vol. 787, pp. 68–84. Springer (1994)
16. Cygan, M., Fomin, F.V., Kowalik, L., Lokshtanov, D., Marx, D., Pilipczuk, M., Pilipczuk, M., Saurabh, S.: Parameterized Algorithms. Springer, 1st edn. (2015)
17. Drucker, A.: New limits to classical and quantum instance compression. SIAM J. Comput. **44**(5), 1443–1479 (2015)
18. Feder, T., Hell, P., Klein, S., Nogueira, L., Protti, F.: List matrix partitions of chordal graphs. Theor. Comput. Sci. **349**(1), 52–66 (2005)
19. Gajarský, J., Lampis, M., Ordyniak, S.: Parameterized algorithms for modular-width. In: Gutin, G.Z., Szeider, S. (eds.) Proc. IPEC 2013. LNCS, vol. 8246, pp. 163–176. Springer (2013)
20. Gallai, T.: Transitiv orientierbare graphen. Acta Math. Acad. Sci. Hung. **18**, 25–66 (1967)
21. Ganian, R.: Twin-cover: Beyond vertex cover in parameterized algorithmics. In: Marx, D., Rossmanith, P. (eds.) Proc. IPEC 2011. LNCS, vol. 7112, pp. 259–271. Springer (2012)

22. Høgemo, S., Telle, J.A., Vågset, E.R.: Linear mim-width of trees. In: Proc. WG 2019. LNCS, vol. 11789, pp. 218–231. Springer (2019)
23. Lampis, M.: Algorithmic meta-theorems for restrictions of treewidth. Algorithmica **64**, 549–560 (2010)
24. Mannino, C., Oriolo, G., Ricci, F., Chandran, S.: The stable set problem and the thinness of a graph. Oper. Res. Lett. **35**, 1–9 (2007)
25. McConnell, R.M., Spinrad, J.P.: Modular decomposition and transitive orientation. Discrete Math. **201**(1), 189–241 (1999)
26. Olariu, S.: An optimal greedy heuristic to color interval graphs. Inf. Process. Lett. **37**, 21–25 (1991)
27. Ramalingam, G., Pandu Rangan, C.: A unified approach to domination problems on interval graphs. Inf. Process. Lett. **27**, 271–274 (1988)
28. Shitov, Y.: Graph thinness, a lower bound and complexity (2021), viXra:2112.0157
29. Tedder, M., Corneil, D., Habib, M., Paul, C.: Simpler linear-time modular decomposition via recursive factorizing permutations. In: Aceto, L., Damgård, I., Goldberg, L.A., Halldórsson, M.M., Ingólfsdóttir, A., Walukiewicz, I. (eds.) Proc. ICALP 2008. LNCS, vol. 5125, pp. 634–645. Springer (2008)