A Dividing Line for Structural Kernelization of Component Order Connectivity via Distance to Bounded Pathwidth

Jakob Greilhuber
CISPA Helmholtz Center for Information Security, Saarbrücken, Germany
Saarland University, Saarbrücken, Germany

Roohani Sharma
Discrete Mathematics Group, Institute for Basic Science (IBS), Daejeon, South Korea

Abstract

In this work we study a classic generalization of the ubiquitous Vertex Cover (VC) problem, called the Component Order Connectivity (COC) problem. In COC, given an undirected graph G, integers $d \geq 1$ and k, the goal is to determine if there is a set of at most k vertices whose deletion results in a graph where each connected component has at most d vertices. When $d = 1$, this is exactly VC.

This work is inspired by polynomial kernelization results with respect to structural parameters for VC. On one hand, Jansen & Bodlaender [*TOCS* 2013] show that VC admits a polynomial kernel when the parameter is the distance to treewidth-1 graphs, on the other hand Cygan, Lokshtanov, Pilipczuk, Pilipczuk & Saurabh [*TOCS* 2014] showed that VC does not admit a polynomial kernel when the parameter is distance to treewidth-2 graphs.

Greilhuber & Sharma [*IPEC* 2024] showed that, for any $d \geq 2$, d-COC cannot admit a polynomial kernel when the parameter is distance to a forest of pathwidth 2. Here, d-COC is the variant of COC where d is a fixed constant rather than part of the input. We complement this result and show that, analogously to the VC setting, where distance to treewidth-1 graphs versus distance to treewidth-2 graphs is the dividing line between structural parameterizations that admit and respectively do not admit polynomial kernelization, for COC this dividing line lies between distance to pathwidth-1 graphs and distance to pathwidth-2 graphs. The main technical result of this work is that COC admits a polynomial kernel parameterized by distance to pathwidth-1 graphs plus d.

The problem d-COC can also be expressed as an $\mathcal{F}$-MinorDeletion problem for an appropriate graph family $\mathcal{F}$. One of the central questions around $\mathcal{F}$-MinorDeletion is for which families $\mathcal{F}$ and minor-closed graph classes $\mathcal{G}$ the problem admits a polynomial kernel when parameterized by the distance to $\mathcal{G}$. For some families $\mathcal{F}$ complete dichotomies answering this question are known [Bougeret et al., *SIDMA* 2022][Bougeret et al., *STACS* 2026][Bougeret et al., *arXiv* 2026]. But, these results do not capture the 2-COC problem. We show that, when $d \geq 2$, the line of tractability for polynomial kernelization of d-COC parameterized by the distance to $\mathcal{G}$ is different from the tractability line of the $\mathcal{F}$-MinorDeletion problems for which the currently known dichotomies apply. Thus, with our result, d-COC serves as an outlier in the class of $\mathcal{F}$-MinorDeletion problems when it comes to understanding the dichotomies for polynomial kernelization when parameterizing by the distance to some minor-closed graph class.

2012 ACM Subject Classification Theory of computation → Parameterized complexity and exact algorithms

Keywords and phrases Kernelization, Component Order Connectivity, Caterpillars, Pathwidth, Structural Parameterization

Digital Object Identifier 10.4230/LIPIcs.MFCS.2026.8

Related Version *Full Version*: `https://arxiv.org/abs/2603.22240`

Funding *Roohani Sharma*: Supported by the Young Scientist Fellowship (IBS-R029-C1) of the Institute for Basic Science.

© Jakob Greilhuber and Roohani Sharma;
licensed under Creative Commons License CC-BY 4.0
51st International Symposium on Mathematical Foundations of Computer Science (MFCS 2026).
Editors: Michal Koucký and Daniela Petrişan; Article No. 8; pp. 8:1–8:19
Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

1 Introduction

Vertex cover. The Vertex Cover (VC) problem is one of the 21 classical NP-complete problems identified in the seminal work of Karp [36] and serves as a canonical problem in parameterized complexity that can be considered the paradigmatic example for developing and testing algorithmic techniques in the field [14, 22]. In this problem, given an undirected graph G and an integer k, the question is to decide whether there is a set $S \subseteq V(G)$ of size at most k such that each connected component of $G - S$ has size at most one.

A particularly rich line of inquiry for VC has focused on its kernelization complexity when parameterized by the solution size k. Kernelization (see [22] or [14, Chapter 2] for textbook introductions to the topic) is a rigorous paradigm for studying preprocessing for NP-hard problems. The fundamental question here is to design polynomial-time algorithms that reduce an instance of a parameterized problem to an equivalent instance whose size and parameter are bounded by a function of the parameter of the input. The gold standard is when the output size is a polynomial function of the parameter, in which case the problem is said to admit a polynomial kernel.

Kernelization for VC with respect to the solution size parameter k has inspired a rich array of techniques including the Buss rule [8], crown decompositions [12, 20], the (weighted) expansion lemma [39], and the LP-based kernel using the celebrated Nemhauser-Trotter theorem [11, 38, 39, 43], resulting in the current best kernel on at most $2k - c \log k$ vertices for the problem, for some constant c [40]. These results have inspired a wide array of generalizations and extensions to broader problem families.

Component order connectivity. A natural generalization of VC is the Component Order Connectivity (COC) problem: given a graph G and integers $d \geq 1$ and k, decide whether there is a set $S \subseteq V(G)$ of size at most k such that deleting S from G results in a graph where every connected component has size at most d. Such a set S is called a d-coc set of G. When $d = 1$ this is precisely the VC problem. Apart from being a classic generalization of VC, COC has independently appeared as a natural network vulnerability measure [29, 37]. In this work, we are also interested in the problem family obtained by fixing d to be some constant not part of the input. Concretely, for any integer $d \geq 1$ the input of the problem d-COC is a graph G and an integer k, and one needs to decide whether G has a d-coc set of size at most k. For each fixed $d \geq 1$, the problem d-COC is NP-complete [41]. Notice that Vertex Cover is exactly the same problem as 1-COC.

Interestingly, many kernelization techniques developed for VC such as the Buss rule, crown decomposition, (weighted) expansion lemma, and even the Nemhauser-Trotter theorem also extend to d-COC when $d \geq 2$ [2, 9, 10, 19, 22, 39, 44]. This suggests that positive kernelization results for the solution-size parameterization of VC frequently generalize to d-COC parameterized by the solution size.

However, the kernelization landscape dramatically shifts when we consider structural parameters. One of the most important works in this direction for the VC problem is due to Jansen & Bodlaender [34] who demonstrated the existence of a polynomial kernel for VC when parameterizing by the feedback vertex set number (fvs) of the graph, which is a parameter provably smaller or equal to the size of a minimum solution. The parameter fvs can equivalently be thought of as the (vertex deletion) distance to treewidth-1 graphs. This positive result is complemented by the result of Cygan, Lokshtanov, Pilipczuk, Pilipczuk, and Saurabh [15] who showed that VC does not admit a polynomial kernel parameterized by the distance to treewidth-2 graphs.[1]

[1] All the lower bounds mentioned here are conditional under reasonable complexity assumptions.

In stark contrast to VC, the behavior of d-COC under structural parameterizations is more pessimistic. Greilhuber & Sharma [28] show that, when $d \geq 2$, d-COC does not admit a polynomial kernel when parameterizing by fvs, and even more, no polynomial kernel exists even when parameterizing by the distance to forests of pathwidth 2.[2] The kernelization lower bound with respect to fvs is also implied by the result of Jansen & Donkers [18].

1.1 Our Main Result

In this work, we identify a clean structural dichotomy for d-COC: analogous to the distance to treewidth-1 graphs versus distance to treewidth-2 graphs divide for polynomial kernelization for VC, we establish that *distance to pathwidth-1 graphs versus distance to pathwidth-2 graphs* form a boundary for polynomial kernelization of d-COC. Our main result shows that d-COC parameterized by the distance to graphs of pathwidth at most one admits a polynomial kernel.

We denote this problem as d-COC/$\texttt{pw}$-1. The input is a graph G, an integer k, and a set $M \subseteq V(G)$ such that $G - M$ has pathwidth at most one. The task is to decide whether G has a d-$\texttt{coc}$ set of size at most k, and the parameter is $|M|$. The assumption that the modulator M is given as part of the input is without loss of generality in the sense that one can compute a modulator that is only a constant factor larger than a minimum modulator in polynomial time [30].

In our main result we show that even the problem COC/d+$\texttt{pw}$-1 admits a polynomial kernel. Here, the problem COC/d+$\texttt{pw}$-1 is the same as d-COC/$\texttt{pw}$-1, with the change that d is now part of the input and that the parameter is $|M| + d$. See Appendix A for formal definitions of the COC variants we consider in this paper. Giving a kernel for COC/d+$\texttt{pw}$-1 is significantly stronger than just having the result for d-COC/$\texttt{pw}$-1. For example, a running time of the form $O(|M|^d)$ or kernel size of the form $O(d^d \cdot |M|^4)$ is allowed when aiming for a polynomial kernel for d-COC/$\texttt{pw}$-1, but a similar result for COC/d+$\texttt{pw}$-1 would not yield a polynomial kernel.

▶ **Main Theorem 1.** *The problem* COC/d+$\texttt{pw}$-1 *admits a kernel with* $O(d^7|M|^3 + d^6|M|^4)$ *vertices. For each integer* $d \geq 1$ *the problem* d-COC/$\texttt{pw}$-1 *admits a kernel with* $O(|M|^4)$ *vertices.*

Main Theorem 1 affirmatively answers an open question of Greilhuber & Sharma [28], who asked whether d-COC admits a polynomial kernel when the parameter is deletion distance to a linear forest (a forest in which each connected component is a path).

Minor deletion problems. One of the most important problem families in kernelization (and beyond) is the family of $\mathcal{F}$-MinorDeletion problems. Here, $\mathcal{F}$ is a fixed finite family of graphs. Given a graph G and integer k, the goal of $\mathcal{F}$-MinorDeletion is to decide whether there is a set $S \subseteq V(G)$ of size at most k such that $G - S$ is $\mathcal{F}$-minor free, that is, $G - S$ has no graph from $\mathcal{F}$ as a minor. Numerous fundamental problems including Vertex Cover, Feedback Vertex Set, and Planar Vertex Deletion are captured by this expressive problem family (see [4] for a recent and detailed introduction to $\mathcal{F}$-MinorDeletion). The problem d-COC is no exception, and also covered by the framework. Concretely, VC is the same as $\mathcal{F}$-MinorDeletion when $\mathcal{F} = \{P_2\}$, the problem 2-COC is obtained by setting

[2] The result given in [28] is stated for the parameter distance to subdivided comb graphs, which are easily seen to be forests of pathwidth at most two.

$\mathcal{F} = \{P_3\}$, and for $d \geq 3$ setting $\mathcal{F}$ to be the family of all connected graphs on $d+1$ vertices suffices. With respect to the parameter solution-size polynomial kernels are known to exist when $\mathcal{F}$ contains a planar graph [21], and there is an influential line of work dedicated to understanding the kernelization complexity of $\mathcal{F}$-MinorDeletion problems with respect to structural parameterizations [4–7, 16, 18, 35]. The perhaps most important question that has emerged is the following: For which families $\mathcal{F}$ and minor-closed graph classes $\mathcal{G}$ does $\mathcal{F}$-MinorDeletion parameterized by the distance to $\mathcal{G}$ have a polynomial kernel?

This question is fully settled for some problem families $\mathcal{F}$. By a result of Bougeret, Jansen & Sau [6], when $\mathcal{F} = \{P_2\}$, that is, for VC, a polynomial kernel exists if and only if $\mathcal{G}$ has bounded bridgedepth. Here, bridgedepth is a structural parameter that was introduced for this result. Without going into the formal details, we remark that any forest has bridgedepth at most one. When $\mathcal{F} = \{K_3\}$, that is, for Feedback Vertex Set, a polynomial kernel exists if and only if $\mathcal{G}$ has bounded elimination distance to the class of forests [16]. The result of [16] has been vastly generalized to show that, if $\mathcal{F}$ is a finite set of biconnected graphs on at least 3 vertices containing at least one planar graph, then a polynomial kernel exists if and only if $\mathcal{G}$ has bounded elimination distance to the class of $\mathcal{F}$-minor free graphs [4]. In a recent update to the arXiv version of [4], this result has been extended to also capture some non-biconnected families $\mathcal{F}$, with the dichotomy again relying on whether $\mathcal{G}$ has bounded elimination distance to the class of $\mathcal{F}$-minor free graphs [5]. Hence, all these dichotomies are centered around $\mathcal{G}$ either having bounded bridgedepth, or bounded elimination distance to the class of $\mathcal{F}$-minor free graphs.

The problem family d-COC is not captured by any of these results. In fact Bougeret et al. [4, 5] explicitly mention that the kernelization complexity of 2-COC is unclear, and ask whether bridgedepth plays a role for the problem. As mentioned before, when $d \geq 2$, d-COC does not have a polynomial kernel parameterized by the distance to forests [18, 28], so $\mathcal{G}$ having bounded bridgedepth does certainly not suffice. Our Main Theorem 1 exhibits a polynomial kernel when $\mathcal{G}$ is the class of graphs with pathwidth at most one. This class $\mathcal{G}$ does not have bounded elimination distance to being $\mathcal{F}$-minor free. Therefore, when $d \geq 2$, the dichotomy for d-COC must rely on another, as yet unidentified parameter. To the best of our knowledge, d-COC is even the first (non-trivial) $\mathcal{F}$-MinorDeletion problem where it is known that the dichotomy relies neither on $\mathcal{G}$ having bounded bridgedepth, nor on $\mathcal{G}$ having bounded elimination distance to the class of $\mathcal{F}$-minor free graphs.

Technical challenges. On the technical front, let us compare our main result with other polynomial kernels for structural parameterizations of d-COC. Jansen and Pieterse [35] show that, for each d and each constant η, the problem d-COC has a polynomial kernel parameterized by the size of a treedepth-η modulator. There are other publications that yield polynomial kernels for d-COC [3, 4], but in fact the existence of polynomial kernels for the parameters they consider is already implied by the aforementioned result of Jansen and Pieterse. To the best of our knowledge, no further positive results are known. A key feature of the kernel by Jansen and Pieterse is that it suffices to be able to bound the number of connected components of the graph obtained after deleting the modulator to obtain the result. The size of these connected components can then be bounded iteratively in at most η further rounds by exploiting the structure of graphs with treedepth at most η. Each round involves moving a bounded number of vertices (the roots of the elimination forest) into the modulator, and then bounding the number of connected components.

For our main result, the key technical challenge is bounding the size of these connected components. In particular, long paths have pathwidth one but unbounded treedepth, so the iterative approach used in the previous positive result completely fails. Our result is the first in COC kernelization that designs reduction rules that bound the size of connected components of the graph after removing the modulator corresponding to the parameter. We explain this phase and its challenges in more detail in Section 2.

1.2 Further Results

We complement our contribution with a variety of further results about COC kernelization. While these results are interesting in their own right, they are less technically involved than Main Theorem 1 and most of them follow by either lifting or adjusting known results, or by relatively standard techniques.

On d in the parameter. First, we investigate the role the value d plays in more detail. Recall that in Main Theorem 1 the integer d is either a constant, or at least part of the parameter. We show that this is unavoidable. In Theorem 1.1 we exhibit that COC parameterized by the *vertex cover number* of the graph plus the solution size k cannot have a polynomial kernel, unless $\mathsf{NP} \subseteq \mathsf{coNP/poly}$.

▶ **Theorem 1.1.** *The problem* COC/k+$\mathtt{vc}$ *does not admit a polynomial compression, unless* $\mathsf{NP} \subseteq \mathsf{coNP/poly}$.

The proof of this theorem is inspired by a reduction of Giannopoulou et al. [26], and starts from a compression lower bound for Perfect t-Set Matching by Dell and Marx [17]. In contrast to Theorem 1.1, a simple branching algorithm shows that COC parameterized by the vertex cover number is FPT.

▶ **Theorem 1.2.** COC/$\mathtt{vc}$ *is* FPT.

We additionally show in Theorem 1.3 that for the parameter distance to pathwidth one graphs alone COC is even $\mathsf{W[1]}$-hard. Therefore, there is no hope of obtaining a result similar to Main Theorem 1 when d is neither a constant, nor part of the parameter. We prove Theorem 1.3 by combining a reduction of Ganian et al. [24] with a new gadget that makes it work for the problem we consider.

▶ **Theorem 1.3.** COC/$\mathtt{pw}$-1 *is* $\mathsf{W[1]}$*-hard.*

Parameterizing by the distance to bounded degree graphs. Next, we utilize the result of Main Theorem 1 to obtain a polynomial kernel when parameterizing by the distance to the class of graphs with maximum degree at most two (plus d for the problem COC). We combine this with an NP-hardness result for d-COC on planar graphs with maximum degree 3 to obtain a kernelization dichotomy. This result represents a generalization of a result by Majumdar, Raman & Saurabh [42] for Vertex Cover to the more general problem COC. For a graph class $\mathcal{G}$ and integer $d \geq 1$, the problem d-COC/$\mathtt{dist\text{-}to\text{-}}\mathcal{G}$ is d-COC where the input is provided together with a modulator to $\mathcal{G}$, that is a set $M \subseteq V(G)$ such that $G - M \in \mathcal{G}$, and the parameter is $|M|$. The problem COC/d+$\mathtt{dist\text{-}to\text{-}}\mathcal{G}$ is defined the same way, only that d is part of the input and the parameter is $|M| + d$.

▶ **Theorem 1.4.** *When $\mathcal{G}$ is the class of graphs with maximum degree at most* 2 *the problem* COC/d+$\mathtt{dist\text{-}to\text{-}}\mathcal{G}$ *admits a polynomial kernel, and, for each $d \geq 1$,* d-COC/$\mathtt{dist\text{-}to\text{-}}\mathcal{G}$ *admits a polynomial kernel. When $\mathcal{G}$ is the class of planar graphs with maximum degree* 3 *and d any positive integer, the problem* d-COC/$\mathtt{dist\text{-}to\text{-}}\mathcal{G}$ *admits no kernel of any size, unless* $\mathsf{P} = \mathsf{NP}$.

For the positive result of Theorem 1.4, we first show that we can bound the number of connected components of $G - M$, and then we lift one vertex from each cycle of $G - M$ to the modulator to obtain an instance of COC/d+pw-1 (or d-COC/pw-1). Then, we can apply our kernel from Main Theorem 1. This type of lifting trick is often used to obtain polynomial kernels for Vertex Cover [7,23,33,42]. The negative result is obtained by a simple reduction from Vertex Cover on planar graphs with maximum degree at most 3, which is NP-hard [25, Lemma 1].

Bounding the number of connected components. Finally, as part of our kernelization routine for Main Theorem 1 we need to be able to reduce the number of connected components of $G - M$, where M is the modulator given in the input. To do this we perform a careful, but relatively standard lifting of a result for Vertex Cover due to Hols, Kratsch & Pieterse [33, Theorems 1.3 and 3.12]. We obtain a preprocessing routine that is usable for a much more general class of parameters in Theorem 1.5. In fact, by combining our preprocessing routine with known necessary conditions for the existence of polynomial kernels [28,33], we obtain that for any d and any graph class $\mathcal{G}$ that is (1) closed under taking the disjoint union and (2) such that the problem d-COC/dist-to-$\mathcal{G}$ has a polynomial kernel, our algorithm can be used to reduce the number of connected components of $G - M$ to $|M|^{O(1)}$. Therefore, our algorithm is the ultimate preprocessing algorithm for reducing the number of connected components.

The result in Theorem 1.5 relies on $\mathcal{G}$ having bounded minimal d-blocking sets. A d-blocking set of a graph G is a set $X \subseteq V(G)$ such that no minimum d-coc set of G contains X, and a d-blocking set is minimal if no proper subset of it is a d-blocking set [28]. A graph class $\mathcal{G}$ has bounded minimal d-blocking sets if there is some constant b such that for each graph $G \in \mathcal{G}$ any minimal d-blocking set of G has size at most b. The notion of 1-blocking sets (sometimes implicitly) plays a role in many kernelization results for structural parameterizations of Vertex Cover [6,7,23,32–34], and similar blocking set notions are important for $\mathcal{F}$-MinorDeletion problems in general, see e.g. [4]. Similar to Hols et al. [33], for a graph class $\mathcal{G}$ we define the graph class $\mathcal{G} + d$ to consist of the graphs that have a modulator to $\mathcal{G}$ of size exactly d.

▶ **Theorem 1.5.** *Let $d \geq 1$ be an integer, and $\mathcal{G}$ be a graph class such that minimal d-blocking sets of $\mathcal{G}$ have size at most b for some constant b. Furthermore, assume that*
(1) d-COC *can be solved in polynomial-time on graph class $\mathcal{G} + d$,* or
(2) d-COC *can be solved in polynomial-time on graph class $\mathcal{G}$ and $\mathcal{G}$ is hereditary.*

Then, there exists a polynomial-time algorithm that, given an instance (G, k, M) of d-COC/dist-to-$\mathcal{G}$ outputs an equivalent instance (G', k') of d-COC, where
- *G' is a subgraph of G obtained by deleting connected components of $G - M$, and*
- *the number of connected components of $G' - M$ is at most $((d-1) \cdot b + 1) \cdot |M|^b$, and*
- *$k' \leq k$.*

1.3 Outlook

In this paper we provide several new positive kernelization results for COC. In particular, we develop a novel approach for reducing the size of connected components of $G - M$ when M is a modulator to graphs of pathwidth one. The task of reducing the size of connected components is new in the sense that all previously known positive kernelization results for structural parameterizations of d-COC follow directly from bounding the number of connected components.

Obtaining a full dichotomy for d-COC/$\texttt{dist-to-}\mathcal{G}$ similar to the results which are known for other $\mathcal{F}$-MinorDeletion problems is a natural ultimate goal. However, it seems we do not yet have a sufficiently good understanding of the problem to prove such a result. We believe that, just like it was the case for Vertex Cover, it would be fruitful to first collect various positive results before trying to prove a full dichotomy. Nevertheless, we want to present the following conjecture:

▶ **Conjecture 1.6.** *For minor-closed graph class $\mathcal{G}$ and all constants $d \geq 1$ the problem d-COC/$\texttt{dist-to-}\mathcal{G}$ admits a polynomial kernel if and only if $\mathcal{G}$ has bounded minimal d-blocking sets.*

This conjecture seems plausible since, by a result of Greilhuber & Sharma [28], the problem d-COC has no polynomial kernel parameterized by the distance to $\mathcal{G}$ when $\mathcal{G}$ is closed under the disjoint union and does not have bounded minimal d-blocking sets. Moreover, the conjecture is true for the case $d = 1$ since $\mathcal{G}$ has bounded minimal 1-blocking sets if and only if $\mathcal{G}$ has bounded bridgedepth [6]. And, so far, a bound on minimal d-blocking sets of $\mathcal{G}$ was always eventually turned into a polynomial kernel for d-COC.

2 Technical Overview of Main Theorem 1

In this overview we present the main ideas behind the proof of Main Theorem 1. The focus will especially be on how we can reduce the *size* of connected components of $G - M$, as this is the most innovative part of our work.

A graph has pathwidth at most one if and only if it is a caterpillar forest, that is, a forest in which every connected component is a caterpillar [1, Lemma 2.4]. Caterpillars are connected graphs such that, after deleting each vertex with degree one, a (possibly empty) path is obtained. We start by setting some handy terminology. Then, we define a specific packing of connected subgraphs of $G - M$ that exploits a tight Erdős-Pósa property for connected subgraphs of size at least $d + 1$ in trees. Based on this packing, we identify a subcaterpillar with a large spine (the large central path to which the degree one vertices are attached) that can be replaced with an equivalent smaller one. This replacement is based on a new insight which we call the *essence* of a caterpillar, and which allows one to partition caterpillars into equivalence classes such that the caterpillars in the same equivalence class can be replaced with each other. An interesting feature about our proof is showing an explicit bound on the size of small representatives of this equivalence class via relations to abstract algebra. All the above allows one to reduce the length of the spine of a caterpillar of $G - M$. To bound the number of caterpillars of $G - M$, we exploit that the size of minimal d-blocking sets of caterpillars is at most two for all $d \geq 1$. Finally, we apply a known kernel for the parameter solution-size (plus d) to guarantee that the remaining reduced instance is small.

Caterpillars and spines. For every caterpillar C, we assume for convenience that we are given $\mathsf{spine}(C)$, which is a fixed nonempty subpath of C, such that each vertex of C that is not in $\mathsf{spine}(C)$ has degree one. Vertices which are not part of the spine are called pendants, and we use $\mathsf{pendants}(C)$ to denote the set of all pendants of C. For each pendant v of C, we call the unique neighbor of v the parent of v. Moreover, we assume that the spine has a fixed order, and we refer to one end of the spine as the left end, and to the other end as the right end. This order remains intact even when taking connected subgraphs C' of the caterpillar C, where the spine of the resulting caterpillar C' is the spine of C restricted to the vertices of C'. A spine vertex v is left of spine vertex w if v is closer to the left end than

w, and left of a pendant w if v is left of the parent of w. A pendant v is left of vertex w if the parent of v is left of w. We similarly define the notions of a vertex/vertex set/subgraph being left/right of another vertex/vertex set/subgraph.

Now, we proceed to how we reduce the size of connected components of $G - M$. On a very high level, the strategy is to find a large subcaterpillar of $G - M$ which we can safely replace with a smaller caterpillar. Unfortunately, this is quite a difficult challenge: not only can the concrete structure of a subcaterpillar affect solutions in different ways, but the subcaterpillar can also have many connections to the modulator, which need to be taken into account.

Solution-tight packings. For figuring out which subcaterpillars we can replace with others, we utilize specific maximum-size packings of vertex disjoint connected graphs on at least $d + 1$ vertices. These packings are useful because they exhibit a tight Erdős-Pósa property as stated below. For a graph H, we denote the size of a minimum d-coc set of H as $\mathsf{OPT}_d(H)$. The following lemma directly follows from the work of Cockayne et al. [13, Theorem 1].

▶ **Lemma 2.1** (Follows from Cockayne et al. [13, Theorem 1]). *Let T be a tree, and let $\mathcal{P}$ be a maximum packing of pairwise vertex-disjoint connected subgraphs of T with at least $d + 1$ vertices each. Then, $\mathsf{OPT}_d(T) = |\mathcal{P}|$. In particular, any optimal d-coc set of T must select exactly one vertex of each graph of $\mathcal{P}$.*

Our initial interest in such packings was inspired by the use of perfect matchings in the Vertex Cover kernel of Jansen and Bodlaender [34]. Now, we introduce the notion of a solution-tight packing, which builds upon Lemma 2.1. A solution-tight packing is a specific packing of vertex-disjoint connected graphs of size at least $d + 1$ each.

▶ **Definition 2.2** (Solution-tight Packing). *Let C be a caterpillar with $\mathsf{spine}(C) = v_1, \ldots, v_n$, where v_1 is the left end and v_n the right end of the spine. A set $\mathcal{P}$ of pairwise vertex-disjoint connected subgraphs of C on at least $d + 1$ vertices each is a* solution-tight packing *if*

1. *$|\mathcal{P}| = \mathsf{OPT}_d(C)$, and*
2. *for each $p \in \mathsf{pendants}(C)$ if the parent p' of p is part of a graph H of $\mathcal{P}$, then p is also part of H, and*
3. *there exists an integer $i \in [0, n]$ such that each spine vertex v_j with $j \leq i$ is part of some graph of $\mathcal{P}$, and no spine vertex v_j with $j > i$ is part of a graph of $\mathcal{P}$, and*
4. *for each $H \in \mathcal{P}$, the set of size one containing the rightmost spine vertex of H is a d-coc set of H.*

Let F be a caterpillar forest, and $\mathcal{C}$ the set of connected components of F. Then, a packing $\mathcal{P}$ is a solution-tight packing of F if $\mathcal{P} = \bigcup_{C' \in \mathcal{C}} \mathcal{P}_{C'}$, where for each $C' \in \mathcal{C}$ we have that $\mathcal{P}_{C'}$ is a solution-tight packing of C'.

The properties a solution-tight packing must fulfill are sufficient to ensure any caterpillar forest (where the spine of each caterpillar in the forest is fixed with a fixed order) has a unique solution-tight packing. For any caterpillar forest, it is easy to compute its solution-tight packing using a greedy algorithm. A solution-tight packing of $G - M$ is powerful for kernelization, because we know that any optimum solution must select at least one vertex of each graph of the packing, and that there exists a solution for $G - M$ that selects exactly one vertex of each packed graph. The further properties of solution-tight packings give us other useful guarantees, for example, if some graph $H \subseteq V(G - M)$ has a solution-tight packing that contains all of its vertices, then there exists an optimal solution for H that selects the rightmost spine vertex of H. We thus have particularly much knowledge about such graphs H.

Given this intuition, the subcaterpillars we replace are created by merging several graphs of the solution-tight packing. Formally we introduce the notion of α-merged packings, which are created by merging each α consecutive graphs of the solution-tight packing together.

▶ **Definition 2.3** (α-merged Packing). *Let C be a caterpillar, $\mathcal{P} = \{H_1, \dots, H_{|\mathcal{P}|}\}$ be the solution-tight packing of C, where the graphs of $\mathcal{P}$ are ordered by their appearance along the spine such that graph H_i is left of graph H_{i+1} for all $i \in [1, |\mathcal{P}| - 1]$. Let $\alpha \geq 1$ be an integer. Then, the α-merged packing of $\mathcal{P}$ is the packing consisting of $t = \left\lfloor \frac{|\mathcal{P}|}{\alpha} \right\rfloor$ graphs $H'_0, \dots, H'_{t-1}$ where H'_i is the subgraph of C induced by the vertex set $\bigcup_{j \in [1,\alpha]} V(H_{i \cdot \alpha + j})$ for each $i \in [0, t-1]$.*

When F is a caterpillar forest, then the α-merged packing of F is obtained by taking the union of the α-merged packings of each connected component of F.

We also define the related notion of merged graphs. Observe that each graph of the α-merged packing is a merged graph.

▶ **Definition 2.4** (Merged Graphs). *Let F be a caterpillar forest with solution-tight packing $\mathcal{P}$. An induced subgraph C of F is a* merged graph *if $V(C) = \bigcup_{C' \in \mathcal{P}'} V(C')$ for some $\mathcal{P}' \subseteq \mathcal{P}$.*

Subcaterpillars with no connections to the modulator. Now, we want to elaborate on how we can use these packings to replace subcaterpillars of $G - M$. To get an initial intuition, let us first consider a graph C of the α-merged packing of $G - M$, for some suitably large value of α that we will elaborate on later, such that C has no edges to the modulator.[3] Let $\hat{C}$ be the connected component of $G - M$ of which C is a subgraph. Since C has no edges to the modulator we do not need to worry about the modulator having a significant effect on the solution within C. However, depending on the exact structure of C, the caterpillar C can significantly affect the solution in the rest of $\hat{C}$.

Since C lies in $\hat{C}$, we can naturally partition $\hat{C}$ into three parts, one part is the connected component L of $\hat{C} - C$ that is left of C, the second part is C, and the third part R is the connected component of $\hat{C} - C$ that is right of C. Of course, L, R might not exist, but let us assume that they do for this overview. Consider some d-coc set S of G of size at most k. Then, $S_L = S \cap V(L)$ is a d-coc set of L, $S_C = S \cap V(C)$ is a d-coc set of C, and $S_R = S \cap V(R)$ is a d-coc set of R. When we replace C with a different graph C', we want to ensure that there is a sufficiently small solution $S_{C'}$ for C' that combines nicely with S_L and S_R. Here, by replacing C with C' we mean deleting C, adding C', adding an edge from the rightmost spine vertex of L to the leftmost spine vertex of C', and adding an edge from the leftmost spine vertex of R to the rightmost spine vertex of C'.

In $L - S_L$ the rightmost spine vertex of L is in a connected component of size x (set $x = 0$ if the rightmost spine vertex of L is in S_L). Then, these x vertices are not "taken care of" by S_L, it is the job of S_C to make sure they do not end up in connected components that are too large. Therefore, S_C is not only a d-coc set of C, but even a d-coc set of the graph C_x we obtain when adding x additional pendants to the leftmost spine vertex of C. If we consider the graph R, we find a similar phenomenon: the rightmost spine vertex of C is in some connected component of $C - S_C$ that has size y (set $y = 0$ if the rightmost spine vertex

[3] We remark that the actual kernelization does not have a specific reduction rule to handle this case. However, elaborating on this case is useful to gain an intuition for how the actual reduction rule we use proceeds.

of C is selected). Then, it is the job of S_R to make sure that these y vertices do not cause any problems. We can observe that, among all solutions for C_x of size $|S_C|$ we would prefer using the solution that minimizes the respective value of y. If $|S_C| > \mathsf{OPT}_d(C)$, then we are also allowed to choose $S_{C'}$ to be larger than $\mathsf{OPT}_d(C')$, which is an easy task since C' will have a d-coc set of size at most $\mathsf{OPT}_d(C') + 1$ that selects the leftmost and rightmost spine vertex of C'. Therefore, we are mostly interested in how small the value y can get when only considering solutions of size $\mathsf{OPT}_d(C)$ for the graph C_x.

For exactly this purpose, we introduce the notion of an essence of a caterpillar. An essence is a function we can compute in polynomial-time which precisely captures how solutions in C can interact with solutions of the rest of the graph.

▶ **Definition 2.5** (Essence of a Caterpillar). *Let C be a caterpillar such that the solution-tight packing of C contains all vertices of C. Let the spine of C be $v_1, \ldots, v_n$ where v_1 is the left end and v_n the right end of the spine.*

For each integer $x \geq 0$, define C_x to be the caterpillar obtained from C by adding x additional pendants to v_1 while keeping the spine the same. For each integer $x \geq 0$ and for any d-coc set Y of C_x we define $\beta_x(Y)$ to be 0 if $v_n \in Y$, and otherwise define $\beta_x(Y)$ to be the number of vertices of the connected component of $C_x - Y$ that contains v_n. For all integers $x \geq 0$, let $\mathcal{D}_x$ be the set of all minimum d-coc sets of C_x.

The essence of C (relative to d) is the function $\gamma : [0, d+1] \to [0, d+1]$ where

$$\gamma(x) = \begin{cases} \min\{\beta_x(Y) \mid Y \in \mathcal{D}_x\} & \text{if } x \leq d \text{ and } \mathsf{OPT}_d(C_x) = \mathsf{OPT}_d(C), \\ d+1 & \text{otherwise.} \end{cases}$$

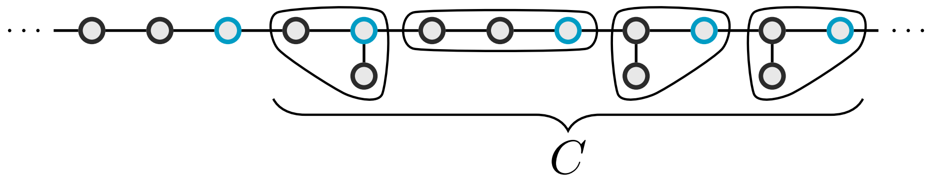
Later, the functional composition of essences will play an important role and is also the reason why we define $d + 1$ to be a fixed point. The idea behind essences is to replace C with a smaller caterpillar C' that has the same essence as C. Figure 1 illustrates such a replacement. The figure also showcases specific selections depending on how many unselected vertices of the caterpillar to the left of C are in the same connected component with the leftmost spine vertex of C (and C'). The key property of essences is that replacement is safe.

▶ **Lemma 2.6.** *Let (G, d, k, M) be an instance of* COC/d+$\mathtt{pw}$-$\mathtt{1}$*, and P be a connected merged subgraph of $G - M$. Let γ be the essence of P, and P' be an arbitrary caterpillar with essence γ. Create G' from G by replacing P with P'. If there is a d-coc set S of G of size at most k, then there is a d-coc set S' of G' of size at most $k' = k - \mathsf{OPT}_d(P) + \mathsf{OPT}_d(P')$.*

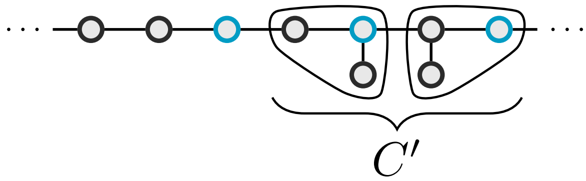
The intuition behind the safety of this replacement is that, recalling our split of $\hat{C}$ into L, C, R, no matter the value x based on the set S_L and the corresponding value y of S_C, we have a solution $S_{C'}$ of appropriate size for C'_x such that the rightmost spine vertex of $C' - S_{C'}$ is in a connected component of size at most y (or selected if $y = 0$). Hence, we find that $(S \setminus S_C) \cup S_{C'}$ is a solution for G' of size at most $|S| - \mathsf{OPT}_d(P) + \mathsf{OPT}_d(P')$.

We have established that replacing a subcaterpillar C that has no connections to the modulator with a different caterpillar C' with the same essence is a safe operation. However, it is unclear how we can quickly find such a caterpillar C' which is smaller than C, such that a reduction rule using such an operation would actually decrease the size of the instance.

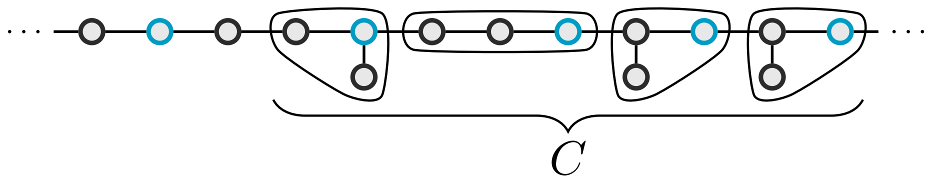
Computing caterpillars with specific essences. Note that for any fixed d the number of possible essences relative to d is bounded by a function of d. Therefore, for the problem d-COC the kernel could hardcode, for each possible essence γ, the smallest caterpillar that has essence γ. However, our result is for the problem COC where d is not a constant. So,



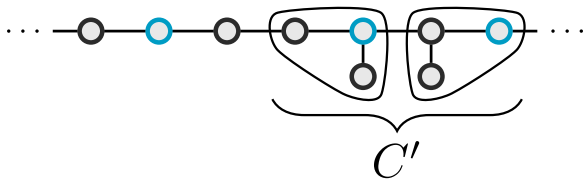
(a) The selection before the replacement when $x = 0$.



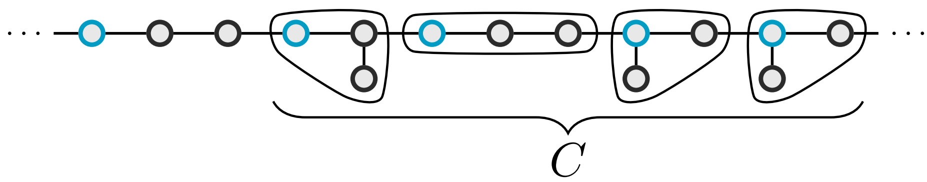
(b) The selection after the replacement when $x = 0$.



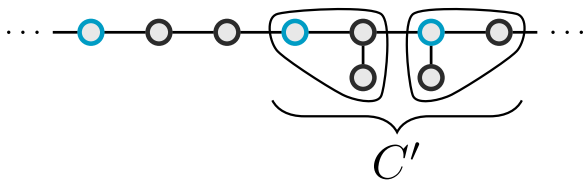
(c) The selection before the replacement when $x = 1$.



(d) The selection after the replacement when $x = 1$.



(e) The selection before the replacement when $x = 2$.



(f) The selection after the replacement when $x = 2$.

Figure 1 The replacement of a subcaterpillar C with a caterpillar C' that has the same essence γ. In the example $d = 2$. The solution-tight packings of C and C' are highlighted. The essence γ fulfills $\gamma(0) = 0$, $\gamma(1) = 0$, $\gamma(2) = 1$, and $\gamma(3) = 3$. For any possible number x of vertices of the caterpillar left of C (or C') which are in a connected component with the leftmost spine vertex of C (or C') we draw the corresponding solution within C and C' that also witnesses the values of the essence.

we must be able to compute small graphs with specific essences on the fly. Let us now explain how we can do that. This part of our work has interesting connections to abstract algebra. Observe that a caterpillar essence γ is non-decreasing in the sense that if $x \leq x'$ then $\gamma(x) \leq \gamma(x')$, and fulfills $\gamma(0) = 0$ and $\gamma(d+1) = d+1$. Keeping these properties in mind, we define the essence monoid[4] as follows.

▶ **Definition 2.7** (The Essence Monoid). *Let d be a positive integer, and $\mathcal{M}_d$ be the set of all non-decreasing functions $\gamma : [0, d+1] \to [0, d+1]$ where $\gamma(0) = 0$, and $\gamma(d+1) = d+1$. Let $\mathsf{id}^d : [0, d+1] \to [0, d+1]$ be the identity function. Then, $(\mathcal{M}_d, \circ, \mathsf{id}^d)$, where $\circ$ is the function composition operation, is the essence monoid (of d).*

It is not difficult to confirm that this structure is indeed a monoid with neutral element id^d. We show that for any d, and any function f of the essence monoid of d, we can efficiently compute at most d^3 simple functions whose function composition is f. We refer to these simple functions as basic functions and define them next.

▶ **Definition 2.8** (Basic Functions). *Let d be a positive integer. For any $i \in [1, d]$ let $\mathsf{dec}_i^d : [0, d+1] \to [0, d+1]$ be the function defined by*

$$\mathsf{dec}_i^d(x) = \begin{cases} x - 1 & \text{if } x = i, \\ x & \text{otherwise.} \end{cases}$$

Let $\mathsf{inc}^d : [0, d+1] \to [0, d+1]$ be defined by

$$\mathsf{inc}^d(x) = \begin{cases} x + 1 & \text{if } 1 \leq x \leq d, \\ x & \text{otherwise.} \end{cases}$$

The set of basic functions (relative to d) is the set $\{\mathsf{id}^d\} \cup \{\mathsf{dec}_i^d \mid i \in [1, d]\} \cup \{\mathsf{inc}^d\}$.

We now state the algorithmic result for computing functions as a composition of basic functions.

▶ **Lemma 2.9.** *There is an algorithm that takes a non-decreasing function $f : [0, d+1] \to [0, d+1]$ with $f(0) = 0$ and $f(d+1) = d+1$ as input and outputs $g_1, \ldots, g_\ell$, where $\ell \leq d^3$, such that $f = g_\ell \circ g_{\ell-1} \circ \cdots \circ g_1$, and the function g_i is a basic function relative to d for all $i \in [1, \ell]$. The running time of the algorithm is polynomial in d.*

Although it seems that we cannot directly employ a result given in the literature in a black-box fashion, the approach we utilize to prove Lemma 2.9 closely follows insights of Higgins [31] and Gomes & Howie [27] given in the abstract algebra literature. This means that our paper marks one of these exciting cases where "old" results in a seemingly unrelated field (semigroup theory) suddenly lead to state-of-the-art algorithmic results. From an algebraic point of view it is an interesting question to characterize which functions we can use for a generator of $\mathcal{M}_d$, and what the least number of functions is that need to be composed to generate any function $f \in \mathcal{M}_d$. Settling this question could improve our kernel size.

We now elaborate further on the links between the essence monoid and caterpillars. It turns out that not only is each caterpillar essence γ in the essence monoid, but also for any function f in the essence monoid there is a caterpillar with essence f. We find the

---

[4] A monoid is an algebraic structure $(M, \oplus, e)$, where M is a set, $\oplus$ an associative binary operator, M is closed under $\oplus$, and $e \in M$ is an element such that $m \oplus e = m$ and $e \oplus m = m$ for all $m \in M$.

fact that all conceivable essences have a caterpillar with that essence quite interesting, as it shows that the interaction of a subcaterpillar with the caterpillars to the left and right of it can be "as complicated as imaginable". To show this, we prove that the operation of composing functions f_1 and f_2 of the monoid is virtually the same as connecting the spines of caterpillars with essence f_1 and f_2. Formally, we prove Lemma 2.10.

▶ **Lemma 2.10.** *Let C_1 be a caterpillar with a solution-tight packing that contains all vertices of C_1. Let the essence of C_1 be γ_1, and let the spine of C_1 be $v_1, \ldots, v_n$. Similarly, let C_2 be a caterpillar with a solution-tight packing that contains all vertices of C_2, the essence of C_2 be γ_2, and its spine be $w_1, \ldots, w_m$.*

Create caterpillar C by taking the disjoint union of C_1 and C_2 and adding an edge from v_n to w_1. The spine of C is $v_1, \ldots, v_n, w_1 \ldots, w_m$. Then, the essence γ of C fulfills $\gamma = \gamma_2 \circ \gamma_1$.

Using this fact, we can relatively easily compute a caterpillar with a specific essence by using Lemma 2.9, assuming that we have caterpillars that have the basic functions as their essence available. So, we show that we can easily compute such caterpillars next.

▶ **Lemma 2.11.** *There is an algorithm that takes any basic function f (relative to some $d \geq 1$) as input and outputs a caterpillar C with essence f that has a solution-tight packing of size one (that therefore contains only C) in time polynomial in d.*

By combining Lemmas 2.9 and 2.11 and using Lemma 2.10 we can easily compute a small caterpillar with a specific essence.

▶ **Lemma 2.12.** *There is an algorithm that takes a non-decreasing function $\gamma : [0, d+1] \to [0, d+1]$ such that $\gamma(0) = 0, \gamma(d+1) = d+1$, as input, and outputs a caterpillar C that has essence γ, such that the solution-tight packing $\mathcal{C}$ of C contains all vertices of C, and $|\mathcal{C}| \leq d^3$ in time polynomial in d.*

The caterpillars we want to replace should have solution-tight packings of size at least $d^3 + 1$, as we can then use Lemma 2.12 to compute a caterpillar with the same essence that has a strictly smaller solution-tight packing. Hence, it makes sense to focus on replacing caterpillars of the $(d^3 + 1)$-merged packing of $G - M$.

Handling edges to the modulator. We have already seen that, if C has no edges to the modulator, then we can quickly replace C with a smaller graph that has the same essence. But what if C has edges to the modulator?

Without going into full detail, in our actual reduction rule we identify some large subcaterpillar C of $G - M$ that we replace with a smaller caterpillar C' that has the same essence as C, such that C has some further useful guarantees that help us deal with the potential edges to M. We also set $k' = k - \mathsf{OPT}_d(C) + \mathsf{OPT}_d(C')$. The forward direction of safety of the rule follows directly from Lemma 2.6. For the backward direction, let S be a d-coc set of the reduced graph of size at most k'. If $N(C) \cap (M \setminus S) = \emptyset$, the backward direction of safety again follows from Lemma 2.6. Otherwise, the additional guarantees that C provides come into play. They ensure that C is actually a subgraph of some significantly larger caterpillar $\hat{C}$. Furthermore, we have the guarantee that there exists an optimal solution $S_{\hat{C}}$ for $\hat{C}$ selecting all vertices of $N(M \setminus S) \cap V(\hat{C})$. The proof of this guarantee builds on an argument of Jansen and Bodlaender [34]. Now, let $\hat{C}'$ be the caterpillar obtained when replacing C of $\hat{C}$ with C', clearly $\hat{C}'$ is a subgraph of the reduced instance. We ensure that each vertex $v \in N(C) \cap (M \setminus S)$ has $d + 1$ neighbors in $\hat{C}$ which are left of C, and $d + 1$ neighbors in $\hat{C}$ which are right of C. Therefore, there are also $d + 1$ neighbors of v in $\hat{C}'$ that are left and right of C'. Hence, S must select at least one of these neighbors on each side of C'. These selected vertices effectively cut $\hat{C}'$ into three parts, one of them, call it $P^{*\prime}$, contains C'. Let P^* be the graph $P^{*\prime}$ where C' is replaced with C. It can then be shown that $(S \setminus V(P^{*\prime})) \cup (V(P^*) \cap S_{\hat{C}})$ is a d-`coc` set of the input graph of size at most k, showing that the rule is safe.

Finishing the kernel. We can apply the sketched reduction rule to bound the length of the spine of components of $G - M$ as we can always find suitable graphs C and $\hat{C}$ if the α-merged packing (for a specific value of α that is polynomial in the parameter) of $G - M$ is sufficiently large. However, $G - M$ could still contain an unbounded number of connected components. By exploiting that caterpillars have minimal d-blocking sets of size at most two for each integer d, one can then bound the number of connected components using relatively standard techniques going back to Jansen & Bodlaender [34]. We use a more involved approach based on the expansion lemma that lifts a result of Hols et al. [33] to COC. This results in an approach that is also similar to the strategy used by Bhyravarapu et al. [3] for d-COC parameterized by the size of a c-`coc` set, where $c \geq d$.

We exhaustively apply the previous rules to obtain an instance (G_2, d, k_2, M) that is equivalent to the input instance. At this point, the size of $\mathsf{spine}(G_2 - M)$, the graph consisting of the spines of each component of $G_2 - M$, is bounded by a polynomial in the parameter. In particular, the set $S = M \cup V(\mathsf{spine}(G_2 - M))$ has size polynomial in $d + |M|$, and as $G_2 - S$ is an independent set, the set S is a d-`coc` set of G_2. If $|S| \leq k_2$, then (G_2, d, k_2, M) is a yes-instance, and we output a constant-sized yes-instance. Otherwise, we have $k_2 < |S|$, and therefore k_2 is bounded by a polynomial in the parameter. Then, we can apply the kernel of Xiao [44] for COC parameterized by $k + d$ to finish our kernelization routine.

References

1 Stefan Arnborg, Andrzej Proskurowski, and Detlef Seese. Monadic second order logic, tree automata and forbidden minors. In Egon Börger, Hans Kleine Büning, Michael M. Richter, and Wolfgang Schönfeld, editors, *Computer Science Logic, 4th Workshop, CSL '90, Heidelberg, Germany, October 1-5, 1990, Proceedings*, volume 533 of *Lecture Notes in Computer Science*, pages 1–16. Springer, 1990. `doi:10.1007/3-540-54487-9_49`.

2 Samuel Baguley, Tobias Friedrich, Aneta Neumann, Frank Neumann, Marcus Pappik, and Ziena Zeif. Fixed parameter multi-objective evolutionary algorithms for the w-separator problem. *Algorithmica*, 87(4):537–571, 2025. `doi:10.1007/s00453-024-01290-9`.

3 Sriram Bhyravarapu, Satyabrata Jana, Saket Saurabh, and Roohani Sharma. Difference determines the degree: Structural kernelizations of component order connectivity. In Neeldhara Misra and Magnus Wahlström, editors, *18th International Symposium on Parameterized and Exact Computation, IPEC 2023, September 6-8, 2023, Amsterdam, The Netherlands*, volume 285 of *LIPIcs*, pages 5:1–5:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023. `doi:10.4230/LIPIcs.IPEC.2023.5`.

4 Marin Bougeret, Eric Brandwein, and Ignasi Sau. Kernelization dichotomies for hitting minors under structural parameterizations. In Meena Mahajan, Florin Manea, Annabelle McIver, and Kim Thang Nguyen, editors, *43rd International Symposium on Theoretical Aspects of Computer Science, STACS 2026, Grenoble, France, March 9-13, 2026*, volume 364 of *LIPIcs*, pages 17:1–17:19. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2026. `doi:10.4230/LIPIcs.STACS.2026.17`.

5 Marin Bougeret, Eric Brandwein, and Ignasi Sau. Kernelization dichotomies for hitting minors under structural parameterizations, 2026. `arXiv:2512.13210v2`.

6 Marin Bougeret, Bart M. P. Jansen, and Ignasi Sau. Bridge-depth characterizes which minor-closed structural parameterizations of vertex cover admit a polynomial kernel. *SIAM J. Discret. Math.*, 36(4):2737–2773, 2022. doi:10.1137/21m1400766.

7 Marin Bougeret and Ignasi Sau. How much does a treedepth modulator help to obtain polynomial kernels beyond sparse graphs? *Algorithmica*, 81(10):4043–4068, 2019. doi:10.1007/s00453-018-0468-8.

8 Jonathan F. Buss and Judy Goldsmith. Nondeterminism within P. *SIAM J. Comput.*, 22(3):560–572, 1993. doi:10.1137/0222038.

9 Katrin Casel, Tobias Friedrich, Davis Issac, Aikaterini Niklanovits, and Ziena Zeif. Balanced crown decomposition for connectivity constraints. In Petra Mutzel, Rasmus Pagh, and Grzegorz Herman, editors, *29th Annual European Symposium on Algorithms, ESA 2021, September 6-8, 2021, Lisbon, Portugal (Virtual Conference)*, volume 204 of *LIPIcs*, pages 26:1–26:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021. doi:10.4230/LIPIcs.ESA.2021.26.

10 Katrin Casel, Tobias Friedrich, Aikaterini Niklanovits, Kirill Simonov, and Ziena Zeif. Combining crown structures for vulnerability measures. In Édouard Bonnet and Pawel Rzazewski, editors, *19th International Symposium on Parameterized and Exact Computation, IPEC 2024, September 4-6, 2024, Royal Holloway, University of London, Egham, United Kingdom*, volume 321 of *LIPIcs*, pages 1:1–1:15. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.IPEC.2024.1.

11 Jianer Chen, Iyad A. Kanj, and Weijia Jia. Vertex cover: Further observations and further improvements. *J. Algorithms*, 41(2):280–301, 2001. doi:10.1006/jagm.2001.1186.

12 Benny Chor, Mike Fellows, and David W. Juedes. Linear kernels in linear time, or how to save k colors in $O(n^2)$ steps. In Juraj Hromkovic, Manfred Nagl, and Bernhard Westfechtel, editors, *Graph-Theoretic Concepts in Computer Science, 30th International Workshop, WG 2004, Bad Honnef, Germany, June 21-23, 2004, Revised Papers*, volume 3353 of *Lecture Notes in Computer Science*, pages 257–269. Springer, 2004. doi:10.1007/978-3-540-30559-0_22.

13 Ernest J. Cockayne, Stephen T. Hedetniemi, and Peter J. Slater. Matchings and transversals in hypergraphs, domination and independence-in trees. *J. Comb. Theory B*, 26(1):78–80, 1979. doi:10.1016/0095-8956(79)90044-3.

14 Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015. doi:10.1007/978-3-319-21275-3.

15 Marek Cygan, Daniel Lokshtanov, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. On the hardness of losing width. *Theory Comput. Syst.*, 54(1):73–82, 2014. doi:10.1007/s00224-013-9480-1.

16 David Dekker and Bart M. P. Jansen. Kernelization for feedback vertex set via elimination distance to a forest. *Discret. Appl. Math.*, 346:192–214, 2024. doi:10.1016/J.DAM.2023.12.016.

17 Holger Dell and Dániel Marx. Kernelization of packing problems. In Yuval Rabani, editor, *Proceedings of the Twenty-Third Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2012, Kyoto, Japan, January 17-19, 2012*, pages 68–81. SIAM, 2012. doi:10.1137/1.9781611973099.6.

18 Huib Donkers and Bart M. P. Jansen. A turing kernelization dichotomy for structural parameterizations of f-minor-free deletion. *J. Comput. Syst. Sci.*, 119:164–182, 2021. doi:10.1016/j.jcss.2021.02.005.

19 Pål Grønås Drange, Markus S. Dregi, and Pim van 't Hof. On the computational complexity of vertex integrity and component order connectivity. *Algorithmica*, 76(4):1181–1202, 2016. doi:10.1007/s00453-016-0127-x.

20 Michael R. Fellows. Blow-ups, win/win's, and crown rules: Some new directions in FPT. In Hans L. Bodlaender, editor, *Graph-Theoretic Concepts in Computer Science, 29th International Workshop, WG 2003, Elspeet, The Netherlands, June 19-21, 2003, Revised Papers*, volume 2880 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2003. doi:10.1007/978-3-540-39890-5_1.

21 Fedor V. Fomin, Daniel Lokshtanov, Neeldhara Misra, and Saket Saurabh. Planar f-deletion: Approximation, kernelization and optimal FPT algorithms. In *53rd Annual IEEE Symposium on Foundations of Computer Science, FOCS 2012, New Brunswick, NJ, USA, October 20-23, 2012*, pages 470–479. IEEE Computer Society, 2012. doi:10.1109/FOCS.2012.62.

22 Fedor V. Fomin, Daniel Lokshtanov, Saket Saurabh, and Meirav Zehavi. *Kernelization: theory of parameterized preprocessing*. Cambridge University Press, 2019. doi:10.1017/9781107415157.

23 Fedor V. Fomin and Torstein J. F. Strømme. Vertex cover structural parameterization revisited. In Pinar Heggernes, editor, *Graph-Theoretic Concepts in Computer Science - 42nd International Workshop, WG 2016, Istanbul, Turkey, June 22-24, 2016, Revised Selected Papers*, volume 9941 of *Lecture Notes in Computer Science*, pages 171–182, 2016. doi:10.1007/978-3-662-53536-3_15.

24 Robert Ganian, Fabian Klute, and Sebastian Ordyniak. On structural parameterizations of the bounded-degree vertex deletion problem. *Algorithmica*, 83(1):297–336, January 2021. doi:10.1007/s00453-020-00758-8.

25 M. R. Garey and David S. Johnson. The rectilinear steiner tree problem is NP complete. *SIAM Journal of Applied Mathematics*, 32:826–834, 1977.

26 Archontia C. Giannopoulou, Bart M. P. Jansen, Daniel Lokshtanov, and Saket Saurabh. Uniform kernelization complexity of hitting forbidden minors. *ACM Trans. Algorithms*, 13(3):35:1–35:35, 2017. doi:10.1145/3029051.

27 Gracinda M. S. Gomes and John M. Howie. On the ranks of certain semigroups of order-preserving transformations. *Semigroup Forum*, 45(1):272–282, 1992. doi:10.1007/BF03025769.

28 Jakob Greilhuber and Roohani Sharma. Component order connectivity admits no polynomial kernel parameterized by the distance to subdivided comb graphs. In Édouard Bonnet and Pawel Rzazewski, editors, *19th International Symposium on Parameterized and Exact Computation, IPEC 2024, September 4-6, 2024, Royal Holloway, University of London, Egham, United Kingdom*, volume 321 of *LIPIcs*, pages 21:1–21:17. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2024. doi:10.4230/LIPIcs.IPEC.2024.21.

29 Daniel Gross, Monika Heinig, Lakshmi Iswara, L. William Kazmierczak, Kristi Luttrell, John T. Saccoman, and Charles Suffel. A survey of component order connectivity models of graph theoretic networks. *WSEAS Transactions on Mathematics*, 12(9):895–910, 2013.

30 Anupam Gupta, Euiwoong Lee, Jason Li, Pasin Manurangsi, and Michal Wlodarczyk. Losing treewidth by separating subsets. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 1731–1749. SIAM, 2019. doi:10.1137/1.9781611975482.104.

31 Peter M. Higgins. Combinatorial results for semigroups of order-preserving mappings. *Mathematical Proceedings of the Cambridge Philosophical Society*, 113(2):281–296, 1993. doi:10.1017/S0305004100075964.

32 Eva-Maria C. Hols and Stefan Kratsch. Smaller parameters for vertex cover kernelization. In Daniel Lokshtanov and Naomi Nishimura, editors, *12th International Symposium on Parameterized and Exact Computation, IPEC 2017, September 6-8, 2017, Vienna, Austria*, volume 89 of *LIPIcs*, pages 20:1–20:12. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2017. doi:10.4230/LIPIcs.IPEC.2017.20.

33 Eva-Maria C. Hols, Stefan Kratsch, and Astrid Pieterse. Elimination distances, blocking sets, and kernels for vertex cover. *SIAM J. Discret. Math.*, 36(3):1955–1990, 2022. doi:10.1137/20m1335285.

34 Bart M. P. Jansen and Hans L. Bodlaender. Vertex cover kernelization revisited - upper and lower bounds for a refined parameter. *Theory Comput. Syst.*, 53(2):263–299, 2013. doi:10.1007/s00224-012-9393-4.

35 Bart M. P. Jansen and Astrid Pieterse. Polynomial kernels for hitting forbidden minors under structural parameterizations. *Theor. Comput. Sci.*, 841:124–166, 2020. doi:10.1016/j.tcs.2020.07.009.

36 Richard M. Karp. Reducibility among combinatorial problems. In Raymond E. Miller and James W. Thatcher, editors, *Proceedings of a symposium on the Complexity of Computer Computations, held March 20-22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, USA*, The IBM Research Symposia Series, pages 85–103. Plenum Press, New York, 1972. doi:10.1007/978-1-4684-2001-2_9.

37 Lawrence William Kazmierczak. *On the relationship between connectivity and component order connectivity.* PhD thesis, Stevens Institute of Technology, 2003. URL: https://dl.acm.org/doi/10.5555/959796.

38 Samir Khuller. Algorithms column: the vertex cover problem. *SIGACT News*, 33(2):31–33, 2002. doi:10.1145/564585.564598.

39 Mithilesh Kumar and Daniel Lokshtanov. A 2lk kernel for l-component order connectivity. In Jiong Guo and Danny Hermelin, editors, *11th International Symposium on Parameterized and Exact Computation, IPEC 2016, August 24-26, 2016, Aarhus, Denmark*, volume 63 of *LIPIcs*, pages 20:1–20:14. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2016. doi:10.4230/LIPIcs.IPEC.2016.20.

40 Michael Lampis. A kernel of order 2 k-c log k for vertex cover. *Inf. Process. Lett.*, 111(23-24):1089–1091, 2011. doi:10.1016/J.IPL.2011.09.003.

41 John M. Lewis and Mihalis Yannakakis. The node-deletion problem for hereditary properties is np-complete. *J. Comput. Syst. Sci.*, 20(2):219–230, 1980. doi:10.1016/0022-0000(80)90060-4.

42 Diptapriyo Majumdar, Venkatesh Raman, and Saket Saurabh. Polynomial kernels for vertex cover parameterized by small degree modulators. *Theory Comput. Syst.*, 62(8):1910–1951, 2018. doi:10.1007/s00224-018-9858-1.

43 Arezou Soleimanfallah and Anders Yeo. A kernel of order 2k-c for vertex cover. *Discret. Math.*, 311(10-11):892–895, 2011. doi:10.1016/j.disc.2011.02.014.

44 Mingyu Xiao. Linear kernels for separating a graph into components of bounded size. *J. Comput. Syst. Sci.*, 88:260–270, 2017. doi:10.1016/j.jcss.2017.04.004.

A Problem Definitions of Component Order Connectivity Variants

In this section we collect and formally restate the definitions of the COC variants we consider in the paper. Recall that for an integer $d \geq 1$ and a graph G, a d-coc set of G is a set $S \subseteq V(G)$ such that each connected component of $G - S$ has size at most d.

Component Order Connectivity (COC)

Input:	Graph G, integer $d \geq 1$, integer k
Question:	Is there a d-coc set S of G with $\lvert S \rvert \leq k$?

If d is a fixed constant, we instead obtain the problem d-COC. For each fixed d we have a different problem.

d-COC

Input:	Graph G, integer k
Question:	Is there a d-coc set S of G with $\lvert S \rvert \leq k$?

The problem Vertex Cover (VC) is the same as 1-COC. Next, we consider structural parameterizations. In the following, $\mathcal{G}$ always denotes a fixed graph class, and we parameterize by the size of a modulator M to $\mathcal{G}$. For the problem COC we additionally consider d as part of the parameter.

COC/d+dist-to-$\mathcal{G}$

Input:	Graph G, integer $d \geq 1$, integer k, set $M \subseteq V(G)$ such that $G - M \in \mathcal{G}$
Parameter:	$d + \|M\|$
Question:	Is there a d-coc set S of G with $\|S\| \leq k$?

When we consider the problem d-COC instead of COC, d is a constant, and thus it is naturally no longer part of the parameter.

d-COC/dist-to-$\mathcal{G}$

Input:	Graph G, integer k, set $M \subseteq V(G)$ such that $G - M \in \mathcal{G}$
Parameter:	$\|M\|$
Question:	Is there a d-coc set S of G with $\|S\| \leq k$?

Our result in Main Theorem 1 is for the problem COC/d+pw-1, which is COC/d+dist-to-$\mathcal{G}$ where $\mathcal{G}$ is fixed to be the class of graphs with pathwidth at most 1, that is, the class of caterpillar forests.

COC/d+pw-1

Input:	Graph G, integer $d \geq 1$, integer k, set $M \subseteq V(G)$ such that $G - M$ is a caterpillar forest
Parameter:	$d + \|M\|$
Question:	Is there a d-coc set S of G with $\|S\| \leq k$?

The result also applies to the problem where d is fixed.

d-COC/pw-1

Input:	Graph G, integer k, set $M \subseteq V(G)$ such that $G - M$ is a caterpillar forest
Parameter:	$\|M\|$
Question:	Is there a d-coc set S of G with $\|S\| \leq k$?

We additionally provide results for the following problems where d is not part of the parameter.

COC/vc

Input:	Graph G, integer $d \geq 1$, integer k, set $M \subseteq V(G)$ such that $G - M$ has no edges
Parameter:	$\|M\|$
Question:	Is there a d-coc set S of G with $\|S\| \leq k$?

COC/k+vc

Input:	Graph G, integer $d \geq 1$, integer k, set $M \subseteq V(G)$ such that $G - M$ has no edges
Parameter:	$k + \|M\|$
Question:	Is there a d-coc set S of G with $\|S\| \leq k$?

COC/`pw`-1

Input:	Graph G, integer $d \geq 1$, integer k, set $M \subseteq V(G)$ such that $G - M$ is a caterpillar forest
Parameter:	$\lvert M \rvert$
Question:	Is there a d-`coc` set S of G with $\lvert S \rvert \leq k$?