Parameterized Complexity of Vertex Splitting to Pathwidth at Most 1

Jakob Baumann, Matthias Pfretzschner(✉), and Ignaz Rutter

Universität Passau, 94032 Passau, Germany
{baumannjak,pfretzschner,rutter}@fim.uni-passau.de

Abstract. Motivated by the planarization of 2-layered straight-line drawings, we consider the problem of modifying a graph such that the resulting graph has pathwidth at most 1. The problem Pathwidth-One Vertex Explosion (POVE) asks whether such a graph can be obtained using at most k vertex explosions, where a *vertex explosion* replaces a vertex v by $\deg(v)$ degree-1 vertices, each incident to exactly one edge that was originally incident to v. For POVE, we give an FPT algorithm with running time $O(4^k \cdot m)$ and an $O(k^2)$ kernel, thereby improving over the $O(k^6)$-kernel by Ahmed et al. [2] in a more general setting. Similarly, a *vertex split* replaces a vertex v by two distinct vertices v_1 and v_2 and distributes the edges originally incident to v arbitrarily to v_1 and v_2. Analogously to POVE, we define the problem variant Pathwidth-One Vertex Splitting (POVS) that uses the split operation instead of vertex explosions. Here we obtain a linear kernel and an algorithm with running time $O((6k+12)^k \cdot m)$. This answers an open question by Ahmed et al. [2].

Keywords: Vertex Splitting · Vertex Explosion · Pathwidth 1

1 Introduction

Crossings are one of the main aspects that negatively affect the readability of drawings [20]. It is therefore natural to try and modify a given graph in such a way that it can be drawn without crossings while preserving as much of the information as possible. We consider three different operations.

A *deletion operation* simply removes a vertex from the graph. A *vertex explosion* replaces a vertex v by $\deg(v)$ degree-1 vertices, each incident to exactly one edge that was originally incident to v. Finally, a *vertex split* replaces a vertex v by two distinct vertices v_1 and v_2 and distributes the edges originally incident to v arbitrarily to v_1 and v_2.

Nöllenburg et al. [18] have recently studied the vertex splitting problem, which is known to be NP-complete [11]. In particular, they gave a non-uniform FPT-algorithm for deciding whether a given graph can be planarized with at most k splits. We observe that, since degree-1 vertices can always be inserted into a planar drawing, the vertex explosion model and the vertex deletion model are

D. Paulusma and B. Ries (Eds.): WG 2023, LNCS 14093, pp. 30–43, 2023.
https://doi.org/10.1007/978-3-031-43380-1_3

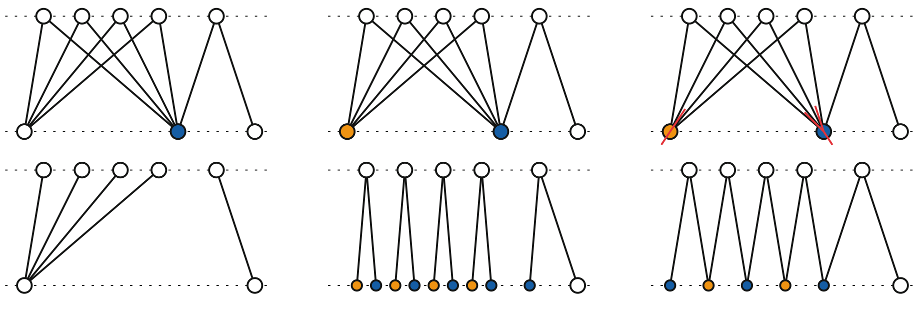


Fig. 1. Given the shown bipartite graph, a crossing-free 2-layered drawing can be obtained using one vertex deletion (a), two vertex explosions (b), or three vertex splits (c).

equivalent for obtaining planar graphs. Note that this is not necessarily the case for other target graph classes (see, for example, Fig. 1). The problem of deleting vertices to obtain a planar graph is also known as Vertex Planarization and has been studied extensively in the literature [13,15–17]. In particular, Jansen et al. [13] gave an FPT-algorithm with running time $O(2^{O(k \log k)} \cdot n)$.

Ahmed et al. [2] investigated the problem of splitting the vertices of a bipartite graph so that it admits a 2-layered drawing without crossings. They assume that the input graph is bipartite and only the vertices of one of the two sets in the bipartition may be split. Under this condition, they give an $O(k^6)$-kernel for the vertex explosion model, which results in an $O(2^{O(k^6)}m)$-time algorithm. They ask whether similar results can be obtained in the vertex splitting model. Figure 1 illustrates the three operations in the context of 2-layered drawings[1].

We note that a graph admits a 2-layer drawing without crossings if and only if it has pathwidth at most 1, i.e., it is a disjoint union of caterpillars [3,9]. Motivated by this, we more generally consider the problem of turning a graph $G = (V, E)$ into a graph of pathwidth at most 1 by the above operations. In order to model the restriction of Ahmed et al. [2] that only one side of their bipartite input graph may be split, we further assume that we are given a subset $S \subseteq V$, to which we may apply modification operations as part of the input. We define that the new vertices resulting from an operation are also included in S.

More formally, we consider the following problems, all of which have been shown to be NP-hard [1,19].

[1] In this context, minimizing the number of vertex explosions is equivalent to minimizing the number of vertices that are split, since it is always best to split a vertex as often as possible.

Pathwidth-One Vertex Explosion (POVE)
Input: An undirected graph $G = (V, E)$, a set $S \subseteq V$, and a positive integer k.
Question: Is there a set $W \subseteq S$ with $|W| \leq k$ such that the graph resulting from exploding all vertices in W has pathwidth at most 1?

Pathwidth-One Vertex Splitting (POVS)
Input: An undirected graph $G = (V, E)$, a set $S \subseteq V$, and a positive integer k.
Question: Is there a sequence of at most k splits on vertices in S such that the resulting graph has pathwidth at most 1?

We note that the analogous problem with the deletion operation has been studied extensively [8,19,23]. Here, a branching algorithm with running time $O(3.888^k \cdot n^{O(1)})$ [23] and a quadratic kernel [8] are known. Our results are as follows.

First, in Sect. 3, we show that POVE admits a kernel of size $O(k^2)$ and an algorithm with running time $O(4^k m)$, thereby improving over the results of Ahmed et al. [2] in a more general setting.

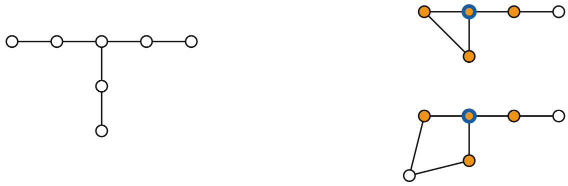
Second, in Sect. 4, we show that POVS has a kernel of size $16k$ and it admits an algorithm with running time $O((6k+12)^k \cdot m)$. This answers the open question of Ahmed et al. [2].

Finally, in Sect. 5, we consider the problem Π Vertex Splitting(Π-VS), the generalized version of the splitting problem where the goal is to obtain a graph of a specific graph class Π using at most k split operations. Eppstein et al. [10] recently studied the similar problem of deciding whether a given graph G is *k-splittable*, i.e., whether it can be turned into a graph of Π by splitting every vertex of G at most k times. For graph classes Π that can be expressed in monadic second-order graph logic (MSO_2, see [7]), they gave an FPT algorithm parameterized by the solution size k and the treewidth of the input graph. We adapt their algorithm for the problem Π-VS, resulting in an FPT algorithm parameterized by the solution size k for MSO_2-definable graph classes Π of bounded treewidth. Using a similar algorithm, we obtain the same result for the problem variant using vertex explosions.

2 Preliminaries

A parameterized problem L with parameter k is *non-uniformly fixed-parameter tractable* if, for every value of k, there exists an algorithm that decides L in time $f(k) \cdot n^{O(1)}$ for some computable function f. If there is a single algorithm that satisfies this property for all values of k, then L is *(uniformly) fixed-parameter tractable*.

Given a graph G, we let n and m denote the number of vertices and edges of G, respectively. Since we can determine the subgraph of G that contains no isolated vertices in $O(m)$ time, we assume, without loss of generality, that $n \in O(m)$. For a vertex $v \in V(G)$, we let $N(v) := \{u \in V(G) \mid \mathrm{adj}(v, u)\}$ and $N[v] := N(v) \cup \{v\}$ denote the open and closed neighborhood of v in G, respectively.



(a) (b)

Fig. 2. (a) The graph T_2. (b) Two graphs that do not contain T_2 as a subgraph, but both contain N_2 (marked in orange) as a substructure. (Color figure online)

We refer to vertices of degree 1 as *pendant* vertices. For a vertex v of G, we let $\deg^*(v) := |\{u \in N(v) \mid \deg(u) > 1\}|$ denote the degree of v ignoring its pendant neighbors. If $\deg^*(v) = d$, we refer to v as a vertex of *degree* d*. A graph is a *caterpillar* (respectively a *pseudo-caterpillar*), if it consists of a simple path (a simple cycle) with an arbitrary number of adjacent pendant vertices. The path (the cycle) is called the *spine* of the (pseudo-)caterpillar.

Philip et al. [19] mainly characterized the graphs of pathwidth at most 1 as the graphs containing no cycles and no T_2 (three simple paths of length 2 that all share an endpoint; see Fig. 2a) as a subgraph. We additionally use slightly different sets of forbidden substructures. An N_2 *substructure* consists of a *root* vertex r adjacent to three distinct vertices of degree at least 2. Note that every T_2 contains an N_2 substructure, however, the existence of an N_2 substructure does not generally imply the existence of a T_2 subgraph; see Fig. 2b. In the following proposition, we state the different characterizations for graphs of pathwidth at most 1 that we use in this work.

Proposition 1 ($\star$[2]). *For a graph G, the following statements are equivalent.*

a) G has pathwidth at most 1
b) every connected component of G is a caterpillar
c) G is acyclic and contains no T_2 subgraph
d) G is acyclic and contains no N_2 substructure
e) G contains no N_2 substructure and no connected component that is a pseudo-caterpillar.

We define the *potential* of $v \in V(G)$ as $\mu(v) := \max(\deg^*(v) - 2, 0)$. The *global potential* $\mu(G) := \sum_{v \in V(G)} \mu(v)$ is defined as the sum of the potentials of all vertices in G. Observe that $\mu(G) = 0$ if and only if G contains no N_2 substructure. The global potential thus indicates how far away we are from eliminating all N_2 substructures from the instance.

Recall that, for the problems POVE and POVS, the set $S \subseteq V(G)$ marks the vertices of G that may be chosen for the respective operations. We say that a set $W \subseteq S$ is a *pathwidth-one explosion set* (POES) of G, if the graph resulting from exploding all vertices in W has pathwidth at most 1.

---
[2] The proofs of results marked with a star can be found in the full version [4].

3 FPT Algorithms for Pathwidth-One Vertex Explosion

In this section, we first show that POVE can be solved in time $O(4^k \cdot m)$ using bounded search trees. Subsequently, we develop a kernelization algorithm for POVE that yields a quadratic kernel in linear time.

3.1 Branching Algorithm

We start by giving a simple branching algorithm for POVE, similar to the algorithm by Philip et al. [19] for the deletion variant of the problem. For an N_2 substructure X, observe that exploding vertices not contained in X cannot eliminate X, because the degrees of the vertices in X remain the same due to the new degree-1 vertices resulting from the explosion. To obtain a graph of pathwidth at most 1, it is therefore always necessary to explode one of the four vertices of every N_2 substructure by Proposition 1. Our branching rule thus first picks an arbitrary N_2 substructure from the instance and then branches on which of the four vertices of the N_2 substructure belongs to the POES. Recall that S denotes the set of vertices of the input graph that can be exploded.

Branching Rule 1. *Let r be the root of an N_2 substructure contained in G and let x, y, and z denote the three neighbors of r in N_2. For every vertex $v \in \{r, x, y, z\} \cap S$, create a branch for the instance $(G', S \setminus \{v\}, k-1)$, where G' is obtained from G by exploding v.*
If $\{r, x, y, z\} \cap S = \emptyset$, reduce to a trivial no-instance instead.

Note that an N_2 substructure can be found in $O(m)$ time by checking, for every vertex v in G, whether v has at least three neighbors of degree at least 2. Also note that vertex explosions do not increase the number of edges of the graph. Since Branching Rule 1 creates at most four new branches, each of which reduces the parameter k by 1, exhaustively applying the rule takes $O(4^k \cdot m)$ time. By Proposition 1, it subsequently only remains to eliminate connected components that are a pseudo-caterpillar. Since a pseudo-caterpillar can (only) be turned into a caterpillar by exploding a vertex of its spine, the remaining instance can be solved in linear time.

Theorem 1. *The problem* Pathwidth-One Vertex Explosion *can be solved in time $O(4^k \cdot m)$.*

3.2 Quadratic Kernel

We now turn to our kernelization algorithm for POVE. In this section, we develop a kernel of quadratic size, which can be computed in linear time.

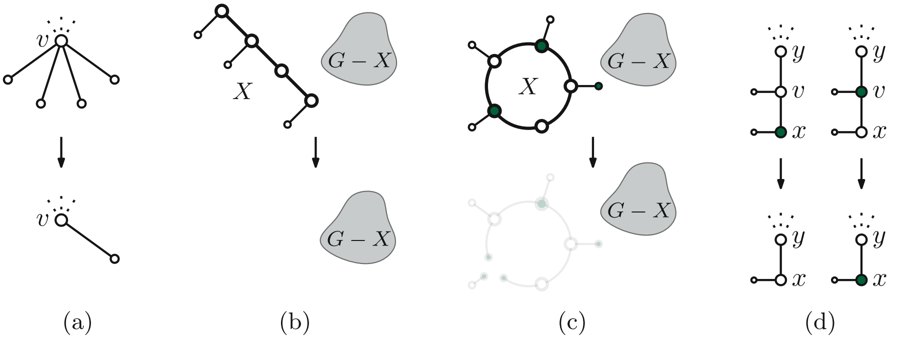


Fig. 3. Examples for Reduction Rules 1 (a), 2 (b), 3 (c), and 4 (d). The vertices of S are marked in green (Color figure online).

We adopt our first two reduction rules from the kernelization of the deletion variant by Philip et al. [19] and show that these rules are also safe for the explosion variant. The first rule reduces the number of pendant neighbors of each vertex to at most one; see Fig. 3a.

Reduction Rule 1. (⋆). *If G contains a vertex v with at least two pendant neighbors, remove all pendant neighbors of v except one to obtain the graph G' and reduce the instance to $(G',\ S \cap V(G'),\ k)$.*

Since a caterpillar has pathwidth at most 1 by Proposition 1, we can safely remove any connected component of G that forms a caterpillar; see Fig. 3b for an example.

Reduction Rule 2. *If G contains a connected component X that is a caterpillar, remove X from G and reduce the instance to $(G - X,\ S \setminus V(X),\ k)$.*

If G contains a connected component that is a pseudo-caterpillar, then exploding an arbitrary vertex of its spine yields a caterpillar. If the spine contains no vertex of S, the spine is a cycle that cannot be broken by a vertex explosion. However, by Proposition 1, acyclicity is a necessary condition for a graph of pathwidth at most 1. Hence we get the following reduction rule; see Fig. 3c for an illustration.

Reduction Rule 3. *Let X denote a connected component of G that is a pseudo-caterpillar. If the spine of X contains a vertex of S, remove X from G and reduce the instance to $(G - X,\ S \setminus V(X),\ k - 1)$. Otherwise reduce to a trivial no-instance.*

Recall that the degree* of a vertex is the number of its non-pendant neighbors. Our next goal is to shorten paths of degree*-2 vertices to at most two vertices. If we have a path x, y, z of degree*-2 vertices, we refer to y as a *2-enclosed* vertex. Note that exploding a 2-enclosed vertex y cannot eliminate any

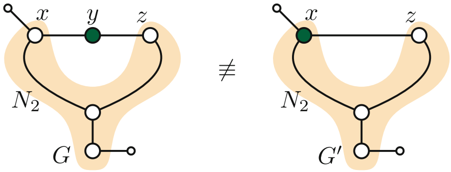


Fig. 4. A graph G that has no POES, because the highlighted N_2 substructure contains no vertex of S. For the graph G' resulting from contracting y into x, the set $\{x\}$ is a POES. The two instances are therefore not equivalent.

N_2 substructures from the instance. By Proposition 1, vertex y can thus only be part of an optimal solution if exploding y breaks cycles. If we want to shorten the chain x, y, z by contracting y into one of its neighbors, we therefore need to ensure that the shortened chain contains a vertex of S if and only if the original chain contained a vertex of S. If $y \in S$, we cannot simply add one of its neighbors, say x, to S in the reduced instance, because exploding x may additionally remove an N_2 substructure; see Fig. 4 for an example. While shortening paths of degree*-2 vertices to at most three vertices is simple, shortening them to length at most 2 (i.e., eliminating all 2-enclosed vertices) is therefore more involved. In the following, we briefly sketch how this can be achieved in linear time. For the specific reduction rules and the corresponding correctness proofs, we refer to the full version of the paper [4].

Lemma 1 (⋆). *Given an instance of* POVE*, an equivalent instance without 2-enclosed vertices can be computed in* $O(m)$ *time.*

Sketch of Proof. Given a 2-enclosed vertex y, we show that we can decide greedily whether y is contained in an optimal solution or not. This means that we can either immediately explode y, or we can safely contract it into one of its degree*-2 neighbors. Since y is 2-enclosed, y is not contained in any N_2 substructures and we thus only have to consider cycles containing y. If there exists a cycle C in G with $C \cap S = \{y\}$ (i.e., y is the only splittable vertex of C), then we can immediately explode y. Otherwise, every cycle containing y contains at least one additional vertex of S. In this case, we can show that there exists a minimum POES of G that does not contain y, thus we can remove y from S and contract it into one of its neighbors, thereby preserving all cycles of the instance. To achieve linear running time, we can show that the set of 2-enclosed vertices that should be exploded can be computed globally using a specialized spanning tree. □

To simplify the instance even further, the following reduction rule removes all degree*-2 vertices v that are adjacent to a vertex x of degree* 1; see Fig. 3d for an illustration. Roughly speaking, since v cannot be contained in a cycle and x substitutes v in all N_2 substructures v is contained in, all forbidden substructures are preserved.

Reduction Rule 4 (⋆). *Let v be a degree*-2 vertex of G with non-pendant neighbors x and y, such that x has degree* 1. Remove v from G and add a new edge xy. If $v \in S$, reduce to $(G - v + xy, (S \setminus \{v\}) \cup \{x\}, k)$. Otherwise reduce to $(G - v + xy, S \setminus \{x\}, k)$.*

Recall that the global potential $\mu(G)$ indicates how far away we are from our goal of eliminating all N_2 substructures from G. With the following lemma, we show that our reduction rules ensure that the number of vertices in the graph G is bounded linearly in the global potential of G.

Lemma 2. *After exhaustively applying Reduction Rules 1–4 and Lemma 1, it holds that $|V(G)| \leq 8 \cdot \mu(G)$.*

Proof. Reduction Rule 2 ensures that G contains no vertices of degree* 0. For $i \in \{1, 2\}$, let V_i denote the set of non-pendant degree*-i vertices of G and let V_3 denote the set of vertices with degree* at least 3. Recall that we defined the global potential as

$$\mu(G) = \sum_{v \in V(G)} \mu(v) = \sum_{v \in V(G)} \max(0, \deg^*(v) - 2).$$

Since all vertices of V_1 and V_2 have degree* at most 2, their potential is 0 and we get

$$\mu(G) = \sum_{v \in V_3} (\deg^*(v) - 2) = \sum_{v \in V_3} \deg^*(v) - 2 \cdot |V_3|.$$

Note that $|V_3| \leq \mu(G)$, because each vertex of degree* at least 3 contributes at least 1 to the global potential. We therefore get

$$\sum_{v \in V_3} \deg^*(v) \leq 3 \cdot \mu(G). \tag{1}$$

By Lemma 1, every vertex in $v \in V_2$ is adjacent to a vertex of $V_1 \cup V_3$, since otherwise, v would be 2-enclosed. However, Reduction Rule 4 additionally ensures that vertices of V_2 cannot be adjacent to vertices of V_1, thus every vertex of V_2 must be adjacent to a vertex of V_3. Note that two adjacent vertices of V_1 would form a caterpillar, which is prohibited by Reduction Rule 2. Therefore, every vertex of V_1 is also adjacent to a vertex of V_3.

Overall, every vertex of V_1 and V_2 is thus adjacent to a vertex of V_3. Note that every vertex $v \in V_1$ must additionally have a pendant neighbor, because otherwise, v itself would be a pendant vertex. Hence every vertex of V_1 and V_2 has degree at least 2 and thus contributes to the degree* of its neighbor in V_3. We therefore have $|V_1| + |V_2| \leq \sum_{v \in V_3} \deg^*(v)$, hence $|V_1| + |V_2| \leq 3 \cdot \mu(G)$ by Eq. 1. Recall that $|V_3| \leq \mu(G)$, thus $|V_1| + |V_2| + |V_3| \leq 4 \cdot \mu(G)$. By Reduction Rule 1, each of these vertices can have at most one pendant neighbor and thus $|V(G)| \leq 8 \cdot \mu(G)$.

With Lemma 2, it now only remains to find an upper bound for the global potential $\mu(G)$. We do this using the following two reduction rules.

Reduction Rule 5. *Let v be a vertex of G with potential $\mu(v) > k$. If $v \in S$, explode v to obtain the graph G' and reduce the instance to $(G',\ S \setminus \{v\},\ k-1)$. Otherwise reduce to a trivial no-instance.*

Proof of Safeness. Since exploding a vertex $u \in V(G) \setminus \{v\}$ decreases $\mu(v)$ by at most one, after exploding at most k vertices in $V(G) \setminus \{v\}$ we still have $\mu(v) > 0$. Because $\mu(v) > 0$ implies that G contains an N_2 substructure, it is therefore always necessary to explode vertex v by Proposition 1. ☐

Reduction Rule 6. *If $\mu(G) > 2k^2 + 2k$, reduce to a trivial no-instance.*

Proof of Safeness. By Reduction Rule 5 we have $\mu(v) \leq k$ and consequently $\deg^*(v) \leq k + 2$ for all $v \in V(G)$. Hence exploding a vertex v decreases the potential of v by at most k and the potential of each of its non-pendant neighbors by at most 1. Overall, k vertex explosions can therefore only decrease the global potential $\mu(G)$ by at most $k \cdot (2k + 2)$. ☐

Because Reduction Rule 6 gives us an upper bound for the global potential $\mu(G)$, we can now use Lemma 2 to obtain the kernel.

Theorem 2 ($\star$). *The problem* Pathwidth-One Vertex Explosion *admits a kernel of size* $16k^2 + 16k$*. It can be computed in time* $O(m)$*.*

4 FPT Algorithms for Pathwidth-One Vertex Splitting

In this section, we briefly outline how the results from Sect. 3 can be adapted for the split operation. For detailed proofs, we refer to the full version [4].

4.1 Linear Kernel

One can prove that Reduction Rules 1–4 and Lemma 1 we used for POVE are also safe for the problem POVS. Since only these are needed to establish the upper bound of $|V(G)| \leq 8 \cdot \mu(G)$ in Lemma 2, the lemma also applies for POVS.

The main difference to the kernelization of POVE lies in the way the global potential changes due to splits. While a vertex explosion can decrease the global potential linearly in k, we can show that a single vertex split decreases $\mu(G)$ by at most 2. If $\mu(G) > 2k$, we can thus again reduce to a trivial no-instance. Using Lemma 2 with $\mu(G) \leq 2k$, we obtain the following result.

Theorem 3 ($\star$). *The problem* Pathwidth-One Vertex Splitting *admits a kernel of size* $16k$*. It can be computed in time* $O(m)$*.*

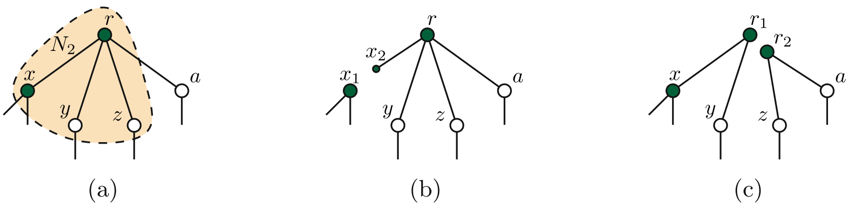


Fig. 5. (a) An N_2 substructure $\{r, x, y, z\}$. (b)-(c) Two possible branches eliminating the N_2 substructure. The former splits off edge rx at x, the latter splits off the edges rz and ra at r.

4.2 Branching Algorithm

As in Sect. 3.1, our branching algorithm for POVS eliminates every N_2 substructure of G by branching on which of its four vertices should be split. In this case, however, we need to additionally consider the possible ways to split a single vertex. The following lemma helps us limit the number of suitable splits.

Lemma 3 (⋆). *For every instance of* POVS*, there exists a minimum sequence of splits such that every split operation splits off at most two edges.*

Theorem 4 (⋆). *The problem* POVS *can be solved in time* $O((6k + 12)^k \cdot m)$*.*

Sketch of Proof. From the kernelization, we use Reduction Rule 1 reducing pendant vertices, and the above rule that yields the bound $\mu(G) \leq 2k$. Together, these two rules ensure that each vertex has degree at most $2k+3$. We now branch on the way of splitting an N_2 substructure with root r and neighbors $\{x, y, z\}$ as above (see Fig. 5). If we split r, then, by Lemma 3, we may assume that we split off one of the neighbors $\{x, y, z\}$, together with at most one other neighbor of r; these are $3 \cdot (2k+3)$ choices. If we split a vertex $v \in \{x, y, z\}$, then it is necessary that we only split off the edge rv at v, thus there is only one possibility for each of them. Overall, we thus find a branching vector of size $6k + 12$. □

5 FPT Algorithms for Splitting and Exploding to MSO$_2$-Definable Graph Classes of Bounded Treewidth

While Sect. 4 focused on the problem of obtaining graphs of pathwidth at most 1 using at most k vertex splits on the input graph, we now consider the problem of splitting vertices to obtain other graph classes. With the following problem, we generalize the problem POVS.

Π VERTEX SPLITTING(*Π*-VS)

Input: An undirected graph $G = (V, E)$, a set $S \subseteq V$, and a positive integer k.

Question: Is there a sequence of at most k splits on vertices in S such that the resulting graph is contained in Π?

Nöllenburg et al. [18] showed that, for any minor-closed graph class Π, the graph class Π_k containing all graphs that can be modified to a graph in Π using at most k vertex splits is also minor-closed. Robertson and Seymour [21] showed that every minor-closed graph class has a constant-size set of forbidden minors and that it can be tested in cubic time whether a graph contains a given fixed graph as a minor. Since Π_k is minor-closed, this implies the existence of a non-uniform FPT-algorithm for the problem Π-VS. Because the graphs of pathwidth at most 1 form a minor-closed graph class, this includes the problem POVS.

Proposition 2 ([18]). *For every minor-closed graph class Π, the problem Π-VS is non-uniformly FPT parameterized by the solution size k.*

We say that a graph class Π is *MSO_2-definable*, if there exists an MSO_2 (monadic second-order graph logic, see [7]) formula φ such that $G \models \varphi$ if and only if $G \in \Pi$. In the following, we show that the problem Π-VS is uniformly FPT parameterized by k if Π is MSO_2-definable and has bounded treewidth. Since every minor-closed graph class is MSO_2-definable, this improves the result from Proposition 2 for graph classes of bounded treewidth.

Eppstein et al. [10] showed that the problem of deciding whether a given graph G can be turned into a graph of class Π by splitting each vertex of G at most k times can be expressed as an MSO_2 formula on G, if Π itself is MSO_2-definable. Using Courcelle's Theorem [6], this yields an FPT-algorithm parameterized by k and the treewidth of the input graph. Their algorithm exploits the fact that the split operations create at most k copies of each vertex in the graph. Since the same also applies for the problem Π-VS, where we may apply at most k splits overall, their algorithm can be straightforwardly adapted for Π-VS, thereby implying the following result.

Corollary 1. *For every MSO_2-definable graph class Π, the problem Π-VS is FPT parameterized by the solution size k and the treewidth of the input graph.*

For a graph class Π of bounded treewidth, we let $\mathrm{tw}(\Pi)$ denote the maximum treewidth among all graphs in Π. With the following lemma, we show that, if the target graph class Π has bounded treewidth, then every yes-instance of Π-VS must also have bounded treewidth.

Proposition 3. *For a graph class Π of bounded treewidth, let $\mathcal{I} = (G, S, k)$ be an instance of Π-VS. If $\mathrm{tw}(G) > k + \mathrm{tw}(\Pi)$, then $\mathcal{I}$ is a no-instance.*

Proof. We first show that a single split operation can reduce the treewidth of G by at most 1. Assume, for the sake of contradiction, that we can obtain a graph G' of treewidth less than $\mathrm{tw}(G)-1$ by splitting a single vertex v of G into vertices v_1 and v_2 of G'. Let $\mathcal{T}$ denote a minimum tree decomposition of G'. Remove all occurences of v_1 and v_2 in $\mathcal{T}$ and add v to every bag of $\mathcal{T}$. Observe that the result is a tree decomposition of size less than $\mathrm{tw}(G)$ for G, a contradiction. A single split operation thus decreases the treewidth of the graph by at most 1. Since every graph $G' \in \Pi$ has $\mathrm{tw}(G') \leq \mathrm{tw}(\Pi)$, it is thus impossible to obtain a graph of Π with at most k vertex splits if $\mathrm{tw}(G) > k + \mathrm{tw}(\Pi)$. $\square$

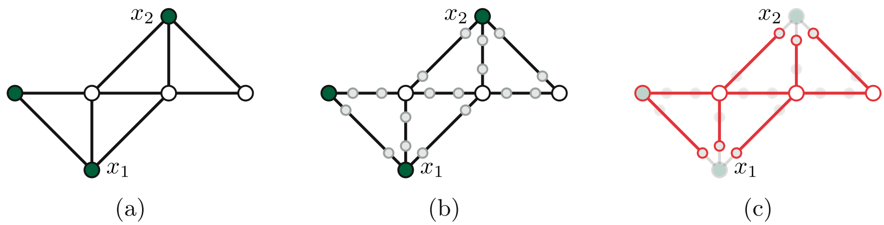


Fig. 6. (a) An instance $(G, S, 2)$ of Π-VE. (b) The corresponding auxiliary graph $G^\times$ obtained by subdividing each edge in G twice. (c) The graph obtained by exploding $\{x_1, x_2\}$ in G is the highlighted minor of $G^\times$. Since Π is MSO_2-definable, one can express Π-VE using an MSO_2 formula on $G^\times$.

Given a graph class Π of bounded treewidth, we first determine in time $f(k + \text{tw}(\Pi)) \cdot n$ whether the treewidth of G is greater than $k + \text{tw}(\Pi)$ [5]. If this is the case, then we can immediately report a no-instance by Proposition 3. Otherwise, we know that $\text{tw}(G) \leq k + \text{tw}(\Pi)$. Since $\text{tw}(\Pi)$ is a constant, we have $\text{tw}(G) \in O(k)$, and thus Corollary 1 yields the following result.

Theorem 5. *For every MSO_2-definable graph class Π of bounded treewidth, the problem Π-VS is FPT parameterized by the solution size k.*

Vertex Explosion. We now briefly sketch how these results extend to the problem variant Π Vertex Explosion(Π-VE) using vertex explosions instead of vertex splits. In this case, for minor-closed graph classes Π, the set of yes-instances of Π-VE is not minor-closed in general, thus the non-uniform FPT algorithm used to obtain Proposition 2 does not work for Π-VE. Additionally, the FPT-algorithm by Eppstein et al. [10] for MSO_2-definable graph classes cannot be straightforwardly adapted for Π-VE, since the number of new vertices resulting from explosions is not bounded by a function in k. However, using the approach illustrated in Fig. 6, we obtain the following results.

Lemma 4 ($\star$). *For every MSO_2-definable graph class Π, the problem Π-VE is FPT parameterized by the treewidth of the input graph.*

Theorem 6 ($\star$). *For every MSO_2-definable graph class Π of bounded treewidth, the problem Π-VE is FPT parameterized by the solution size k.*

We remark that, for arbitrary graph classes Π, the question whether a graph of Π can be obtained by applying arbitrarily many vertex splits to at most k vertices in the input graph is not equivalent to Π-VE.

6 Conclusion

In this work, we studied the problems Pathwidth-One Vertex Explosion and Pathwidth-One Vertex Splitting, obtaining an efficient branching algorithm and a small kernel for each variant. Subsequently, we more generally considered the problem of obtaining a graph of a specific graph class Π using at most k vertex splits (respectively explosions). For MSO_2-definable graph classes Π of bounded treewidth, we obtained an FPT algorithm parameterized by the solution size k. These graph classes include, for example, the outerplanar graphs, the pseudoforests, and the graphs of treewidth (respectively pathwidth) at most c for some constant c.

Instead of splitting vertices to obtain a graph of pathwidth at most 1, one can also consider obtaining graphs of treewidth at most 1, i.e., forests. Since, in this context, the degree-1 vertices resulting from an explosion can simply be reduced, the explosion model is equivalent to the problem Feedback Vertex Set, a well-studied NP-complete [14] problem that admits a quadratic kernel [22]. In the full version of this paper [4], we show that the problem of splitting vertices of a graph to obtain a forest is equivalent to the problem Feedback Edge Set, which asks whether a given graph can be made acyclic using at most k edge deletions; a problem that can be solved by computing an arbitrary spanning forest of the graph. Firbas [12] independently obtained the same result.

References

1. Ahmed, R., et al.: Splitting vertices in 2-layer graph drawings. IEEE Comput. Graph. Appl. **43**(3), 24–35 (2023). https://doi.org/10.1109/MCG.2023.3264244
2. Ahmed, R., Kobourov, S.G., Kryven, M.: An FPT algorithm for bipartite vertex splitting. In: Angelini, P., von Hanxleden, R. (eds.) Graph Drawing and Network Visualization - 30th International Symposium, GD 2022. LNCS, vol. 13764, pp. 261–268. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-22203-0_19
3. Arnborg, S., Proskurowski, A., Seese, D.: Monadic second order logic, tree automata and forbidden minors. In: Börger, E., Kleine Büning, H., Richter, M.M., Schönfeld, W. (eds.) CSL 1990. LNCS, vol. 533, pp. 1–16. Springer, Heidelberg (1991). https://doi.org/10.1007/3-540-54487-9_49
4. Baumann, J., Pfretzschner, M., Rutter, I.: Parameterized complexity of vertex splitting to pathwidth at most 1. CoRR abs/2302.14725 (2023). https://doi.org/10.48550/arXiv.2302.14725
5. Bodlaender, H.L.: A linear time algorithm for finding tree-decompositions of small treewidth. In: Kosaraju, S.R., Johnson, D.S., Aggarwal, A. (eds.) Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing, pp. 226–234. ACM (1993). https://doi.org/10.1145/167088.167161
6. Courcelle, B.: The monadic second-order logic of graphs. i. recognizable sets of finite graphs. Inf. Comput. **85**(1), 12–75 (1990). https://doi.org/10.1016/0890-5401(90)90043-H
7. Cygan, M.: Parameterized Algorithms. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-21275-3
8. Cygan, M., Pilipczuk, M., Pilipczuk, M., Wojtaszczyk, J.O.: An improved FPT algorithm and a quadratic kernel for pathwidth one vertex deletion. Algorithmica **64**(1), 170–188 (2012). https://doi.org/10.1007/s00453-011-9578-2
9. Eades, P., McKay, B.D., Wormald, N.C.: On an edge crossing problem. In: Proceedings of the 9th Australian Computer Science Conference, vol. 327, p. 334 (1986)

10. Eppstein, D., et al.: On the planar split thickness of graphs. Algorithmica **80**(3), 977–994 (2017). https://doi.org/10.1007/s00453-017-0328-y
11. Faria, L., de Figueiredo, C.M.H., Mendonça, C.F.X.: Splitting number is NP-complete. In: Hromkovič, J., Sýkora, O. (eds.) WG 1998. LNCS, vol. 1517, pp. 285–297. Springer, Heidelberg (1998). https://doi.org/10.1007/10692760_23
12. Firbas, A.: Establishing Hereditary Graph Properties via Vertex Splitting. Diploma thesis, Technische Universität Wien (2023). https://doi.org/10.34726/hss.2023.103864
13. Jansen, B.M.P., Lokshtanov, D., Saurabh, S.: A near-optimal planarization algorithm. In: Chekuri, C. (ed.) Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 1802–1811. SIAM (2014). https://doi.org/10.1137/1.9781611973402.130
14. Karp, R.M.: Reducibility among combinatorial problems. In: Miller, R.E., Thatcher, J.W. (eds.) Proceedings of a Symposium on the Complexity of Computer Computations, pp. 85–103. The IBM Research Symposia Series, Plenum Press, New York (1972). https://doi.org/10.1007/978-1-4684-2001-2_9
15. Kawarabayashi, K.: Planarity allowing few error vertices in linear time. In: 50th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2009, pp. 639–648. IEEE Computer Society (2009). https://doi.org/10.1109/FOCS.2009.45
16. Lewis, J.M., Yannakakis, M.: The node-deletion problem for hereditary properties is NP-complete. J. Comput. Syst. Sci. **20**(2), 219–230 (1980). https://doi.org/10.1016/0022-0000(80)90060-4
17. Marx, D., Schlotter, I.: Obtaining a planar graph by vertex deletion. In: Brandstädt, A., Kratsch, D., Müller, H. (eds.) WG 2007. LNCS, vol. 4769, pp. 292–303. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-74839-7_28
18. Nöllenburg, M., Sorge, M., Terziadis, S., Villedieu, A., Wu, H., Wulms, J.: Planarizing graphs and their drawings by vertex splitting. In: Angelini, P., von Hanxleden, R. (eds.) GD 2022. LNCS, vol. 13764, pp. 232–246. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-22203-0_17
19. Philip, G., Raman, V., Villanger, Y.: A quartic kernel for pathwidth-one vertex deletion. In: Thilikos, D.M. (ed.) WG 2010. LNCS, vol. 6410, pp. 196–207. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-16926-7_19
20. Purchase, H.C., Cohen, R.F., James, M.: Validating graph drawing aesthetics. In: Brandenburg, F.J. (ed.) GD 1995. LNCS, vol. 1027, pp. 435–446. Springer, Heidelberg (1996). https://doi.org/10.1007/BFb0021827
21. Robertson, N., Seymour, P.D.: Graph minors. XIII. The disjoint paths problem. J. Comb. Theory, Ser. B **63**(1), 65–110 (1995). https://doi.org/10.1006/jctb.1995.1006
22. Thomassé, S.: A $4k^2$ kernel for feedback vertex set. ACM Trans. Algorithms **6**(2), 32:1–32:8 (2010). https://doi.org/10.1145/1721837.1721848
23. Tsur, D.: Faster algorithm for pathwidth one vertex deletion. Theor. Comput. Sci. **921**, 63–74 (2022). https://doi.org/10.1016/j.tcs.2022.04.001