

A Framework for Parameterized Subexponential Algorithms for Generalized Cycle Hitting Problems on Planar Graphs*

Dániel Marx[†] Pranabendu Misra[‡] Daniel Neuen[§] Prafullkumar Tale[¶]

Abstract

Subexponential parameterized algorithms are known for a wide range of natural problems on planar graphs, but the techniques are usually highly problem specific. The goal of this paper is to introduce a framework for obtaining $n^{\mathcal{O}(\sqrt{k})}$ time algorithms for a family of graph modification problems that includes problems that can be seen as generalized cycle hitting problems.

Our starting point is the NODE UNIQUE LABEL COVER problem (that is, given a CSP instance where each constraint is a permutation of values on two variables, the task is to delete k variables to make the instance satisfiable). We introduce a variant of the problem where k vertices have to be deleted such that every 2-connected component of the remaining instance is satisfiable. Then we extend the problem with cardinality constraints that restrict the number of times a certain value can be used (globally or within a 2-connected component of the solution). We show that there is an $n^{\mathcal{O}(\sqrt{k})}$ time algorithm on planar graphs for any problem that can be formulated this way, which includes a large number of well-studied problems, for example, ODD CYCLE TRANSVERSAL, SUBSET FEEDBACK VERTEX SET, GROUP FEEDBACK VERTEX SET, SUBSET GROUP FEEDBACK VERTEX SET, VERTEX MULTIWAY CUT, and COMPONENT ORDER CONNECTIVITY.

For those problems that admit appropriate (quasi)polynomial kernels (that increase the parameter only linearly and preserve planarity), our results immediately imply $2^{\mathcal{O}(\sqrt{k} \cdot \text{polylog}(k))} \cdot n^{\mathcal{O}(1)}$ time parameterized algorithms on planar graphs. In particular, we use or adapt known kernelization results to obtain $2^{\mathcal{O}(\sqrt{k} \cdot \text{polylog}(k))} \cdot n^{\mathcal{O}(1)}$ time (randomized) algorithms for VERTEX MULTIWAY CUT, GROUP FEEDBACK VERTEX SET, and SUBSET FEEDBACK VERTEX SET.

Our algorithms are designed with possible generalization to H -minor free graphs in mind. To obtain the same $n^{\mathcal{O}(\sqrt{k})}$ time algorithms on H -minor free graphs, the only missing piece is the vertex version of a contraction decomposition theorem that we currently have only for planar graphs.

1 Introduction

For most NP-hard graph problems, restriction to planar graphs does not seem to impact complexity: they remain NP-hard. However, in many cases, planarity can be exploited to obtain improved algorithms compared to what is possible for general graphs. In terms of approximability, polynomial-time approximation schemes (PTAS) are known for the planar restrictions of many problems that are APX-hard (or worse) on general graphs [2–8, 10, 12, 21, 27, 28, 34, 35]. Planarity can decrease also the exponential time needed to compute exact solutions. The Exponential-Time Hypothesis (ETH) implies for many NP-hard problems that they cannot be solved in time $2^{\mathcal{O}(n)}$ on n -vertex graphs. An n -vertex planar graph has treewidth $\mathcal{O}(\sqrt{n})$ and there is a long list of problems that can be solved in time $2^{\tilde{\mathcal{O}}(t)} \cdot n^{\mathcal{O}(1)}$ on graphs with treewidth at most t (the $\tilde{\mathcal{O}}$ -notation hides polylogarithmic factors). Putting together these two results gives $2^{\tilde{\mathcal{O}}(\sqrt{n})}$ time algorithms on n -vertex planar graphs. This form of running time can be considered essentially optimal, as usually it can be shown that no $2^{o(\sqrt{n})}$ time algorithm exists, assuming the ETH. Therefore, the appearance of the square root in the exponent is the natural behavior of the running time for many problems.

*Research supported by the European Research Council (ERC) consolidator grant No. 725978 SYSTEMATICGRAPH. The full version of the paper can be accessed at <https://arxiv.org/abs/2110.15098>

[†]CISPA Helmholtz Center for Information Security, Germany.

[‡]Chennai Mathematical Institute, India.

[§]CISPA Helmholtz Center for Information Security, Germany.

[¶]CISPA Helmholtz Center for Information Security, Germany.

This “square root phenomenon” has been observed also in the context of parameterized problems: subexponential running times of the form $2^{\tilde{\mathcal{O}}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$ and $f(k) \cdot n^{\tilde{\mathcal{O}}(\sqrt{k})}$ are known for a wide range of problems, with matching lower bounds ruling out $2^{o(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$ and $f(k) \cdot n^{o(\sqrt{k})}$ time algorithms, assuming the ETH [11, 14, 23, 36, 37, 41–45]. The lower bounds usually follow a well-understood methodology: either they can be obtained from known planar NP-hardness proofs or they are based on grid-like W[1]-hardness proofs (see [13, Section 14.4.1]). On the other hand, there is no generic argument for obtaining subexponential running times with the square root in the exponent. As treewidth can be much larger than the parameter k (representing for example the size of the solution we are looking for), the fact that planar graphs have treewidth $\mathcal{O}(\sqrt{n})$ does not help us on its own. While there are certain algorithmic ideas that were used multiple times, the algorithms are highly problem-specific and they do not give a general understanding of why planar parameterized problems should have subexponential algorithms on planar graphs. In fact, there is an example of a natural problem, STEINER TREE, which was shown not to have $2^{o(k)} \cdot n^{\mathcal{O}(1)}$ time algorithms on planar graphs, assuming the ETH [43]. Thus we cannot take it for granted that every reasonable parameterized problem has subexponential algorithms on planar graphs.

There is a set of basic problems for which we can obtain subexponential parameterized algorithms using an argument called *bidimensionality* [15, 17, 18, 22, 24–26]. Let us consider for example the FEEDBACK VERTEX SET problem, where given a graph G and an integer k , the task is to find a set Z of at most k vertices that hits all the cycles, that is, $G - Z$ is a forest. It is known that if a planar graph G has treewidth t , then G contains an $\Omega(t) \times \Omega(t)$ grid minor, which shows that there are $\Omega(t^2)$ vertex-disjoint cycles, requiring at least that many vertices for the solution. Therefore, there is a constant c such that if G has treewidth at least $c\sqrt{k}$, then we can safely answer “no”; otherwise, if treewidth is at most $c\sqrt{k}$, then a standard $2^{\tilde{\mathcal{O}}(t)} \cdot n^{\mathcal{O}(1)}$ treewidth based algorithm achieves running time $2^{\tilde{\mathcal{O}}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$. There are a few other problems that can be handled by similar arguments (for example, INDEPENDENT SET, DOMINATING SET, LONGEST PATH, LONGEST CYCLE), but the technique is not very robust and does not extend to most generalizations of the problems. Two well-studied generalizations of FEEDBACK VERTEX SET are ODD CYCLE TRANSVERSAL (OCT, hit every cycle of odd length) and SUBSET FEEDBACK VERTEX SET (SFVS, hit every cycle that contains at least one vertex of the terminal set T). Now a large grid minor does not imply an immediate answer to the problem: it could be that every cycle in the grid minor is of even length or they do not contain any terminal vertices.

The main contribution of this paper is formulating a family of deletion-type problems and showing that each such problem can be solved in time $n^{\mathcal{O}(\sqrt{k})}$ on planar graphs. Our original motivation comes from cycle-hitting problems such as OCT and SFVS, but our framework is much wider than that: it also includes problems such as VERTEX MULTIWAY CUT and COMPONENT ORDER CONNECTIVITY (where the task is to delete k vertices such that every component has size at most t). A key feature of our framework is the ability to reason about problems that are defined in terms of the 2-connected components of the graph $G - Z$ resulting after the deletions. For example, SFVS can be defined as saying that in $G - Z$ every 2-connected component is either of size at most 2 or terminal free. ODD SUBSET FEEDBACK VERTEX SET, the common generalization of OCT and SFVS can be defined as saying that every 2-connected component is either bipartite or terminal free. The most novel technical ideas of our proofs are related to handling these constraints on 2-connected components.

It may look underwhelming that we are targeting running time $n^{\mathcal{O}(\sqrt{k})}$ for the above-mentioned problems: after all, they are known to be fixed-parameter tractable, many of them with $2^{\tilde{\mathcal{O}}(k)} \cdot n^{\mathcal{O}(1)}$ time algorithms, hence the natural expectation is that they can be solved in time $2^{\tilde{\mathcal{O}}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$ on planar graphs. However, let us observe that a polynomial kernelization together with an $n^{\mathcal{O}(\sqrt{k})}$ time algorithm delivers precisely this running time. A *polynomial kernelization* of problem Π is a polynomial-time algorithm that, given an instance x of Π with parameter k , produces an equivalent instance x' of Π with parameter k' such that $|x'| = |k'|^{\mathcal{O}(1)}$ and $k' = k^{\mathcal{O}(1)}$. In other words, the kernelization compresses the problem to a small instance x' whose size is polynomially bounded in the new parameter k' (hence also in the original parameter k). We need two additional properties of the kernelization for our applications. First, we require that the parameter be increased only linearly, that is, $k' = \mathcal{O}(k)$. Second, we need that if x is a planar instance, then x' is also planar. In general, a kernelization for a problem defined on general graphs does not need to keep planarity during the compression; therefore, we need a kernelization for the *planar version* of the problem. If we perform such a kernelization and then we solve the resulting instance (x', k') using our algorithm, then the running time is $|x'|^{\tilde{\mathcal{O}}(\sqrt{k'})} = k'^{\tilde{\mathcal{O}}(\sqrt{k})} = 2^{\tilde{\mathcal{O}}(\sqrt{k})}$, plus the polynomial running time of the kernelization. Thus our $n^{\mathcal{O}(\sqrt{k})}$ time algorithm can be seen as an important step

towards obtaining subexponential FPT algorithms. As we shall see, for some problems the required kernelization results already exist (or existing results can be adapted to preserve planarity), hence new subexponential FPT algorithms follow from our work.

In order to appreciate our technical contributions, it is useful to take the following perspective. We can classify the problems considered here into three groups, characterized by a distinctive set of tools required to obtain subexponential FPT algorithms. Below we briefly highlight the technical tools that become relevant, and then proceed with a more detailed explanation. The first technique is well known, for the second we need to put together known pieces and extend them, and we use a significantly novel approach for the third technique.

- **Technique 1: Bidimensionality.** As explained above, for some problems (e.g., FEEDBACK VERTEX SET), we can exploit the fact that the answer is trivial if there is a grid of size $\Omega(\sqrt{k}) \times \Omega(\sqrt{k})$, hence we may assume that treewidth is $\mathcal{O}(\sqrt{k})$. Then a $2^{\tilde{\mathcal{O}}(t)} \cdot n^{\mathcal{O}(1)}$ algorithm on graphs of treewidth t gives an algorithm with running time $2^{\tilde{\mathcal{O}}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$.
- **Technique 2: Kernelization, contraction decomposition, guessing.** A contraction decomposition theorem can be used to find a set A that has intersection $\mathcal{O}(\sqrt{k})$ with the solution Z . After guessing this intersection $A \cap Z$, we can remove it from the graph. We know that each component of $A \setminus Z$ is disjoint from the solution and in some problems such as OCT, the component can be contracted and represented by a single vertex (after appropriate modifications). If the contraction decomposition theorem ensures that the contracted graph has treewidth $\mathcal{O}(\sqrt{k})$, then the procedure involves guessing $n^{\mathcal{O}(\sqrt{k})}$ possibilities, followed by a $2^{\tilde{\mathcal{O}}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$ time treewidth algorithm. Together with a kernelization, this is sufficient for our purposes. For edge-deletion problems, the required contraction decomposition theorems are readily available. While the technique is simple and may be implicit in earlier work, our main technical contribution for this group of problems is precisely formulating the required contraction decomposition theorem for vertex-deletion problems and proving it for planar graphs.
- **Technique 3: Guessing the 2-connected components.** The technique of solving OCT sketched above essentially relied on the assumption that if a set of edges is disjoint from the solution, then they can be contracted and represented by a single vertex. For problems that are defined through the 2-connected components of the remaining graph $G - Z$, it is not clear how such a contraction could be performed, as it could significantly change the structure of 2-connected components. Nevertheless, our main contribution for this group of problems is a way of making the approach via contraction work. If a connected set I of vertices is disjoint from the solution Z , then the vertices of I are contained in some number of 2-connected components of $G - Z$. We show that we can identify polynomially many possibilities for the union B of all (but one) of these components. Then we can contract I to a vertex v_I and implement the problem as a binary constraint satisfaction (CSP) instance on the contract graph. The variable v_I stores the correct choice of this union B and constraints ensure that this choice is consistent with the 2-connected structure of $G - Z$.

To formally state our main technical results, we need to introduce the framework of constraint satisfaction problems (CSPs). A *binary CSP* (or *2-CSP*) instance consists of a set X of variables, a domain D , and a set of constraints on two variables. A constraint $((x_1, x_2), R)$ requires that in a satisfying assignment $\alpha : X \rightarrow D$, we have $(\alpha(x_1), \alpha(x_2)) \in R$. We say that a relation R is a *permutation* if for every $x \in D$ there is at most one $y_1 \in D$ with $(y_1, x) \in R$ and at most one $y_2 \in D$ with $(x, y_2) \in R$. *Permutation CSP* is an instance where every relation is a permutation. Given a CSP instance the *constraint graph* or *primal graph* is the graph with vertex set X where two vertices are adjacent if there is a constraint on them. With some abuse of notation, we often identify the variables and constraints with the vertices and edges of the constraint graph, respectively.

The most basic question about CSP is satisfiability. Observe that this is trivial for a permutation CSP, as the value of a variable uniquely determines the values of the neighboring variables. However, for unsatisfiable instances, maximizing the number of satisfied constraints or minimizing the number of unsatisfied constraints is still a non-trivial task. We can also define these optimization problems using the language of deletion problems, which is more natural in our applications. In the (PERM) CSP EDGE DELETION problem, we need to make a (permutation) CSP instance satisfiable by removing at most k constraints. This problem is often called the UNIQUE LABEL COVER problem in the approximation algorithms literature [1, 32, 33, 40]. In (PERM) CSP

VERTEX DELETION, the instance needs to be made satisfiable by removing at most k variables (together with all the constraints appearing on them); this problem is also called NODE UNIQUE LABEL COVER. Given a tree decomposition of width at most t of the constraint graph, these CSP deletion problems can be solved in time $|D|^{\mathcal{O}(t)} \cdot n^{\mathcal{O}(1)}$ using standard dynamic programming techniques. This algorithm will be a basic tool in our results.

To express a wider range of problems, we augment the CSP deletion problems with size constraints. Let G be the constraint graph and let Z be the set of edges or vertices that we delete from the instance. Let C be a connected component of the constraint graph after the removal of Z . A size constraint may restrict the size of such a component, or the number of certain types of vertices that appear in a component, or the number of times certain values can appear in the component, etc. We define size constraints in the following very general way. Let $w : X \times D \rightarrow \mathbb{N}$ be a function and let $q \in \mathbb{N}$. Then an assignment $\alpha : X \rightarrow D$ satisfies the \leq -constraint (w, q) on component C if $\sum_{x \in C} w(x, \alpha(x)) \leq q$. We say that the constraint is Q -bounded if $q \leq Q$. A \geq -constraint (w, q) is defined similarly. A CSP deletion instance can be augmented with more than one size constraint, in particular, a \leq - and a \geq - constraint together can express equality.

Our first main technical result considers permutation CSP deletion problems and formalizes Technique 2 described above.

THEOREM 1.1. *PERM CSP EDGE DELETION and PERM CSP VERTEX DELETION with a set \mathcal{S} of Q -bounded \leq - or \geq -constraints can be solved in time $(|X| + |D| + Q^{|\mathcal{S}|})^{\mathcal{O}(\sqrt{k})}$ if the constraint graph is planar.*

By simple reductions, we can obtain the following corollaries as example applications:

COROLLARY 1.1. *The following problems can be solved in time $n^{\mathcal{O}(\sqrt{k})}$ on planar graphs: OCT, GROUP FEEDBACK VERTEX SET (where the group is part of the input), VERTEX MULTIWAY CUT WITH UNDELETABLE TERMINALS, VERTEX MULTIWAY CUT WITH DELETABLE TERMINALS, COMPONENT ORDER CONNECTIVITY (where t is part of the input), and the edge-deletion versions of all these problems.*

As mentioned earlier, if appropriate kernels are available, then Corollary 1.1 implies subexponential FPT algorithms on planar graphs. In particular, the kernelization needs to preserve planarity, which is not necessarily a goal in kernelization algorithms designed to work on general graphs. We use/adapt (quasi)polynomial (randomized) kernels from the literature to obtain subexponential FPT algorithms for the following problems (see Section 3.3 for more details):

COROLLARY 1.2. *The following problems can be solved in randomized time $2^{\tilde{\mathcal{O}}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$ on planar graphs: ODD CYCLE TRANSVERSAL, EDGE MULTIWAY CUT, VERTEX MULTIWAY CUT, GROUP FEEDBACK VERTEX SET (for a fixed group), GROUP FEEDBACK EDGE SET (where the size of the group is $k^{\mathcal{O}(1)}$).*

Note that a subexponential FPT algorithm is known for OCT on planar graphs, but it uses more problem specific arguments [41]. In particular, it is not based on a polynomial kernel for OCT, which was not known when [41] was published.

Let us turn now our attention towards Technique 3, which allows us to handle problems defined by properties of the 2-connected components of the graph. In order to model such problems, we introduce first the 2-connected version of the permutation CSP deletion problem. In 2CONN PERM CSP EDGE/VERTEX DELETION, we need to delete at most k constraints/variables such that the remaining instance is satisfiable on every 2-connected component of the constraint graph. Note that even if every 2-connected component is satisfiable, this does not mean that the whole instance is satisfiable: a cut vertex appearing in 2-connected components C_1 and C_2 may need to receive different values in satisfying assignments of the two components. Similarly to PERM CSP EDGE/VERTEX DELETION, we augment the problem with size constraints, but now the constraints restrict the value of $\sum_{v \in C} w(v, \alpha(v))$ on each 2-connected component C and for technical reasons we allow only \leq -constraints. The main technical result is the following:

THEOREM 1.2. *2CONN PERM CSP EDGE DELETION and 2CONN PERM CSP VERTEX DELETION with a set \mathcal{S} of Q -bounded \leq -constraints can be solved in time $\mathcal{O}(|X| + |D| + Q^{|\mathcal{S}|})^{\mathcal{O}(\sqrt{k})}$ if the constraint graph is planar.*

Again by simple reductions, we can obtain the following corollaries:

COROLLARY 1.3. *The following problems can be solved in time $n^{\mathcal{O}(\sqrt{k})}$ on planar graphs: SUBSET FEEDBACK VERTEX SET, TWO SUBSET FEEDBACK VERTEX SET, SUBSET OCT, SUBSET GROUP FEEDBACK VERTEX SET (where the group is part of the input), 2CONN COMPONENT ORDER CONNECTIVITY (where t is part of the input), and the edge-deletion versions of all these problems.*

We can adapt the SUBSET FEEDBACK VERTEX SET kernel of Hols and Kratsch [30] to preserve planarity, yielding a subexponential FPT algorithm when combined with Corollary 1.3.

THEOREM 1.3. *SUBSET FEEDBACK VERTEX SET can be solved in randomized time $2^{\tilde{\mathcal{O}}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$ on planar graphs.*

Currently no polynomial kernel is known for SUBSET ODD CYCLE TRANSVERSAL. Note that this problem is a common generalization of OCT and SUBSET FEEDBACK VERTEX SET. We could use such a (planarity preserving) kernel and Corollary 1.3 to obtain a subexponential FPT algorithm for SUBSET ODD CYCLE TRANSVERSAL on planar graphs, which would be a common generalization of the subexponential FPT algorithms for OCT and SUBSET FEEDBACK VERTEX SET.

We would like to emphasize that our framework could be applied to a much wider range of problems than those explicitly stated in Corollaries 1.1 and 1.3. For example, we obtain $n^{\mathcal{O}(\sqrt{k})}$ time algorithms on planar graphs for problems of the type “delete k vertices such that every 2-connected component is either (1) bipartite or (2) contains at most 3 red and at most 5 blue terminals.” A useful feature of our framework is that if two properties can be expressed as the specified permutation CSP with size constraints, then the OR of the properties can be expressed as well (by a permutation CSP on the union of the two domains) and also the AND of the properties (by a permutation CSP on the product of the two domains). However, these results give subexponential FPT algorithms only if suitable polynomial kernels are also available. Our work decouples the question of kernels (which is an interesting question on its own right) from the question of obtaining subexponential running time in planar graphs.

1.1 Technical Overview In this section, we give an intuitive overview of the main technical ideas for subexponential FPT algorithms that will be developed in detail in later sections.

Bidimensionality. Earlier we briefly recalled how bidimensionality can be used to obtain subexponential parameterized algorithms on planar graphs for some problems. As our results concern problems where bidimensionality is not applicable, we do not want to elaborate further on this technique. The following bidimensional problems are listed only to contrast them with later problems that do not have this property.

- **FEEDBACK VERTEX SET** (remove k vertices to make the graph acyclic). As we have seen, the existence of a $c\sqrt{k} \times c\sqrt{k}$ grid implies that there is no solution of size k .
- **EVEN CYCLE TRANSVERSAL** (remove k vertices to destroy every even cycle). It can be shown that a $c\sqrt{k} \times c\sqrt{k}$ grid contains k vertex-disjoint even cycles, hence implying that there is no solution of size k .
- $\geq \ell$ -**CYCLE TRANSVERSAL** (remove k vertices to destroy every cycle of length at least ℓ). Again, for fixed ℓ , a $c_\ell\sqrt{k} \times c_\ell\sqrt{k}$ grid minor implies that there is no solution of size k ,
- **COMPONENT ORDER CONNECTIVITY** (remove k vertices such that every component has size at most t). For fixed t , a $c_t\sqrt{k} \times c_t\sqrt{k}$ grid minor implies that there is no solution of size k .
- **2CONN COMPONENT ORDER CONNECTIVITY** (remove k vertices such that every 2-connected component has size at most t). For fixed t , a $c_t\sqrt{k} \times c_t\sqrt{k}$ grid minor implies that there is no solution of size k .

We remark that if ℓ is part of the input, then $\geq \ell$ -CYCLE TRANSVERSAL is NP-hard even for $k = 0$, as it contains the HAMILTONIAN CYCLE problem. For (2CONN) COMPONENT ORDER CONNECTIVITY, if t is part of the input, then it becomes W[1]-hard (see the full version for details), but our techniques show that they can be solved in time $n^{\mathcal{O}(\sqrt{k})}$.

Edge-Deletion Problems. The edge-deletion version of OCT (delete k edges to make the graph bipartite, also called EDGE BIPARTIZATION) will be a convenient example to explain the main ideas behind the $n^{\mathcal{O}(\sqrt{k})}$ time algorithm following from Technique 2. For the clean treatment of the problem, let us introduce the following two minor extensions:

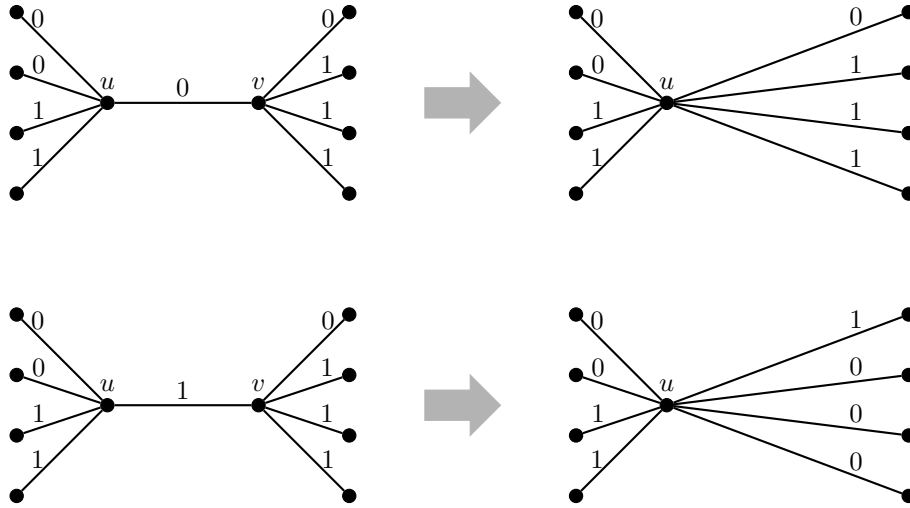


Figure 1: Contracting the undeletable edge uv in the EDGE BIPARTIZATION problem. The numbers on the edges show their parity.

- The input contains a set $U \subseteq E(G)$ of *undeletable edges* and the solution $Z \subseteq E(G)$ has to be disjoint from U .
- The edges are of two types, odd and even, and the task is to destroy every cycle whose total parity is odd.

Our first observation is that if we have an undeletable edge uv , then we can simplify the instance: let us remove v , and for every edge vw , let us introduce an edge uw whose parity is the parity of uv plus the parity of vw (see Figure 1). It is not difficult to observe that this does not change the problem: for any closed walk going through u in the new graph, there is a corresponding closed walk with the same parity in the original graph.

The second key step is the use of the following contraction decomposition theorem:

THEOREM 1.4. ([19, 20, 34, 35]) *Let G be a planar graph and $\ell \geq 1$. In polynomial-time, we can find a partition of the edge set $E(G) = E_1 \uplus E_2 \uplus \dots \uplus E_\ell$ such that, for every $i \in [\ell]$, it holds that*

$$\text{tw}(G/E_i) = \mathcal{O}(\ell).$$

Let us invoke the algorithm of Theorem 1.4 with $\ell = \sqrt{k}$ and let $Z \subseteq E(G)$ be a hypothetical solution of size at most k . Then for some $i \in [\ell]$, we have $|Z \cap E_i| \leq \sqrt{k}$. Let us guess this value of i (we have \sqrt{k} possibilities) and let us guess the set of edges $Z_i = Z \cap E_i$ (we have $n^{\mathcal{O}(\sqrt{k})}$ possibilities). Now we can remove Z_i from the instance and mark every edge in $E_i \setminus Z$ as undeletable. Then, as described above, we can modify the instance by contracting every edge in $E_i \setminus Z$. We argue that the resulting graph $G' = G/(E_i \setminus Z)$ has treewidth $\mathcal{O}(\sqrt{k})$. By Theorem 1.4, the graph G/E_i has treewidth $\mathcal{O}(\sqrt{k})$ and G/E_i can be obtained from $G/(E_i \setminus Z_i)$ by $|Z_i|$ further contractions. As contracting an edge can decrease treewidth only by at most 1 and $|Z_i| \leq \sqrt{k}$, it follows that $\text{tw}(G/(E_i \setminus Z))$ has treewidth $\mathcal{O}(\sqrt{k})$. Therefore, using known $2^{\mathcal{O}(\text{tw}(G))} \cdot n^{\mathcal{O}(1)}$ time algorithms, we can solve the resulting equivalent instance on G' in time $2^{\mathcal{O}(\sqrt{k})} \cdot n^{\mathcal{O}(1)}$.

Vertex-Deletion Problems. We can try to handle vertex-deletion problems such as OCT in a similar way, but we need to overcome some technical difficulties. Again, we can extend the problem with parities of the edges and a set $U \subseteq E(G)$ of undeletable edges, meaning that if $uv \in U$, then neither endpoint of uv can be deleted. Then we would need a vertex-partition version of a contraction decomposition, where we contract each connected component of a vertex set V_i into a single vertex (which is the same as saying that we contract every edge induced by V_i).

THEOREM 1.5. *Let G be a planar graph and $\ell \geq 1$. In polynomial time, we can find a partition of the vertex set $V(G) = V_1 \uplus V_2 \uplus \dots \uplus V_\ell$ such that, for every $i \in [\ell]$, it holds that*

$$\text{tw}(G/E(G[V_i])) = \mathcal{O}(\ell).$$

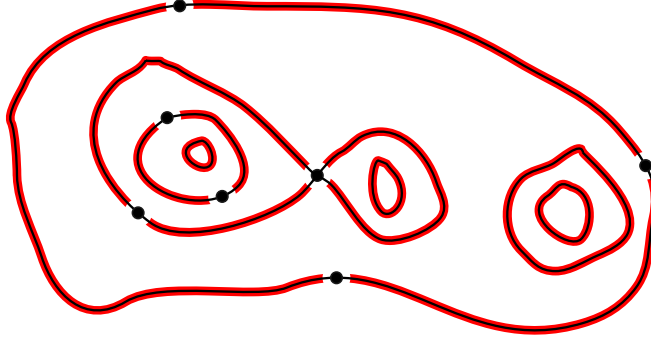


Figure 2: The 7 vertices of $Z \cap V_i$ split V_i into 12 segments.

Now we can proceed similarly as in the edge version: we guess an i where $Z_i := V_i \cap Z$ has size at most \sqrt{k} , guess Z_i , and mark every edge in $G[V_i \setminus Z_i]$ as undeletable. Then we can again contract these edges and arrive to an instance on $G' = G/E(G[V_i \setminus Z_i])$. We would need to connect the treewidth of G' with the treewidth of $G/E(G[V_i])$, but this is not as obvious as in the edge-deletion case. We prove the following extension of Theorem 1.5 that makes the connection apparent. Intuitively, we can think of V_i as collection of some kind of nested layers (see Figure 2). Accordingly, we call each component I of $V_i \setminus Z$ a *segment*.

THEOREM 1.6. *Let G be a planar graph and $\ell \geq 1$. In polynomial time, we can find a partition of the vertex set $V(G) = V_1 \uplus V_2 \uplus \dots \uplus V_\ell$ such that, for every $i \in [\ell]$ and every $W \subseteq V_i$, it holds that*

$$\text{tw}(G/E(G[V_i \setminus W])) = \mathcal{O}(\ell + |W|).$$

This extended stronger bound on treewidth allows the algorithm to go through for the vertex-deletion case without any difficulty. To make this formal, the following corollary combines Theorem 1.6 with the guessing of i and $V_i \cap Z$.

COROLLARY 1.4. *Let G be a planar graph and $k \geq 1$. In time $n^{\mathcal{O}(\sqrt{k})}$, we can find a sequence $V_1, \dots, V_h \subseteq V(G)$ of sets of vertices with $h = n^{\mathcal{O}(\sqrt{k})}$ such that for every $Z \subseteq V(G)$ with $|Z| \leq k$, the following holds for at least one $1 \leq i \leq h$:*

1. $V_i \cap Z = \emptyset$, and
2. $\text{tw}(G/E(G[V_i])) = \mathcal{O}(\sqrt{k})$.

That is, by trying every V_i , we are guaranteed to find one that can be safely contracted (as it is disjoint from the solution) and the contracted graph has treewidth $\mathcal{O}(\sqrt{k})$.

Permutation CSPs. The arguments presented above can be generalized to every CSP deletion problem with permutation constraints, that is, for PERM CSP VERTEX DELETION. There are two problem-specific points in the algorithm:

- (1) We used a $2^{\mathcal{O}(t)} \cdot n^{\mathcal{O}(1)}$ time algorithm on graphs of treewidth t .
- (2) We need a way of contracting the vertices in a component C of undeletable edges to a single vertex.

Condition (1) certainly holds for permutation CSP deletion problems over a fixed domain D : using standard dynamic programming techniques, we can find a minimum set of deletions that make the instance satisfiable. For (2), let us observe that if C is a set of variables that are connected by an undeletable set of edges, then knowing the value of any variable $x \in C$ allows us to deduce the value of any other variable $y \in C$: the value of a variable uniquely determines the value of all its neighbors. Therefore, we can represent all the variables in C by one of the variables, say x , and every constraint involving some $y \in C$ can be replaced by an equivalent constraint involving x . Therefore, we get $n^{\mathcal{O}(\sqrt{k})}$ time algorithms for PERM CSP EDGE/VERTEX DELETION on planar graphs the same way as we get such algorithms for EDGE BIPARTIZATION and OCT.

Size Constraints. Our goal is to extend the permutation CSP deletion problems with size constraints that restrict the number of appearances of certain values in *each connected component* of $G - Z$. We need to check if (1) and (2) above still holds if we extend the problem with a set \mathcal{S} of \leq - or \geq -constraints, as defined earlier in the introduction. For (1), it is routine to extend a dynamic programming algorithm over a tree decomposition to enforce that a size constraint holds *globally* for $G - Z$. If there are $|\mathcal{S}|$ size constraints, and each size constraint bounds a value up to Q , then this adds a factor of $Q^{|\mathcal{S}|}$ overhead to the number of states of the dynamic programming table. However, we need to ensure that the size constraints hold for each component separately, and we cannot bound the number of components of $G - Z$. But from the viewpoint of dynamic programming, all that matters is that in a tree decomposition of width t , every bag can intersect $t + 1$ of the components of $G - Z$. Therefore, it is sufficient to extend the dynamic programming with counters that keep track of each size constraint in at most $t + 1$ components. Therefore, the $|\mathcal{S}|$ size constraints add only a factor of $Q^{\mathcal{O}(t|\mathcal{S}|)}$ to the running time.

For (2), let us first observe that every variable of C is in the same component of $G - Z$. Therefore, knowing the value of one variable in C allows us to tell how the variables of C contribute to the size constraint. Therefore, when contracting C to a single representative vertex, we can define the size constraint on that variable in a way that faithfully represents the contribution of C to the size constraints.

2-Connected Components. To explain Technique 3, let us consider for example the SUBSET FEEDBACK VERTEX SET problem, where we require that every 2-component of $G - Z$ be either of size at most 2, or terminal free. Now if I is a segment of undeletable edges, then contracting I is highly problematic, as this could change the 2-connected components of $G - Z$ (see Figure 3). For example, in the figure the color of the rectangles represent the 2-connected components where they belong, but after the contraction of I it is no longer possible to recover that rectangles of different color are supposed to be in different 2-connected components. Nevertheless, we manage to do the seemingly impossible: given a set V_i of Corollary 1.4, we represent the problem as a CSP instance whose constraint graph is $G/E(G[V_i])$ and hence has treewidth $\mathcal{O}(\sqrt{k})$.

A key idea in our handling of this issue is that guessing of a small number of vertices allows us to recover all (but one) of the 2-connected components visited by I in $G - Z$. For example, in the figure, if we know that a, b, c are precisely the vertices of Z that are adjacent to these components, and u, v, w are precisely the cut vertices where these components are connected to the rest of the graph, then this information is sufficient to recover all the 2-connected components visited by I in $G - Z$. If we can argue that there is only a constant number of such vertices that are important to localizing the 2-connected components, then there is only polynomially many possibilities for the 2-connected components where I appears in $G - Z$. Our main technical contribution is showing that (after some preprocessing and careful choice of V_i in Theorem 1.6), we may assume that indeed there is only a constant number of such vertices for every segment I of $V_i \setminus Z$.

Let us make these ideas more formal. Let us consider the decomposition of $G - Z$ into 2-connected components; we may assume that this is a rooted decomposition (see Figure 4). Let I be a set of vertices such that $G[I]$ is connected, and let us consider the 2-connected components that contain at least one vertex of I . There is a unique such component closest to the root, which we call the *root* $r(I)$ of I . The union of the vertices of every other 2-connected component intersected by I is called the *body* $B(I)$ of I (it is possible that $B(I) = \emptyset$). Note that a 2-connected component of $G - Z$ can be the root of multiple segments, can be both the root of some segment and the body of some other, but it can be in the body of only a single component: if a 2-connected component C is in the body of I , then I contains the vertex of C joining it with the parent component. The following lemma can be seen as an extension of Corollary 1.4: for each V_i , we also produce a collection \mathcal{B}_i of sets that are possible candidates for the bodies of the components of $G[V_i]$.

LEMMA 1.1. *Given a planar graph and an integer k , we can compute in time $n^{\mathcal{O}(\sqrt{k})}$ a sequence $(V_1, \mathcal{B}_1), \dots, (V_h, \mathcal{B}_h)$ with $h = n^{\mathcal{O}(\sqrt{k})}$, where $V_i \subseteq V(G)$ and \mathcal{B}_i contains $n^{\mathcal{O}(1)}$ sets of vertices, such that the following holds. For every $Z \subseteq V(G)$ with $|Z| \leq k$, there is at least one $1 \leq i \leq h$ such that*

1. $V_i \cap Z = \emptyset$,
2. $\text{tw}(G/E(G[V_i])) = \mathcal{O}(\sqrt{k})$, and
3. for every component I of $G[V_i]$, the body of I in $G - Z$ is a set from \mathcal{B}_i .

We remark that it seems unavoidable that only the body of I appears in Lemma 1.1 and not the union of every 2-connected component visited by I . The reason is that it may happen that $G - Z$ is 2-connected, in which case

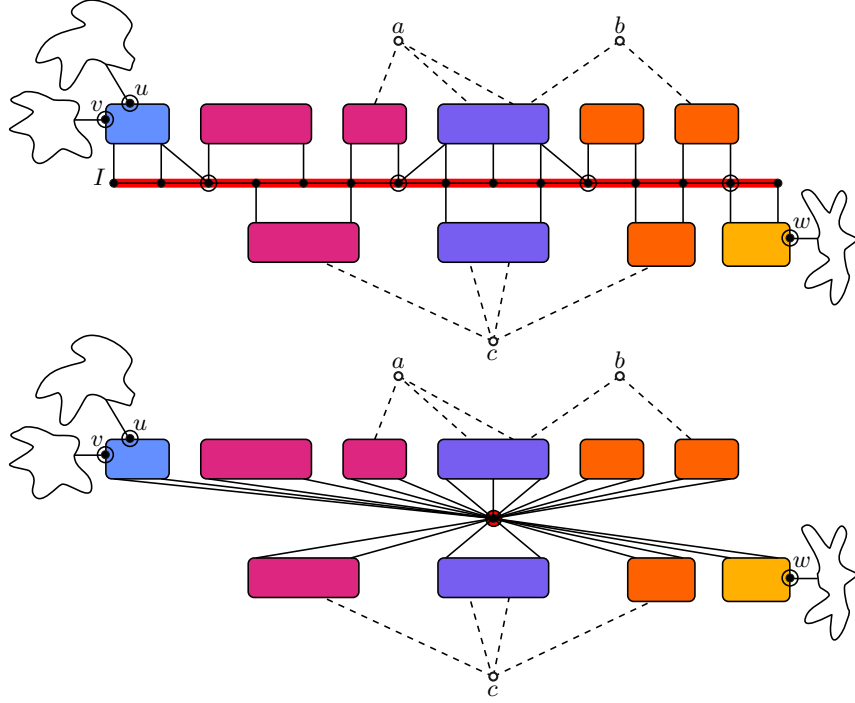


Figure 3: Top: After removing $Z = \{a, b, c\}$, segment I is contained in 5 different 2-connected components of $G - Z$, as shown by the colors. Cut vertices are circled black dots. In particular, cut vertices u, v, w connect the 2-connected components visited by I to the rest of the graph. Bottom: After contracting I , it is no longer possible to recognize these 2-connected components.

some collection \mathcal{B}_i would need to contain precisely the set $V(G) \setminus Z$. But it is certainly easy to construct an instance that has $n^{O(k)}$ solutions Z for which $G - Z$ is 2-connected. For such an instance, the output of Lemma 1.1 would need to contain all these $n^{O(k)}$ sets. The definition of body avoids this issue: if $G - Z$ is 2-connected, then the body of I is empty.

Representing a Hypothetical Solution. With Lemma 1.1 at hand, we can translate a 2CONN PERM CSP VERTEX DELETION instance to a CSP deletion problem the following way. We invoke Lemma 1.1, and try every $1 \leq i \leq h$. For a fixed i , we define a CSP instance whose constraint graph is $G' = \text{tw}(G/E(G[V_i]))$. We may assume that every set $B \in \mathcal{B}_i$ has the property that every 2-connected component induced by B is satisfiable in the permutation CSP instance: if not, then this B cannot be the body of any segment in the solution $G - Z$. Let us consider a hypothetical solution $G - Z$, the decomposition of $G - Z$ into 2-connected components, and satisfying assignments of the permutation CSP instances induced by each 2-connected component of $G - Z$. We describe how the variables can represent this solution. If variable v_I corresponds to the contraction of some component I of $G[V_i]$, then it needs to store the following pieces of information:

- the value of one of the variables in $r(I) \cap I$ in a satisfying assignment of $r(I)$ (which also determines the value of every other variable in $r(I) \cap I$), and
- the body $B \in \mathcal{B}_i$ of I in the hypothetical solution Z that we are looking for.

Note that v_I *does not* contain any information about the values of the variables of I outside $r(I)$. However, this information is not necessary if the guess of the body B is correct: by our assumption on \mathcal{B}_i , every 2-connected component induced by B is satisfiable in the permutation CSP. It is the job of the variables corresponding to the vertices of $B \setminus I$ to enforce that the guess of v_I about the body B is correct. This is only possible if the information stored at a variable $v \in B \setminus I$ is sufficient to determine whether v is in the same 2-connected component as a segment I , or more precisely, whether v is in the body of I . Therefore, if $v \notin V_i$, then v stores the following pieces

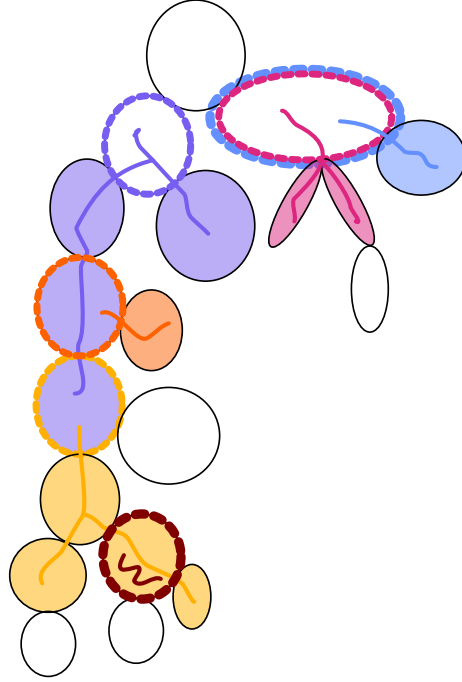


Figure 4: The rooted decomposition of the 2-connected components of $G - Z$ with 6 segments (shown by colored lines). The shading shows the bodies of the segments, the root components are shown by highlighted boundaries.

of information:

- The value of v in the satisfying assignment of the 2-connected component where v appears.¹
- The segment I whose body contains v (if exists).
- The body B of this segment I (if exists).
- The cut vertex that joins the 2-connected component of v to its parent component.
- The level of the 2-connected component of v in the rooted decomposition into 2-connected components.

We show that if the variables store all this information, then binary constraints can ensure that the stored information consistently describe the 2-connected structure of the instance and ensure the correctness of the hypothetical solution Z . Therefore, we can solve the 2CONN PERM CSP VERTEX DELETION instance by checking whether the constructed CSP instance can be made satisfiable by the removal of k variables. As the primal graph of this instance is $G/E(G[V_i])$, Lemma 1.1 guarantees that it has treewidth $\mathcal{O}(\sqrt{k})$, leading to an $n^{\mathcal{O}(\sqrt{k})}$ time algorithm.

There is a technical detail, which will become important when introducing size constraints. We said that the values of the variables in the CSP instance describe a decomposition of $G - Z$ into 2-connected components. However, this decomposition may be coarser than the actual decomposition into 2-connected components. Fortunately, this does not cause any problem for 2CONN PERM CSP VERTEX DELETION: if the instance is satisfiable on every component of the coarser decomposition, then it follows that it is satisfiable on every subset of each such component.

¹This is not well defined for cut vertices, which appear in more than one 2-connected component. Further tricks are needed to handle the value of cut vertices.

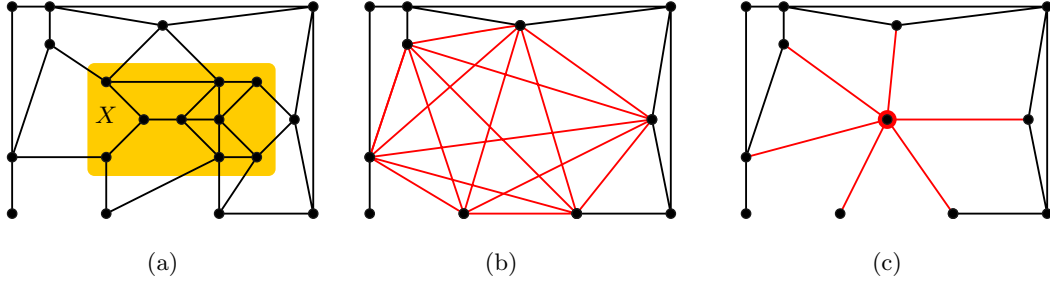


Figure 5: (a) A set X of vertices that is disjoint from the solution. (b) *Torso*: removing X and adding a clique on $N(X)$. (c) *Contract to Undeletable*: set X is contracted to a single undeletable vertex.

Size Constraints for 2-Connected Components. In the 2CONN PERM CSP VERTEX DELETION problem, we want to introduce size constraints that restrict the appearance of the values in each 2-connected component. Given that the CSP instance constructed above can identify the 2-connected components of $G - Z$, we can augment the dynamic programming algorithm for CSP deletion with size constraints on the 2-connected components. The technical issue mentioned in the previous paragraph does not matter for \leq -constraints: if such constraints are satisfied for a superset of a 2-connected component, then they are satisfied for the component as well. However, because of this issue, we cannot introduce \geq -constraints 2CONN PERM CSP VERTEX DELETION. In particular, we do not have an $n^{\mathcal{O}(\sqrt{k})}$ time algorithm for the version of 2CONN COMPONENT ORDER CONNECTIVITY where each 2-connected component has to be of size exactly t , with t being part of the input.

Kernelization. There is a somewhat counterintuitive phenomenon when considering kernelization algorithms for restricted graph classes. Even though problem PLANAR II is a special case of problem II, a polynomial kernelization for II *does not* imply a polynomial kernelization for PLANAR II. The reason is that a kernelization algorithm for general graphs need not produce a planar output when the input is planar, thus it does not necessarily produce an instance of PLANAR II.

There is a particular kernelization step, which we call *Mark and Torso*, that was used for many of the problems considered in the paper. Unfortunately, this step can ruin planarity (see Figure 5). Given a set X that is guaranteed to be disjoint from the solution, we put a clique on $N(X)$ and remove X from the graph. Intuitively, we introduce all possible shortcuts for the paths that could go through X , which means that X is no longer necessary in the graph. Clearly, adding a large clique ruins planarity. To obtain planarity-preserving kernels, we replace *Torso* with what we call *Contract to Undeletable*: we contract X to a single vertex and mark it undeletable. This step also retains every path going through X , but leaves a somewhat larger graph. However, after some processing (removing vertices with the same neighborhood), we can prove a combinatorial bound showing that in planar graphs (and more generally, in H -minor-free graphs) the number of such undeletable vertices that we introduce is polynomial in the size of the rest of the graph.

A minor technical issue is that this technique requires that we can mark vertices as undeletable in the output instance, but some of the known kernelization algorithms do not work if we extend the input with undeletable vertices. This does not cause a problem for our algorithmic applications, but it means that some of the kernelization results should be stated as a compression into the problem extended with undeletable vertices (see Section 3.3 for details).

There are additional adaptations that need to be done, and additional results that we use from the literature, which we only briefly highlight here.

- In the SUBSET FVS kernel of Hols and Kratsch [30], there is another step beyond *Mark and Torso* that does not preserve planarity. We need to carefully redo that part using different arguments that use and preserve planarity.
- For GROUP FEEDBACK VERTEX SET, contracting X to a single vertex and bounding the number of such vertices is more complicated, as we have to take into account the group elements on the edges.
- The recent work of Wahlström [49] provides quasipolynomial kernels (which is suitable for our applications),

but only for edge-deletion problems.

- The work of Jansen et al. [31] provides deterministic kernels, but only for planar graphs and no generalization is known for H -minor-free graphs.

Extension to H -Minor-Free Graphs. Graphs excluding a fixed graph H as a minor are often considered to be a generalization of planar and bounded-genus graphs [16, 19]. Indeed, many of the good algorithmic and combinatorial properties of planar graphs appear to be generalizable to H -minor-free graphs for every fixed H : the Graph Minors Structure Theorem of Robertson and Seymour [47] provides a roadmap for proving such results.

While results for H -minor-free graphs are (understandably) more complicated to prove than for planar graphs, often the increase of complexity is hidden in “black box” results that can be used as convenient tools. Therefore, in addition to increasing the generality of the result, there is another potential motivation for the generalization to H -minor-free graphs: to develop cleaner and more robust arguments. Arguments about planar graphs typically rely on topological intuition, which can be misleading and formal proofs require very careful treatment of all possible situations. On the other hand, if an argument for H -minor-free graphs only uses some black box results, then it can be actually simpler than its planar counterpart and could reveal better the nature of the problem.

With this goal in mind, we developed our results in a way that would work also for H -minor-free graphs. The only point where we use specific geometric properties of planar graphs is in the proof of Theorem 1.6 and indeed the only missing piece is the generalization of Theorem 1.6 to H -minor-free graphs. For the edge partition version, such a generalization is known:

THEOREM 1.7. (DEMAINE ET AL. [19]) *For every fixed H , there is a $c_H > 0$ such that the following holds. Let G be an H -minor-free graph and $\ell \geq 1$. In polynomial time, we can find a partition of the edge set $E(G) = E_1 \uplus E_2 \uplus \dots \uplus E_\ell$ such that, for every $i \in [\ell]$, it holds that*

$$\text{tw}(G/E_i) = c_H \cdot \ell.$$

This allows us to generalize for example the edge-deletion version of Theorem 1.1 to H -minor-free graphs. However, we need the vertex partition version of Theorem 1.7 for two reasons: to handle vertex-deletion problems and to handle the 2-connected versions of the problems (where the cut vertices make it necessary to work with a vertex partition, even in the edge-deletion version). We formulate as a conjecture that Theorem 1.6 can be generalized in a similar way to H -minor-free graphs.

CONJECTURE 1.1. *For every fixed H , there is a $c_H > 0$ such that the following holds. Let G be an H -minor-free graph and $\ell \geq 1$. In polynomial time, we can find a partition of the vertex set $V(G) = V_1 \uplus V_2 \uplus \dots \uplus V_\ell$ such that, for every $i \in [\ell]$ and every $W \subseteq V_i$, it holds that*

$$\text{tw}(G/E(G[V_i \setminus W])) = c_H \cdot (\ell + |W|).$$

It is likely that the techniques in the proof of Theorem 1.7, together with additional ideas from the proof of Theorem 1.6 could lead to a proof of Conjecture 1.1. However, this would require going into the details of the proof of Theorem 1.7 (see [19]), carefully checking and adapting every step to the vertex-partition case, which is beyond the scope of this paper. Therefore, in the rest of the paper, we point out whenever a result can be generalized to H -minor-free graphs assuming Conjecture 1.1. We also make it clear which of the kernelization results keep not only planarity, but H -minor-freeness as well.

1.2 Organization The paper is organized as follows. Section 2 introduces basic notions related to graph theory, parameterized algorithms, and CSPs. Section 3 formally defines the problems considered in the paper, states our results for the technical CSP problems, and briefly goes through problems of interest that can be reduced to these technical problems. Additionally, we state the kernelization results known from the literature or adapted in this paper, and obtain subexponential FPT algorithms as corollaries.

The next two sections provide technical details of our results. In Section 4, we prove our results on contraction decompositions by partitioning the vertices (Theorem 1.6) and the guessing of the bodies of the segments (Lemma 1.1). Section 5 first presents a dynamic programming algorithm for CSP deletion with size constraints on graphs of bounded treewidth. Then we show how to use this tool to obtain the algorithms for (2CONN) PERM CSP EDGE/VERTEX DELETION.

2 Preliminaries

2.1 Basics We use $\mathbb{N} = \{0, 1, 2, 3, \dots\}$ to denote the natural numbers starting from 0. For a positive integer q , we denote the set $\{1, 2, \dots, q\}$ by $[q]$.

In this article, we consider simple graphs with a finite number of vertices. For an undirected graph G , sets $V(G)$ and $E(G)$ denote its set of vertices and edges, respectively. Unless otherwise specified, we use n to denote the number of vertices of the input graph G . We denote an edge e with two endpoints u, v as $e = uv$. Two vertices u, v in $V(G)$ are *adjacent* to each other if $uv \in E(G)$. The open neighborhood of a vertex v , denoted by $N_G(v)$, is the set of vertices adjacent to v and its degree $\deg_G(v)$ is $|N_G(v)|$. The closed neighborhood of a vertex v , denoted by $N_G[v]$, is the set $N(v) \cup \{v\}$. We omit the subscript in the notation for neighborhood and degree if the graph under consideration is clear. For $S \subseteq V(G)$, we define $N[S] = \bigcup_{v \in S} N[v]$ and $N(S) = N[S] \setminus S$.

For $S \subseteq V(G)$, we denote the graph obtained by deleting S from G by $G - S$ and the subgraph of G induced on the set S by $G[S]$. For two subsets $S_1, S_2 \subseteq V(G)$, the set $E(S_1, S_2)$ denotes the edges with one endpoint in S_1 and another one in S_2 . We say S_1, S_2 are adjacent if $E(S_1, S_2) \neq \emptyset$. For a subset $F \subseteq E(G)$ of edges we write $V(F)$ to denote the collection of endpoints of edges in F . The subgraph of G with $V(F)$ as its set of vertices and F as its set of edges is denoted by $G[F]$.

A set of vertices $S \subseteq V(G)$ is said to be an *independent set* in G if no two vertices in S are adjacent to each other. A set of vertices S is a *vertex cover* of G if $V(G) \setminus S$ is an independent set in G . A *tree-decomposition* of a graph G is a pair (T, β) where T is a tree and $\beta: V(T) \rightarrow 2^{V(G)}$ such that (i) $\bigcup_{t \in V(T)} \beta(t) = V(G)$, (ii) for every edge $uv \in E(G)$ there is a $t \in V(T)$ such that $\{u, v\} \subseteq \beta(t)$, and (iii) for every vertex $v \in V(G)$ the subgraph of T induced by the set $\{t \mid v \in \beta(t)\}$ is connected. The *width* of a tree decomposition is $\max_{t \in V(T)} \{|\beta(t)| - 1\}$ and the *treewidth* of G , denoted by $\text{tw}(G)$, is the minimum width over all tree decompositions of G .

For a tree decomposition (T, β) we distinguish one vertex r of T which will be the root of T . This introduces natural parent-child and ancestor-descendant relations in the tree T .

A *path* $P = (v_1, \dots, v_\ell)$ is a sequence of distinct vertices such that any pair of consecutive vertices are adjacent with each other. The vertex set of P , denoted by $V(P)$, is the set $\{v_1, \dots, v_\ell\}$. The vertices v_1 and v_ℓ are called *endpoints* of the path whereas the other vertices in $V(P)$ are called *internal vertices*. For two vertices $u, v \in V(G)$, we use $\text{dist}_G(u, v)$ to denote the length of the shortest path with endpoints u, v . A *cycle* $C = (v_1, \dots, v_\ell)$ is a sequence of distinct vertices such that any pair of consecutive vertices and v_1, v_ℓ are adjacent with each other. A graph is *connected* if there is a path between every pair of its vertices and it is *disconnected* otherwise. A subset S of $V(G)$ is a *connected set of vertices* if $G[S]$ is connected. A *connected component* of a graph G is a maximal connected set of vertices. A *cut vertex* of a graph G is a vertex v such that the number of connected components of $G - \{v\}$ is strictly larger than that of G . A connected graph that has no cut vertex is called *2-connected*. A *2-connected component* of G is a maximal subset $C \subseteq V(G)$ such that $G[C]$ is 2-connected.

2.2 Planar Graphs and Minors A planar graph is a graph that can be embedded in the Euclidean plane, that is, there exists a mapping from every vertex to a point on a plane, and from every edge to a plane curve on that plane, such that the extreme points of each curve are the points mapped to the endpoints of the corresponding edge, and all curves are disjoint except on their extreme points. A plane graph G is a planar graph with a fixed embedding. Its faces are the regions bounded by the edges, including the outer infinitely large region.

The *contraction* of an edge uv in G deletes vertices u and v from G , and adds a new vertex which is adjacent to vertices that were adjacent to u or v . This process does not introduce self-loops or parallel edges. The resulting graph is denoted by G/e . For $F \subseteq E(G)$, the graph G/F denotes the graph obtained from G by contracting each connected component in the subgraph $G[F]$ to a vertex. Similarly, for $S \subseteq V(G)$, the graph G/S denotes the graph obtained from G by contracting each connected component in the subgraph $G[S]$ to a vertex.

A graph H obtained by a sequence of such edge contractions starting from G is said to be a contraction of G . A graph H is a minor of G if H is a subgraph of some contraction of G . For a graph H , a graph G is said to be *H -minor-free* if G can not be contracted to H . We use the following result about *H -minor-free* graphs.

THEOREM 2.1. (KOSTOCHKA [38], THOMASON [48]) *Let G be an H -minor-free graph. Then there is a constant d_H such that*

$$\sum_{v \in V(G)} \deg_G(v) \leq d_H |V(G)|.$$

Moreover, one can choose $d_H = \mathcal{O}(h\sqrt{\log h})$ where h denotes the number of vertices of H .

2.3 Parameterized Algorithms and Kernelization An instance of a *parameterized problem* is of the form $(I; k)$ where I is an instance of a (classical) decision problem and $k \in \mathbb{N}$ is the *parameter*. A parameterized problem is said to be *fixed parameter tractable* (FPT) if there exists an algorithm \mathcal{A} , a computable function f , and a constant c such that, given any instance $(I; k)$ of the parameterized problem, the algorithm \mathcal{A} correctly decides whether $(I; k)$ is a YES-instance in time $\mathcal{O}(f(k)|I|^c)$. An accompanying theory of hardness can be used to identify parameterized problems that are unlikely to admit FPT algorithms. For the purpose of this article, we call the class of such problems as W[1]-Hard.

Instances $(I; k)$ and $(I'; k')$ of a parameterized problem are *equivalent* if $(I; k)$ is a YES-instance if and only if $(I'; k')$ is a YES-instance. A *compression* for a parameterized problem is an algorithm \mathcal{B} that, given an instance (I, k) of the problem, works in time $\mathcal{O}((|I| + k)^c)$ (for some constant c) and returns an equivalent instance (I', k') of some problem. A compression is said to be a *kernel* if (I', k') is an instance of the same problem. If there exists a computable function g such that size of an output obtained by algorithm \mathcal{B} for (I, k) is at most $g(k)$, we say that problem admits a compression of size $g(k)$. If $g(k)$ is a polynomial function, then we say the problem admits a polynomial compression. If $g(k) = k^{\text{polylog}(k)}$, then we say the problem admits a quasipolynomial compression. Consider the parameterized problems whose input contains a graph. We say a compression is *minor-preserving* if the graph in the reduced instance (I', k') is H -minor-free whenever the graph in the input instance (I, k) is H -minor-free. We define similar notion for kernels.

We refer readers to [13] for a detailed exposition on the subject.

2.4 Constraint Satisfaction Problems A *CSP-instance* is a tuple $\Gamma = (X, D, \mathcal{C})$ where X is a finite set of variables, D is a finite domain, and \mathcal{C} is a set of constraints $c = ((x_1, \dots, x_{a(c)}), R)$ where $x_1, \dots, x_{a(c)} \in X$ and $R \subseteq D^{a(c)}$. We refer to $a(c)$ as the *arity* of constraint c . An assignment $\alpha: X \rightarrow D$ *satisfies* a constraint $c = ((x_1, \dots, x_{a(c)}), R) \in \mathcal{C}$ if $(\alpha(x_1), \dots, \alpha(x_{a(c)})) \in R$. If α does not satisfy c , then we say that α *violates* c . An assignment $\alpha: X \rightarrow D$ *satisfies* Γ if it satisfies every constraint $c \in \mathcal{C}$. The instance Γ is *satisfiable* if there is a satisfying assignment.

We say that a CSP-instance $\Gamma = (X, D, \mathcal{C})$ is *binary* if $a(c) \leq 2$ for all $c \in \mathcal{C}$. In the remainder of this work, we restrict ourselves to binary CSP-instances. Let $c \in \mathcal{C}$ be a constraint. We call c a *unary* constraint if $a(c) = 1$ and a *binary* constraint if $a(c) = 2$. A binary constraint $c = ((x_1, x_2), R) \in \mathcal{C}$ is called a *permutation constraint* if for every $a \in D$ it holds that $|\{b \in D \mid (a, b) \in R\}| \leq 1$ and $|\{b \in D \mid (b, a) \in R\}| \leq 1$. A binary CSP-instance is called is *Permutation-CSP-instance* if every binary constraint is a permutation constraint.

Let $\Gamma = (X, D, \mathcal{C})$ be a binary CSP-instance. The *constraint graph* of Γ is defined as the graph $G(\Gamma)$ with vertex set $V(G(\Gamma)) := X$ and edge set

$$E(G(\Gamma)) := \{xy \mid ((x, y), R) \in \mathcal{C}\}.$$

Let $Y \subseteq X$ be a subset of the variables. We define $\Gamma[Y] := (Y, D, \mathcal{C}[Y])$ to be the *induced subinstance* of Γ where $\mathcal{C}[Y]$ contains all constraints $c = ((x_1, \dots, x_{a(c)}), R) \in \mathcal{C}$ such that $x_i \in Y$ for all $i \in [a(c)]$. Observe that $G(\Gamma[Y]) = (G(\Gamma))[Y]$. Also, for $F \subseteq E(G(\Gamma))$, we define $\mathcal{C}[F]$ to be the set of all binary constraints $c = ((x_1, x_2), R) \in \mathcal{C}$ such that $x_1x_2 \in F$. Moreover, we define $\mathcal{C} - F := \mathcal{C} \setminus \mathcal{C}[F]$ and $\Gamma - F := (X, D, \mathcal{C} - F)$.

3 Permutation CSPs with Size Constraints

3.1 Problem Definitions In this work, we shall be interested in vertex- and edge-deletion problems on Permutation-CSP-instances. In the following, we formally define the problems on CSP-instances and state the main algorithmic results on planar input instances (more details on the algorithms are given in Section 5). Then, we demonstrate the wide applicability of these problems by reducing various well-known cycle hitting problems to the defined problems on Permutation-CSP-instances. Finally, we list several kernelization results which, in combination with our algorithmic results for vertex- and edge deletion problems on Permutation-CSP-instances, lead to various subexponential parameterized algorithms for certain cycle hitting problems on planar graphs.

The Basic Edge-Deletion Problem. We start by considering the basic edge deletion problem for Permutation-CSPs.

PERM CSP EDGE DELETION

Parameter: k

Input: A Permutation-CSP-instance $\Gamma = (X, D, \mathcal{C})$, a set of undeletable edges $U \subseteq E(G(\Gamma))$, and an integer k

Question: Is there a set $Z \subseteq E(G(\Gamma)) \setminus U$ such that $|Z| \leq k$ and $\Gamma - Z := (X, D, \mathcal{C} - Z)$ is satisfiable?

In this work, we are only interested in CSP-instances Γ where $G(\Gamma)$ satisfies a certain property. We call a binary CSP-instance Γ *planar* if the graph $G(\Gamma)$ is planar. Similarly, a binary CSP-instance Γ is *H-minor-free* if $G(\Gamma)$ is H-minor-free. This allows to define the problems PLANAR PERM CSP EDGE DELETION and H-MINOR-FREE PERM CSP EDGE DELETION where we restrict ourselves to input CSP-instances Γ that are planar or H-minor-free. We remark here that we follow similar naming conventions for other vertex- and edge deletion problems without explicitly defining them.

The next theorem is essentially a simple consequence of the contraction decompositions for H-minor-free graphs due to Demaine et al. [19]. Still, in combination with recent kernelization results due to Wahlström [49], it already leads to improved FPT algorithms for certain edge deletion problems on H-minor-free graphs.

THEOREM 3.1. *There is an algorithm solving H-MINOR-FREE PERM CSP EDGE DELETION in time $(|X| + |D|)^{\mathcal{O}(c_H \sqrt{k})}$ where c_H is a constant depending only on H.*

Size Constraints for Connected Components. Next, we turn our attention to vertex-deletion problems. To increase the applicability of our results, we also extend the CSP-instances with certain size constraints. In this section, we focus on size constraints that bound the total weight of a connected component (after removing the solution).

Let $\Gamma = (X, D, \mathcal{C})$ be a binary CSP instance. A *1cc-size constraint* is a triple (w, q, op) where $w: X \times D \rightarrow \mathbb{N}$ is a weight function, $q \in \mathbb{N}$, and $\text{op} \in \{\leq, \geq\}$. An assignment $\alpha: X \rightarrow D$ satisfies (w, q, op) on Γ if

$$\left(\sum_{v \in C} w(v, \alpha(v)), q \right) \in \text{op}$$

for every connected component C of $G(\Gamma)$. (Here, $(p, q) \in \leq$ if $q \leq p$ and $(p, q) \in \geq$ if $q \geq p$.)

A *Permutation-CSP-instance with 1cc-size constraints* is a pair (Γ, \mathcal{S}) where $\Gamma = (X, D, \mathcal{C})$ is a Permutation-CSP-instance and \mathcal{S} is a set of 1cc-size constraints. We say that (Γ, \mathcal{S}) is *satisfiable* if there is an assignment $\alpha: X \rightarrow D$ which satisfies Γ as well as every constraint $(w, q, \text{op}) \in \mathcal{S}$ on Γ . Also, we define

$$\|\mathcal{S}\| := \prod_{(w, q, \text{op}) \in \mathcal{S}} (2 + q).$$

PERM CSP VERTEX DELETION WITH SIZE CONSTRAINTS

Parameter: k

Input: A Permutation-CSP-instance with 1cc-size constraints (Γ, \mathcal{S}) with $\Gamma = (X, D, \mathcal{C})$, a set of undeletable variables $U \subseteq X$, and an integer k .

Question: Is there a set $Z \subseteq X \setminus U$ such that $|Z| \leq k$ and $(\Gamma[X \setminus Z], \mathcal{S})$ is satisfiable?

THEOREM 3.2. *There is an algorithm solving PLANAR PERM CSP VERTEX DELETION WITH SIZE CONSTRAINTS in time $(|X| + |D| + \|\mathcal{S}\|)^{\mathcal{O}(\sqrt{k})}$.*

Satisfiable 2-Connected Components. Next, we state the corresponding problems and results for 2-connected components.

Let $\Gamma = (X, D, \mathcal{C})$ be a binary CSP instance. A *2cc-size constraint* is a pair (w, q) where $w: X \times D \rightarrow \mathbb{N}$ is a weight function, and $q \in \mathbb{N}$. Let $W \subseteq X$. An assignment $\alpha: W \rightarrow D$ satisfies (w, q) on W if

$$\sum_{v \in W} w(v, \alpha(v)) \leq q.$$

Observe that, in comparison to 1cc-size constraints, we only allow to check for upper bounds on the weighted size of a set $W \subseteq X$.

A *Permutation-CSP-instance with 2cc-size constraints* is a pair (Γ, \mathcal{S}) where $\Gamma = (X, D, \mathcal{C})$ is a Permutation-CSP-instance and \mathcal{S} is a set of 2cc-size constraints. For $W \subseteq X$, we say that (Γ, \mathcal{S}) is *satisfiable on W* if there is an assignment $\alpha: W \rightarrow D$ that satisfies $\Gamma[W]$ as well as all 2cc-size constraints $(w, q) \in \mathcal{S}$ on W . As before, we define

$$\|\mathcal{S}\| := \prod_{(w, q, \text{op}) \in \mathcal{S}} (2 + q).$$

2CONN PERM CSP VERTEX DELETION WITH SIZE CONSTRAINTS

Parameter: k

Input: A Permutation-CSP-instance with 2cc-size constraints (Γ, \mathcal{S}) with $\Gamma = (X, D, \mathcal{C})$, a set of undeletable variables $U \subseteq X$, and an integer k .

Question: Is there a set $Z \subseteq X \setminus U$ such that $|Z| \leq k$ and (Γ, \mathcal{S}) is satisfiable on W for every 2-connected component W of the graph $G(\Gamma[X \setminus Z])$.

THEOREM 3.3. *There is an algorithm solving PLANAR 2CONN PERM CSP VERTEX DELETION WITH SIZE CONSTRAINTS in time $(|X| + |D| + \|\mathcal{S}\|)^{O(\sqrt{k})}$.*

3.2 Reductions Next, we list several standard vertex and edge deletion problems can be interpreted as special cases of the problems discussed above. In all the cases, the constraint graph of the CSP-instance we construct is the same as the input graph for the problem in question. In particular, restrictions of the input graph immediately translate to corresponding restrictions of the constructed CSP instance. Also, if the problem in question marks certain vertices as undeletable, this immediately translates to a set of undeletable variables in the CSP instance. Here, we only focus on vertex deletion problems. However, the edge deletion versions of all problems naturally translate to the edge deletion versions of the CSP problems.

We start by covering some problems that can be reduced to PERM CSP VERTEX DELETION WITH SIZE CONSTRAINTS (more details on the reductions can be found in the full version). We first consider some problems where the size constraints are not required to build the reduction. We start with the problem of eliminating all odd cycles in a graph.

ODD CYCLE TRANSVERSAL (OCT)

Parameter: k

Input: A graph G , and an integer k

Question: Does there exist a set $Z \subseteq V(G)$ of size at most k that hits all odd cycles, i.e., $G - Z$ is bipartite?

This problem can be translated into a Permutation-CSP-instance $\Gamma = (X, D, \mathcal{C})$ with

- $X := V(G)$,
- $D := \{0, 1\}$, and
- binary constraints $((v, w), R_{\neq})$ for all $vw \in E(G)$ where $R_{\neq} := \{(0, 1), (1, 0)\}$.

It is easy to verify that $Z \subseteq V(G)$ is a solution for (G, k) if and only if Z is a solution for (Γ, k) .

More generally, the same approach can be used to reduce GROUP FEEDBACK VERTEX SET to PERM CSP VERTEX DELETION WITH SIZE CONSTRAINTS. Let Σ be a group. A Σ -labeled graph is a pair (G, λ) where G is a graph and $\lambda: \{(v, w), (w, v) \mid vw \in E(G)\} \rightarrow \Sigma$ is a mapping satisfying $\lambda(v, w) = (\lambda(w, v))^{-1}$ for all $vw \in E(G)$. A cycle $C = (v_1, \dots, v_\ell)$ is a *non-null cycle* if $\lambda(C) := \lambda(v_1, v_2)\lambda(v_2, v_3) \dots \lambda(v_{\ell-1}, v_\ell)\lambda(v_\ell, v_1) \neq 1_\Sigma$ where 1_Σ denotes the identity element of Σ (this notion is well-defined; see [29]).

GROUP FEEDBACK VERTEX SET (GROUP FVS)

Parameter: k

Input: A Σ -labeled graph G , and an integer k

Question: Does there exist a set $Z \subseteq V(G)$ of size at most k that hits all non-null cycles?

Next, we consider two problems where the reductions use the size constraints.

VERTEX MULTIWAY CUT

Parameter: k

Input: A graph G , a set $T \subseteq V(G)$, and an integer k

Question: Does there exist a set $Z \subseteq V(G) \setminus T$ of size at most k such that all vertices from T lie in different connected components of $G - Z$?

Consider the instance (Γ, \mathcal{S}, U) with $\Gamma = (X, D, C)$ and

- $X := V(G)$,
- $D := \{\otimes\}$,
- binary constraints $((v, w), \{(\otimes, \otimes)\})$ for all $vw \in E(G)$,
- $U := T$ is the set of undeletable variables, and
- $\mathcal{S} = \{(w, 1, \leq)\}$ where $w: X \times D \rightarrow \mathbb{N}$ is defined via $w(v, \otimes) = 1$ for all $v \in T$, and $w(v, \otimes) = 0$ for all $v \in V(G) \setminus T$.

Again, it is easy to verify that Z is a VERTEX MULTIWAY CUT solution for (G, T, k) if and only if Z is a solution for $(\Gamma, \mathcal{S}, U, k)$.

A second example is the problem COMPONENT ORDER CONNECTIVITY.

COMPONENT ORDER CONNECTIVITY **Parameter:** k
Input: A graph G and integers k, t
Question: Does there exist a set $Z \subseteq V(G)$ of size at most k such that $|C| \leq t$ for every connected component C of $G - Z$?

Next, we turn to problems that can be reduced to 2CONN PERM CSP VERTEX DELETION WITH SIZE CONSTRAINTS. The first example is the SUBSET FEEDBACK VERTEX SET problem which asks to eliminate all cycles visiting a terminal vertex $t \in T$ in a graph G .

SUBSET FEEDBACK VERTEX SET (SUBSET FVS) **Parameter:** k
Input: A graph G , a set $T \subseteq V(G)$, and an integer k
Question: Does there exist a set $Z \subseteq V(G)$ of size at most k that hits all T -cycles?

A set $Z \subseteq V(G)$ is a solution for SUBSET FVS if and only if $|W| \leq 2$ or $W \cap T = \emptyset$ for every 2-connected component W of $G - Z$. Consider the CSP-instance $\Gamma = (X, D, C)$ with

- $X := V(G)$,
- $D := \{\otimes\} \uplus V(G) \uplus E(G)$,
- unary constraints $(v, \{e \in E(G) \mid v \in e\} \cup \{v\})$ for all $v \in T$,
- unary constraints $(v, \{\otimes\} \cup \{e \in E(G) \mid v \in e\})$ for all $v \notin T$, and
- binary constraints $((v, w), R_=)$ for all $vw \in E(G)$ where $R_:= \{(a, a) \mid a \in D\}$.

It is not difficult to show that Z is a SUBSET FVS solution for (G, T, k) if and only if Z is a solution for (Γ, k) .

Using the framework of Permutation-CSPs, we can also cover more specialized problems such as the following variant of SUBSET FVS.

TWO SUBSET FEEDBACK VERTEX SET (TWO SUBSET FVS) **Parameter:** k
Input: A graph G , a set $T \subseteq V(G)$, and an integer k
Question: Does there exist a set $Z \subseteq V(G)$ of size at most k that hits all cycles C such that $|C \cap T| \geq 2$?

Another example is the SUBSET GROUP FEEDBACK VERTEX SET problem which in particular contains SUBSET OCT as a special case.

SUBSET GROUP FEEDBACK VERTEX SET (SUBSET GROUP FVS) **Parameter:** k
Input: A Σ -labeled graph G , a set $T \subseteq V(G)$, and an integer k
Question: Does there exist a set $Z \subseteq V(G)$ of size at most k that hits all non-null T -cycles?

As the last example, we consider 2CONN COMPONENT ORDER CONNECTIVITY which again relies on the size constraints.

Input: A graph G and integers k, t

Question: Does there exist a set $Z \subseteq V(G)$ of size at most k such that $|W| \leq t$ for every 2-connected component W of $G - Z$?

This problem can be formulated as an instance (Γ, \mathcal{S}) with $\Gamma = (X, D, \mathcal{C})$ and

- $X := V(G)$,
- $D := \{\otimes\}$,
- binary constraints $((v, w), \{(\otimes, \otimes)\})$ for all $vw \in E(G)$, and
- $\mathcal{S} = \{(w, t)\}$ where $w: X \times D \rightarrow \mathbb{N}$ is defined via $w(v, \otimes) = 1$ for all $v \in V(G)$.

We complete this part by explicitly listing some algorithmic consequences of the reductions described above and Theorems 3.2 and 3.3.

COROLLARY 3.1. *There are algorithms for the following problems:*

1. PLANAR COMPONENT ORDER CONNECTIVITY running in time $(n + t)^{\mathcal{O}(\sqrt{k})}$,
2. PLANAR TWO SUBSET FVS running in time $n^{\mathcal{O}(\sqrt{k})}$,
3. PLANAR SUBSET GROUP FVS running in time $(n + |\Sigma|)^{\mathcal{O}(\sqrt{k})}$, and
4. PLANAR 2CONN COMPONENT ORDER CONNECTIVITY running in time $(n + t)^{\mathcal{O}(\sqrt{k})}$.

For the other problems, we combine Theorems 3.2 and 3.3 with kernelization results to obtain subexponential FPT algorithms.

3.3 Kernels and Subexponential FPT Algorithms We can combine the subexponential parameterized algorithms for the problems discussed above with existing kernelization results to also obtain subexponential FPT algorithms. For this to work, we need two additional properties of the kernelization. First, we require that the parameter is only increased linearly. Second, if the input graph is planar (resp. H -minor-free), then the graph of the reduced instance also has to be planar (resp. H -minor-free). In many cases, existing kernelizations do not satisfy the second property. However, it is usually possible to modify the kernels appropriately. In the following, we list the kernelization results relevant for this work, and briefly comment on where modifications to existing kernels are required. All details for these results can be found in the full version.

We start by considering edge deletion versions of the problems above (to refer to the edge deletion version of a problem, we simply replace the word VERTEX by EDGE in the problem name, or add the word EDGE if the word VERTEX does not appear). In a recent work, Wahlström [49] obtained quasipolynomial kernels for the following problems. Also, it can be checked that all these kernels preserve minors of the input graph and do not increase the parameter in question.

THEOREM 3.4. (WAHLSTRÖM [49]) *The following problems have randomized quasipolynomial kernels with failure probability $\mathcal{O}(2^{-n})$:*

1. EDGE MULTIWAY CUT parameterized by solution size,
2. GROUP FEEDBACK EDGE SET parameterized by solution size, for any group, such that the group remains the same in the reduced instance,
3. SUBSET FEEDBACK EDGE SET with undeletable edges, parameterized by solution size.

Moreover, if the graph in the input instance is H -minor-free then the graph in the reduced instance is also H -minor-free, and the parameter does not increase in the reduced instance.

Since all the reductions presented above also go through for the edge deletion versions of the problems, we obtain the following corollary by combining Theorem 3.1 and the last theorem.

COROLLARY 3.2. *There are randomized algorithms solving the following problems with failure probability $\mathcal{O}(2^{-n})$:*

1. *H-MINOR-FREE EDGE MULTIWAY CUT in time $2^{\mathcal{O}(c_H \cdot \sqrt{k} \cdot \text{polylog}(k))} n^{\mathcal{O}(1)}$, and*
2. *H-MINOR-FREE GROUP FEEDBACK EDGE SET in time $|\Sigma|^{\mathcal{O}(c_H \cdot \sqrt{k} \cdot \text{polylog}(k))} n^{\mathcal{O}(1)}$.*

Here, c_H denotes a constant that only depends on H .

We remark that a subexponential fpt algorithm for PLANAR EDGE MULTIWAY CUT was already given in [46].

Observe that SUBSET FEEDBACK EDGE SET is not covered by the corollary since restrictions need to be enforced on 2-connected components (after removing the solution). However, we can still obtain a subexponential FPT algorithm on planar graphs by reducing the edge deletion version to the vertex deletion version with undeletable vertices. Observe that we can easily introduce undeletable vertices into the above reduction by using the set of undeletable variables in the definition of PERM CSP VERTEX DELETION WITH SIZE CONSTRAINTS and 2CONN PERM CSP VERTEX DELETION WITH SIZE CONSTRAINTS. So we obtain the following corollary from combining Theorems 3.3 and 3.4.

COROLLARY 3.3. *There is a randomized algorithm with failure probability $\mathcal{O}(2^{-n})$ solving PLANAR SUBSET FEEDBACK EDGE SET WITH UNDELETABLE EDGES in time $2^{\mathcal{O}(\sqrt{k} \cdot \text{polylog}(k))} n^{\mathcal{O}(1)}$.*

Next, let us come back to vertex deletion problems. Here, the situation is generally more complicated since existing kernels for the problems we are interested in usually do not preserve H -minor-freeness. We present modifications to several kernels from [30, 39]. These modifications require us to introduce undeletable vertices into the problem definition. Strictly speaking, this means that our results are not kernelizations, but only compressions. However, for the purpose of designing subexponential FPT algorithms, this does not pose any additional problems since all reductions presented above can trivially be extended to take undeletable vertices into account. For any problem Π discussed above, we use Π WITH UNDELETABLE VERTICES to refer to the variant of the problem where undeletable vertices are added. Also, VERTEX MULTIWAY CUT WITH DELETABLE TERMINAL refers to the variant of VERTEX MULTIWAY CUT where terminal vertices may be deleted in the solution.

THEOREM 3.5. *Let Π be VERTEX MULTIWAY CUT WITH DELETABLE TERMINAL or SUBSET FEEDBACK VERTEX SET. There is an algorithm that given an instance (G, T, k) of H -MINOR FREE Π constructs an equivalent instance (G', T, F, k') of H -MINOR FREE Π WITH UNDELETABLE VERTICES in randomized polynomial time and with failure probability $\mathcal{O}(2^{-|V(G)|})$ such that $|V(G')| \in (c_H \cdot k)^{\mathcal{O}(1)}$ and $k' \leq k$.*

THEOREM 3.6. *There is an algorithm that given an instance $(G, \lambda, \Sigma, T, k)$ of H -MINOR-FREE GROUP FEEDBACK VERTEX SET constructs an equivalent instance $(G', \lambda', \Sigma, T, F, k')$ of H -MINOR-FREE GROUP FEEDBACK VERTEX SET WITH UNDELETABLE VERTICES in randomized polynomial time and with failure probability $\mathcal{O}(2^{-|V(G)|})$ such that $|V(G')| \in k^{\mathcal{O}(|\Sigma| \cdot |V(H)|)}$ and $k' \leq k$.*

We get the following result by combining Theorems 3.3 and 3.5.

COROLLARY 3.4. *There is a randomized algorithm solving PLANAR SUBSET FEEDBACK VERTEX SET with failure probability $\mathcal{O}(2^{-n})$ in time $2^{\mathcal{O}(\sqrt{k} \cdot \log k)} n^{\mathcal{O}(1)}$ where k denotes the solution size.*

Similarly, we can get the following result by combining Theorems 3.2 and 3.6.

COROLLARY 3.5. *There is a randomized algorithm solving PLANAR GROUP FEEDBACK VERTEX SET with failure probability $\mathcal{O}(2^{-n})$ in time $2^{\mathcal{O}(\sqrt{k} \cdot \log k \cdot |\Sigma|)} n^{\mathcal{O}(1)}$ where k denotes the solution size and Σ the input group.*

For VERTEX MULTIWAY CUT on planar graphs, we can actually get a stronger result by exploiting existing kernels on planar graphs.

THEOREM 3.7. (JANSEN ET AL. [31]) *PLANAR VERTEX MULTIWAY CUT parameterized by solution size admits a deterministic polynomial kernel where the parameter in the reduced instance is not increased.*

COROLLARY 3.6. *There is a (deterministic) algorithm solving PLANAR VERTEX MULTIWAY CUT in time $2^{\mathcal{O}(\sqrt{k} \cdot \log k)} n^{\mathcal{O}(1)}$ where k denotes the solution size.*

4 Structural Analysis

The next two sections are devoted to the proofs of Theorems 3.1, 3.2 and 3.3. In this section, we start by providing all necessary structural tools. In particular, we prove Theorem 1.6 and Lemma 1.1.

4.1 Contraction Decompositions

4.1.1 Edge Partitions We first concern ourselves with the contraction decompositions for planar and H -minor-free graphs. For the edge deletion problem PERM CSP EDGE DELETION our algorithm relies on a contraction decomposition due to Demaine et al. [19] (see Theorem 1.7).

THEOREM 4.1. (DEMAINE, HAJIAGHAYI, KAWARABAYASHI [19]) *Let G be an H -minor free graph and $k \geq 1$. Then there is a partition of the edge set $E(G) = E_1 \uplus E_2 \uplus \dots \uplus E_k$ such that, for every $i \in [k]$, it holds that*

$$\text{tw}(G/E_i) \leq c_H k$$

for some constant c_H depending only on H . Moreover, given the graph G and $k \geq 1$, the partition $E(G) = E_1 \uplus E_2 \uplus \dots \uplus E_k$ can be computed in polynomial time.

To be more precise, the algorithm from Theorem 3.1 relies on the following corollary which allows us to declare certain edges to be uncontractible.

COROLLARY 4.1. *Let G be an H -minor free graph and $k \geq 1$. Then there is a partition of the edge set $E(G) = E_1 \uplus E_2 \uplus \dots \uplus E_k$ such that, for every $i \in [k]$ and every $W \subseteq E_i$, it holds that*

$$\text{tw}(G/(E_i \setminus W)) \leq c_H k + |W|$$

for some constant c_H depending only on H . Moreover, given the graph G and $k \geq 1$, the partition $E(G) = E_1 \uplus E_2 \uplus \dots \uplus E_k$ can be computed in polynomial time.

Proof. The statement follows from Theorem 4.1 by observing that the contraction of a single edge decreases the treewidth of a graph by at most one. \square

4.1.2 Vertex Partitions Next, we turn to the vertex version of the contraction decomposition. In this work, we prove such a statement only for planar graphs. More precisely, we obtain the following theorem which implies Theorem 1.6 (here, we give an explicit upper bound on the treewidth of the contracted graph and avoid using \mathcal{O} -notation).

THEOREM 4.2. *Let G be a planar graph and $k \geq 1$. Then there is a partition of the vertex set $V(G) = V_1 \uplus V_2 \uplus \dots \uplus V_k$ such that, for every $i \in [k]$ and every $W \subseteq V_i$, it holds that*

$$\text{tw}(G/E(G[V_i \setminus W])) \leq 3k + 14|W| + 2.$$

Moreover, given the graph G and $k \geq 1$, the partition $V(G) = V_1 \uplus V_2 \uplus \dots \uplus V_k$ can be computed in polynomial time.

The remainder of this subsection is devoted to proving the theorem. First consider the case $k = 1$. Then W forms a vertex cover of the graph $G/E(G[V_1 \setminus W])$ and thus, $\text{tw}(G/E(G[V_1 \setminus W])) \leq |W|$. So suppose $k \geq 2$.

Let G be a planar graph and let us fix an embedding of G in the plane. Let $F(G)$ denote the faces of the embedding. We define the *radial graph* $R := R(G)$ with vertex set $V(R) := V(G) \cup F(G)$ and edge set

$$E(R) := \{vf \mid v \in V(G), f \in F(G), v \text{ is incident to } f\}.$$

Note that R is a connected, bipartite graph with bipartition $(V(G), F(G))$. Also pick f_0 to be the exterior face of the embedding. For $j \geq 1$ define

$$L_j := \{v \in V(R) \mid \text{dist}_R(f_0, v) = 2j - 1\}$$

to be the j -th *vertex layer*. Note that $L_j \subseteq V(G)$ for all $j \geq 1$. For a vertex $v \in V(G)$ define $L(v) := j$ for the unique $j \geq 1$ such that $v \in L_j$.

OBSERVATION 4.1. Let $vw \in E(G)$. Then $|L(v) - L(w)| \leq 1$.

Proof. There is a face $f \in F(G)$ such that v and w are incident to f . Hence, $\text{dist}_R(v, w) \leq 2$. This implies the observation. \square

For later reference we give a second description of the layers (with respect to the fixed embedding of G). For $j \geq 1$ define L'_j to be the set of vertices incident to the exterior face of the embedding after deleting all vertices from $L'_1 \cup \dots \cup L'_{j-1}$. The following observation follows from a simple induction on $j \geq 1$.

OBSERVATION 4.2. $L_j = L'_j$ for all $j \geq 1$.

DEFINITION 4.1. An embedding of a graph G is 1-outerplanar if it is planar and all vertices lie on the exterior face. For $k \geq 2$ an embedding of a graph G is k -outerplanar if it is planar and, after deleting all vertices on the exterior face, the resulting embedding is $(k - 1)$ -outerplanar. A graph is k -outerplanar if it has a k -outerplanar embedding.

We first record a property of 1-outplanar graphs that plays an important role later on.

LEMMA 4.1. Let G be a 1-outerplanar graph and let $W \subseteq V(G)$. Also fix some 1-outerplanar embedding of G and let f_0 denote the exterior face. Then

$$\sum_{f \in F(G) \setminus \{f_0\}} \max\{|N_{R(G)}(f) \cap W| - 1, 0\} \leq 13|W|.$$

Proof. Consider the radial graph $R := R(G)$ which is planar and bipartite. Let $H := R[W \cup F(G)]$ be the induced subgraph on W and $F(G)$. For $i \geq 0$ let $F_i := \{f \in F(G) \mid |N_{R(G)}(f) \cap W| = i\}$. Then

$$\sum_{i \geq 3} i \cdot |F_i| \leq |E(H)| \leq 2 \left(\sum_{i \geq 3} |F_i| + |W| \right) - 4$$

which implies that

$$\sum_{i \geq 3} (i - 2) \cdot |F_i| \leq 2|W|.$$

In particular,

$$\sum_{i \geq 3} (i - 1) \cdot |F_i| \leq 4|W|.$$

It remains to bound the size of F_2 . Consider the graph $H_2 := R[W \cup F_2]$ in which every vertex in F_2 has exactly two neighbors. By outerplanarity, for each $f \in F_2$, it holds that $|\{f' \in F_2 \mid N_R(f) = N_R(f')\}| \leq 3$. Moreover, there are at most $3|W|$ elements from F_2 with pairwise distinct neighborhoods since H_2 is planar (every planar graph with n vertices has at most $3n - 6$ edges). Hence, $|F_2| \leq 9|W|$.

So in total

$$\sum_{f \in F(G) \setminus \{f_0\}} \max\{|N_{R(G)}(f) \cap W| - 1, 0\} = |F_2| + \sum_{i \geq 3} (i - 1) \cdot |F_i| \leq 13|W|.$$

\square

OBSERVATION 4.3. The graph $G[L_j \cup L_{j+1} \cup \dots \cup L_{j+k}]$ is $(k + 1)$ -outerplanar.

Proof. This follows from Observation 4.2 and the definition of the sets L'_j , $j \geq 1$. \square

An important feature of k -outerplanar graphs is that their treewidth is bounded by a linear function in k .

THEOREM 4.3. (BODLAENDER [9]) Let G be a k -outerplanar graph. Then $\text{tw}(G) \leq 3k - 1$.

This implies that

$$(4.1) \quad \text{tw}(G[L_j \cup L_{j+1} \cup \dots \cup L_{j+k}]) \leq 3k + 2$$

for all $j \geq 1$. Next, we define

$$V_i := \bigcup_{j \equiv i \pmod k} L_j$$

for all $i \in [k]$. Note that this partition can be computed in polynomial time since an embedding for a planar graph can be computed in polynomial time. Fix $i \in [k]$ and $W \subseteq V_i$.

Let C_1, \dots, C_ℓ be the connected components of $G[V_i \setminus W]$. We refer to these components as the *articulation points*. Note that $C_r \subseteq L_j$ for some $j \equiv i \pmod k$ by Observation 4.1 (recall that $k \geq 2$). Consistent with previous notation, let $L(C_r) := j$ for the unique $j \geq 1$ such that $C_r \subseteq L_j$.

Now let G' be the graph with vertex set

$$V(G') := (V(G) \setminus V_i) \cup \{C_r, C'_r \mid r \in [\ell]\}$$

and edge set

$$\begin{aligned} E(G') := & E(G[V(G) \setminus V_i]) \\ & \cup \{vC_r \mid L(v) + 1 = L(C_r) \wedge \exists w \in C_r: vw \in E(G)\} \\ & \cup \{vC'_r \mid L(v) = L(C_r) + 1 \wedge \exists w \in C_r: vw \in E(G)\}. \end{aligned}$$

LEMMA 4.2. $(G/E(G[V_i \setminus W])) - W = G'/\{C_r C'_r \mid r \in [\ell]\}$.

Proof. Clearly, $(G/E(G[V_i \setminus W])) - W$ is obtained from G by contracting C_r to a single vertex for all $r \in [\ell]$, and deleting all vertices from W . Similarly, $G'/\{C_r C'_r \mid r \in [\ell]\}$ is obtained from G by contracting C_r to a single vertex for all $r \in [\ell]$, and deleting all vertices from W . Note that all edges from the graph $(G/E(G[V_i \setminus W])) - W$ are present by Observation 4.1. \square

In order to bound the treewidth of $G/E(G[V_i \setminus W])$ we proceed in three steps. For the first step we provide an upper bound on the treewidth of G' . Then, in the second step, we prove that contraction of pairs of articulation points does not increase the treewidth significantly. Finally, all vertices from W are added to all bags which increases the treewidth only by $|W|$.

Let D_1, \dots, D_t be the connected components of G' and let $p \in [t]$. For $j \equiv i \pmod k$ define

$$U_j := \{C'_r \mid L(C_r) = j\} \cup L_{j+1} \cup \dots \cup L_{j+k-1} \cup \{C_r \mid L(C_r) = j+k\}.$$

Clearly, there is some $j \equiv i \pmod k$ such that $D_p \cap U_j \neq \emptyset$. Moreover, there are no outgoing edges from the set U_j in the graph G' by Observation 4.1. Hence, $D_p \subseteq U_j$.

This implies that $G'[D_p]$ is a minor of the graph $G[L_j \cup L_{j+1} \cup \dots \cup L_{j+k}]$. So $\text{tw}(G'[D_p]) \leq 3k + 2$ by Equation (4.1).

For the second step consider the following DAG H with vertex $V(H) := \{D_1, \dots, D_t\} \cup \{C_1, \dots, C_\ell\}$ and edges

$$E(H) := \{(C_r, D_p) \mid C_r \in D_p\} \cup \{(D_p, C_r) \mid C'_r \in D_p\}.$$

Note that, for edges (D_p, C_r) and $(C_r, D_{p'})$, it holds that $D_p \subseteq U_j$ and $D_{p'} \subseteq U_{j+k}$ for some $j \equiv i \pmod k$. Moreover, each articulation point has exactly one incoming and one outgoing edge. Together this proves that H is a DAG.

For $p \in [t]$ we denote by $\deg_H^-(D_p)$ the number of incoming edges for the vertex D_p in the graph H .

LEMMA 4.3. $\sum_{p \in [t]: \deg_H^-(D_p) \geq 2} \deg_H^-(D_p) - 1 \leq 13|W|$.

Proof. Fix $j \equiv i \pmod k$ and let $S_j := L_j \cup L_{j+1} \cup \dots \cup L_{j+k}$. Also fix A to be the vertex set of a connected component of $G[S_j]$. We define $U_{j,A}$ to be the ‘‘intersection’’ of U_j and A , i.e.,

$$U_{j,A} := \{C'_r \mid L(C_r) = j, C_r \subseteq A\} \cup (L_{j+1} \cap A) \cup \dots \cup (L_{j+k-1} \cap A) \cup \{C_r \mid L(C_r) = j+k, C_r \subseteq A\}.$$

We prove that

$$(4.2) \quad \sum_{p \in [t]: D_p \subseteq U_{j,A} \wedge \deg_H^-(D_p) \geq 2} \deg_H^-(D_p) - 1 \leq 13|W \cap A \cap L_j|.$$

First observe that this implies the lemma since, for each $p \in [t]$, there is some $j \equiv i \pmod k$ and some component A of the graph $G[S_j]$ such that $D_p \subseteq U_{j,A}$.

By Observation 4.2 it holds that $L_j \cap A$ are exactly those vertices incident to the exterior face of $G[A]$. In particular, $G[L_j \cap A]$ is connected and 1-outerplanar. Let B be the vertex set of a connected component of $G[A \setminus L_j]$. We define

$$N_j(B) := \{v \in L_j \mid \exists w \in B: \text{dist}_R(v, w) \leq 2\}.$$

It is easy to verify that $G[N_j[B]]$ is a cycle. More precisely, $N_j(B)$ forms a face cycle of $G[L_j \cap A]$.

For a connected component B of $G[A \setminus L_j]$ define the *articulation degree* $\text{adeg}(B)$ to be the number of articulation points C_r such that $N_j[B] \cap C_r \neq \emptyset$. We argue that

$$\sum_B \max\{\text{adeg}(B) - 1, 0\} \leq 13|W \cap A \cap L_j|$$

where B ranges over connected components of $G[A \setminus L_j]$. Note that this implies Equation (4.2). For a component B it holds that $\text{adeg}(B) \leq \max\{|N_j(B) \cap W|, 1\}$. Hence, it suffices to prove that

$$\sum_B \max\{|N_j(B) \cap W| - 1, 0\} \leq 13|W \cap A \cap L_j|.$$

But this follows from Lemma 4.1 since each $N_j(B)$ forms a face cycle of $G[L_j \cap A]$. \square

Now let \mathcal{C} be a set of articulation points such that $|\mathcal{C}| \leq 13|W|$ and $\deg_{H-\mathcal{C}}^-(D_p) \leq 1$ for all $p \in [t]$. Then $H - \mathcal{C}$ is a tree. This means there a tree decomposition of $(G/E(G[V_i \setminus W])) - (\mathcal{C} \cup W)$ of width $3k + 2$ by stitching the tree decompositions for the components D_p , $p \in [t]$, together along the tree structure of $H - \mathcal{C}$. Adding the vertices from $\mathcal{C} \cup W$ to all bags then gives a tree decomposition of $G/E(G[V_i \setminus W])$ of width $3k + 14|W| + 2$. This completes the proof of Theorem 4.2.

4.2 Guessing Bodies of Segments Next, we concern ourselves with the guessing of the bodies which is related to problems defined over 2-connected components. In particular, we prove Lemma 1.1. More precisely, in Lemma 1.1, the sets V_i are obtained from the Contraction Decomposition Theorem (Theorem 4.2) proved in the last subsection. Here, we concern ourselves with the problem of finding the body sets \mathcal{B}_i . Towards this end, we assume that we are already given a suitable partition of the vertex set (later, this partition is replaced with the partition computed in Theorem 4.2).

So let G be a graph and fix $Z \subseteq V(G)$ to be a *solution* of size $|Z| \leq k$. Also fix a partition $V(G) = V_1 \uplus \dots \uplus V_\ell$.

We refer to the connected components of the graph $G[V_i \setminus Z]$ as the *i-segments*. A *segment* is an *i*-segment for some $i \in [\ell]$. Let \mathcal{I} denote the set of all segments.

In the following, we are interested in the 2-connected components of $G - Z$. We shall represent the 2-connected components of $G - Z$ as a pair (\mathfrak{F}, γ) where \mathfrak{F} is a forest and $\gamma: V(\mathfrak{F}) \rightarrow 2^{V(G)}$ is a function as follows. Let us first consider the block-cut tree \mathfrak{T}' which contains a node t_C for every 2-connected component C of $G - Z$, and a node s_v for every cut vertex v of $G - Z$. Also, there is an edge between t_C and s_v if and only if $v \in C$. It is well known that \mathfrak{T}' is a forest where each connected component of \mathfrak{T}' corresponds to some connected component of $G - Z$. For each connected component of \mathfrak{T}' we fix a unique root node t_C that corresponds to some 2-connected component of $G - Z$. For simplicity, let us fix a ‘‘canonical’’ rooting as follows. We assume that the vertices of the input graph G are ordered (the order is arbitrary). Now, we can order the 2-connected components of $G - Z$ lexicographically. For each connected component T of \mathfrak{T}' , we pick the lexicographically smallest 2-connected component as the root of T .

Now, \mathfrak{F} is obtained from \mathfrak{T}' by identifying each node s_v with its parent t_C , i.e., \mathfrak{F} has a node t_C for every 2-connected component of $G - Z$. We set $\gamma(t_C) = C$. For $t \in V(\mathfrak{F})$ define $\text{Desc}(t)$ to be the set of descendants of t , including t itself.

Now, consider a segment $I \in \mathcal{I}$. We need to define several objects based on the decomposition (\mathfrak{T}, γ) into the 2-connected components of $G - Z$. First, let $r_Z(I)$ denote the unique node $t \in V(\mathfrak{T})$ which is closest to a root node t_0 and for which $I \cap \gamma(t) \neq \emptyset$. Also, let $B_Z(I) := \{t \in V(\mathfrak{T}) \setminus \{r_Z(I)\} \mid I \cap \gamma(t) \neq \emptyset\}$ be the *body* of I . Observe that $r_Z(I) \notin B_Z(I)$. Moreover, let $W_Z(I) := \text{Desc}(r_Z(I)) \setminus \{r_Z(I)\}$ denote the set of all nodes that are descendants of $r_Z(I)$, excluding $r_Z(I)$. Observe that $r_Z(I) \notin W_Z(I)$, but $B_Z(I) \subseteq W_Z(I)$.

Finally, we define $\widehat{r}_Z(I) := \gamma(r_Z(I))$, $\widehat{B}_Z(I) := \bigcup_{t \in R_Z(I)} \gamma(t)$ and $\widehat{W}_Z(I) := \bigcup_{t \in W_Z(I)} \gamma(t)$ to be the corresponding sets of vertices in the graph G . To simplify notation, we will regularly omit the index Z if it is clear from context.

The next theorem allows us to compute the body sets. Actually, for technical reasons, the objects we guess are slightly more complicated and also need to contain information about the sets $\widehat{r}_Z(I) \cap I$.

THEOREM 4.4. *Let G be an H -minor-free graph and let V_1, \dots, V_ℓ be a partition of the vertex set $V(G)$. Also let $Z \subseteq V(G)$ be an (unknown) set of vertices of size $|Z| = k$ and let (\mathfrak{T}, γ) be the (rooted) decomposition into 2-connected components of $G - Z$.*

Then there is a constant c_H , an index $i \in [\ell]$, and a sequence of vertices $\bar{s} = (s_1, \dots, s_r) \in (V(G))^r$ of length $r = \mathcal{O}(c_H \frac{k}{\ell})$ such that the following conditions are satisfied:

(I) $V_i \cap Z = \{s_1, \dots, s_j\}$ for some $j \in \{0, \dots, r\}$, and

(II) *given $G, V_1, \dots, V_\ell, i, \bar{s}, j$ one can compute in time $n^{\mathcal{O}(c_H)}$, for each i -segment I , a family of sets $\mathcal{F}(I) \subseteq 2^{V(G)} \times 2^{V(G)}$ of size $|\mathcal{F}(I)| \leq n^{c_H}$ such that $(\widehat{B}_Z(I), \widehat{r}_Z(I) \cap I) \in \mathcal{F}(I)$.*

Observe that this is non-trivial since both $\widehat{B}_Z(I)$ and $\widehat{r}_Z(I) \cap I$ depend on Z which is only partially given to the algorithm. The remainder of this subsection is devoted to proving the theorem.

Let $d_H = \mathcal{O}(h\sqrt{\log h})$ denote the constant from Theorem 2.1, i.e., $\sum_{v \in V(G)} \deg_G(v) \leq d_H |V(G)|$ holds for every H -minor-free graph G .

LEMMA 4.4. *Let G be an H -minor-free graph and suppose $V(G) = U \uplus Z$ such that $\deg_G(u) > 2d_H$ for all $u \in U$. Then $|U| \leq |Z|$.*

Proof. Suppose towards a contradiction that $|U| > |Z|$. Then $|E(G)| \geq \frac{1}{2}|U|2d_H = d_H|U|$. Hence,

$$\sum_{v \in V(G)} \deg_G(v) = 2|E(G)| \geq 2d_H|U| > d_H|V(G)|$$

which is a contradiction to Theorem 2.1. \square

Now, let us fix an H -minor-free graph G , a partition V_1, \dots, V_ℓ of the vertex set $V(G)$, and an unknown solution set $Z \subseteq V(G)$ of size $|Z| = k$. Also, let (\mathfrak{T}, γ) denote the (rooted) decomposition into 2-connected components of $G - Z$.

LEMMA 4.5. *Let I_1, I_2 be two distinct segments. Then $B(I_1) \cap B(I_2) = \emptyset$.*

Proof. Suppose $t \in B(I_1)$ and let s be the parent node of t . Then $s \in B(I_1) \cup \{r(I_1)\}$. Let $v \in \gamma(t) \cap \gamma(s)$. Since $G[I_1]$ is connected it holds that $v \in I_1$. So $v \notin I_2$ since $I_1 \cap I_2 = \emptyset$. Hence, $t \notin B(I_2)$. \square

Let I be a segment. We define $\text{val}(I)$ to be the maximum number of paths from $\widehat{B}(I)$ to Z in the graph $G[\widehat{W}(I) \cup Z]$ that are pairwise disjoint except on $\widehat{B}(I)$. Observe that $\text{val}(I) = 0$ whenever $B(I) = \emptyset$.

LEMMA 4.6. *Suppose G is H -minor-free. Then there is a constant c_H such that*

$$\sum_{I \in \mathcal{I}} \max(\text{val}(I) - c_H, 0) \leq c_H \cdot |Z|.$$

Proof. We define $c_H := 6d_H$. Let $\mathcal{I}^* := \{I \in \mathcal{I} \mid \text{val}(I) > c_H\}$. We first partition \mathcal{I}^* into three sets $\mathcal{I}_0, \mathcal{I}_1$ and \mathcal{I}_2 such that, for all $\mu \in \{0, 1, 2\}$ and all $I, I' \in \mathcal{I}_\mu$ it holds that

$$r(I) \notin B(I')$$

and

$$r(I) \neq r(I') \Rightarrow E_{\mathfrak{T}}(B(I) \cup \{r(I)\}, B(I') \cup \{r(I')\}) = \emptyset.$$

Actually, let us remark that the first condition is implied by the second one, but we still state it explicitly for later reference.

The partition can be computed inductively as follows. Throughout the induction we maintain the property that segments I with the same root bag $r(I)$ are assigned to the same partition class. For each $I \in \mathcal{I}$ define the *height* of I to be the distance from $r(I)$ to the root of the corresponding connected component of \mathfrak{T} . First, we assign all segments of height 0 to the class \mathcal{I}_0 . Clearly, this meets the above requirements. Now suppose all segments up to height h have been partitioned and let I be a segment of height $h + 1$. Let $t := r(I)$ and let s be the parent of t . First suppose $t \in B(I')$ for some segment I' of height at most h . By Lemma 4.5 there is a unique I' of height at most h such that $t \in B(I')$. Note that $s \in B(I') \cup \{r(I')\}$. Suppose $I' \in \mathcal{I}_{\mu'}$. Moreover, by Lemma 4.5, there is at most one I'' such that $s \in B(I'')$. Assume $I'' \in \mathcal{B}_{\mu''}$ (if it exists). We assign I to a partition class \mathcal{I}_μ for some $\mu \in \{0, 1, 2\} \setminus \{\mu', \mu''\}$. Moreover, we assign all segments with the same root bag t to the same partition class. It can be easily checked that the requirements are satisfied.

In the following we argue that $\sum_{I \in \mathcal{I}_\mu} (\text{val}(I) - c_H) \leq \frac{c_H}{3} \cdot |Z|$ for all $\mu \in \{0, 1, 2\}$ which implies the lemma.

Without loss of generality assume $\mu = 0$. For each $I \in \mathcal{I}_0$ we associate a subset $U(I)$ of $V(G) \setminus Z$ as follows. In the first step, all elements from $I \cup \widehat{B}(I)$ are added to $U(I)$. Note that all sets $U(I)$ are connected and pairwise disjoint. Let X be those vertices that are not assigned so far. Next, all vertices from $\widehat{W}(I) \cap X$ that are reachable from $\widehat{B}(I)$ in the graph $G[X \cup \widehat{B}(I)]$ are added to $U(I)$. Observe that all sets $U(I)$ remain connected and pairwise disjoint.

Now consider the following graph F with vertex set $V(F) := \mathcal{I}_0 \uplus Z$ and edge set

$$E(F) := \{II' \mid I \neq I' \in \mathcal{I}_0, E_G(U(I), U(I')) \neq \emptyset\} \cup \{Iz \mid I \in \mathcal{I}_0, z \in Z, E_G(U(I), \{z\}) \neq \emptyset\}$$

(F is not a multigraph, i.e., there is for only a single edge between I and I' (resp. I and z) even if $E(U(I), U(I'))$ (resp. $E(U(I), \{z\})$) contains more than one element). Note that F is a minor of G .

We claim that $\text{val}(I) \leq \deg_F(I)$ for all $I \in \mathcal{I}_0$. Let us first complete the proof assuming the claim holds true. Then $|\mathcal{I}_0| \leq |Z|$ by Lemma 4.4. This means that

$$\sum_{I \in \mathcal{I}_0} (\text{val}(I) - c_H) \leq \sum_{v \in V(F)} \deg_F(v) \leq d_H |V(F)| \leq 2d_H |Z| \leq \frac{c_H}{3} |Z|.$$

To complete the proof it remains show that $\text{val}(I) \leq \deg_F(I)$ for all $I \in \mathcal{I}_0$. Let us fix some $I \in \mathcal{I}_0$ and a set of paths $P_1^I, \dots, P_{\text{val}(I)}^I$ from $\widehat{B}(I)$ to Z in the graph $G[\widehat{W}(I) \cup Z]$ that are pairwise disjoint except on $\widehat{B}(I)$. Without loss of generality assume that no internal vertex of a path P_i^I is contained in Z . Also define $Z(I) \subseteq Z$ to be set of all endpoints of the paths $P_1^I, \dots, P_{\text{val}(I)}^I$. For $z \in Z(I)$ we also write P_z^I for the unique path P_i^I which ends in z .

We say that $z \in Z(I)$ is *directly reachable from I* if P_z^I does not visit any of the sets $U(I')$ for $I \neq I' \in \mathcal{I}_0$. Let $Z^*(I) \subseteq Z(I)$ be those vertices directly reachable from I . Let $z \in Z^*(I)$. By definition, all internal vertices from P_z^I are contained in the set $U(I)$. Hence, $Iz \in E(F)$. On the other hand, for $z \in Z(I) \setminus Z^*(I)$, consider the first vertex w_z on P_z^I that is not contained in $U(I)$. Then $w_z \in I_z$ for some $I \neq I_z \in \mathcal{I}_0$. Hence, $w_z \in U(I_z)$ and $II_z \in E(F)$. It remains to argue that, for a second $z \neq z' \in Z(I) \setminus Z^*(I)$, we have that $I_z \neq I_{z'}$. Suppose otherwise. Clearly, $r(I) \neq r(I_z)$. By the properties of the set \mathcal{I}_0 , it holds that $r(I_z) \notin B(I)$. Since P_z^I and $P_{z'}^I$ are vertex-disjoint except on $\widehat{B}(I)$ we have that $w_z \neq w_{z'}$. Moreover, $w_z \in I_z \cap \widehat{r}(I_z)$ and $w_{z'} \in I_{z'} \cap \widehat{r}(I_{z'})$. Since P_z^I and $P_{z'}^I$ are vertex-disjoint except on $\widehat{B}(I)$ this is only possible if $r(I_z)$ is a child of some node $t \in B(I)$. Hence, $E_T(\{r(I_z)\}, B(I)) \neq \emptyset$ which contradicts the second property of the set \mathcal{I}_0 . Hence, $I_z \neq I_{z'}$ for all distinct $z, z' \in Z(I) \setminus Z^*(I)$.

Together, this means that $\deg_F(I) \geq |Z(I)|$ for all $I \in \mathcal{I}_0$ which completes the proof of the lemma. \square

For $i \in [\ell]$ we define

$$\text{val}^*(V_i) := \sum_{I \text{ is } i\text{-segment}} \max(\text{val}(I) - c_H, 0).$$

We claim there exists some $i \in [\ell]$ such that

1. $|V_i \cap Z| \leq 2\frac{k}{\ell}$, and
2. $\text{val}^*(V_i) \leq 2c_H\frac{k}{\ell}$.

Let $A_1 := \{i \in [\ell] \mid |V_i \cap Z| > 2\frac{k}{\ell}\}$ be the set of indices that violate the first condition. Then $k = |Z| > |A_1| \cdot 2\frac{k}{\ell}$ which means that $|A_1| < \frac{\ell}{2}$.

Similarly, let $A_2 := \{i \in [\ell] \mid \text{val}^*(V_i) > 2c_H\frac{k}{\ell}\}$ be the set of indices that violate the second condition. By the lemma

$$c_H k = c_H |Z| \geq \sum_{i \in A_2} \text{val}^*(V_i) > |A_2| 2c_H \frac{k}{\ell}$$

which implies that $|A_2| < \frac{\ell}{2}$. Hence, there is some $i \in [\ell]$ such that neither $i \in A_1$ nor $i \in A_2$. Let $j := |V_i \cap Z|$ and define $\{s_1, \dots, s_j\} = V_i \cap Z$ (the vertices are enumerated arbitrarily).

Now let I be an i -segment and consider the graph $G[\widehat{W}(I) \cup Z]$. In this graph, there are at most $\text{val}(I)$ many paths from $\widehat{B}(I)$ to Z that are pairwise disjoint except on $\widehat{B}(I)$. Hence, by Menger's theorem, there is a set $S_I \subseteq \widehat{W}(I) \cup Z$ of size $|S_I| = \text{val}(I)$ such that $S_I \cap \widehat{B}(I) = \emptyset$ and every path from $\widehat{B}(I)$ to Z in the graph $G[\widehat{W}(I) \cup Z]$ visits some vertex from S_I . Suppose that $S_I = \{w_1^I, \dots, w_{\text{val}(I)}^I\}$.

We define

$$\{s_{j+1}, \dots, s_r\} = \{w_{c_H+1}^I, \dots, w_{\text{val}(I)}^I \mid I \text{ is } i \text{ segment with } \text{val}(I) > c_H\}.$$

Note that $r \leq |V_i \cap Z| + \text{val}^*(V_i) \leq 3(c_H + 1)\frac{k}{\ell}$. This completes the description of the sequence $\bar{s} = (s_1, \dots, s_r)$.

Clearly, Property (I) of Theorem 4.4 is satisfied. Hence, in order to complete the proof of Theorem 4.4 it remains to verify Property (II), i.e., we have to argue how to compute the families $\mathcal{F}(I)$ for all i -segments I . Clearly, given access to $G, V_1, \dots, V_\ell, i, \bar{s}, j$ we can compute the set of all i -segments in polynomial time. So fix I to be an i -segment. The algorithm first guesses $\text{val}(I)$ by iterating through all possible values.

If $\text{val}(I) \leq c_H$ the algorithm iterates through all subsets $S \subseteq V(G) \setminus I$ of size $\text{val}(I)$ and computes the decomposition $(\mathfrak{T}_S, \gamma_S)$ into 2-connected components of the graph $G - S$. Also, given a subset $S \subseteq V(G) \setminus I$ of size $\text{val}(I)$, the algorithm guesses a root node t_0^S for the connected component of $(\mathfrak{T}_S, \gamma_S)$ that contains I . Finally, let $r_S(I)$ denote the unique node $t \in V(\mathfrak{T}_S)$ which is closest to the root node t_0^S and for which $I \cap \gamma_S(t) \neq \emptyset$. Also, let $B_S(I) := \{t \in V(\mathfrak{T}_S) \setminus \{r_S(I)\} \mid I \cap \gamma_S(t) \neq \emptyset\}$. Finally, we add the pair

$$\left(\bigcup_{t \in B_S(I)} \gamma_S(t), \gamma_S(r_S(I)) \cap I \right)$$

to the family $\mathcal{F}(I)$. For $S = S_I$ together with a suitable choice of t_0^S we get that $\widehat{B}(I) = \bigcup_{t \in B_S(I)} \gamma_S(t)$.

For $\text{val}(I) > c_H$ the algorithm first guesses $L_I = \{w_{c_H+1}^I, \dots, w_{\text{val}(I)}^I\}$. By appropriately ordering the sequence \bar{s} and marking certain elements (for example, by repeating the element), this adds a multiplicative factor of at most n . Having guessed L_I the algorithm again iterates through all subsets of $S \subseteq V(G) \setminus I$ of size c_H and computes the decomposition $(\mathfrak{T}_S, \gamma_S)$ into 2-connected components of the graph $G - (S \cup L_I)$. Now the algorithm proceeds as in the previous case by guessing a root of $(\mathfrak{T}_S, \gamma_S)$ and adding the pair

$$\left(\bigcup_{t \in B_S(I)} \gamma_S(t), \gamma_S(r_S(I)) \cap I \right)$$

to the family $\mathcal{F}(I)$. Again, for $S = S_I \setminus L_I$ together with a suitable choice of t_0^S we get that $\widehat{B}(I) = \bigcup_{t \in B_S(I)} \beta(t)$.

This completes the proof of Theorem 4.4.

4.3 Obtaining Segmented Graphs with Body Sets Having proved Theorems 4.2 and 4.4, we can now combine both results to prove Lemma 1.1. Actually, we shall use a different formulation of Lemma 1.1 which is slightly more convenient due to technical reasons. We leave it as an exercise for the reader to derive Lemma 1.1 from Corollary 4.2 below (all algorithms making use of this result do so via Corollary 4.2). Towards this end, we first need to introduce some additional notation on segments that is also heavily used in the next section.

A *segmented graph* is a pair (G, \mathcal{I}) where G is a graph and $\mathcal{I} = \{I_1, \dots, I_\ell\}$ is a set of pairwise disjoint, connected subsets of $V(G)$. Moreover, we generally assume that each segment $I \in \mathcal{I}$ is equipped with a linear order $<_I$. We refer to the sets I_1, \dots, I_ℓ as the *segments* of (G, \mathcal{I}) . Also, define $V(\mathcal{I}) := \bigcup_{I \in \mathcal{I}} I$ as the set of vertices appearing in some segment. We define G/\mathcal{I} to be the graph obtained from G by contracting each segment to a single vertex. Note that G/\mathcal{I} is a minor of G since all sets $I \in \mathcal{I}$ are connected. We always use v_I to denote the vertex of G/\mathcal{I} that corresponds to segment I . Also, $V_{\mathcal{I}} := \{v_I \mid I \in \mathcal{I}\}$. For $U \subseteq V(G)$ we use $\text{shr}(U)$ to denote the set of all vertices $v \in V(G/\mathcal{I})$ that correspond to some $u \in U$. In particular, $v_I \in \text{shr}(U)$ if $U \cap I \neq \emptyset$. In the other direction, for $U \subseteq V(G/\mathcal{I})$, we use $\text{ext}(U)$ to denote the set of all vertices $v \in V(G)$ that correspond to some $u \in U$. In particular, if $v_I \in U$ then $I \subseteq \text{ext}(U)$. If $U = \{u\}$ consists of a single vertex, then we also write $\text{shr}(u)$ and $\text{ext}(u)$ instead of $\text{shr}(\{u\})$ and $\text{ext}(\{u\})$.

COROLLARY 4.2. *There is a polynomial-time algorithm that, given a planar graph G , an integer k , and a sequence of vertices $(v_1, \dots, v_r) \in (V(G))^r$, computes a set of pairwise vertex-disjoint segments \mathcal{I} and a function \mathcal{F} mapping every $I \in \mathcal{I}$ to a set $\mathcal{F}(I) \subseteq 2^{V(G)} \times 2^{V(G)}$ of size $|\mathcal{F}(I)| = n^{\mathcal{O}(1)}$ such that the following property is satisfied:*

For every $Z \subseteq V(G)$ of size $|Z| \leq k$ there is some sequence $(v_1, \dots, v_r) \in (V(G))^r$ of length $r \in \mathcal{O}(\sqrt{k})$ such that, if $(\mathcal{I}, \mathcal{F})$ is the output of the algorithm on input $(G, k, (v_1, \dots, v_r))$, then

1. $V(\mathcal{I}) \cap Z = \emptyset$,
2. $\text{tw}(G/\mathcal{I}) = \mathcal{O}(\sqrt{k})$, and
3. $(\widehat{B}_Z(I), \widehat{r}_Z(I) \cap I) \in \mathcal{F}(I)$ for every $I \in \mathcal{I}$.

Proof. Let $(G, k, (v_1, \dots, v_r))$ be the input to the algorithm. The algorithm first fixes an arbitrary numbering of the vertices of G , i.e., $V(G) = \{u_1, \dots, u_n\}$. Let $\ell := \lceil \sqrt{k} \rceil$. The algorithm applies Theorem 4.2 and obtains a partition V_1, \dots, V_ℓ of $V(G)$. Let $i \in [\ell]$ be the unique index such that $v_1 \in V_i$. Also, let $j \in \{0, \dots, n-1\}$ such that $v_2 = u_{j+1}$. We define $Z_i := \{v_3, \dots, v_{j+2}\}$ and compute \mathcal{I} as the set of connected components of $G[V_i \setminus Z_i]$. Also, we define $\bar{s} := (v_3, \dots, v_r)$. Finally, the algorithm computes the function \mathcal{F} using the algorithm from Theorem 4.4, Item (II) on input $(G, V_1, \dots, V_\ell, i, \bar{s}, j)$. Clearly, this algorithm runs in polynomial time.

So let $Z \subseteq V(G)$ of size $|Z| \leq k$. We need to argue about the existence of a sequence $(v_1, \dots, v_r) \in (V(G))^r$ of length $r \in \mathcal{O}(\sqrt{k})$ satisfying the properties stated in the corollary. By Theorem 4.4 there is some $i \in [\ell]$, a sequence $\bar{s} = (s_1, \dots, s_r) \in (V(G))^r$ for $r \in \mathcal{O}(\sqrt{k})$, and an integer $j \in [r]$ such that $V_i \cap Z = \{s_1, \dots, s_j\}$. We pick $v_1 \in V_i$ arbitrarily, $v_2 := u_{j+1}$, and $v_{p+2} := s_p$ for all $p \in [r]$. Let $(\mathcal{I}, \mathcal{F})$ denote the output of the above algorithm on input $(G, k, (v_1, \dots, v_r))$. Clearly, $V(\mathcal{I}) \cap Z = \emptyset$ by definition. Also, $\text{tw}(G/\mathcal{I}) = \mathcal{O}(\ell + |V_i \cap Z|) = \mathcal{O}(\sqrt{k})$. Finally, $(\widehat{B}_Z(I), \widehat{r}_Z(I) \cap I) \in \mathcal{F}(I)$ for every $I \in \mathcal{I}$ by Theorem 4.4, Property (II). \square

For later reference, we also formulate a version of the last corollary for H -minor-free graphs assuming Conjecture 1.1 holds.

COROLLARY 4.3. *Assuming Conjecture 1.1, there is an algorithm that, given an H -minor-free graph G , an integer k , and a sequence of vertices $(v_1, \dots, v_r) \in (V(G))^r$, computes a set of pairwise vertex-disjoint segments \mathcal{I} and a function \mathcal{F} mapping every $I \in \mathcal{I}$ to a set $\mathcal{F}(I) \subseteq 2^{V(G)} \times 2^{V(G)}$ of size $|\mathcal{F}(I)| = n^{c_H}$ for some constant c_H depending only on H such that the following property is satisfied:*

For every $Z \subseteq V(G)$ of size $|Z| \leq k$ there is some sequence $(v_1, \dots, v_r) \in (V(G))^r$ of length $r \in \mathcal{O}(c_H \sqrt{k})$ such that, if $(\mathcal{I}, \mathcal{F})$ is the output of the algorithm on input $(G, k, (v_1, \dots, v_r))$, then

1. $V(\mathcal{I}) \cap Z = \emptyset$,
2. $\text{tw}(G/\mathcal{I}) = \mathcal{O}(\sqrt{k})$, and
3. $(\widehat{B}_Z(I), \widehat{r}_Z(I) \cap I) \in \mathcal{F}(I)$ for every $I \in \mathcal{I}$.

Moreover, the algorithm runs in time $n^{\mathcal{O}(c_H)}$.

Proof. The proof is analogous to the proof of Corollary 4.2 replacing the application of Theorem 4.2 with Conjecture 1.1. \square

5 Algorithms for CSP Deletion Problems on Planar Graphs

In this section, we discuss the proofs of Theorems 3.1, 3.2 and 3.3. All proofs follow essentially the same high-level strategy. First, we use the Contraction Decomposition Theorem to partition the vertex or edge set of the input constraint graph (depending on the problem in question). Then, we guess some partition class that has small intersection with the solution as well as the intersection of the solution with said partition class. This gives rise to a segmented graph (G, \mathcal{I}) where the segments are the connected components of the chosen partition class after removing all solution vertices. Now, we translate the initial Permutation-CSP-instance (with constraint graph G) to an equivalent binary CSP-instance with constraint graph G/\mathcal{I} . For 2CONN PERM CSP VERTEX DELETION WITH SIZE CONSTRAINTS, this translation also builds on Theorem 4.4 to guess the body sets of the segments. Finally, we use dynamic programming to solve the constructed binary CSP-instance over constraint graph G/\mathcal{I} exploiting that G/\mathcal{I} has small treewidth.

We remark that the algorithms presented in Sections 5.1 - 5.3 are standard algorithms, and the corresponding results should not be surprising to a reader familiar with the concept of contraction decompositions as well as algorithmic approaches to CSPs. Indeed, the main algorithmic contribution of this section is the subexponential algorithm for 2CONN PERM CSP VERTEX DELETION WITH SIZE CONSTRAINTS which is presented in Section 5.4.

5.1 CSPs with Size Constraints on Graphs of Bounded Treewidth We start by implementing the last step, i.e., we provide an algorithm deciding satisfiability of binary CSPs $\Gamma = (X, D, \mathcal{C})$ in time $|D|^{\mathcal{O}(\text{tw}(G(\Gamma)))}|X|^{\mathcal{O}(1)}$. It is well-known that such an algorithm can be obtained via a simple dynamic programming strategy along a tree decomposition of G . However, for Theorems 3.2 and 3.3, we also need to be able to take size constraints into account. Here, our strategy is similar, i.e., we translate the size constraints on G to suitable size constraints over G/\mathcal{I} . To be able to use the same subroutine for both Theorem 3.2 and 3.3, the size constraints that we allow on the contracted graph need to be fairly general. In the following, we define suitable size constraints and provide a dynamic programming algorithm for input instances of small treewidth. We remark that the size constraints we introduce may look somewhat unnatural at first glance, but they are designed primarily to fit our needs when proving Theorems 3.2 and 3.3.

Let $\Gamma = (X, D, \mathcal{C})$ be a binary CSP instance. A *global size constraint* is a triple (w, q, op) where $w: X \times D \rightarrow \mathbb{N}$ is a weight function, $q \in \mathbb{N}$ and $\text{op} \in \{\leq, \geq\}$. An assignment $\alpha: X \rightarrow D$ satisfies (w, q) if

$$\left(\sum_{v \in X} w(v, \alpha(v)), q \right) \in \text{op}.$$

Let $F \subseteq D^2$ such that F is symmetric (i.e., $(a, b) \in F$ if and only if $(b, a) \in F$ for all $a, b \in D$) and let \mathcal{D} be a partition of D . An (F, \mathcal{D}) -*local size constraint* is a tuple $(w, w_{\mathcal{D}}, q, \text{op})$ where $w: X \times D \rightarrow \mathbb{N}$ and $w_{\mathcal{D}}: \mathcal{D} \rightarrow \mathbb{N}$ are weight functions, $q \in \mathbb{N}$ and $\text{op} \in \{\leq, \geq\}$. Let $\alpha: X \rightarrow D$ be an assignment. Let $G := G(\Gamma)$ be the constraint graph. We define H_{α} to be the subgraph of G with vertex set $V(H_{\alpha}) := V(G)$ and edge set

$$E(H_{\alpha}) := \{vw \in E(G) \mid \alpha(v)\alpha(w) \in F\}.$$

Let C_1, \dots, C_{ℓ} denote the connected components of H . We say that $\alpha: X \rightarrow D$ satisfies $(w, w_{\mathcal{D}}, q, \text{op})$ if, for every $i \in [\ell]$, there is some $D_i \in \mathcal{D}$ such that

$$\alpha(v) \in D_i$$

for every $v \in C_i$ and

$$\left(w_{\mathcal{D}}(D_i) + \sum_{v \in C_i} w(v, \alpha(v)), q \right) \in \text{op}.$$

Let $\mathcal{S}_{\text{global}}$ be a set of global size constraints and $\mathcal{S}_{\text{local}}$ be a set of (F, \mathcal{D}) -local size constraints. We say that $(\Gamma, \mathcal{S}_{\text{global}}, \mathcal{S}_{\text{local}})$ is *satisfiable* if there is an assignment $\alpha: X \rightarrow D$ that satisfies Γ , all global constraints

$(w, q, \text{op}) \in \mathcal{S}_{\text{global}}$ and all (F, \mathcal{D}) -local size constraints $(w, q, \text{op}) \in \mathcal{S}_{\text{local}}$. We define

$$\|\mathcal{S}_{\text{global}}\| := \prod_{(w, q, \text{op}) \in \mathcal{S}_{\text{global}}} (2 + q)$$

and similarly,

$$\|\mathcal{S}_{\text{local}}\| := \prod_{(w, w_{\mathcal{D}}, q, \text{op}) \in \mathcal{S}_{\text{local}}} (2 + q).$$

THEOREM 5.1. *Let $\Gamma = (X, D, \mathcal{C})$ be a binary CSP instance and let $F \subseteq \binom{D}{2}$ and \mathcal{D} be a partition of D . Also, let $\mathcal{S}_{\text{global}}$ be a set of global size constraints and $\mathcal{S}_{\text{local}}$ be a set of (F, \mathcal{D}) -local size constraints. Moreover, let $k := \text{tw}(G(\Gamma))$. Then there is an algorithm which decides whether $(\Gamma, \mathcal{S}_{\text{global}}, \mathcal{S}_{\text{local}})$ is satisfiable and runs in time*

$$(|D| + k + \|\mathcal{S}_{\text{global}}\| + \|\mathcal{S}_{\text{local}}\|)^{\mathcal{O}(k)} |X|^{\mathcal{O}(1)}.$$

We also need a second result for binary CSPs on graphs of bounded treewidth. Let $\Gamma = (X, D, \mathcal{C})$ be a binary CSP and let $\alpha: X \rightarrow D$ be an assignment. Also, let $w: X^2 \times D^2 \rightarrow \mathbb{N}$ such that $w(x, y, a, b) = w(y, x, b, a)$ for all $x, y \in X$ and $a, b \in D$. We define the *cost* of α as

$$\text{cost}_{\Gamma}(\alpha) := \sum_{x, y \in X: \alpha \text{ violates some constraint } ((x, y), R) \in \mathcal{C}} w(x, y, \alpha(x), \alpha(y)).$$

We remark that the sum also covers unary constraints by setting $x = y$.

THEOREM 5.2. *Let $\Gamma = (X, D, \mathcal{C})$ be a binary CSP instance, $w: X^2 \times D^2 \rightarrow \mathbb{N}$ such that $w(x, y, a, b) = w(y, x, b, a)$ for all $x, y \in X$ and $a, b \in D$, and $m \geq 0$. Also define $k := \text{tw}(G(\Gamma))$. Then there is an algorithm which decides whether there is an assignment $\alpha: X \rightarrow D$ of cost $\text{cost}_{\Gamma}(\alpha) \leq m$ in time*

$$|D|^{\mathcal{O}(k)} |X|^{\mathcal{O}(1)}.$$

5.2 Permutation CSP Edge Deletion Now, we are ready to prove the main algorithmic results from Section 3. In this subsection, we start by proving Theorem 3.1. We restate the problem as well as the main result.

PERM CSP EDGE DELETION

Parameter: k

Input: A Permutation-CSP-instance $\Gamma = (X, D, \mathcal{C})$, a set of undeletable edges $U \subseteq E(G(\Gamma))$, and an integer k

Question: Is there a set $Z \subseteq E(G(\Gamma)) \setminus U$ such that $|Z| \leq k$ and $\Gamma - Z := (X, D, \mathcal{C} - Z)$ is satisfiable?

THEOREM 5.3. *There is an algorithm solving H -MINOR-FREE PERM CSP EDGE DELETION in time $(|X| + |D|)^{\mathcal{O}(c_H \sqrt{k})}$ where c_H is a constant depending only on H .*

Proof. Let $\Gamma = (X, D, \mathcal{C})$ be a Permutation-CSP-instance and let $G := G(\Gamma)$. By combining constraints over the same variables, we may assume without loss of generality that, for every $uv \in E(G)$, there is at most one constraint over variables u and v . Also suppose $Z \subseteq E(G) \setminus U$ is a solution, i.e., $|Z| \leq k$ and $\Gamma - Z = (X, D, \mathcal{C} - Z)$ is satisfiable. Let $\ell := \lceil \sqrt{k} \rceil$. Let E_1, \dots, E_{ℓ} be the partition of the edge $E(G)$ computed by Corollary 4.1. Then there is some $i \in [\ell]$ such that $|E_i \cap Z| \leq \sqrt{k}$. Hence, at an additional multiplicative cost of $\ell |E(G)|^{\sqrt{k}} = |X|^{\mathcal{O}(\sqrt{k})}$, the algorithm can guess i as well as the set $E_i \cap Z$. We set $\Gamma' := (X, D, \mathcal{C} - (E_i \cap Z))$ and $k' := k - |E_i \cap Z|$. Hence, it suffices to determine whether (Γ', k') is a YES-instance.

Let I_1, \dots, I_m denote the connected components of $G[E_i \setminus Z]$ and let $\mathcal{I} := \{I_1, \dots, I_m\}$. Also, for each $I \in \mathcal{I}$, we fix an arbitrary linear order $<_I$ on the set I . This gives rise to a segmented graph (G, \mathcal{I}) . We have that $\text{tw}(G/\mathcal{I}) \leq c_H \sqrt{k}$ by Corollary 4.1 for some constant c_H that only depends on H . Now, the central idea is to translate Γ' to a binary CSP instance over the graph G/\mathcal{I} .

Let $\Gamma^* := (X^*, D, \mathcal{C}^*)$ where $X^* := V(G/\mathcal{I})$, and a set of constraints \mathcal{C}^* defined as follows. For $a \in D$ and $I \in \mathcal{I}$, we define $\alpha_{I,a}: I \rightarrow D$ to be a satisfying assignment such that

- $\alpha_{I,a}(v) = a$ where v denotes the minimal element of I with respect to $<_I$, and
- $\alpha_{I,a}$ satisfies all constraints $((u, v), R) \in \mathcal{C}$ for which $u, v \in I$ and $u = v$ or $uv \in (E_i \setminus Z) \cup U$.

If such an assignment exists, we say that a is valid for I . Observe that, if a is valid for I , then the assignment $\alpha_{I,a}$ is unique since Γ is a Permutation-CSP-instance. To bound the number of violated constraints, we also define a weight function $w: X^2 \times D^2 \rightarrow \mathbb{N}$.

For every $I \in \mathcal{I}$, we introduce a unary constraint (v_I, R_I) to \mathcal{C}^* where

$$R_I = \{a \in D \mid a \text{ is valid for } I\}.$$

For every $a \in R_I$ we define $w(v_I, v_I, a, a)$ to be the number of edges which correspond to some constraint that is violated by $\alpha_{a,I}$ on $\Gamma[I]$.

For every $v \in X \setminus (\bigcup_{I \in \mathcal{I}} I)$ and unary constraint $(v, R) \in \mathcal{C}$, the constraint (v, R) is added to \mathcal{C}^* . Also, we set $w(v, v, a, a) := k' + 1$ (indicating that this constraint can not be violated). Now, let $((u, v), R) \in \mathcal{C}$ be a binary constraint. If $u, v \in X \setminus (\bigcup_{I \in \mathcal{I}} I)$ then we add the constraint $((u, v), R)$ to \mathcal{C}^* . If $uv \in U$ then we set $w(u, v, a, b) := k' + 1$ for all $a, b \in D$. Otherwise, $w(u, v, a, b) := 1$. If $u \in X \setminus (\bigcup_{I \in \mathcal{I}} I)$ and $v \in I$ for some $I \in \mathcal{I}$ then we add the constraint

$$\left((u, v_I), \{(a, a') \mid a \in D, a' \in R_I, (a, \alpha_{a', I}(v)) \in R\} \right).$$

Also, $w(u, v_I, a, b)$ denotes the number of such constraints over variables u and v_I that are violated when assigning a with u and b to v_I . If some undeletable constraint is violated, then we set $w(u, v_I, a, b) := k' + 1$. If $v \in X \setminus (\bigcup_{I \in \mathcal{I}} I)$ and $u \in I$ we proceed analogously. Finally, if $u \in I$ and $v \in I'$ for some distinct $I, I' \in \mathcal{I}$ then we add the constraint

$$\left((v_I, v_{I'}), \{(a, a') \mid a \in R_I, a' \in R_{I'}, (\alpha_{a, I}(v), \alpha_{a', I'}(w)) \in R\} \right).$$

Again, $w(u, v_I, a, b)$ denotes the number of such constraints over variables u and v_I that are violated when assigning a with u and b to v_I . If some undeletable constraint is violated, then we set $w(u, v_I, a, b) := k' + 1$.

Now, it is easy to see that (Γ', k') is a YES-instance if and only if there is an assignment $\alpha^*: X^* \rightarrow D$ of cost $\text{cost}_{\Gamma^*}(\alpha^*) \leq k'$. By Theorem 5.2, the latter can be checked in time $|D|^{\mathcal{O}(\text{tw}(G/\mathcal{I}))} |X|^{\mathcal{O}(1)} = |D|^{\mathcal{O}(c_H \sqrt{k})} |X|^{\mathcal{O}(1)}$. In total, this gives a running time of

$$|X|^{\mathcal{O}(\sqrt{k})} \cdot |D|^{\mathcal{O}(c_H \sqrt{k})} |X|^{\mathcal{O}(1)} = (|X| + |D|)^{\mathcal{O}(c_H \sqrt{k})}.$$

□

5.3 Permutation CSP Deletion with Size Constraints Next, we turn to the proof of Theorem 3.2. Again, we start by restating the problem and the main result. Also, we provide a variant of Theorem 3.2 for H -minor-free graphs assuming Conjecture 1.1 holds.

Let $\Gamma = (X, D, \mathcal{C})$ be a binary CSP instance. A *1cc-size constraint* is a triple (w, q, op) where $w: X \times D \rightarrow \mathbb{N}$ is a weight function, $q \in \mathbb{N}$, and $\text{op} \in \{\leq, \geq\}$. An assignment $\alpha: X \rightarrow D$ satisfies (w, q, op) on Γ if

$$\left(\sum_{v \in C} w(v, \alpha(v)), q \right) \in \text{op}$$

for every connected component C of $G(\Gamma)$.

A *Permutation-CSP-instance with 1cc-size constraints* is a pair (Γ, \mathcal{S}) where $\Gamma = (X, D, \mathcal{C})$ is a Permutation-CSP-instance and \mathcal{S} is a set of 1cc-size constraints. We say that (Γ, \mathcal{S}) is *satisfiable* if there is an assignment $\alpha: X \rightarrow D$ which satisfies Γ as well as every constraint $(w, q, \text{op}) \in \mathcal{S}$ on Γ . Also, we define

$$\|\mathcal{S}\| := \prod_{(w, q, \text{op}) \in \mathcal{S}} (2 + q).$$

PERM CSP VERTEX DELETION WITH SIZE CONSTRAINTS

Parameter: k

Input: A Permutation-CSP-instance with 1cc-size constraints (Γ, \mathcal{S}) with $\Gamma = (X, D, \mathcal{C})$, a set of undeletable variables $U \subseteq X$, and an integer k .

Question: Is there a set $Z \subseteq X \setminus U$ such that $|Z| \leq k$ and $(\Gamma[X \setminus Z], \mathcal{S})$ is satisfiable?

THEOREM 5.4. *There is an algorithm solving PLANAR PERM CSP VERTEX DELETION WITH SIZE CONSTRAINTS in time $(|X| + |D| + \|\mathcal{S}\|)^{\mathcal{O}(\sqrt{k})}$.*

Moreover, assuming Conjecture 1.1, there is an algorithm solving H-MINOR-FREE PERM CSP VERTEX DELETION WITH SIZE CONSTRAINTS in time $(|X| + |D| + \|\mathcal{S}\|)^{\mathcal{O}(c_H \sqrt{k})}$ where c_H is a constant that only depends on H .

Proof. We focus on the first statement. Let $G = G(\Gamma)$ be the constraint graph and let $\ell := \lceil \sqrt{k} \rceil$. Also suppose $Z \subseteq V(G) \setminus U = X \setminus U$ is a solution, i.e., $|Z| \leq k$ and $(\Gamma[X \setminus Z], \mathcal{S})$ is satisfiable. Let V_1, \dots, V_ℓ be the partition of the vertex set $V(G)$ computed by Theorem 4.2. Then there is some $i \in [\ell]$ such that

$$|Z \cap V_i| \leq \frac{k}{\ell} \leq \sqrt{k}.$$

Let $Z_i := Z \cap V_i$. The algorithm first guesses $i \in [\ell]$ as well as the set Z_i by iterating over all possible choices. Observe that the guessing only increases the running time by a factor of $|X|^{(k/\ell)+1} = |X|^{\mathcal{O}(\sqrt{k})}$ which does not pose any problems. So for the remainder of the proof, suppose that i and Z_i are fixed.

Let I_1, \dots, I_m be the connected components of $G[V_i \setminus Z_i]$ and define $\mathcal{I} := \{I_1, \dots, I_m\}$. Also, for each $I \in \mathcal{I}$, we fix an arbitrary linear order $<_I$ on the set I . Overall, this gives rise to the segmented graph (G, \mathcal{I}) . Now, the central idea is to translate the instance (Γ, \mathcal{S}) to a binary CSP instance with global and local size constraints over the graph G/\mathcal{I} .

Let $\Gamma^* := (X^*, D^*, \mathcal{C}^*)$ where $X^* := V(G/\mathcal{I})$, $D^* := D \uplus \{\text{sol}\}$, and a set of constraints \mathcal{C}^* defined as follows. For $I \in \mathcal{I}$ and $a \in D$, we define $\alpha_{I,a}: I \rightarrow D$ to be an assignment such that

- $\alpha_{I,a}(v) = a$ where v denotes the minimal element of I with respect to $<_I$,
- $\alpha_{I,a}$ satisfies all unary constraints $(v, R) \in \mathcal{C}$ for which $v \in I$, and
- $\alpha_{I,a}$ satisfies all binary constraints $((v, w), R) \in \mathcal{C}$ for which $v, w \in I$ and $vw \in E(G)$.

If such an assignment exists, we say that a is valid for I . Observe that, if a is valid for I , then the assignment $\alpha_{I,a}$ is unique since Γ is a Permutation-CSP-instance and $G[I]$ is connected.

For every $I \in \mathcal{I}$, we introduce a unary constraint (v_I, R_I) to \mathcal{C}^* where

$$R_I = \{a \in D \mid a \text{ is valid for } I\}.$$

For every $v \in X \setminus (Z_i \cup \bigcup_{I \in \mathcal{I}} I)$ and unary constraint $(v, R) \in \mathcal{C}$, the constraint $(v, R \cup \{\text{sol}\})$ is added to \mathcal{C}^* . For every $v \in Z_i$, we introduce the unary constraint $(v, \{\text{sol}\})$ to \mathcal{C}^* . Also, for every $v \in U \setminus (\bigcup_{I \in \mathcal{I}} I)$, we introduce the unary constraint (v, D) to \mathcal{C}^* . Now, let $((v, w), R) \in \mathcal{C}$ be a binary constraint. If $v, w \in X \setminus (\bigcup_{I \in \mathcal{I}} I)$ then we add the constraint

$$\left((v, w), R \cup (D \times \{\text{sol}\}) \cup (\{\text{sol}\} \times D) \cup (\{\text{sol}\} \times \{\text{sol}\}) \right)$$

to \mathcal{C}^* . If $v \in X \setminus (\bigcup_{I \in \mathcal{I}} I)$ and $w \in I$ for some $I \in \mathcal{I}$ then we add the constraint

$$\left((v, v_I), \{(a, a') \mid a \in D, a' \in R_I, (a, \alpha_{a', I}(w)) \in R\} \cup (\{\text{sol}\} \times R_I) \right)$$

If $w \in X \setminus (\bigcup_{I \in \mathcal{I}} I)$ and $v \in I$ we proceed analogously. Finally, if $v \in I$ and $w \in I'$ for some distinct $I, I' \in \mathcal{I}$ then we add the constraint

$$\left((v_I, v_{I'}), \{(a, a') \mid a \in R_I, a' \in R_{I'}, (\alpha_{a, I}(v), \alpha_{a', I'}(w)) \in R\} \right).$$

Next, we define the size constraints for Γ^* . First, we add a global size constraint to force that at most k variables $x \in X^*$ are assigned the value sol . More precisely, we define $\mathcal{S}_{\text{global}} := \{(w_{\text{sol}}, k, \leq)\}$ where $w_{\text{sol}}: X^* \times D^* \rightarrow \mathbb{N}$ is defined via $w_{\text{sol}}(v, \text{sol}) = 1$ and $w_{\text{sol}}(v, a) = 0$ for every $a \in D$ and $v \in X^*$. For the local constraints, we define $F := \{dd' \mid d, d' \in D\} \subseteq (D^*)^2$ and $\mathcal{D} := \{D^*\}$. Also, we define $0_{\mathcal{D}}$ to denote the function with domain $\mathcal{D} = \{D^*\}$ that maps D^* to value 0.

We translate every lcc-size constraint $(w, q, \text{op}) \in \mathcal{S}$ into an F -local constraint $(w^*, 0_{\mathcal{D}}, q, \text{op}) \in \mathcal{S}_{\text{local}}$ as follows. We set

- $w^*(v, \text{sol}) := \begin{cases} q & \text{if } \text{op} = \geq \\ 0 & \text{if } \text{op} = \leq \end{cases}$ for all $v \in X^*$,
- $w^*(v, a) := w(v, a)$ for all $a \in D$ and $v \in X^* \setminus V_{\mathcal{I}}$, and
- $w^*(v_I, a) := \begin{cases} \sum_{u \in I} w(u, \alpha_{a,I}(u)) & \text{if } a \text{ is valid for } I \\ 0 & \text{otherwise} \end{cases}$ for all $a \in D$ and $v_I \in V_{\mathcal{I}}$.

CLAIM 5.1. $(\Gamma, \mathcal{S}, U, k)$ is a YES-instance if and only if $(\Gamma^*, \mathcal{S}_{\text{global}}, \mathcal{S}_{\text{local}})$ is satisfiable.

Proof. First suppose $(\Gamma, \mathcal{S}, U, k)$ is a YES-instance, i.e., there is a set $Z \subseteq X \setminus U$ of size $|Z| \leq k$ and an assignment $\alpha: X \setminus Z \rightarrow D$ that satisfies $\Gamma[X \setminus Z]$ as well as every 1cc-size constraint $(w, q, \text{op}) \in \mathcal{S}$ on $\Gamma[X \setminus Z]$. Moreover, by the above assumptions, $Z \cap V_i = Z_i$. We construct an assignment $\alpha^*: X^* \rightarrow D^*$ defined via

- $\alpha^*(v_I) := \alpha(v)$ where v is the minimal element of I with respect to $<_I$,
- $\alpha^*(v) := \text{sol}$ for every $v \in Z$, and
- $\alpha^*(v) := \alpha(v)$ for all every $v \in X \setminus (Z \cup \bigcup_{I \in \mathcal{I}} I)$.

We claim that α^* satisfies $(\Gamma^*, \mathcal{S}_{\text{global}}, \mathcal{S}_{\text{local}})$. First of all, for every $I \in \mathcal{I}$, $\alpha(v)$ is valid for I where v is the minimal element of I with respect to $<_I$. In particular, the unary constraint (v_I, R_I) is satisfied. Moreover,

$$\alpha_{\alpha(v), I}(w) = \alpha(w)$$

for every $w \in I$. This implies that α^* satisfies all constraints in the set \mathcal{C}^* . Hence, it remains to consider the size constraints. Clearly, the global size constraint $(w_{\text{sol}}, k, \leq)$ is satisfied since $|Z| \leq k$. So consider some constraint $(w, q, \text{op}) \in \mathcal{S}$ and its corresponding (F, \mathcal{D}) -local constraint $(w^*, 0_{\mathcal{D}}, q, \text{op})$. Let $G^* := G(\Gamma^*)$ be the constraint graph of Γ^* and define H^* to be the graph with $V(H^*) := V(G^*) = X^*$ and

$$E(H^*) := \{vw \mid \alpha^*(v)\alpha^*(w) \in F\}$$

Let C_1^*, \dots, C_ℓ^* denote the connected components of H^* . Fix $i \in [\ell]$. Since $\mathcal{D} = \{D^*\}$ there is clearly some $D_i \in \mathcal{D}$ such that $\alpha^*(v) \in D_i$ for every $v \in C_i^*$. So we need to show that

$$(5.3) \quad \left(\sum_{v \in C_i^*} w^*(v, \alpha^*(v)), q \right) \in \text{op}.$$

If there is some $v \in C_i$ such that $\alpha^*(v) = \text{sol}$ then $C_i = \{v\}$ by definition of the set F . In particular, Equation (5.3) is satisfied by definition of w^* . Otherwise, let $C_i := \text{ext}(C_i^*)$ be the corresponding set of vertices in G . Observe that C_i is a connected component of $G - Z$. Also, for $I \in \mathcal{I}$, define $a_I \in D$ such that $\alpha^*(v_I) = a_I$. Hence,

$$\begin{aligned} \sum_{v \in C_i^*} w^*(v, \alpha^*(v)) &= \left(\sum_{v_I \in C_i^* \cap V_{\mathcal{I}}} \sum_{u \in I} w(u, \alpha_{a_I, I}(u)) \right) + \sum_{v \in C_i^* \setminus V_{\mathcal{I}}} w(v, \alpha(v)) \\ &= \left(\sum_{v_I \in C_i^* \cap V_{\mathcal{I}}} \sum_{u \in I} w(u, \alpha(u)) \right) + \sum_{v \in C_i^* \setminus V_{\mathcal{I}}} w(v, \alpha(v)) \\ &= \sum_{v \in C_i} w(v, \alpha(v)). \end{aligned}$$

So Equation (5.3) is satisfied since α satisfies the 1cc-size constraint (w, q, op) on $\Gamma[X \setminus Z]$.

For the other direction, let $\alpha^*: X^* \rightarrow D^*$ be a satisfying assignment for $(\Gamma^*, \mathcal{S}_{\text{global}}, \mathcal{S}_{\text{local}})$. Let $Z := \{v \in X^* \mid \alpha^*(v) = \text{sol}\}$. First observe that $Z \subseteq X$ since segment vertices can not be assigned the value sol due to the unary constraints. Also, $Z \cap V_i = Z_i$ and $Z \cap U = \emptyset$ again by the unary constraints. Moreover, $|Z| \leq k$ by the global size constraint $(w_{\text{sol}}, k, \leq)$.

We construct an assignment $\alpha: X \setminus Z \rightarrow D$ via

- $\alpha(v) := \alpha^*(v)$ for every $v \in X \setminus (Z \cup \bigcup_{I \in \mathcal{I}} I)$, and
- $\alpha(v) := \alpha_{a_I, I}(v)$ for every $I \in \mathcal{I}$ and $v \in I$ where $a_I := \alpha^*(v_I)$.

It is not difficult to check that α satisfies $\Gamma[X \setminus Z]$. So let $(w, q, \text{op}) \in \mathcal{S}$ and C a connected component of $G - Z$. Also, let $C^* := \text{shr}(C)$ be the corresponding set in G/\mathcal{I} . Observe that $Z \cap I = \emptyset$ for every $I \in \mathcal{I}$ and hence, $I \subseteq C$ for every $I \in \mathcal{I}$ for which $C \cap I \neq \emptyset$. As before, let $G^* := G(\Gamma^*)$ be the constraint graph of Γ^* and define H^* to be the graph with $V(H^*) := V(G^*) = X^*$ and

$$E(H^*) := \{vw \mid \alpha^*(v)\alpha^*(w) \in F\}$$

Then C^* is a connected component of H^* . Using the same calculations as above, it follows that

$$\left(\sum_{v \in C} w(v, \alpha(v)), q \right) \in \text{op}$$

because

$$\left(\sum_{v \in C^*} w^*(v, \alpha^*(v)), q \right) \in \text{op}$$

due to the fact that α^* satisfies the (F, \mathcal{D}) -local constraint $(w^*, 0_{\mathcal{D}}, q, \text{op})$. \square

By the previous claim, we can now use the algorithm from Theorem 5.1 to decide whether $(\Gamma, \mathcal{S}, U, k)$ is a YES-instance. This completes the description of the algorithm. It only remains to analyze its running time. Clearly, the instance $(\Gamma^*, \mathcal{S}_{\text{global}}, \mathcal{S}_{\text{local}})$ can be computed in polynomial time given $(\Gamma, \mathcal{S}, U, k, i, Z_i)$. Also, $\text{tw}(G/\mathcal{I}) = \mathcal{O}(\ell + |Z_i|) = \mathcal{O}(\sqrt{k})$ by Theorem 4.2. So the algorithm from Theorem 5.1 runs in time

$$(|D^*| + \mathcal{O}(\sqrt{k}) + k + \|\mathcal{S}\|)^{\mathcal{O}(\sqrt{k})} |X^*|^{\mathcal{O}(1)} = (|D| + k + \|\mathcal{S}\|)^{\mathcal{O}(\sqrt{k})} |X|^{\mathcal{O}(1)}.$$

Overall, this result in a running time of

$$|X|^{\mathcal{O}(\sqrt{k})} \cdot (|D| + k + \|\mathcal{S}\|)^{\mathcal{O}(\sqrt{k})} |X|^{\mathcal{O}(1)} = (|X| + |D| + \|\mathcal{S}\|)^{\mathcal{O}(\sqrt{k})}.$$

For the second statement of the theorem, the algorithm is essentially identical, but it relies on Conjecture 1.1 instead of Theorem 4.2. In this case $\text{tw}(G/\mathcal{I}) = \mathcal{O}(c_H(\ell + |Z_i|)) = \mathcal{O}(c_H\sqrt{k})$. Hence, the algorithm from Theorem 5.1 runs in time

$$(|D| + c_H k + \|\mathcal{S}\|)^{\mathcal{O}(c_H\sqrt{k})} |X|^{\mathcal{O}(1)} = (|D| + k + \|\mathcal{S}\|)^{\mathcal{O}(c'_H\sqrt{k})} |X|^{\mathcal{O}(1)}$$

for some suitable constant c'_H that only depends on c_H . \square

The next theorem provides a variant of the last theorem where we partition the vertex set into $k + 1$ classes. This result may also be of interest.

THEOREM 5.5. *There is an algorithm solving PLANAR PERM CSP VERTEX DELETION WITH SIZE CONSTRAINTS in time $(|D| + k + \|\mathcal{S}\|)^{\mathcal{O}(k)} |X|^{\mathcal{O}(1)}$.*

Moreover, assuming Conjecture 1.1, there is an algorithm solving H-MINOR-FREE PERM CSP VERTEX DELETION WITH SIZE CONSTRAINTS in time $(|D| + k + \|\mathcal{S}\|)^{\mathcal{O}(c_H k)} |X|^{\mathcal{O}(1)}$ where c_H is a constant that only depends on H .

Proof. The algorithm is exactly the same as in the previous theorem, but it computes a partition V_1, \dots, V_ℓ into $\ell := k + 1$ blocks. This implies there is some $i \in [\ell]$ such that $Z \cap V_i = \emptyset$, and the entire algorithm runs in time $(|D| + k + \|\mathcal{S}\|)^{\mathcal{O}(k)} |X|^{\mathcal{O}(1)}$.

Again, the same analysis provides the statement for H -minor-free graphs assuming Conjecture 1.1 holds.

\square

5.4 Permutation CSP on Two-Connected Components Finally, we turn to the proof of Theorem 3.3. The proof of this theorem is the most complicated one of this section since we need to incorporate the guessing of the bodies of the segments and we need to verify that all guesses are consistent. As before, we start by recalling the problem at hand.

Let $\Gamma = (X, D, \mathcal{C})$ be a binary CSP instance. A *2cc-size constraint* is a pair (w, q) where $w: X \times D \rightarrow \mathbb{N}$ is a weight function, and $q \in \mathbb{N}$. Let $W \subseteq X$. An assignment $\alpha: W \rightarrow D$ satisfies (w, q) on W if

$$\sum_{v \in W} w(v, \alpha(v)) \leq q.$$

Observe that, in comparison to 1cc-size constraints, we only allow to check for upper bounds on the weighted size of a set $W \subseteq X$.

A *Permutation-CSP-instance with 2cc-size constraints* is a pair (Γ, \mathcal{S}) where $\Gamma = (X, D, \mathcal{C})$ is a Permutation-CSP-instance and \mathcal{S} is a set of 2cc-size constraints. For $W \subseteq X$, we say that (Γ, \mathcal{S}) is *satisfiable on W* if there is an assignment $\alpha: W \rightarrow D$ that satisfies $\Gamma[W]$ as well as all 2cc-size constraints $(w, q) \in \mathcal{S}$ on W . As before, we define

$$\|\mathcal{S}\| := \prod_{(w, q, \text{op}) \in \mathcal{S}} (2 + q).$$

2CONN PERM CSP VERTEX DELETION WITH SIZE CONSTRAINTS

Parameter: k

Input: A Permutation-CSP-instance with 2cc-size constraints (Γ, \mathcal{S}) with $\Gamma = (X, D, \mathcal{C})$, a set of undeletable variables $U \subseteq X$, and an integer k .

Question: Is there a set $Z \subseteq X \setminus U$ such that $|Z| \leq k$ and (Γ, \mathcal{S}) is satisfiable on W for every 2-connected component W of the graph $G(\Gamma[X \setminus Z])$.

To prove Theorem 3.3, we need to build an algorithm solving PLANAR 2CONN PERM CSP VERTEX DELETION WITH SIZE CONSTRAINTS and which runs in time $(|X| + |D| + \|\mathcal{S}\|)^{\mathcal{O}(\sqrt{k})}$. For this, we build on similar ideas as in the previous subsection, but we need to incorporate information about the bodies of segments $I \in \mathcal{I}$ as obtained in Theorem 4.4. Let us start by recalling some notation.

Let (G, \mathcal{I}) be a segmented graph. We define $V(\mathcal{I}) := \bigcup_{I \in \mathcal{I}} I$ as the set of vertices appearing in some segment. We always use v_I to denote the vertex of G/\mathcal{I} that corresponds to segment I . Also, $V_{\mathcal{I}} := \{v_I \mid I \in \mathcal{I}\}$. For $U \subseteq V(G)$ we use $\text{shr}(U)$ to denote the set of all vertices $v \in V(G/\mathcal{I})$ that correspond to some $u \in U$. In particular, $v_I \in \text{shr}(U)$ if $U \cap I \neq \emptyset$. In the other direction, for $U \subseteq V(G/\mathcal{I})$, we use $\text{ext}(U)$ to denote the set of all vertices $v \in V(G)$ that correspond to some $u \in U$. In particular, if $v_I \in U$ then $I \subseteq \text{ext}(U)$. If $U = \{u\}$ consists of a single vertex, then we also write $\text{shr}(u)$ and $\text{ext}(u)$ instead of $\text{shr}(\{u\})$ and $\text{ext}(\{u\})$.

Next, let $Z \subseteq V(G) \setminus V(\mathcal{I})$ denote a “solution set”. Let (\mathfrak{T}, γ) be the decomposition into the 2-connected components of $G - Z$. Recall that \mathfrak{T} is a forest where each connected component is equipped with a root node. For $t \in V(\mathfrak{T})$ we denote by $\text{Desc}(t)$ the set of descendants of t , including t itself.

We recall the following objects defined for every segment $I \in \mathcal{I}$. We have that $r_Z(I)$ denote the unique node $t \in V(\mathfrak{T})$ which is closest to a root node t_0 and for which $I \cap \gamma(t) \neq \emptyset$. Also, $B_Z(I) := \{t \in V(\mathfrak{T}) \setminus \{r(I)\} \mid I \cap \gamma(t) \neq \emptyset\}$ denotes the *body* of I . Observe that $r_Z(I) \notin B_Z(I)$. Moreover, $\hat{r}_Z(I) := \gamma(r_Z(I))$ and $\hat{B}_Z(I) := \bigcup_{t \in B_Z(I)} \gamma(t)$ denote the corresponding sets of vertices in the graph G . To simplify notation, we usually omit the index Z if it is clear from context.

To obtain the information on the bodies of segments $I \in \mathcal{I}$ we build on Corollary 4.2 which provides suitable segments and their bodies after guessing $\mathcal{O}(\sqrt{k})$ many vertices. Hence, we may assume that this information is additionally given to algorithm. This is formulated by the following intermediate problem.

2CONN PERM CSP VERTEX DELETION WITH BODIES

Parameter: $\text{tw}(G/\mathcal{I}) + \|\mathcal{F}\|$

Input: A Permutation-CSP-instance with 2cc-size constraints (Γ, \mathcal{S}) with $\Gamma = (X, D, \mathcal{C})$, a set of undeletable variables $U \subseteq X$, a set of segments \mathcal{I} of the graph $G(\Gamma)$, a function \mathcal{F} on domain \mathcal{I} , and an integer k .

Question: Is there a set $Z \subseteq X \setminus (V(\mathcal{I}) \cup U)$ such that $|Z| \leq k$ and, for (\mathfrak{T}, γ) being a rooted decomposition into 2-connected components of $G(\Gamma) - Z$, (Γ, \mathcal{S}) is satisfiable on $\gamma(t)$ for all $t \in V(\mathfrak{T})$.

Guarantee: If $(\Gamma, \mathcal{S}, U, \mathcal{I}, \mathcal{F}, k)$ is a YES-instance, then there is a solution such that additionally $(\hat{B}_Z(I), \hat{r}_Z(I) \cap I) \in \mathcal{F}(I)$ for all $I \in \mathcal{I}$.

Here, we define $\|\mathcal{F}\| := \sum_{I \in \mathcal{I}} |\mathcal{F}(I)|$. To be able to use an algorithm for this problem as a subroutine later on, let us clarify the behavior in case the guarantee is not satisfied. If $(\Gamma, \mathcal{S}, U, k)$ is a YES-instance, an algorithm may output either answer if the guarantee does not hold. However, if $(\Gamma, \mathcal{S}, U, k)$ is a NO-instance, an algorithm may never output YES independent of whether the guarantee is satisfied or not.

THEOREM 5.6. *There is an algorithm solving 2CONN PERM CSP VERTEX DELETION WITH BODIES in time*

$$(|X| + |D| + \|\mathcal{S}\| + \|\mathcal{F}\|)^{\mathcal{O}(\text{tw}(G/\mathcal{I}))}.$$

Proof Idea. We give a polynomial-time algorithm that translates an instance of the problem above into an equivalent CSP-instance $\Gamma^* = (X^*, D^*, \mathcal{C}^*)$ with global size constraints $\mathcal{S}_{\text{global}}$ and (F^*, D^*) -local size constraints $\mathcal{S}_{\text{local}}$ (for some suitable choice of (F^*, D^*)) such that $G(\Gamma^*) = (G(\Gamma))/\mathcal{I}$ and

$$|D^*| = (|X| + |D| + \|\mathcal{F}\|)^{\mathcal{O}(1)}.$$

Let $G := G(\Gamma)$. Towards this end, we describe a solution to $(\Gamma, \mathcal{S}, U, k)$ by a series of functions defined on $X^* := V(G/\mathcal{I})$.

Let $I \in \mathcal{I}$ be a segment and let $(B, J) \in \mathcal{F}(I)$. We say that (B, J) is *valid* if

- (I) (Γ, \mathcal{S}) is satisfiable on C for every 2-connected component C of $G[B]$,
- (II) $I \subseteq B \cup J$,
- (III) every vertex $v \in B \cap J$ is a cut vertex in $G[B \cup J]$,
- (IV) $G[J]$ is connected, and
- (V) $B \cap I \neq \emptyset$ and $G[B \cup I]$ is connected.

We start by removing all elements (B, J) from $\mathcal{F}(I)$ that are not valid. Observe that this can be done in polynomial time.

Now let $\tilde{\mathcal{F}} := \bigcup_{I \in \mathcal{I}} \{I\} \times \mathcal{F}(I)$. We describe a solution to the input instance by the following functions:

- $\alpha: X^* \rightarrow D \uplus \{\text{sol}\} \uplus \tilde{F}$ such that $\alpha(v_I) \neq \text{sol}$ for every $I \in \mathcal{I}$, and $\alpha(v) \neq \text{sol}$ for every $v \in U \setminus V(\mathcal{I})$,
- $\alpha^+: X^* \rightarrow D$,
- $h: X^* \rightarrow [0, |X|]$,
- $\text{cut}: X^* \rightarrow X \cup \{\top\}$,
- $\rho: \mathcal{I} \rightarrow \bigcup_{I \in \mathcal{I}} \mathcal{F}(I)$ such that $\rho(I) \in \mathcal{F}(I)$ for all $I \in \mathcal{I}$,

These functions correspond to a satisfying assignment for Γ^* . Consequently, we set

$$|D^*| := (D \uplus \{\text{sol}\} \cup \tilde{\mathcal{F}}) \times D \times [0, |X|] \times (X \cup \{\top\}) \times \left(\bigcup_{I \in \mathcal{I}} \mathcal{F}(I) \right).$$

This allows us to associate every tuple of functions $(\alpha, \alpha^+, h, \text{cut}, \rho)$ with some assignment $\alpha^*: X^* \rightarrow D^*$, and vice versa. Here, we can interpret \mathcal{I} (the domain of the function ρ) as a subset of X^* in the natural way, and view ρ as a function with domain X^* where vertices $v \in V(G/\mathcal{I}) \setminus V(\mathcal{I})$ are assigned arbitrary values.

Now, let us first describe how to translate a solution $Z \subseteq V(G) \setminus (V(\mathcal{I}) \cup U)$ for $(\Gamma, \mathcal{S}, U, k)$ into a satisfying assignment $(\alpha, \alpha^+, h, \text{cut}, \rho)$ for $(\Gamma^*, \mathcal{S}_{\text{global}}, \mathcal{S}_{\text{local}})$. (The construction of the instance $(\Gamma^*, \mathcal{S}_{\text{global}}, \mathcal{S}_{\text{local}})$ is completed afterwards in such a way that $(\alpha, \alpha^+, h, \text{cut}, \rho)$ indeed provides a satisfying assignment.)

Suppose $Z \subseteq V(G) \setminus (V(\mathcal{I}) \cup U)$ is a solution for $(\Gamma, \mathcal{S}, U, k)$ such that $(\hat{B}_Z(I), \hat{r}_Z(I) \cap I) \in \mathcal{F}(I)$ for all $I \in \mathcal{I}$. Also, let (\mathfrak{T}, γ) denote the corresponding rooted decomposition into 2-connected components of $G - Z$. First, we fix $\alpha(v) := \text{sol}$ for every $v \in Z$ and define $\alpha^+(v)$ arbitrarily, $h(v) = 0$ and $\text{cut}(v) = \top$. For all other vertices, the functions $(\alpha, \alpha^+, h, \text{cut}, \rho)$ are defined as follows (see Figure 6 for an example).

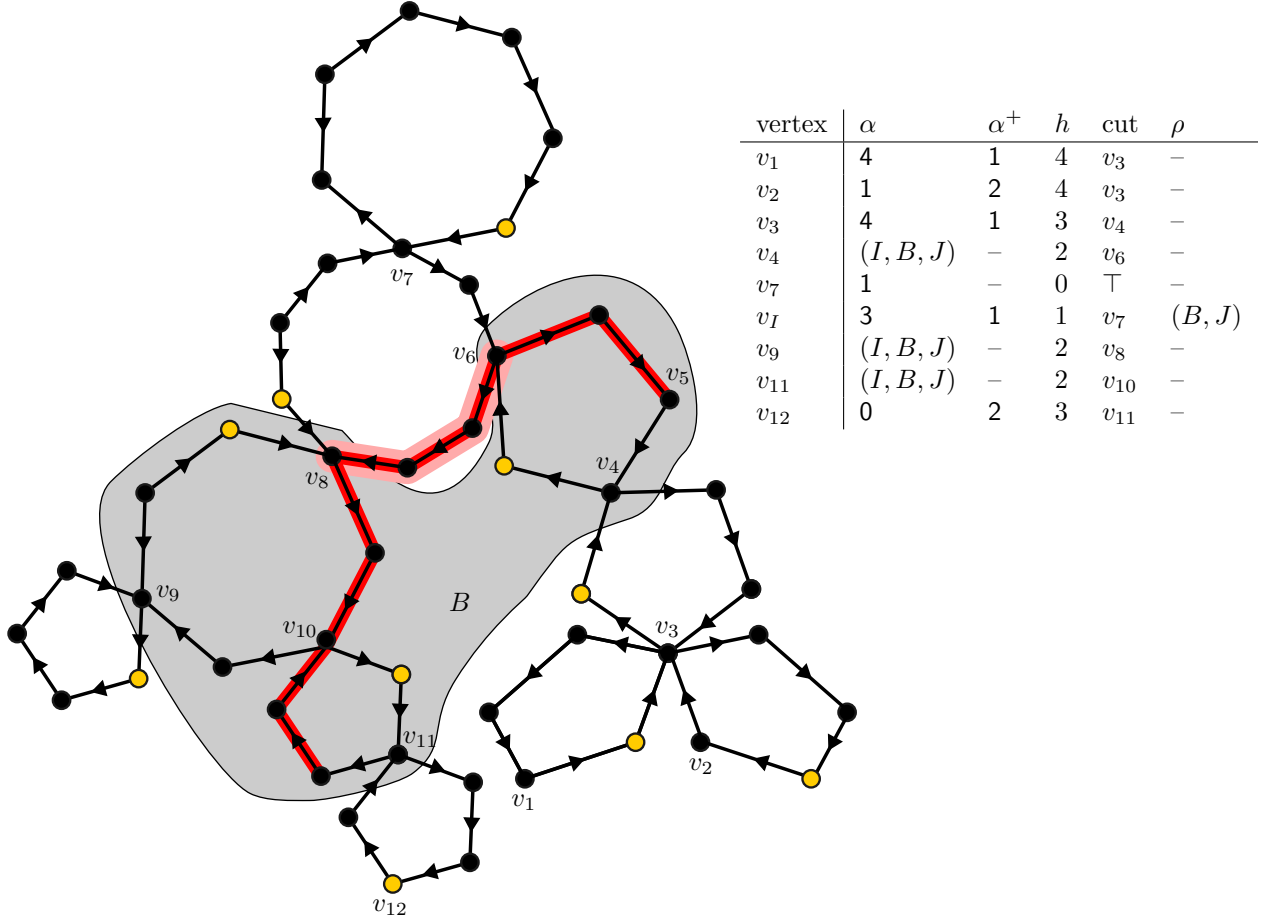


Figure 6: A permutation CSP instance remaining after removing variables Z , where each 2-connected component is satisfiable. The domain is $D = \{0, 1, 2, 3, 4\}$. Each directed edge $x \rightarrow y$ represents the relation $y = x + 1 \pmod{5}$. Each yellow vertex appears in a unary constraint forcing it to 0. Thus each 2-connected component has a unique satisfying assignment (note that these assignments do not agree on the cut vertices). Set B is shown in gray, segment I is red, and set J is highlighted in light red. We assume that v_6 is the minimal element of J on I . The table show the values of certain vertices in a consistent tuple $(\alpha, \alpha^+, h, \text{cut}, \rho)$.

ρ : We first define $\rho(I) := (\widehat{B}_Z(I), \widehat{r}_Z(I) \cap I)$ for every $I \in \mathcal{I}$. It is easy to see that $(\widehat{B}_Z(I), \widehat{r}_Z(I) \cap I)$ is valid for every $I \in \mathcal{I}$.

h : For every $v \in V(G/\mathcal{I}) \setminus Z$ we define $r(v) := t$ for the unique highest node $t \in V(\mathfrak{T})$ such that $\gamma(t) \cap \text{ext}(v) \neq \emptyset$. Let $t_0, \dots, t_\ell = r(v)$ denote the unique path from a root of \mathfrak{T} to $r(v)$. For every $I \in \mathcal{I}$ it holds that $\{t_0, \dots, t_\ell\} \cap B_Z(I) = \{t_i, t_{i+1}, \dots, t_j\}$ for some i, j , i.e., the intersection forms a subpath of t_0, \dots, t_ℓ . Let t'_0, \dots, t'_h denote the path obtained from t_0, \dots, t_ℓ by contradicting all these subpaths to single vertices for every segment $I \in \mathcal{I}$. We define $h(v) := h$.

(Intuitively speaking, $h(v)$ is the height of $t = r(v)$ in the tree \mathfrak{T} . However, for technical reasons, $h(v)$ is defined slightly differently where single body sets are viewed as a single 2-connected component to determine the height; see Figure 6).

cut: If $h(v) = 0$ we define $\text{cut}(v) := \top$. Otherwise, $\text{cut}(v)$ is set to be the cut vertex to the parent of $r(v)$, i.e., $\text{cut}(v) := x$ where $x \in \gamma(t) \cap \gamma(s)$ is the unique element for $t = r(v)$ and s being the parent of t in \mathfrak{T} .

α **and** α^+ : Fix some $t \in V(\mathfrak{I})$. We distinguish two cases. First suppose that $t \in B_Z(I)$ for some segment $I \in \mathcal{I}$. For every $v \in V(G/\mathcal{I})$ such that $r(v) = t$ we set $\alpha(v) := (I, \widehat{B}_Z(I), \widehat{r}_Z(I) \cap I)$. Also, we define $\alpha^+(v)$ arbitrarily.

In the other case, there is no $I \in \mathcal{I}$ such that $t \in B_Z(I)$. Let $\alpha_t: \gamma(t) \rightarrow D$ denote a satisfying assignment for (Γ, \mathcal{S}) on $\gamma(t)$. Let $v \in V(G/\mathcal{I})$ such that $r(v) = t$. If $v \notin V_{\mathcal{I}}$ then we define $\alpha(v) := \alpha_t(v)$. If $v = v_I \in V_{\mathcal{I}}$ then $I \cap \gamma(t) = \widehat{r}_Z(I) \cap I =: J$ by definition. Let v' denote the minimal element contained in J according to the linear order on I . We define $\alpha(v_I) := \alpha_t(v')$. Observe that $\text{cut}(v) \notin I$ for every $I \in \mathcal{I}$ (as otherwise $t \in B_Z(I)$). We define $\alpha^+(v) := \alpha_t(\text{cut}(v))$ (if $\text{cut}(v) = \top$ then $\alpha^+(v)$ is defined arbitrarily).

This completes the description of the tuple $(\alpha, \alpha^+, h, \text{cut}, \rho)$.

Now, let us complete the construction of $(\Gamma^*, \mathcal{S}_{\text{global}}, \mathcal{S}_{\text{local}})$. Here, the basic idea is that every solution Z for the input problem should provide a satisfying assignment $(\alpha, \alpha^+, h, \text{cut}, \rho)$ as described above, and every satisfying assignment $(\alpha, \alpha^+, h, \text{cut}, \rho)$ gives raise to a solution $Z := \alpha^{-1}(\text{sol})$. We need to introduce the following additional notation. Let $I \in \mathcal{I}$. Observe that $\rho(I) = (B, J)$ for some $B \subseteq V(G)$ and $J \subseteq I$. We denote by $\rho_1(I) = B$ the first component of $\rho(I)$, and $\rho_2(I) = J$ denotes the second component of $\rho(I)$.

Next, let $J \subseteq I$ such that $G[J]$ is connected and let $a \in D$. We define $\alpha_{J,a}: J \rightarrow D$ to the unique assignment for which $\alpha_{J,a}(v) = a$ where v denotes the minimal element of J (where elements of J are ordered according to the linear order on I), and such that $\alpha_{J,a}$ is a satisfying assignment of $\Gamma[J]$. If there is no such assignment, we say that a is *invalid* for J and $\alpha_{J,a}$ may be defined in an arbitrary manner.

Now, we describe a series of conditions that enforces any tuple $(\alpha, \alpha^+, h, \text{cut}, \rho)$ of function to have the intended meaning described above. In particular, the tuple of functions constructed from a solution set Z as above satisfies all these conditions. We say that $(\alpha, \alpha^+, h, \text{cut}, \rho)$ is *consistent* if the following conditions are satisfied. For every $v \in V(G/\mathcal{I})$ such that $\alpha(v) \neq \text{sol}$ it holds that

1. $h(v) = 0$ if and only if $\text{cut}(v) = \top$,
2. if $\alpha(v) \in D$ and $v \notin V_{\mathcal{I}}$ then $\alpha(v) \in R$ for unary every constraint $(v, R) \in \mathcal{C}$,
3. if $\alpha(v) \in D$ and $v = v_I \in V_{\mathcal{I}}$ then $\alpha(v)$ is valid for $\rho_2(I)$,
4. $\alpha^+(v) \in R$ for every unary constraint $(\text{cut}(v), R) \in \mathcal{C}$ (if $\text{cut}(v) = \top$, this condition is ignored),
5. if $\alpha(v) = (I, B, J)$ then $\text{cut}(v) \in B \cap J$,
6. if $\text{cut}(v) \notin I$ for every $I \in \mathcal{I}$, then $\alpha(v) \in D$,
7. if $\text{cut}(v) \in I$, then $\alpha(v) = (I, B, J)$ for some $(B, J) \in \mathcal{F}(I)$,
8. if $\alpha(v) = (I, B, J)$ and $v \notin V_{\mathcal{I}}$ then $v \in B$,
9. if $\alpha(v) = (I, B, J)$ and $v = v_{I'} \in V_{\mathcal{I}}$ then $I \neq I'$ and $\rho_2(I') = B \cap I'$,
10. $\text{cut}(v) \notin \text{ext}(v)$.

Also, for every $vw \in E(G/\mathcal{I})$ such that $\alpha(v) \neq \text{sol}$ it holds that

11. if $v = v_I \in V_{\mathcal{I}}$, $w \notin V_{\mathcal{I}}$, $\rho(I) = (B, J)$ and $w \in B$ then $\alpha(w) = (I, B, J)$ and $h(w) = h(v) + 1$,
12. if $v = v_I \in V_{\mathcal{I}}$, $w = v_{I'} \in V_{\mathcal{I}}$, $\rho(I) = (B, J)$ and $I' \cap B \neq \emptyset$ then $\alpha(w) = (I, B, J)$ and $h(w) = h(v) + 1$,
13. if $\alpha(v) = (I, B, J)$, $w \notin V_{\mathcal{I}}$ and $w \in B$ then $\alpha(w) = (I, B, J)$ and $h(w) = h(v)$,
14. if $\alpha(v) = (I, B, J)$, $w = v_{I'} \in V_{\mathcal{I}}$, $I \neq I'$ and $I' \cap B \neq \emptyset$ then $\alpha(w) = (I, B, J)$ and $h(w) = h(v)$,
15. if $\alpha(w) = (I, B, J)$ and $v = v_I$ then $\rho(I) = (B, J)$.

(Intuitively speaking, these constraints enforce that, if $\rho(I) = (B, J)$, then the set of vertices v that set $\alpha(v) := (I, B', J')$ corresponds to B , and $\alpha(v) := (I, B, J)$ for all such vertices. In other words, the constraints enforce that bodies are assigned consistently.) And finally, for every $vw \in E(G/\mathcal{I})$ such that $\alpha(v) \neq \text{sol}$ and $\alpha(w) \neq \text{sol}$ it holds that

16. $h(v) \in \{h(w) + 1, h(w), h(w) - 1\}$,

17. if $h(v) = h(w)$ then

(a) $\text{cut}(v) = \text{cut}(w)$ and $\alpha^+(v) = \alpha^+(w)$,

(b) if $v = v_I \in V_{\mathcal{I}}$ and $\rho(I) = (B, J)$ then $N(\text{ext}(w)) \cap I \subseteq J$,

(c) if $\alpha(v) \in \tilde{F}$ then $\alpha(w) = \alpha(v)$

(d) if $\alpha(v) \in D$ and $v, w \notin V_{\mathcal{I}}$ then $(\alpha(v), \alpha(w))$ satisfies all constraints $((v, w), R) \in \mathcal{C}$,

(e) if $\alpha(v) \in D$ and $v = v_I \in V_{\mathcal{I}}$, $w \notin V_{\mathcal{I}}$ then $a := \alpha(v)$ is valid for $\rho_2(I)$, and $(\alpha_{a, \rho_2(I)}(v'), \alpha(w))$ satisfies all constraints $((v', w), R) \in \mathcal{C}$ where $v' \in \rho_2(I)$,

(f) if $\alpha(v) \in D$ and $v = v_I \in V_{\mathcal{I}}$, $w = w_{I'} \in V_{\mathcal{I}}$ then $a := \alpha(v)$ is valid for $\rho_2(I)$, $a' := \alpha(w)$ is valid for $\rho_2(I')$, and $(\alpha_{a, \rho_2(I)}(v'), \alpha_{a', \rho_2(I')}(w'))$ satisfies all constraints $((v', w'), R) \in \mathcal{C}$ where $v' \in \rho_2(I)$ and $w' \in \rho_2(I')$,

18. if $h(w) = h(v) + 1$ then

(a) $\text{cut}(w) \in \text{ext}(v)$,

(b) if $w = v_I \in V_{\mathcal{I}}$ and $\rho(I) = (B, J)$ then $N(\text{ext}(v)) \cap I \subseteq J$,

(c) if $v, w \notin V_{\mathcal{I}}$, then $(\alpha^+(w), \alpha(w))$ satisfies all constraints $((v, w), R) \in \mathcal{C}$,

(d) if $v \notin V_{\mathcal{I}}$ and $w = v_I \in V_{\mathcal{I}}$, then $a := \alpha(w)$ is valid for $\rho_2(I)$, and $(\alpha^+(w), \alpha_{a, \rho_2(I)}(w'))$ satisfies all constraints $((v, w'), R) \in \mathcal{C}$ where $w' \in \rho_2(I)$.

(Here, we mostly ensure that all binary constraints are satisfied, and the root part J for $\rho(I) = (B, J)$ interacts with the rest of graph in a consistent way, see 17b and 18b.)

Now, the existence of a consistent tuple can be formulated as a binary CSP-instance $\Gamma^* = (X^*, D^*, C^*)$ such that $G(\Gamma^*) = (G(\Gamma))/\mathcal{I}$ and

$$|D^*| = (D \uplus \{\text{sol}\} \cup \tilde{F}) \times D \times [0, |X|] \times (X \cup \{\top\}) \times \left(\bigcup_{I \in \mathcal{I}} \mathcal{F}(I) \right).$$

It remains to define the global and local size constraints. Since we need to define weight function on $X^* \times D^*$, we interpret elements $\bar{a} \in D^*$ as vectors of dimension 5, and use \bar{a}_i to denote the i -th entry. In this sense, \bar{a}_1 corresponds to the function α , \bar{a}_2 corresponds to the function α^+ , \bar{a}_3 corresponds to the function h , \bar{a}_4 corresponds to the function cut , and \bar{a}_5 corresponds to the function ρ . First, we add a global size constraint to force that at most k variables $x \in X^*$ are assigned the value sol (under the function α). More precisely, we define $\mathcal{S}_{\text{global}} := \{(w_{\text{sol}}, k, \leq)\}$ where $w_{\text{sol}}: X^* \times D^* \rightarrow \mathbb{N}$ is defined via

$$w_{\text{sol}}(v, \bar{a}) = \begin{cases} 1 & \text{if } \bar{a}_1 = \text{sol} \\ 0 & \text{otherwise} \end{cases}$$

for every $\bar{a} \in D^*$ and $v \in X^*$, where \bar{a}_i denotes the i -th component of \bar{a} .

Next, we turn to the local size constraints. We define

$$F^* = \left\{ (\bar{a}, \bar{a}') \in (D^*)^2 \mid \bar{a}_1 \in D, \bar{a}'_1 \in D, \bar{a}_3 = \bar{a}'_3 \right\}.$$

Also, we define the partition \mathcal{D}^* of D^* in such a way that two elements $\bar{a}, \bar{a}' \in D^*$ end up in the same partition class of \mathcal{D}^* if and only if

$$\bar{a}_2 = \bar{a}'_2 \wedge \bar{a}_3 = \bar{a}'_3 \wedge \bar{a}_4 = \bar{a}'_4.$$

Now, consider a 2cc-size constraint $(w, q) \in \mathcal{S}$. We translate (w, q) into a (F^*, \mathcal{D}^*) -local size constraint $(w^*, w_{\mathcal{D}}^*, q, \leq) \in \mathcal{S}_{\text{local}}$ as follows. For $D' \in \mathcal{D}^*$ there are $a \in D$, $h \in [0, |X|]$, and $v \in (X \cup \{\top\})$ such that $\bar{a}_2 = a$, $\bar{a}_3 = h$, and $\bar{a}_4 = v$ for all $\bar{a} \in D'$. We define

$$w_{\mathcal{D}}^*(D') := \begin{cases} 0 & \text{if } v = \top \\ w(v, a) & \text{otherwise} \end{cases}.$$

Next, we define the function w^* . Let $v \in X^* = V(G/\mathcal{I})$ and $\bar{a} \in D^*$. If $v \notin V_{\mathcal{I}}$ we define

$$w^*(v, \bar{a}) := \begin{cases} w(v, \bar{a}_1) & \text{if } \bar{a}_1 \in D \\ 0 & \text{otherwise} \end{cases}.$$

Also, if $v = v_I \in V_{\mathcal{I}}$ we define

$$w^*(v, \bar{a}) := \begin{cases} \sum_{u \in J} w(u, \alpha_{J, \bar{a}_1}(u)) & \text{if } \bar{a}_1 \in D, \bar{a}_5 = (B, J), \bar{a}_1 \text{ is valid for } J \\ 0 & \text{otherwise} \end{cases}.$$

This completes the description of the instance $(\Gamma^*, \mathcal{S}_{\text{global}}, \mathcal{S}_{\text{local}})$.

First observe that it can be computed in time $(|D| + |X| + |\mathcal{S}| + \|\mathcal{F}\|)^{\mathcal{O}(1)}$. Also, $|X^*| \leq |X|$, $\|\mathcal{S}_{\text{global}}\| = k$, $\|\mathcal{S}_{\text{local}}\| = \|\mathcal{S}\|$, and

$$|D^*| = (|X| + |D| + \|\mathcal{F}\|)^{\mathcal{O}(1)}.$$

Now, we claim that the instance $(\Gamma^*, \mathcal{S}_{\text{global}}, \mathcal{S}_{\text{local}})$ is equivalent to the input instance. More precisely, it is possible to prove the following two statements:

- (E.1) If $(\Gamma^*, \mathcal{S}_{\text{global}}, \mathcal{S}_{\text{local}})$ is satisfiable with a consistent tuple $(\alpha, \alpha^+, h, \text{cut}, \rho)$ then $Z := \alpha^{-1}(\text{sol})$ is a solution for the instance $(\Gamma, \mathcal{S}, U, k)$.
- (E.2) If $Z \subseteq V(G) \setminus (V(\mathcal{I}) \cup U)$ is a solution for $(\Gamma, \mathcal{S}, U, k)$ such that $(\widehat{B}_Z(I), \widehat{r}_Z(I) \cap I) \in \mathcal{F}(I)$ for all $I \in \mathcal{I}$, then $(\Gamma^*, \mathcal{S}_{\text{global}}, \mathcal{S}_{\text{local}})$ is satisfiable.

Hence, it suffices to check whether $(\Gamma^*, \mathcal{S}_{\text{global}}, \mathcal{S}_{\text{local}})$ is satisfiable. By Theorem 5.1 this can be done in time

$$(|D^*| + \text{tw}(G/\mathcal{I}) + \|\mathcal{S}_{\text{global}}\| + \|\mathcal{S}_{\text{local}}\|)^{\mathcal{O}(\text{tw}(G/\mathcal{I}))} |X^*|^{\mathcal{O}(1)} = (|X| + |D| + \|\mathcal{S}\| + \|\mathcal{F}\|)^{\mathcal{O}(\text{tw}(G/\mathcal{I}))}.$$

□

Now, we can combine the previous theorem with Corollary 4.2 to obtain an algorithm for PLANAR 2CONN PERM CSP VERTEX DELETION WITH SIZE CONSTRAINTS.

THEOREM 5.7. *There is an algorithm solving PLANAR 2CONN PERM CSP VERTEX DELETION WITH SIZE CONSTRAINTS in time $(|X| + |D| + \|\mathcal{S}\|)^{\mathcal{O}(\sqrt{k})}$.*

Moreover, assuming Conjecture 1.1, there is an algorithm solving H-MINOR-FREE 2CONN PERM CSP VERTEX DELETION WITH SIZE CONSTRAINTS in time $(|X| + |D| + \|\mathcal{S}\|)^{\mathcal{O}(c_H \sqrt{k})}$ for some constant c_H that only depends on H .

Proof. Consider the first part of the theorem. Let $(\Gamma, \mathcal{S}, U, k)$ be the input instance. Also, let $G := G(\Gamma)$ be the constraint graph. We choose $r = \mathcal{O}(\sqrt{k})$ according to Lemma 4.2 and iterate over all tuples $(v_1, \dots, v_r) \in (V(G))^r$. For $(v_1, \dots, v_r) \in (V(G))^r$, we compute a set of segments \mathcal{I} and a function \mathcal{F} using the algorithm from 4.2. If $\text{tw}(G/\mathcal{I}) = \omega(\sqrt{k})$, the algorithm moves to the next tuple. Otherwise, we apply Theorem 5.6 to the input $(\Gamma, \mathcal{S}, U, \mathcal{I}, \mathcal{F}, k)$. If the algorithm returns YES then we also output YES. Otherwise, we move to the next tuple $(v_1, \dots, v_r) \in (V(G))^r$. If we never outputs YES for any tuple $(v_1, \dots, v_r) \in (V(G))^r$, then we return NO.

Let us first analyse the running time of the algorithm. Clearly, in each iteration, $|\mathcal{I}| \leq |X|$ and $|\mathcal{F}| = |X|^{\mathcal{O}(1)}$ by Corollary 4.2. Also, the computation of $(\mathcal{I}, \mathcal{F})$ requires time $|X|^{\mathcal{O}(1)}$. Also, we can determine whether $\text{tw}(G/\mathcal{I}) = \mathcal{O}(\sqrt{k})$ in time $2^{\mathcal{O}(\sqrt{k})} |X|^{\mathcal{O}(1)}$. Next, the algorithm from Theorem 5.6 runs in time

$$(|X| + |D| + \|\mathcal{S}\| + \|\mathcal{F}\|)^{\mathcal{O}(\text{tw}(G/\mathcal{I}))} = (|X| + |D| + \|\mathcal{S}\|)^{\mathcal{O}(\sqrt{k})}.$$

Overall, this gives a running time of

$$|X|^r \left(|X|^{\mathcal{O}(1)} + 2^{\mathcal{O}(\sqrt{k})} |X|^{\mathcal{O}(1)} + (|X| + |D| + \|\mathcal{S}\|)^{\mathcal{O}(\sqrt{k})} \right) = (|X| + |D| + \|\mathcal{S}\|)^{\mathcal{O}(\sqrt{k})}.$$

For the correctness of the algorithm, first observe that we only output YES if $(\Gamma, \mathcal{S}, U, k)$ is indeed a YES-instance. So suppose that $(\Gamma, \mathcal{S}, U, k)$ is a YES-instance and let $Z \subseteq X = V(G)$ be a solution. Observe that $|Z| \leq k$. By Corollary 4.2, there is some tuple $(v_1, \dots, v_r) \in (V(G))^r$ such that

1. $V(\mathcal{I}) \cap Z = \emptyset$,
2. $\text{tw}(G/\mathcal{I}) = \mathcal{O}(\sqrt{k})$, and
3. $(\widehat{B}_Z(I), \widehat{r}_Z(I) \cap I) \in \mathcal{F}(I)$ for every $I \in \mathcal{I}$.

where $(\mathcal{I}, \mathcal{F})$ denotes the output of the algorithm from Corollary 4.2 on input $(G, (v_1, \dots, v_r))$. For this tuple, the algorithm from Theorem 5.6 outputs YES.

For the second part of the theorem, we proceed in the same way, but we replace the application of Corollary 4.2 with Corollary 4.3. \square

In particular, the last theorem implies Theorem 3.3. We complete this section by considering the edge deletion version of the problem above.

2CONN PERM CSP EDGE DELETION WITH SIZE CONSTRAINTS

Parameter: k

Input: A Permutation-CSP-instance with 2cc-size constraints (Γ, \mathcal{S}) with $\Gamma = (X, D, \mathcal{C})$, a set of undeletable edges $U \subseteq E(G(\Gamma))$, and an integer k .

Question: Is there a set $Z \subseteq E(G(\Gamma)) \setminus U$ such that $|Z| \leq k$ and $((X, D, \mathcal{C} - Z), \mathcal{S})$ is satisfiable on W for every 2-connected component W of the graph $G(\Gamma) - Z$.

By reducing this problem to the corresponding vertex deletion version, we obtain the following theorem.

THEOREM 5.8. *There is an algorithm solving PLANAR 2CONN PERM CSP EDGE DELETION WITH SIZE CONSTRAINTS in time $(|X| + |D| + \|\mathcal{S}\|)^{\mathcal{O}(\sqrt{k})}$.*

Moreover, assuming Conjecture 1.1, there is an algorithm solving H-MINOR-FREE 2CONN PERM CSP EDGE DELETION WITH SIZE CONSTRAINTS in time $(|X| + |D| + \|\mathcal{S}\|)^{\mathcal{O}(c_H \sqrt{k})}$ for some constant c_H that only depends on H .

References

- [1] Sanjeev Arora, Boaz Barak, and David Steurer. Subexponential algorithms for unique games and related problems. *J. ACM*, 62(5):42:1–42:25, 2015.
- [2] MohammadHossein Bateni, Chandra Chekuri, Alina Ene, Mohammad Taghi Hajiaghayi, Nitish Korula, and Dániel Marx. Prize-collecting steiner problems on planar graphs. In Dana Randall, editor, *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, San Francisco, California, USA, January 23-25, 2011*, pages 1028–1049. SIAM, 2011.
- [3] MohammadHossein Bateni, Erik D. Demaine, MohammadTaghi Hajiaghayi, and Dániel Marx. A PTAS for planar group steiner tree via spanner bootstrapping and prize collecting. In Daniel Wichs and Yishay Mansour, editors, *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 570–583. ACM, 2016.
- [4] MohammadHossein Bateni, Alireza Farhadi, and MohammadTaghi Hajiaghayi. Polynomial-time approximation scheme for minimum k-cut in planar and minor-free graphs. 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 1055–1068. SIAM, 2019.
- [5] MohammadHossein Bateni, Mohammad Taghi Hajiaghayi, and Dániel Marx. Approximation schemes for steiner forest on planar graphs and graphs of bounded treewidth. *J. ACM*, 58(5):21:1–21:37, 2011.
- [6] MohammadHossein Bateni, MohammadTaghi Hajiaghayi, Philip N. Klein, and Claire Mathieu. A polynomial-time approximation scheme for planar multiway cut. 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 639–655. SIAM, 2012.
- [7] Amariah Becker, Philip N. Klein, and David Saulpic. A quasi-polynomial-time approximation scheme for vehicle routing on planar and bounded-genus graphs. In Kirk Pruhs and Christian Sohler, editors, *25th Annual European Symposium on Algorithms, ESA 2017, September 4-6, 2017, Vienna, Austria*, volume 87 of *LIPICs*, pages 12:1–12:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.

- [8] Amariah Becker, Philip N. Klein, and Aaron Schild. A PTAS for bounded-capacity vehicle routing in planar graphs. In Zachary Friggstad, Jörg-Rüdiger Sack, and Mohammad R. Salavatipour, editors, *Algorithms and Data Structures - 16th International Symposium, WADS 2019, Edmonton, AB, Canada, August 5-7, 2019, Proceedings*, volume 11646 of *Lecture Notes in Computer Science*, pages 99–111. Springer, 2019.
- [9] Hans L. Bodlaender. A partial k -arboretum of graphs with bounded treewidth. *Theor. Comput. Sci.*, 209(1-2):1–45, 1998.
- [10] Glencora Borradaile, Philip N. Klein, and Claire Mathieu. An $O(n \log n)$ approximation scheme for steiner tree in planar graphs. *ACM Trans. Algorithms*, 5(3):31:1–31:31, 2009.
- [11] Rajesh Hemant Chitnis, Andreas Emil Feldmann, Mohammad Taghi Hajiaghayi, and Dániel Marx. Tight bounds for planar strongly connected steiner subgraph with fixed number of terminals (and extensions). *SIAM J. Comput.*, 49(2):318–364, 2020.
- [12] Vincent Cohen-Addad, Anupam Gupta, Philip N. Klein, and Jason Li. A quasipolynomial $(2 + \epsilon)$ -approximation for planar sparsest cut. In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 1056–1069. ACM, 2021.
- [13] Marek Cygan, Fedor V. Fomin, Lukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. *Parameterized Algorithms*. Springer, 2015.
- [14] Éric Colin de Verdière. Multicuts in planar and bounded-genus graphs with bounded number of terminals. *Algorithmica*, 78(4):1206–1224, 2017.
- [15] Erik D. Demaine, Fedor V. Fomin, Mohammad Taghi Hajiaghayi, and Dimitrios M. Thilikos. Bidimensional parameters and local treewidth. *SIAM J. Discret. Math.*, 18(3):501–511, 2004.
- [16] Erik D. Demaine, Fedor V. Fomin, Mohammad Taghi Hajiaghayi, and Dimitrios M. Thilikos. Subexponential parameterized algorithms on bounded-genus graphs and H -minor-free graphs. *J. ACM*, 52(6):866–893, 2005.
- [17] Erik D. Demaine and MohammadTaghi Hajiaghayi. The bidimensionality theory and its algorithmic applications. *Comput. J.*, 51(3):292–302, 2008.
- [18] Erik D. Demaine and MohammadTaghi Hajiaghayi. Linearity of grid minors in treewidth with applications through bidimensionality. *Comb.*, 28(1):19–36, 2008.
- [19] Erik D. Demaine, MohammadTaghi Hajiaghayi, and Ken-ichi Kawarabayashi. Contraction decomposition in h -minor-free graphs and algorithmic applications. In Lance Fortnow and Salil P. Vadhan, editors, *Proceedings of the 43rd ACM Symposium on Theory of Computing, STOC 2011, San Jose, CA, USA, 6-8 June 2011*, pages 441–450. ACM, 2011.
- [20] Erik D. Demaine, MohammadTaghi Hajiaghayi, and Bojan Mohar. Approximation algorithms via contraction decomposition. *Comb.*, 30(5):533–552, 2010.
- [21] David Eisenstat, Philip N. Klein, and Claire Mathieu. An efficient polynomial-time approximation scheme for steiner forest in planar graphs. 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 626–638. SIAM, 2012.
- [22] Fedor V. Fomin, Petr A. Golovach, and Dimitrios M. Thilikos. Contraction bidimensionality: The accurate picture. In Amos Fiat and Peter Sanders, editors, *Algorithms - ESA 2009, 17th Annual European Symposium, Copenhagen, Denmark, September 7-9, 2009. Proceedings*, volume 5757 of *Lecture Notes in Computer Science*, pages 706–717. Springer, 2009.
- [23] Fedor V. Fomin, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. Subexponential parameterized algorithms for planar and apex-minor-free graphs via low treewidth pattern covering. In Irit Dinur, editor, *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 515–524. IEEE Computer Society, 2016.
- [24] Fedor V. Fomin, Daniel Lokshtanov, Venkatesh Raman, and Saket Saurabh. Bidimensionality and EPTAS. In Dana Randall, editor, *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2011, San Francisco, California, USA, January 23-25, 2011*, pages 748–759. SIAM, 2011.
- [25] Fedor V. Fomin, Daniel Lokshtanov, and Saket Saurabh. Bidimensionality and geometric graphs. 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 1563–1575. SIAM, 2012.
- [26] Fedor V. Fomin, Daniel Lokshtanov, Saket Saurabh, and Dimitrios M. Thilikos. Bidimensionality and kernels. *SIAM J. Comput.*, 49(6):1397–1422, 2020.
- [27] Kyle Fox, Philip N. Klein, and Shay Mozes. A polynomial-time bicriteria approximation scheme for planar bisection. In Rocco A. Servedio and Ronitt Rubinfeld, editors, *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 841–850. ACM, 2015.
- [28] Eli Fox-Epstein, Philip N. Klein, and Aaron Schild. Embedding planar graphs into low-treewidth graphs with applications to efficient approximation schemes for metric problems. 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 1069–1088. SIAM, 2019.

- [29] Sylvain Guillemot. FPT algorithms for path-transversal and cycle-transversal problems. *Discret. Optim.*, 8(1):61–71, 2011.
- [30] Eva-Maria C. Hols and Stefan Kratsch. A randomized polynomial kernel for subset feedback vertex set. *Theory Comput. Syst.*, 62(1):63–92, 2018.
- [31] Bart M. P. Jansen, Marcin Pilipczuk, and Erik Jan van Leeuwen. A deterministic polynomial kernel for odd cycle transversal and vertex multiway cut in planar graphs. In Rolf Niedermeier and Christophe Paul, editors, *36th International Symposium on Theoretical Aspects of Computer Science, STACS 2019, March 13-16, 2019, Berlin, Germany*, volume 126 of *LIPICs*, pages 39:1–39:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2019.
- [32] Subhash Khot. On the power of unique 2-prover 1-round games. In *Proceedings of the 17th Annual IEEE Conference on Computational Complexity, Montréal, Québec, Canada, May 21-24, 2002*, page 25. IEEE Computer Society, 2002.
- [33] Subhash Khot. On the unique games conjecture (invited survey). In *Proceedings of the 25th Annual IEEE Conference on Computational Complexity, CCC 2010, Cambridge, Massachusetts, USA, June 9-12, 2010*, pages 99–121. IEEE Computer Society, 2010.
- [34] Philip N. Klein. A linear-time approximation scheme for planar weighted TSP. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS 2005), 23-25 October 2005, Pittsburgh, PA, USA, Proceedings*, pages 647–657. IEEE Computer Society, 2005.
- [35] Philip N. Klein. A subset spanner for planar graphs, : with application to subset TSP. In Jon M. Kleinberg, editor, *Proceedings of the 38th Annual ACM Symposium on Theory of Computing, Seattle, WA, USA, May 21-23, 2006*, pages 749–756. ACM, 2006.
- [36] Philip N. Klein and Dániel Marx. Solving planar k -terminal cut in $O(n^{c \sqrt{k}})$ time. In Artur Czumaj, Kurt Mehlhorn, Andrew M. Pitts, and Roger Wattenhofer, editors, *Automata, Languages, and Programming - 39th International Colloquium, ICALP 2012, Warwick, UK, July 9-13, 2012, Proceedings, Part I*, volume 7391 of *Lecture Notes in Computer Science*, pages 569–580. Springer, 2012.
- [37] Philip N. Klein and Dániel Marx. A subexponential parameterized algorithm for subset TSP on planar graphs. In Chandra Chekuri, editor, *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 1812–1830. SIAM, 2014.
- [38] Alexandr V. Kostochka. Lower bound of the hadwiger number of graphs by their average degree. *Comb.*, 4(4):307–316, 1984.
- [39] Stefan Kratsch and Magnus Wahlström. Representative sets and irrelevant vertices: New tools for kernelization. *J. ACM*, 67(3):16:1–16:50, 2020.
- [40] Daniel Lokshtanov, M. S. Ramanujan, and Saket Saurabh. A linear-time parameterized algorithm for node unique label cover. In Kirk Pruhs and Christian Sohler, editors, *25th Annual European Symposium on Algorithms, ESA 2017, September 4-6, 2017, Vienna, Austria*, volume 87 of *LIPICs*, pages 57:1–57:15. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2017.
- [41] Daniel Lokshtanov, Saket Saurabh, and Magnus Wahlström. Subexponential parameterized odd cycle transversal on planar graphs. In Deepak D’Souza, Telikepalli Kavitha, and Jaikumar Radhakrishnan, editors, *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, FSTTCS 2012, December 15-17, 2012, Hyderabad, India*, volume 18 of *LIPICs*, pages 424–434. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2012.
- [42] Dániel Marx. A tight lower bound for planar multiway cut with fixed number of terminals. In Artur Czumaj, Kurt Mehlhorn, Andrew M. Pitts, and Roger Wattenhofer, editors, *Automata, Languages, and Programming - 39th International Colloquium, ICALP 2012, Warwick, UK, July 9-13, 2012, Proceedings, Part I*, volume 7391 of *Lecture Notes in Computer Science*, pages 677–688. Springer, 2012.
- [43] Dániel Marx, Marcin Pilipczuk, and Michal Pilipczuk. On subexponential parameterized algorithms for steiner tree and directed subset TSP on planar graphs. In Mikkel Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 474–484. IEEE Computer Society, 2018.
- [44] Dániel Marx and Michal Pilipczuk. Optimal parameterized algorithms for planar facility location problems using voronoi diagrams. In Nikhil Bansal and Irene Finocchi, editors, *Algorithms - ESA 2015 - 23rd Annual European Symposium, Patras, Greece, September 14-16, 2015, Proceedings*, volume 9294 of *Lecture Notes in Computer Science*, pages 865–877. Springer, 2015.
- [45] Jesper Nederlof. Detecting and counting small patterns in planar graphs in subexponential parameterized time. In Konstantin Makarychev, Yury Makarychev, Madhur Tulsiani, Gautam Kamath, and Julia Chuzhoy, editors, *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, pages 1293–1306. ACM, 2020.
- [46] Marcin Pilipczuk, Michal Pilipczuk, Piotr Sankowski, and Erik Jan van Leeuwen. Network sparsification for steiner problems on planar and bounded-genus graphs. *ACM Trans. Algorithms*, 14(4):53:1–53:73, 2018.
- [47] Neil Robertson and Paul D. Seymour. Graph minors. XVI. excluding a non-planar graph. *J. Comb. Theory, Ser. B*, 89(1):43–76, 2003.

- [48] Andrew Thomason. An extremal function for contractions of graphs. *Math. Proc. Cambridge Philos. Soc.*, 95(2):261–265, 1984.
- [49] Magnus Wahlström. On quasipolynomial multicut-mimicking networks and kernelization of multiway cut problems. *CoRR*, abs/2002.08825, 2020.