

# Multiplicative Parameterization Above a Guarantee

FEDOR V. FOMIN and PETR A. GOLOVACH, University of Bergen, Norway

DANIEL LOKSHTANOV, University of California, Santa Barbara, United States

FAHAD PANOLAN, Indian Institute of Technology Hyderabad, India

SAKET SAURABH, The Institute for Mathematical Sciences, HBNI, India and  
University of Bergen, Norway

MEIRAV ZEHAZI, Ben-Gurion University, Israel

---



Parameterization above a guarantee is a successful paradigm in Parameterized Complexity. To the best of our knowledge, all fixed-parameter tractable problems in this paradigm share an *additive form* defined as follows. Given an instance  $(I, k)$  of some (parameterized) problem  $\Pi$  with a *guarantee*  $g(I)$ , decide whether  $I$  admits a solution of size at least (or at most)  $k + g(I)$ . Here,  $g(I)$  is usually a lower bound on the minimum size of a solution. Since its introduction in 1999 for MAX SAT and MAX CUT (with  $g(I)$  being half the number of clauses and half the number of edges, respectively, in the input), analysis of parameterization above a guarantee has become a very active and fruitful topic of research.

We highlight a *multiplicative* form of parameterization above (or, rather, times) a guarantee: Given an instance  $(I, k)$  of some (parameterized) problem  $\Pi$  with a guarantee  $g(I)$ , decide whether  $I$  admits a solution of size at least (or at most)  $k \cdot g(I)$ . In particular, we study the LONG CYCLE problem with a multiplicative parameterization above the girth  $g(I)$  of the input graph, which is the most natural guarantee for this problem, and provide a fixed-parameter algorithm. Apart from being of independent interest, this exemplifies how parameterization above a multiplicative guarantee can arise naturally. We also show that, for any fixed constant  $\varepsilon > 0$ , multiplicative parameterization above  $g(I)^{1+\varepsilon}$  of LONG CYCLE yields para-NP-hardness, thus our parameterization is tight in this sense. We complement our main result with the design (or refutation of the existence) of fixed-parameter algorithms as well as kernelization algorithms for additional problems parameterized multiplicatively above girth.

CCS Concepts: • **Theory of computation** → **Fixed parameter tractability**;

Additional Key Words and Phrases: Parameterized complexity, above-guarantee, multiplicative above-guarantee, girth, long cycle

---

A preliminary version of this paper appeared in the proceedings of ITCS 2020. The research leading to these results has received funding from the Research Council of Norway via the projects “CLASSIS” and “MULTIVAL”, the European Research Council (ERC) via grant LOPPRE, reference 819416, the Swarnajayanti Fellowship grant DST/SJF/MSA-01/2017-18, the Israel Science Foundation (ISF) grant no. 1176/18, the United States – Israel Binational Science Foundation (BSF) grant no. 2018302, and the National Science Foundation (NSF) grant no. 2008838.  

Authors' addresses: F. V. Fomin and P. A. Golovach, University of Bergen, Thormøhlensgate, Bergen, Norway; emails: fomin@ii.uib.no, petr.golovach@uib.no; D. Lokshtanov, University of California, Santa Barbara, Santa Barbara, CA 93106; email: daniello@ucsb.edu; F. Panolan, Indian Institute of Technology Hyderabad, IITH Main Road, Sangareddy, Kandi, Telangana, India; email: fahad@iith.ac.in; S. Saurabh, The Institute for Mathematical Sciences, HBNI, Chennai, India, University of Bergen, Bergen, Norway; email: saket@imsc.res.in; M. Zehavi, Ben-Gurion University, Ben-Gurion Street, Beer-sheba, Israel; email: meiravze@bgu.ac.il.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](mailto:permissions@acm.org).

© 2021 Association for Computing Machinery.

1942-3454/2021/08-ART18 \$15.00

<https://doi.org/10.1145/3460956>

**ACM Reference format:**

Fedor V. Fomin, Petr A. Golovach, Daniel Lokshantov, Fahad Panolan, Saket Saurabh, and Meirav Zehavi. 2021. Multiplicative Parameterization Above a Guarantee. *ACM Trans. Comput. Theory* 13, 3, Article 18 (August 2021), 16 pages.

<https://doi.org/10.1145/3460956>

**1 INTRODUCTION**

The goal of parameterized complexity is to find ways of solving NP-hard problems more efficiently than brute force: our aim is to restrict the combinatorial explosion to a parameter that is hopefully much smaller than the input size. Formally, a *parameterization* of a problem is the assignment of an integer  $k$  to each input instance, and we say that a parameterized problem is **fixed-parameter tractable (FPT)** if there is an algorithm (called a *fixed-parameter algorithm*) that solves the problem in time  $f(k) \cdot n^{O(1)}$ , where  $n$  is the size of the input and  $f$  is an arbitrary computable function depending on the parameter  $k$  only. Further, we say that a parameterized problem admits a polynomial kernel if there exists a polynomial-time algorithm that, given an instance of the problem, outputs an equivalent instance of the problem whose size is bounded by a polynomial in  $k$ . There is a long list of NP-hard problems that are in FPT under various parameterizations: finding a vertex cover of size  $k$ , finding a cycle of length  $k$ , finding a maximum independent set in a graph of treewidth at most  $k$ , and so on (see, e.g., [15, 18, 24]).

Choosing a suitable parameter plays an important role in the field of parameterized complexity. Traditionally, the *solution size* has been the most sought after parameter. However, in various circumstances this is not a good parameter. To illustrate this, consider the following problems: MAX SAT and MAX CUT. Observe that there always exists a truth assignment that satisfies half of the clauses, and there is always a max-cut containing at least half the edges. Thus, if we choose solution size as the parameter, then we get the following trivial algorithm: if  $k \leq m/2$ , then return yes; else  $m \leq 2k$ , so now any brute-force algorithm is a fixed-parameter algorithm. Thus, if we want to design a non-trivial fixed-parameter algorithm, then solution size is *not a suitable parameter*. In particular, a general message here is as follows.

The natural parameterization of a maximization/minimization problem by the solution size is not satisfactory if there is a lower bound for the solution size that is sufficiently large.

Thus, for such cases, it is more natural to parameterize the problem by *the difference between the solution size and the bound*. This perspective is known as “above guarantee” parameterization. This approach was introduced by Mahajan and Raman [42] for the MAX SAT and MAX CUT problem. This approach was successfully applied to various problems (see, e.g., [1, 13, 14, 19, 23, 31–33, 43] for a few illustrative examples).

In some of the above examples (such as the lower bound of  $m/2$  for MAX SAT and MAX CUT), there is an explicit lower bound on the solution size, given in terms of the input size. However, in many cases, we do not have explicit lower bounds. We substantiate this with the example of the classic VERTEX COVER problem. In this problem, we are given a graph  $G$  and a positive integer  $k$ , and the goal is to test whether there exists a set of vertices  $C$  of size at most  $k$  that is a vertex cover (i.e., every edge has at least one endpoint in  $C$ ). Clearly, the size of a *maximum matching* of  $G$  or the value of the linear programming relaxation of an integer linear program for VERTEX COVER is a lower bound on the size of a vertex cover of  $G$ . Observe that these lower bounds are graph-dependent and not a direct function of the input size. This is what we mean by implicit lower bounds. Coming back to VERTEX COVER, we note that the aforementioned

bounds have led to the study of the problem VERTEX COVER ABOVE LP (or VERTEX COVER ABOVE MATCHING), which has played a central role in the development of the field of parameterized complexity [15].

In this article, we take the philosophy of above guarantee parameterization a significant conceptual step forward. In particular,

The goal of this article is to study another classical problem—namely, LONG CYCLE—from the viewpoint of a new implicit parameterization, that is neither standard nor an additive above guarantee parameterization, and which is natural for that problem.

The LONG PATH problem is to decide, given a directed or undirected  $n$ -vertex graph  $G$  and an integer  $k$ , whether  $G$  contains a path on at least  $k$  vertices, that is, a self-avoiding walk with at least  $k$  vertices. Similarly, the LONG CYCLE problem is to decide whether  $G$  contains a cycle of length at least  $k$ , that is, a closed self-avoiding walk with at least  $k$  vertices. These problems are natural generalizations of the classical HAMILTONIAN PATH and HAMILTONIAN CYCLE problems and have been actively studied. In particular, there is a plethora of results about parameterized complexity of LONG PATH and LONG CYCLE (see, e.g., [6, 7, 11, 12, 26, 29, 37, 39, 40, 47]) since the early work of Monien [44]. Let us just mention here that the fastest known randomized algorithm for LONG PATH is due to Björklund et al. [6] and runs in time  $1.657^k \cdot n^{O(1)}$ , whereas the fastest known deterministic algorithm is due to Tsour [46] and runs in time  $2.554^k \cdot n^{O(1)}$ . For LONG CYCLE, the randomized algorithm with the currently best running time of  $4^k \cdot n^{O(1)}$  was given by Zehavi [49], and the best deterministic algorithm was given by Fomin et al. [28] and runs in time  $4.884^k \cdot n^{O(1)}$ .

For LONG PATH, the investigation of above guarantee parameterizations was initiated by Bezáková et al. [3]. Let  $s$  and  $t$  be two vertices of a graph  $G$ . Clearly, the length of any  $(s, t)$ -path in  $G$  is lower-bounded by the distance (being the length of a shortest path),  $d(s, t)$ , between these vertices. Based on this straightforward observation, Bezáková et al. [3] introduced the LONGEST DETOUR problem that asks, given a graph  $G$ , two vertices  $s, t$ , and a positive integer  $k$ , whether  $G$  has an  $(s, t)$ -path with at least  $d(s, t) + k$  vertices. They proved that for undirected graphs, this problem can be solved in time  $2^{O(k)} \cdot n^{O(1)}$ . That is, it is fixed-parameter tractable parameterized by  $k$ . For the variant of the problem where the question is whether  $G$  has an  $(s, t)$ -path with *exactly*  $d(s, t) + k$  vertices, a randomized algorithm with running time  $2.746^k \cdot n^{O(1)}$  and a deterministic algorithm with running time  $6.745^k \cdot n^{O(1)}$  were obtained. In a recent work, Fomin et al. [25] studied the parameterization of LONG PATH and LONG CYCLE above the degeneracy  $d$  of the input graph  $G$ . Formally, a graph  $G$  has degeneracy at most  $d$  if every subgraph of  $G$  has a vertex of degree at most  $d$ . A classic result by Erdős and Gallai [21] from 1959 states that any graph of degeneracy  $d > 1$  has a cycle (and hence, also a path) on at least  $d$  vertices. If the graph  $G$  is not guaranteed to be 2-connected, then deciding whether  $G$  contains a cycle of length  $d + 2$  is already NP-hard, and therefore parameterization above degeneracy does not make sense. However, when we add the requirement that  $G$  is 2-connected, then deciding whether  $G$  contains a cycle of length  $d + k$  parameterized by  $k$  can be done in time  $2^{O(k)} \cdot n^{O(1)}$ , and hence is fixed-parameter tractable parameterized by  $k$ . A similar situation holds for LONG PATH where connectivity replaces 2-connectivity.

### 1.1 Multiplicative Above Guarantee Parameterization

To the best of our knowledge, all successful above guarantee parameterizations are additive. Roughly speaking, this means that if we have a lower bound  $\tau$  on the optimal solution size, then

we ask whether we can find a solution of size (at least or at most)  $\tau + k$  in time  $f(k) \cdot n^{O(1)}$ . Our main message of this article is the following.

The goal of this article is to introduce the notion of multiplicative above guarantee parameterization. In multiplicative above guarantee parameterization, we ask whether we can find a solution of size (at least or at most)  $\tau \cdot k$  in time  $g(k) \cdot n^{O(1)}$ . We illustrate our new definition by designing several fixed-parameter algorithms parameterized multiplicatively above a guarantee.

We remark that Serna and Thilikos [45] stated a few problems in a way fitting parameterization above a multiplicative guarantee. However, these were shown to be para-NP-hard [34, 35].

Recall that the *girth* of a graph  $G$ , denoted by  $\gamma(G)$ , is the length of the shortest cycle in  $G$ . First, consider the problem where we are given a graph  $G$  and an integer  $k$  and the objective is to test whether there is a cycle of length at least  $\gamma(G) + k$  in  $G$ . If  $\gamma(G) \leq 2k + 6$ , then clearly we can solve the problem in time  $2^{O(k)} \cdot n^{O(1)}$  by using, for example, the algorithm of Zehavi [48] (adapted to find a cycle, rather than a path, of specific length). We now show that when  $2k + 6 < \gamma(G)$ , the problem is solvable in polynomial time. To this end, let  $(G, k)$  be a Yes-instance, and let  $C$  be a hypothetical solution. That is,  $C$  is a cycle of length at least  $\gamma(G) + k$ . Let  $g = \gamma(G)$ . Let  $u, v, w, x \in V(C)$  be such that when we traverse  $C$  in some order from  $u$ , we encounter  $v$  at distance  $\lfloor \frac{g}{2} \rfloor - 1$  from  $u$ , next we encounter  $w$  at additional distance  $\lfloor \frac{g}{2} \rfloor - 1$  from  $v$ , and lastly we encounter  $x$  at additional distance  $k' = (g+k) - (2\lfloor \frac{g}{2} \rfloor - 2)$  from  $w$ . Notice that  $k' \in \{k+2, k+3\}$ . Since the girth of  $G$  is  $g$  and  $k+3 < \frac{g}{2}$ , we have that shortest paths between  $u$  and  $v$ ,  $v$  and  $w$ , and  $w$  and  $x$  are unique and they are part of the cycle  $C$ . Therefore, we may compute the shortest paths between the pairs above, and later check whether  $u$  is reachable from  $x$  after deleting these shortest paths. Going over every possible choice of  $u, v, w, x \in V(C)$ , this leads to a polynomial time algorithm when  $2k + 6 < \gamma(G)$ . This implies that it can be decided in time  $2^{O(k)} \cdot n^{O(1)}$  whether a graph has a cycle with at least  $\gamma(G) + k$  vertices.

The informal argument above shows that LONG CYCLE parameterized additively above girth is not more “interesting” than just normal LONG CYCLE, where the input lower bound on solution size is the parameter. In a sense, it hints that the guarantee may be strengthened. In light of this, we introduce a new above guarantee version of LONG CYCLE (and LONG PATH), termed LONG CYCLE (LONG PATH, respectively) parameterized multiplicatively above girth, as follows. The input consists of a graph  $G$  and an integer  $k$ , and the objective is to decide whether there is a cycle (path, respectively) on at least  $k \cdot \gamma(G)$  vertices.

We note that the choice of girth as a guarantee for LONG CYCLE is very natural. Indeed, being the smallest length of a cycle in the graph and thereby the first guarantee that one could think of for LONG CYCLE, and also being computable in polynomial time, makes girth the perfect choice.<sup>1</sup> Moreover, graphs of large girth have been of interest since the pioneering result in probabilistic graph theory of Erdős [20] in the late 1950s [20]. They find applications in practice and have been extensively studied since then (see, e.g., [2, 4, 36]).

## 1.2 Our Contribution

We first prove our main result, which states that LONG CYCLE (and LONG PATH) parameterized multiplicatively above girth is in FPT. Specifically, we prove the following theorem.

<sup>1</sup>In fact, our initial goal was to study LONG CYCLE parameterized additively above girth. After we discovered the simple argument above, we were motivated to introduce a new, more appropriate, form of parameterization, which was the starting point of our work.

**THEOREM 1.1.** *LONG CYCLE (and LONG PATH) parameterized multiplicatively above girth is solvable in time  $2^{O(k^2)}n$ . However, LONG CYCLE (and LONG PATH) parameterized multiplicatively above girth does not admit a polynomial kernel unless  $\text{NP} \subseteq \text{coNP/poly}$ .*

As a complementary result to the above theorem, we assert that our parameterization is tight in the following sense.

**THEOREM 1.2.** *For any fixed constant  $\varepsilon > 0$ , LONG CYCLE (and LONG PATH) parameterized multiplicatively above  $g^{1+\varepsilon}$  is para-NP-hard, where  $g$  is the girth of the input graph.*

Next, we further extend the scope of the applicability of our parameterization by showing that also VERTEX COVER and CONNECTED VERTEX COVER parameterized multiplicatively above girth are in FPT. Given a graph  $G$  and an integer  $k$ , the objectives of these problems are to decide whether  $G$  has a vertex cover of size at most  $k \cdot \gamma(G)$ , or a connected vertex cover (i.e., a vertex cover that induces a connected subgraph) of size at most  $k \cdot \gamma(G)$ , respectively. Here,  $\lceil \gamma(G)/2 \rceil$  is a lower bound on the size of an optimal solution. Notably, for VERTEX COVER we further derive a polynomial kernel.

**THEOREM 1.3.** *VERTEX COVER and CONNECTED VERTEX COVER parameterized multiplicatively above girth are solvable in time  $2^{O(k \log k)}n$ . Moreover, VERTEX COVER parameterized multiplicatively above girth admits a polynomial kernel, while CONNECTED VERTEX COVER parameterized multiplicatively above girth does not admit a polynomial kernel unless  $\text{NP} \subseteq \text{coNP/poly}$ .*

Additionally, we show that MAX INTERNAL SPANNING TREE parameterized multiplicatively above girth is in FPT. Given a graph  $G$  and an integer  $k$ , the objective of this problem is to decide whether  $G$  has a spanning tree with at least  $k \cdot \gamma(G)$  internal vertices. Here,  $\gamma(G) - 1$  is a lower bound on the size of an optimal solution.

**THEOREM 1.4.** *MAX INTERNAL SPANNING TREE parameterized multiplicatively above girth is solvable in time  $2^{O(k \log k)}n$ .*

Taking into consideration only graphs whose girth is at least 4, we show that INDEPENDENT SET parameterized multiplicatively above girth is also in FPT. Given a graph  $G$  and an integer  $k$ , the objective of this problem is to decide whether  $G$  has an independent set of size at least  $k \cdot \gamma(G)$ . Here,  $\lfloor \gamma(G)/2 \rfloor$  is a lower bound on the size of an optimal solution. We remark that the requirement of having girth at least 4 is mandatory, because on general graphs (and, in particular, on graphs of girth 3), INDEPENDENT SET is W[1]-hard parameterized by the sought solution size [15].

**THEOREM 1.5.** *INDEPENDENT SET on graphs of girth at least 4 parameterized multiplicatively above girth is solvable in time  $2^{O(k^2 \log k)}n$ .*

Lastly, we observe that parameterization above girth (even additively) can often yield NP-hardness when  $k$  is a fixed constant. Specifically, we give FEEDBACK VERTEX SET and CYCLE PACKING as illustrative simple examples.

**THEOREM 1.6.** *FEEDBACK VERTEX SET and CYCLE PACKING parameterized additively above girth are para-NP-hard.*

We remark that we further discuss our results, as well as questions left open, in Section 8.

## 2 PRELIMINARIES

For graph-theoretic terminology not explicitly defined here, we refer to the book of Diestel [16]. Throughout the article, we consider finite undirected graphs. For a graph  $G$ , let  $V(G)$  and  $E(G)$  denote its vertex set and edge set, respectively. When  $G$  is clear from context, let  $n = |V(G)|$  and  $m = |E(G)|$ . The maximum degree of a vertex in  $G$  is denoted by  $\Delta(G)$ . For a vertex  $v \in V(G)$ ,

let  $N_G(v)$  denote the neighborhood of  $v$  in  $G$ . Given a subset  $U \subseteq V(G)$ , let  $G[U]$  be the subgraph of  $G$  induced by  $U$ . The *girth* of  $G$  is the length of the shortest cycle in  $G$ ,<sup>2</sup> and is denoted by  $\gamma(G)$ . A *vertex cover* of  $G$  is a set of vertices in  $G$  whose removal from  $G$  yields an edgeless graph. The *vertex cover number* of  $G$  is the smallest size of a vertex cover of  $G$ . A *feedback vertex set* of  $G$  is a set of vertices in  $G$  whose removal from  $G$  yields a forest. The *feedback vertex set number* of  $G$  is the smallest size of a feedback vertex set of  $G$ . For any  $t \in \mathbb{N}$ , let  $C_t$  denote the cycle on  $t$  vertices and  $K_t$  denote the complete graph on  $t$  vertices.

A *subdivision* of an edge  $e = \{u, v\} \in E(G)$  is its replacement by a new degree-2 vertex whose neighbors are  $u$  and  $v$ . A *subdivision* of  $G$  is any graph that can be obtained by subdividing some of the edges of  $G$  (where a single edge can be subdivided multiple times). Let  $\text{Paths}(G)$  be the set of all (simple) paths in  $G$ . Given two (simple) paths  $P_1$  and  $P_2$  that share one or two endpoints and are internally vertex-disjoint, let  $P_1 + P_2$  denote the (simple) path or cycle (if both endpoints are shared) obtained by concatenating  $P_1$  and  $P_2$ . The *size* of a cycle or path is its number of vertices, and its *length* is its number of edges. A graph  $H$  is a *topological minor* of  $G$  if  $G$  contains a subgraph that is isomorphic to some subdivision of  $H$ . More explicitly, this notion is defined as follows.

*Definition 2.1 (Topological Minor).* A graph  $H$  is a *topological minor* of a graph  $G$  if there exist injective functions  $\phi : V(H) \rightarrow V(G)$  and  $\varphi : E(H) \rightarrow \text{Paths}(G)$  such that for all  $e = \{h, h'\} \in E(H)$ , the endpoints of  $\varphi(e)$  are  $\phi(h)$  and  $\phi(h')$ , for all distinct  $e, e' \in E(H)$ , the paths  $\varphi(e)$  and  $\varphi(e')$  are internally vertex-disjoint, and there do not exist a vertex  $v$  in the image of  $\phi$  and an edge  $e \in E(H)$  such that  $v$  is an internal vertex on  $\varphi(e)$ .

The Cartesian product of two graphs is defined as follows.

*Definition 2.2 (Cartesian Product of Graphs).* The *Cartesian product*  $G \times H$  of two graphs  $G$  and  $H$  is the graph whose vertex set is the Cartesian product  $V(G) \times V(H)$ , where any two vertices  $(u, u')$  and  $(v, v')$  are adjacent if and only if one of the following holds: (i)  $u = v$  and  $\{u', v'\} \in E(H)$ ; or (ii)  $u' = v'$  and  $\{u, v\} \in E(G)$ .

Treewidth is a measure of how “treelike” is a graph, formally defined as follows.

*Definition 2.3 (Treewidth).* A *tree decomposition* of a graph  $G$  is a pair  $(T, \beta)$  of a tree  $T$  and  $\beta : V(T) \rightarrow 2^{V(G)}$ , such that

- (1) for any edge  $\{x, y\} \in E(G)$  there exists a node  $v \in V(T)$  such that  $x, y \in \beta(v)$ , and
- (2) for any vertex  $x \in V(G)$ , the subgraph of  $T$  induced by the set  $T_x = \{v \in V(T) : x \in \beta(v)\}$  is a non-empty tree.

The *width* of  $(T, \beta)$  is  $\max_{v \in V(T)} \{|\beta(v)|\} - 1$ . The *treewidth* of  $G$  is the minimum width over all tree decompositions of  $G$ .

The treewidth of a graph can be efficiently approximated up to a constant factor as follows.

**PROPOSITION 2.1 ([10]).** *There exists an algorithm that, given a graph  $G$  and an integer  $t$ , in time  $2^{O(t)}n$  either determines that the treewidth of  $G$  is larger than  $t$ , or outputs a tree decomposition of  $G$  of width at most  $5t + 4$ .*

The following relation between treewidth and feedback vertex set number is folklore.

**PROPOSITION 2.2 ([15]).** *There exists an algorithm that, given a graph  $G$  with a feedback vertex set  $U$ , in time  $O(|U|n)$  outputs a tree decomposition of  $G$  of width  $|U|$ .*

<sup>2</sup>If  $G$  does not contain any cycle, then its girth is defined as  $\infty$ .

**Parameterized Complexity.** Let  $\Pi$  be a decision problem. In the framework of Parameterized Complexity, each instance of  $\Pi$  is associated with a *parameter*  $k$ . Here, the goal is to confine the combinatorial explosion in the running time of an algorithm for  $\Pi$  to depend only on  $k$ . Formally, we say that  $\Pi$  is *FPT* if any instance  $(I, k)$  of  $\Pi$  is solvable in time  $f(k) \cdot |I|^{O(1)}$ , where  $f$  is an arbitrary function of  $k$ . Nowadays, Parameterized Complexity supplies a rich toolkit to design fixed-parameter algorithms [15]. Parameterized Complexity also provides methods to show that a problem is unlikely to be in FPT. The main technique is the one of parameterized reductions analogous to those employed in classical complexity. Here, the concept of  $W[1]$ -hardness replaces the one of NP-hardness, and for reductions we need not only construct an equivalent instance solvable by a fixed-parameter algorithm, but also ensure that the size of the parameter in the new instance depends only on the size of the parameter in the original one. If there exists such a reduction transforming a problem known to be  $W[1]$ -hard to another problem  $\Pi$ , then the problem  $\Pi$  is  $W[1]$ -hard as well. Central  $W[1]$ -hard-problems include, for example, deciding whether a nondeterministic single-tape Turing machine accepts within  $k$  steps, CLIQUE parameterized by solution size, and INDEPENDENT SET parameterized by solution size. If there exists a fixed  $k$  such that  $\Pi$  is NP-hard, then the problem is said to be para-NP-hard, and in particular, it is not in FPT unless  $P=NP$ .

A companion notion to that of fixed-parameter tractability is the one of a polynomial kernel. Formally, a parameterized problem  $\Pi$  is said to admit a *compression* if there exists a not necessarily parameterized problem  $\Pi'$  and a polynomial-time algorithm that given an instance  $(I, k)$  of  $\Pi$ , outputs an equivalent<sup>3</sup> instance  $I'$  of  $\Pi'$  such that  $|I'| \leq p(k)$  where  $p$  is some function that depends only on  $k$ . If the function  $p$  is a polynomial, then the compression is said to be a *polynomial compression*. In case  $\Pi' = \Pi$ , we further say that  $\Pi$  admits a *kernel* or a *polynomial kernel*, depending on whether or not  $p$  is a polynomial. For more information on Parameterized Complexity in general and kernelization in particular, we refer the reader to recent books, such as those by Cygan et al. [15], Downey and Fellows [18], and Fomin et al. [24].

### 3 FIXED-PARAMETER ALGORITHM FOR LONG CYCLE

In this section, we consider the parameterized complexity of LONG CYCLE parameterized multiplicatively above girth, and prove Theorems 1.1 and 1.2. For our positive result, we require the following definition and propositions. First, we give the definition of a graph called a  $t$ -prism, which has  $2t$  vertices and  $3t$  edges (see Figure 1).

*Definition 3.1 (Prism).* For any  $t \in \mathbb{N}$ , the  $t$ -prism is the Cartesian product  $C_t \times K_2$ .

The following proposition asserts that a graph of sufficiently large treewidth necessarily contains a large prism as a topological minor.

PROPOSITION 3.1 ([5]). *For any  $k \in \mathbb{N}$ , any graph  $G$  of treewidth at least  $60k^2$  contains a  $k$ -prism as a topological minor.*<sup>4</sup>

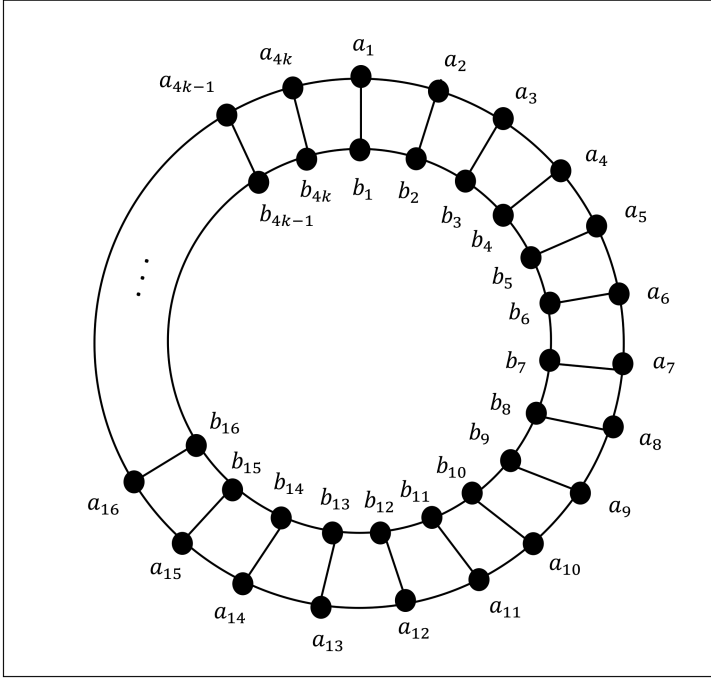
In case the treewidth of the graph is small, we will be able to solve the problem by making use of the following proposition.

PROPOSITION 3.2 ([8]). *There exists an algorithm that, given a graph  $G$  and a tree decomposition of  $G$  of width  $t$ , in time  $2^{O(t)}n$  outputs a cycle (if one exists) and a path in  $G$  of maximum sizes.*

The main combinatorial lemma we need for our positive result is as follows. This lemma handles the case where the treewidth of the graph is large.

<sup>3</sup>That is,  $(I, k)$  is a Yes-instance of  $\Pi$  if and only if  $I'$  is a Yes-instance of  $\Pi'$ .

<sup>4</sup>Although the result is mentioned in terms of minors, the proof given in Birmil e et al. [5] yields the result stated here.

Fig. 1. A  $4k$ -prism.

LEMMA 3.1. For any  $k \in \mathbb{N}$ , any graph  $G$  that contains the  $4k$ -prism as a topological minor also has a cycle on at least  $\gamma(G) \cdot k$  vertices.

PROOF. Let  $H$  be the  $4k$ -prism. Notice that the vertex set of  $H$  can be denoted by  $V(H) = \{a_1, a_2, \dots, a_{4k}, b_1, b_2, \dots, b_{4k}\}$  so that  $H[\{a_1, a_2, \dots, a_{4k}\}]$  and  $H[\{b_1, b_2, \dots, b_{4k}\}]$  are cycles and  $\{\{a_i, b_i\} : i \in \{1, 2, \dots, 4k\}\} \subseteq E(H)$  (see Figure 1). Let  $\phi: V(H) \rightarrow V(G)$  and  $\varphi: E(H) \rightarrow \text{Paths}(G)$  be some two functions that witness that  $H$  is a topological minor of  $G$ . Let  $S_1, \dots, S_{2k}$  be a set of  $2k$  vertex-disjoint cycles of length 4 in  $H$  defined as follows: For each  $i \in \{0, 1, \dots, 2k-1\}$ ,  $S_i = H[\{a_{2i+1}, a_{2i+2}, b_{2i+2}, b_{2i+1}\}]$  (see Figure 1). Additionally, we define two (not vertex-disjoint) cycles  $C$  and  $C'$  in  $H$  as follows:  $C = H[\{a_1, a_2, b_2, b_3, a_3, a_4, \dots, a_{4k-1}, a_{4k}, b_{4k}\}]$  and  $C' = H[\{a_1, b_1, b_2, a_2, a_3, b_3, \dots, b_{4k}, a_{4k}, a_1\}]$  (see Figure 1).

Now, for each  $i \in \{0, 1, \dots, 2k-1\}$ , let  $A_i = \varphi(\{a_{2i+1}, a_{2i+2}\}) + \varphi(\{a_{2i+2}, b_{2i+2}\}) + \varphi(\{b_{2i+2}, b_{2i+1}\}) + \varphi(\{b_{2i+1}, a_{2i+1}\})$ . This notation is well-defined as required to concatenate paths—specifically, here we concatenated internally vertex-disjoint paths sharing exactly one endpoint except for the last concatenation where both endpoints are shared (by the paths  $\varphi(\{a_{2i+1}, a_{2i+2}\}) + \varphi(\{a_{2i+2}, b_{2i+2}\}) + \varphi(\{b_{2i+2}, b_{2i+1}\})$  and  $\varphi(\{b_{2i+1}, a_{2i+1}\})$ ). Roughly speaking,  $A_i$  is the cycle to which  $\varphi$  maps  $S_i$ . Notice that  $\{A_i\}_{i=0}^{2k-1}$  is a collection of  $2k$  vertex-disjoint cycles in  $G$ . Therefore, together they contain at least  $2\gamma(G) \cdot k$  edges.

Additionally, let  $B = \varphi(\{a_1, a_2\}) + \varphi(\{a_2, b_2\}) + \varphi(\{b_2, b_3\}) + \varphi(\{b_3, a_3\}) + \dots + \varphi(\{a_{4k}, b_{4k}\}) + \varphi(\{b_{4k}, a_1\})$ . Again, this notation is well-defined as required to concatenate paths. Similarly, let  $B' = \varphi(\{a_1, b_1\}) + \varphi(\{b_1, b_2\}) + \varphi(\{b_2, a_2\}) + \varphi(\{a_2, a_3\}) + \dots + \varphi(\{b_{4k}, a_{4k}\}) + \varphi(\{a_{4k}, a_1\})$ . Roughly speaking,  $B$  and  $B'$  are the cycles to which  $\varphi$  maps  $C$  and  $C'$ , respectively. Notice that  $E(H) = E(C) \cup E(C')$ . Thus, we deduce that  $\bigcup_{i=0}^{2k-1} E(A_i) \subseteq E(C) \cup E(C')$ . From this, we further deduce that at least one among the cycles  $B$  and  $B'$  must contain at least half the edges in  $\bigcup_{i=0}^{2k-1} E(A_i)$ . Because we already



showed that  $|\bigcup_{i=0}^{2k-1} E(A_i)| \geq 2\gamma(G) \cdot k$ , this means that at least one among the cycles  $B$  and  $B'$  has length (and therefore also size) at least  $\gamma(G) \cdot k$ .  $\square$

By Proposition 3.1, any graph  $G$  of treewidth at least  $60 \cdot (4k)^2 = 960k^2$  contains a  $4k$ -prism as a topological minor. Thus, we derive the following corollary to Lemma 3.1.

**COROLLARY 3.1.** *For any  $k \in \mathbb{N}$ , any graph  $G$  of treewidth at least  $960k^2$  has a cycle on at least  $\gamma(G) \cdot k$  vertices.*

We are now ready to prove our main theorem, restated below.

**THEOREM 1.1.** *LONG CYCLE (and LONG PATH) parameterized multiplicatively above girth is solvable in time  $2^{O(k^2)}n$ . However, LONG CYCLE (and LONG PATH) parameterized multiplicatively above girth does not admit a polynomial kernel unless  $\text{NP} \subseteq \text{coNP/poly}$ .*

**PROOF.** We first consider the fixed-parameter tractability of the problem. Given an instance  $(G, k)$  of LONG CYCLE (LONG PATH), the algorithm is executed as follows. First, it calls the algorithm in Proposition 2.1 with  $t = 960k^2$  in time  $2^{O(t)}n = 2^{O(k^2)}n$  to either determine that the treewidth of  $G$  is larger than  $t$ , or output a tree decomposition of  $G$  of width at most  $5t + 4$ . In the first case, it concludes that  $(G, k)$  is a Yes-instance of LONG CYCLE (LONG PATH, respectively), which is correct by Corollary 3.1. In the second case, it calls the algorithm in Proposition 3.2 to obtain a cycle (path, respectively) of maximum size in  $G$  in time  $2^{O(5t+4)}n = 2^{O(k^2)}n$ , and concludes that  $(G, k)$  is a Yes-instance of LONG CYCLE (LONG PATH, respectively) if and only if the size of this cycle (path, respectively) is at least  $\gamma(G) \cdot k$ . This completes the proof of the first part of the theorem.

For the second part of the theorem, we note that LONG CYCLE (LONG PATH, respectively) is known not to admit a polynomial kernel parameterized by the sought solution size unless  $\text{NP} \subseteq \text{coNP/poly}$  [9, 15]. This is clearly also the case when parameterized by a parameter upper-bounded by the sought solution size, and in particular when parameterized multiplicatively above girth.  $\square$

**THEOREM 1.2.** *For any fixed constant  $\varepsilon > 0$ , LONG CYCLE (and LONG PATH) parameterized multiplicatively above  $g^{1+\varepsilon}$  is para-NP-hard, where  $g$  is the girth of the input graph.*

**PROOF.** Consider some fixed constant  $\varepsilon > 0$ . Our proof is based on a reduction from the HAMILTONIAN CYCLE (HAMILTONIAN PATH, respectively) problem, which is known to be NP-hard [30], to LONG CYCLE (LONG PATH, respectively) parameterized multiplicatively above  $g^{1+\varepsilon}$ . In the HAMILTONIAN CYCLE (HAMILTONIAN PATH, respectively) problem, we are given a graph  $G$  and the objective is to decide whether  $G$  contains a (simple) cycle (path, respectively) on  $n$  vertices. In what follows, we only describe the proof for LONG CYCLE (since the proof for LONG PATH is similar).

We now describe the reduction. To this end, let  $G$  be an instance of HAMILTONIAN CYCLE. Let  $b$  be the smallest positive integer such that  $b(1 + \varepsilon) \in \mathbb{N}$ . (Note that  $b$  depends only on  $\varepsilon$ .) Let  $a = b(1 + \varepsilon) - 1$ . Notice that  $b \leq a$  (since  $b \cdot \varepsilon$  is a positive integer). Now, subdivide each edge in  $G$   $n^a - 1$  times, and let  $G'$  be the resulting graph. Let  $H$  be the disjoint union of  $G'$  and  $C$ , where  $C$  is a cycle of length  $n^b$ . Finally, set  $k = 1$ , and let  $(H, k)$  be the output instance of LONG CYCLE parameterized multiplicatively above  $g^{1+\varepsilon}$ . Notice that here,  $g$  is the girth of  $H$ , i.e.,  $g = \gamma(H)$ .

Notice that for any  $\ell \in \mathbb{N}$ , for any cycle of size  $\ell$  in  $G$ , there is a corresponding cycle of size  $\ell \cdot n^a$  in  $G'$ , and vice versa. This implies that any cycle in  $G'$  has size at least  $3n^a$ . Thus, as  $b \leq a$ , there is a unique shortest cycle in  $H$ , which is  $C$ . Therefore,  $\gamma(H) = n^b$ . Then, any cycle (path, respectively)  $C'$  on at least  $(\gamma(H))^{1+\varepsilon}k = n^{b(1+\varepsilon)} = n^{1+a}$  vertices in  $H$  is fully contained in  $G'$ . From this, we conclude that for any cycle of size  $n$  in  $G$ , there is a corresponding cycle of size  $(\gamma(H))^{1+\varepsilon}k$  in  $H$ , and vice versa. This yields the correctness of the reduction. In particular, a fixed-parameter algorithm

for LONG CYCLE parameterized multiplicatively above  $g^{1+\epsilon}$  would solve  $(H, k)$  in polynomial time (because  $k = 1$ ), and thereby yield a polynomial-time algorithm for HAMILTONIAN CYCLE.  $\square$

#### 4 FIXED-PARAMETER ALGORITHMS FOR (CONNECTED) VERTEX COVER

In this section, we consider the parameterized complexity of (CONNECTED) VERTEX COVER parameterized multiplicatively above girth, and prove Theorem 1.3. The following proposition asserts that every graph contains either a small feedback vertex set or a large number of vertex-disjoint cycles (with a logarithmic gap), which can also be computed efficiently.

PROPOSITION 4.1 ([22, 41]). *For some fixed constant  $c \in \mathbb{N}$ , there exists an algorithm that, given any  $r \in \mathbb{N}$  and a graph  $G$ , in time  $r^{O(1)}n$  outputs either  $r$  vertex-disjoint cycles in  $G$  or a feedback vertex set of  $G$  of size at most  $cr \log r$ .*

In what follows, we use  $c$  to refer to the constant in this proposition. In case the treewidth of the graph is small, we will be able to solve the problem by making use of the following proposition.

PROPOSITION 4.2 ([15, 27]). *There exist algorithms for VERTEX COVER and CONNECTED VERTEX COVER that run in time  $2^{O(t)}n$ , where  $t$  is the treewidth of the input graph.<sup>5</sup>*

Further, in case the feedback vertex set number is small, we will make use of the following proposition for kernelization purposes.

PROPOSITION 4.3 ([38]). *The VERTEX COVER problem admits a polynomial kernel parameterized by the feedback vertex set number  $f$  of the input graph with  $O(f^3)$  vertices.*

The main combinatorial lemma required for our algorithms handles the case where the feedback vertex set number (and hence also treewidth) of the graph is high.

LEMMA 4.1. *For any  $r \in \mathbb{N}$ , any graph  $G$  that contains  $r$  vertex-disjoint cycles has vertex cover number at least  $\frac{\gamma(G)}{2} \cdot r$ .*

PROOF. Let  $C$  be a collection of  $r$  vertex-disjoint cycles of  $G$ . Consider any vertex cover  $U$  of  $G$ . Then,  $U$  must contain at least half the vertices of each cycle in  $C$  in order to contain an endpoint of every edge of that cycle. Since each cycle in  $C$  contains at least  $\gamma(G)$  vertices, the lemma follows.  $\square$

We are now ready to prove our main theorem.

THEOREM 1.3. *VERTEX COVER and CONNECTED VERTEX COVER parameterized multiplicatively above girth are solvable in time  $2^{O(k \log k)}n$ . Moreover, VERTEX COVER parameterized multiplicatively above girth admits a polynomial kernel, while CONNECTED VERTEX COVER parameterized multiplicatively above girth does not admit a polynomial kernel unless  $\text{NP} \subseteq \text{coNP/poly}$ .*

PROOF. We first consider the fixed-parameter tractability of the problems. Given an instance  $(G, k)$  of (CONNECTED) VERTEX COVER, the algorithm is executed as follows. First, the algorithm calls the algorithm in Proposition 4.1 with  $r = 2k + 1$  in time  $r^{O(1)}n = k^{O(1)}n$  to obtain either a feedback vertex set of  $G$  of size at most  $cr \log r$  or  $r > 2k$  vertex-disjoint cycles in  $G$ . In the second case, by Lemma 4.1  $G$  has vertex cover number strictly larger than  $\frac{\gamma(G)}{2} \cdot 2k = \gamma(G) \cdot k$ . Thus, for both VERTEX COVER and CONNECTED VERTEX COVER the algorithm correctly determines that the answer is No. In the first case, the algorithm calls the algorithm in Proposition 2.2 in time  $O(k \log k \cdot n)$  to obtain a tree decomposition of  $G$  of width at most  $cr \log r = O(k \log k)$ . Then, it calls the algorithm in Proposition 4.2 in time  $2^{O(k \log k)}n$  to obtain a (connected) vertex cover of  $G$

<sup>5</sup>Such an algorithm for CONNECTED VERTEX COVER is not given explicitly, but it is easily seen to follow from the approach of dynamic programming over tree decompositions using representative sets of Fomin et al. [27].

of minimum size, and concludes that  $(G, k)$  is a Yes-instance of (CONNECTED) VERTEX COVER if and only if the output (connected) vertex cover has size at most  $\gamma(G) \cdot k$ . This completes the proof of the first part of the theorem.

For the second part of the theorem, first consider VERTEX COVER parameterized multiplicatively above girth. Then, we proceed similarly to the fixed-parameter algorithm above. More precisely, given an instance  $(G, k)$  of VERTEX COVER, the algorithm is executed as follows. First, the algorithm calls the algorithm in Proposition 4.1 with  $r = 2k + 1$  in time  $r^{O(1)}n = k^{O(1)}n$  to obtain either a feedback vertex set of  $G$  of size at most  $cr \log r$  or  $r > 2k$  vertex-disjoint cycles in  $G$ . In the second case, the algorithm correctly determines that the answer is No. In the first case, the algorithm calls the polynomial-time algorithm in Proposition 4.3 to obtain an equivalent instance of VERTEX COVER with  $O((cr \log r)^3) = O((k \log k)^3)$  vertices and in particular of size polynomial in  $k$ .<sup>6</sup>

Lastly, we note that CONNECTED VERTEX COVER is known not to admit a polynomial kernel parameterized by the sought-out solution size unless  $\text{NP} \subseteq \text{coNP/poly}$  [15, 17]. This is clearly also the case when parameterized by a parameter upper-bounded by the sought solution size, and in particular when parameterized multiplicatively above girth.  $\square$

## 5 FIXED-PARAMETER ALGORITHM FOR MAX INTERNAL SPANNING TREE

In this section, we consider the parameterized complexity of MAX INTERNAL SPANNING TREE parameterized multiplicatively above girth, and prove Theorem 1.4.

In case the treewidth of the graph is small, we will be able to solve the problem by making use of the following proposition.

**PROPOSITION 5.1** ([15, 27]). *There exists an algorithm for MAX INTERNAL SPANNING TREE that runs in time  $2^{O(t)}n$ , where  $t$  is the treewidth of the input graph.*<sup>7</sup>

As before, the main combinatorial lemmas required for our algorithm handle the case where the feedback vertex set number (and hence also treewidth) of the graph is high.

**LEMMA 5.1.** *For any  $r \in \mathbb{N}$ , any connected graph  $G$  that contains  $r$  vertex-disjoint cycles has a spanning tree with at least  $(\gamma(G) - 1) \cdot r - 1$  internal vertices.*

**PROOF.** Let  $C$  be a collection of  $r$  vertex-disjoint cycles of  $G$ . Let  $H$  be the graph obtained from  $G$  by replacing each cycle  $C \in C$  by a single new vertex  $v_C$  (that is made adjacent to all vertices previously adjacent to at least one vertex in  $C$ ). Let  $T$  be some spanning tree of  $H$  (whose existence follows from the supposition that  $G$ , and hence also  $H$ , is connected), and root it arbitrarily.

Now, we traverse  $T$  in preorder, and when we encounter a new vertex of the form  $v_C$ , we perform the following operations. Let  $p$  be the parent of  $v_C$  in  $T$  (if it is not the root), and let  $u$  be some vertex in  $C$  that is adjacent to  $p$  (if  $v_C$  is the root, let  $u$  be any vertex in  $C$ ). Then, replace  $v_C$  in  $T$  by a path  $P$  from  $u$  to one of its neighbors,  $x$ , in  $C$  so that this path  $P$  contains exactly all the edges of  $C$  except one (between  $u$  and the chosen neighbor  $x$ ). Here, make  $u$  be the child of  $p$  in  $T$ , and every child  $w$  of  $v_C$  is handled as follows: if  $w \in V(G)$ , then let  $y$  be some vertex in  $V(P)$  such that  $\{y, w\} \in E(G)$ , and otherwise (if  $w \notin V(G)$ ), then let  $y$  be some vertex in  $V(P)$  such that there exists a vertex in the cycle represented by  $w$  that is a neighbor of  $y$ ; then, add the edge  $\{y, w\}$  to  $T$  (this replaces the edge  $\{v_C, w\}$ ), and consider  $y$  to be the parent of  $w$ .<sup>8</sup>

<sup>6</sup>Similarly, the question of the kernelization complexity of MAX INTERNAL SPANNING TREE parameterized multiplicatively above girth boils down to the question of the kernelization complexity of MAX INTERNAL SPANNING TREE parameterized by the feedback vertex set number of the input graph, which is not known.

<sup>7</sup>As before, such an algorithm for MAX INTERNAL SPANNING TREE is not given explicitly, but it is easily seen to follow from the approach of Fomin et al. [27].

<sup>8</sup>Intuitively, the subtree of  $w$  is "hung" from a vertex of the path  $P$  such that  $w$  (or some vertex in the cycle represented by  $w$ , if  $w$  is a new vertex) is adjacent to it in  $G$ .

Notice that at the end of this process, we obtain a spanning tree of  $G$  with the following property. For each cycle  $C \in \mathcal{C}$ , all vertices except possibly one are internal vertices, with the exception of possibly one cycle (if there was a cycle represented by the root of  $T$ ) where all vertices except possibly two are internal vertices. Thus, this spanning tree has at least  $(\gamma(G) - 1) \cdot r - 1$  internal vertices.  $\square$

We are now ready to prove our main theorem.

**THEOREM 1.4.** *MAX INTERNAL SPANNING TREE parameterized multiplicatively above girth is solvable in time  $2^{O(k \log k)} n$ .*

**PROOF.** Given an instance  $(G, k)$  of MAX INTERNAL SPANNING TREE, the algorithm is executed as follows. Without loss of generality, we suppose that  $G$  is connected in the case of MAX INTERNAL SPANNING TREE, else we clearly have a No-instance of this problem. First, the algorithm calls the algorithm in Proposition 4.1 with  $r = 2k + 1$  in time  $r^{O(1)} n = k^{O(1)} n$  to obtain either a feedback vertex set of  $G$  of size at most  $cr \log r$  or  $r > 2k$  vertex-disjoint cycles in  $G$ . In the second case, by Lemma 5.1  $G$  has a spanning tree with at least  $(\gamma(G) - 1) \cdot 2k - 1 \geq \gamma(G) \cdot k$  internal vertices (where the last inequality follows from the facts that  $\gamma(G) \geq 3$  and  $k \in \mathbb{N}$ ). Thus, the algorithm correctly determines that the answer is Yes. In the first case, the algorithm calls the algorithm in Proposition 2.2 in time  $O(k \log k \cdot n)$  to obtain a tree decomposition of  $G$  of width at most  $cr \log r = O(k \log k)$ . Then, it calls the algorithm in Proposition 5.1 in time  $2^{O(k \log k)} n$  to obtain a spanning tree of  $G$  of maximum number of internal vertices, and concludes that  $(G, k)$  is a Yes-instance of MAX INTERNAL SPANNING TREE if and only if the output spanning tree has at least  $\gamma(G) \cdot k$  internal vertices.  $\square$

## 6 FIXED-PARAMETER ALGORITHM FOR INDEPENDENT SET ON GRAPHS OF GIRTH AT LEAST 4

In this section, we consider the parameterized complexity of INDEPENDENT SET on graphs of girth at least 4 parameterized multiplicatively above girth, and prove Theorem 1.5.

In case the treewidth of the graph is small, we will be able to solve the problem by making use of the following proposition.

**PROPOSITION 6.1** ([15]). *There exists an algorithm for INDEPENDENT SET that runs in time  $2^{O(t)} n$ , where  $t$  is the treewidth of the input graph.*

Unlike before, here we distinguish between the case where  $\Delta(G')$  is “large” and the case where it is “small,” where  $G'$  is the subgraph obtained from  $G$  by iteratively removing vertices of degree 1. Only in the latter case, we will consider its treewidth. For the first case, we will make use of the following lemma.

**LEMMA 6.1.** *Any graph  $G$  of minimum degree 2 contains an independent set of size at least  $\frac{\lfloor \gamma(G)/2 \rfloor - 1}{2} \Delta(G)$ .*

**PROOF.** Let  $v$  be a vertex of degree  $\Delta(G)$  in  $G$ . For every  $u \in N_G(v)$ , construct a path  $P_u$  as follows: initialize  $P_u$  to be the path consisting only of  $v$  and  $u$ ; then, if the endpoint of  $P_u$  that is not  $v$  has a neighbor that does not belong to  $P_u$ , extend  $P_u$  using such a neighbor (if there is more than one choice for the neighbor, then choose one arbitrarily). For every  $u \in N_G(v)$ , because the minimum degree of  $G$  is 2, the path  $P_u$  has at least  $\gamma(G)$  vertices. Now, let  $P'_u$  denote the subpath of  $P_u$  that consists of the  $\lfloor \gamma(G)/2 \rfloor$  vertices that are closest to  $v$  (including  $v$ ). Notice that for all distinct  $u, w \in N_G(v)$ , we have that (i)  $V(P'_u) \cap V(P'_w) = \{v\}$  and (ii) there do not exist  $x \in V(P'_u)$  and  $y \in V(P'_w)$  that are adjacent—indeed, if any of these two conditions was false, then the union

of  $P'_u$  and  $P'_v$  would have contained a cycle of length at most  $|V(P'_u)| + |V(P'_v)| - 1 = 2\lfloor \gamma(G)/2 \rfloor - 1 < \gamma(G)$ , which is a contradiction.

Let  $S \subseteq V(G)$  be defined as follows. For every  $u \in N_G(v)$ , insert into  $S$  every vertex in  $V(P'_u)$  whose distance from  $v$  in  $P'_u$  is even (so, we insert the second, fourth, sixth, etc., vertices on  $P'_u$  when the first vertex is supposed to be  $v$ ). Thus, by condition (i) above,  $|S| \geq \frac{\lfloor \gamma(G)/2 \rfloor - 1}{2} \Delta(G)$ . Moreover, by condition (ii) and because every path  $P'_u$ ,  $u \in N_G(v)$ , is an induced path (else we obtain a cycle of length smaller than  $\gamma(G)$ ), we have that  $S$  is an independent set.  $\square$

For the second case and where the feedback vertex set number (and hence also treewidth) is large, we will make use of the following lemma.

**LEMMA 6.2.** *For any  $r \in \mathbb{N}$ , any graph  $G$  that contains  $r$  vertex-disjoint cycles has an independent set of size at least  $\frac{\gamma(G) \cdot r}{\Delta(G)+1}$ .*

**PROOF.** Because  $G$  contains  $r$  vertex-disjoint cycles, we have that  $|V(G)| \geq \gamma(G) \cdot r$ . Observe that every vertex in  $G$  can be adjacent to at most  $\Delta(G)$  other vertices. So,  $G$  must contain an independent set of size at least  $\frac{|V(G)|}{\Delta(G)+1} \geq \frac{\gamma(G) \cdot r}{\Delta(G)+1}$ .  $\square$

We are now ready to prove our main theorem.

**THEOREM 1.5.** **INDEPENDENT SET** *on graphs of girth at least 4 parameterized multiplicatively above girth is solvable in time  $2^{O(k^2 \log k)} n$ .*

**PROOF.** Given an instance  $(G, k)$  of **INDEPENDENT SET** on graphs of girth at least 4, the algorithm is executed as follows. Let  $G'$  be the subgraph of  $G$  obtained by iteratively removing vertices of degree at most 1 from  $G$ . Observe that  $\gamma(G) = \gamma(G')$ . We distinguish between two cases based on  $\Delta(G')$ . First, suppose that  $\Delta(G') \geq 16k$ . Then, by Lemma 6.1,  $G'$  (and hence also  $G$ ) contains an independent set of size at least  $\frac{\lfloor \gamma(G')/2 \rfloor - 1}{2} \Delta(G') \geq \frac{\gamma(G)-3}{4} \cdot 16k = 4\gamma(G)k - 12k \geq \gamma(G)k$ . Here, the last inequality followed because  $\gamma(G) \geq 4$ . So, the algorithm correctly determines that the answer is Yes.

Second, suppose that  $\Delta(G') \leq 16k$ . Then, the algorithm calls the algorithm in Proposition 4.1 with  $r = (16k + 1)k$  in time  $r^{O(1)} n = k^{O(1)} n$  to obtain either a feedback vertex set of  $G$  of size at most  $cr \log r$  or  $r$  vertex-disjoint cycles in  $G$ . First, consider the second (sub)case. Then,  $G'$  has the same  $r$  vertex-disjoint cycles. So, by Lemma 6.2  $G$  has an independent set of size at least  $\frac{\gamma(G') \cdot r}{\Delta(G')+1} \geq \frac{\gamma(G) \cdot (16k+1)k}{16k+1} = \gamma(G) \cdot k$ . Thus, the algorithm correctly determines that the answer is Yes. Now, consider the first (sub)case. Here, the algorithm calls the algorithm in Proposition 2.2 in time  $O(k^2 \log k \cdot n)$  to obtain a tree decomposition of  $G$  of width at most  $cr^2 \log r = O(k^2 \log k)$ . Then, it calls the algorithm in Proposition 6.1 in time  $2^{O(k^2 \log k)} n$  to obtain an independent set of  $G$  of maximum size, and concludes that  $(G, k)$  is a Yes-instance of **INDEPENDENT SET** if and only if the output set has size at least  $\gamma(G) \cdot k$ .  $\square$

## 7 HARDNESS FOR FEEDBACK VERTEX SET AND CYCLE PACKING

Lastly, we prove the correctness of Theorem 1.6.

**THEOREM 1.6.** **FEEDBACK VERTEX SET** and **CYCLE PACKING** *parameterized additively above girth are para-NP-hard.*

**PROOF.** We only prove the lemma for **FEEDBACK VERTEX SET** since the proof for **CYCLE PACKING** follows from symmetric arguments. For this purpose, we give a reduction from **FEEDBACK VERTEX SET** itself, which is an NP-hard problem [30]. Toward this, let  $(G, k)$  be an instance of **FEEDBACK VERTEX SET**. Then, obtain  $H$  from  $G$  by subdividing each edge of  $G$   $k$  times, and adding a new

cycle of size  $k$ . Clearly,  $G$  has a feedback vertex set of size at most  $k$  if and only if  $H$  has a feedback vertex set of size at most  $k + 1$  (the extra 1 is required to hit the new cycle). Moreover, the girth of  $H$  is exactly  $k$ . Thus, an algorithm for FEEDBACK VERTEX SET parameterized additively above girth should solve  $(H, 1)$  in polynomial time (since the parameter is 1), thereby determining whether the feedback vertex set number of  $H$  is at most  $k$ , and consequently solving  $(G, k)$ .  $\square$

## 8 CONCLUSION AND OPEN PROBLEMS

In this article, we highlighted the notion and usefulness of parameterization multiplicatively above a guarantee, demonstrated using girth as the guarantee. As we have argued, girth is (arguably) the most natural guarantee for the LONG CYCLE problem, and indeed this problem was our starting point. After proving that LONG CYCLE problem (as well as LONG PATH) parameterized multiplicatively above a girth is in FPT, we turned to consider other problems under this parameterization, motivated by the extensive body of work existing on graphs of large girth. It turned out that parameterization multiplicatively above girth is quite fruitful—we were able to show that VERTEX COVER, CONNECTED VERTEX COVER, MAX INTERNAL SPANNING TREE and INDEPENDENT SET on graphs of girth at least 4 are in FPT with this parameterization. All of our proofs followed the same win-win approach (having either small treewidth or a specific topological minor), yet the details in its application varied. Additionally, we obtained a polynomial kernel for VERTEX COVER parameterized multiplicatively above a girth.

We conclude this article with directions for further research and open questions.

- The main venue for research that we would like to highlight is the exploration of other guarantees that make sense for multiplicative parameterization.
- Can the time complexities (and kernel size) presented in this article be substantially improved? In particular, for which of the problems considered in this article can we design a fixed-parameter algorithm of time complexity  $2^{O(k)} \cdot n$ , and for which can we refute the existence of such an algorithm (say, under the Exponential Time Hypothesis)?
- We have shown that for FEEDBACK VERTEX SET and CYCLE PACKING parameterization (even additively) above girth is futile. However, for these problems, girth is not a guarantee. Are there problems where girth is a guarantee, yet parameterization (additively or multiplicatively) above it is futile?
- For problems on directed graphs, parameterization multiplicatively above girth (which is now the length of a shortest *directed* cycle) may also make sense. For example, what is the parameterized complexity of DIRECTED LONG CYCLE under this parameterization?

## REFERENCES

- [1] Noga Alon, Gregory Gutin, Eun Jung Kim, Stefan Szeider, and Anders Yeo. 2010. Solving MAX- $r$ -SAT above a tight lower bound. In *Proceedings of the 21st Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'10)*. SIAM, 511–517.
- [2] Mohsen Bayati, Andrea Montanari, and Amin Saberi. 2009. Generating random graphs with large girth. In *Proceedings of the 20th Annual ACM-SIAM Symposium on Discrete Algorithms*. SIAM, 566–575.
- [3] Ivona Bezáková, Radu Curticapean, Holger Dell, and Fedor V. Fomin. 2017. Finding detours is fixed-parameter tractable. In *44th International Colloquium on Automata, Languages, and Programming (ICALP'17) (Leibniz International Proceedings in Informatics (LIPIcs))*, Vol. 80. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 54:1–54:14.
- [4] Norman Biggs. 1998. Constructions for cubic graphs with large girth. *Electron. J. Combin.* (1998), A1–A1.
- [5] E. Birmelé, J. A. Bondy, and B. A. Reed. 2007. Brambles, prisms and grids. In *Graph Theory in Paris: Proceedings of a Conference in Memory of Claude Berge*, Adrian Bondy, Jean Fonlupt, Jean-Luc Fouquet, Jean-Claude Fournier, and Jorge L. Ramírez Alfonsín (Eds.). Birkhäuser Basel, Basel, 37–44.
- [6] Andreas Björklund, Thore Husfeldt, Petteri Kaski, and Mikko Koivisto. 2010. Narrow sieves for parameterized paths and packings. *CoRR* abs/1007.1161 (2010).
- [7] Hans L. Bodlaender. 1993. On linear time minor tests with depth-first search. *J. Algorithms* 14, 1 (1993), 1–23.

- [8] Hans L. Bodlaender, Marek Cygan, Stefan Kratsch, and Jesper Nederlof. 2015. Deterministic single exponential time algorithms for connectivity problems parameterized by treewidth. *Inf. Comput.* 243 (2015), 86–111. <https://doi.org/10.1016/j.ic.2014.12.008>
- [9] Hans L. Bodlaender, Rodney G. Downey, Michael R. Fellows, and Danny Hermelin. 2009. On problems without polynomial kernels. *J. Comput. Syst. Sci.* 75, 8 (2009), 423–434.
- [10] Hans L. Bodlaender, Pål Grønås Drange, Markus S. Dregi, Fedor V. Fomin, Daniel Lokshtanov, and Michal Pilipczuk. 2016. A  $c^k n$  5-approximation algorithm for treewidth. *SIAM J. Comput.* 45, 2 (2016), 317–378. <https://doi.org/10.1137/130947374>
- [11] Jianer Chen, Joachim Kneis, Songjian Lu, Daniel Mölle, Stefan Richter, Peter Rossmanith, Sing-Hoi Sze, and Fenghui Zhang. 2009. Randomized divide-and-conquer: Improved path, matching, and packing algorithms. *SIAM J. Comput.* 38, 6 (2009), 2526–2547. <https://doi.org/10.1137/080716475>
- [12] Jianer Chen, Songjian Lu, Sing-Hoi Sze, and Fenghui Zhang. 2007. Improved algorithms for path, matching, and packing problems. In *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'07)*. SIAM, 298–307.
- [13] Robert Crowston, Michael R. Fellows, Gregory Z. Gutin, Mark Jones, Eun Jung Kim, Fran Rosamond, Imre Z. Ruzsa, Stéphan Thomassé, and Anders Yeo. 2014. Satisfying more than half of a system of linear equations over GF(2): A multivariate approach. *J. Comput. Syst. Sci.* 80, 4 (2014), 687–696.
- [14] Robert Crowston, Mark Jones, Gabriele Muciaccia, Geevarghese Philip, Ashutosh Rai, and Saket Saurabh. 2013. Polynomial kernels for lambda-extendible properties parameterized above the Poljak-Turzik bound. In *IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS'13) (Leibniz International Proceedings in Informatics (LIPIcs))*, Vol. 24. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 43–54.
- [15] Marek Cygan, Fedor V. Fomin, Łukasz Kowalik, Daniel Lokshtanov, Dániel Marx, Marcin Pilipczuk, Michal Pilipczuk, and Saket Saurabh. 2015. *Parameterized Algorithms*. Springer. <https://doi.org/10.1007/978-3-319-21275-3>
- [16] Reinhard Diestel. 2005. *Graph Theory* (3rd ed.). Graduate Texts in Mathematics, Vol. 173. Springer-Verlag, Berlin. xvi+411 pages.
- [17] Michael Dom, Daniel Lokshtanov, and Saket Saurabh. 2014. Kernelization lower bounds through colors and IDs. *ACM Trans. Algorithms (TALG)* 11, 2 (2014), 13.
- [18] Rodney G. Downey and Michael R. Fellows. 2013. *Fundamentals of Parameterized Complexity*. Springer. <https://doi.org/10.1007/978-1-4471-5559-1>
- [19] Zdenek Dvorák and Matthias Mnich. 2017. Large independent sets in triangle-free planar graphs. *SIAM J. Discr. Math.* 31, 2 (2017), 1355–1373.
- [20] Paul Erdős. 1959. Graph theory and probability. *Can. J. Math.* 11 (1959), 34–38.
- [21] P. Erdős and T. Gallai. 1959. On maximal paths and circuits of graphs. *Acta Math. Acad. Sci. Hungar* 10 (1959), 337–356 (unbound insert).
- [22] P. Erdős and L. Pósa. 1965. On independent circuits contained in a graph. *Can. J. Math.* 17 (1965), 347–352.
- [23] Michael Etscheid and Matthias Mnich. 2018. Linear kernels and linear-time algorithms for finding large cuts. *Algorithmica* 80, 9 (2018), 2574–2615.
- [24] F. V. Fomin, D. Lokshtanov, S. Saurabh, and M. Zehavi. 2018. *Kernelization: Theory of Parameterized Preprocessing*. Cambridge University Press.
- [25] Fedor V. Fomin, Petr A. Golovach, Daniel Lokshtanov, Fahad Panolan, Saket Saurabh, and Meirav Zehavi. 2019. Going far from degeneracy. In *27th Annual European Symposium on Algorithms (ESA'19) (Leibniz International Proceedings in Informatics (LIPIcs))*, Michael A. Bender, Ola Svensson, and Grzegorz Herman (Eds.), Vol. 144. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 47:1–47:14. <https://doi.org/10.4230/LIPIcs.ESA.2019.47>
- [26] Fedor V. Fomin, Daniel Lokshtanov, Fahad Panolan, and Saket Saurabh. 2016. Efficient computation of representative families with applications in parameterized and exact algorithms. *J. ACM* 63, 4 (2016), 29:1–29:60. <https://doi.org/10.1145/2886094>
- [27] Fedor V. Fomin, Daniel Lokshtanov, Fahad Panolan, and Saket Saurabh. 2016. Efficient computation of representative families with applications in parameterized and exact algorithms. *J. ACM* 63, 4 (2016), 29:1–29:60.
- [28] Fedor V. Fomin, Daniel Lokshtanov, Fahad Panolan, Saket Saurabh, and Meirav Zehavi. 2018. Long directed  $(s, t)$ -path: FPT algorithm. *Inf. Process. Lett.* 140 (2018), 8–12.
- [29] Harold N. Gabow and Shuxin Nie. 2008. Finding a long directed cycle. *ACM Trans. Algorithms* 4, 1 (2008).
- [30] M. R. Garey and David S. Johnson. 1979. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman.
- [31] Gregory Gutin, Eun Jung Kim, Michael Lampis, and Valia Mitsou. 2011. Vertex cover problem parameterized above and below tight bounds. *Theory Comput. Syst.* 48, 2 (2011), 402–410.

- [32] Gregory Gutin, Leo van Iersel, Matthias Mnich, and Anders Yeo. 2012. Every ternary permutation constraint satisfaction problem parameterized above average has a kernel with a quadratic number of variables. *J. Comput. Syst. Sci.* 78, 1 (2012), 151–163.
- [33] Gregory Z. Gutin and Viresh Patel. 2016. Parameterized traveling salesman problem: Beating the average. *SIAM J. Discrete Math.* 30, 1 (2016), 220–238.
- [34] Gregory Z. Gutin, Arash Rafiey, Stefan Szeider, and Anders Yeo. 2007. The linear arrangement problem parameterized above guaranteed value. *Theory Comput. Syst.* 41, 3 (2007), 521–538.
- [35] Gregory Z. Gutin, Stefan Szeider, and Anders Yeo. 2008. Fixed-parameter complexity of minimum profile problems. *Algorithmica* 52, 2 (2008), 133–152.
- [36] Carlos Hoppen and Nicholas Wormald. 2018. Local algorithms, regular graphs of large girth, and random regular graphs. *Combinatorica* 38, 3 (2018), 619–664.
- [37] Falk Hüffner, Sebastian Wernicke, and Thomas Zichner. 2008. Algorithm engineering for color-coding with applications to signaling pathway detection. *Algorithmica* 52, 2 (2008), 114–132. <https://doi.org/10.1007/s00453-007-9008-7>
- [38] Bart M. P. Jansen and Hans L. Bodlaender. 2013. Vertex cover kernelization revisited: Upper and lower bounds for a refined parameter. *Theory Comput. Syst.* 53, 2 (2013), 263–299.
- [39] Joachim Kneis, Daniel Mölle, Stefan Richter, and Peter Rossmanith. 2008. Divide-and-color. In *Proceedings of the 34th International Workshop Graph-Theoretic Concepts in Computer Science (WG'08)* Lecture Notes in Computer Science, Vol. 4271. Springer, 58–67.
- [40] Ioannis Koutis. 2008. Faster algebraic algorithms for path and packing problems. In *Proceedings of the 35th International Colloquium on Automata, Languages and Programming (ICALP'08)* Lecture Notes in Computer Science, Vol. 5125. Springer, 575–586.
- [41] Daniel Lokshtanov, Amer E. Mouawad, Saket Saurabh, and Meirav Zehavi. 2019. Packing cycles faster than Erdős-Pósa. *SIAM J. Discret. Math.* 33, 3 (2019), 1194–1215.
- [42] Meena Mahajan and Venkatesh Raman. 1999. Parameterizing above guaranteed values: MaxSat and maxcut. *J. Algorithms* 31, 2 (1999), 335–354.
- [43] Meena Mahajan, Venkatesh Raman, and Somnath Sikdar. 2009. Parameterizing above or below guaranteed values. *J. Comput. Syst. Sci.* 75, 2 (2009), 137–153.
- [44] B. Monien. 1985. How to find long paths efficiently. In *Analysis and Design of Algorithms for Combinatorial Problems (Udine, 1982)*. North-Holland Mathematics Studies, Vol. 109. North-Holland, Amsterdam, 239–254. [https://doi.org/10.1016/S0304-0208\(08\)73110-4](https://doi.org/10.1016/S0304-0208(08)73110-4)
- [45] Maria J. Serna and Dimitrios M. Thilikos. 2005. Parameterized complexity for graph layout problems. *Bull. EATCS* 86 (2005), 41–65.
- [46] Dekel Tsur. 2018. Faster deterministic parameterized algorithm for k-Path. *CoRR* abs/1808.04185 (2018). arXiv:1808.04185. <http://arxiv.org/abs/1808.04185>
- [47] Ryan Williams. 2009. Finding paths of length  $k$  in  $O^*(2^k)$  time. *Inf. Process. Lett.* 109, 6 (2009), 315–318.
- [48] Meirav Zehavi. 2015. Mixing color coding-related techniques. In *ESA 2015*, Lecture Notes in Computer Science, Vol. 9294. Springer, 1037–1049.
- [49] Meirav Zehavi. 2016. A randomized algorithm for long directed cycle. *Inf. Process. Lett.* 116, 6 (2016), 419–422. <https://doi.org/10.1016/j.ipl.2016.02.005>

Received November 2020; revised January 2021; accepted March 2021