Problems Hard for Treewidth but Easy for Stable Gonality

Hans L. Bodlaender[1(✉)], Gunther Cornelissen[2], and Marieke van der Wegen[2]

[1] Department of Information and Computing Sciences, Utrecht University, Princetonplein 5, 3584CC Utrecht, The Netherlands
h.l.bodlaender@uu.nl

[2] Department of Mathematics, Utrecht University, P.O. Box 80010, 3508 TA Utrecht, The Netherlands
{g.cornelissen,.vanderwegen}@uu.nl

Abstract. We show that some natural problems that are XNLP-hard (hence W[t]-hard for all t) when parameterized by pathwidth or treewidth, become FPT when parameterized by stable gonality, a novel graph parameter based on optimal maps from graphs to trees. The problems we consider are classical flow and orientation problems, such as Undirected Flow with Lower Bounds, Minimum Maximum Outdegree, and capacitated optimization problems such as Capacitated (Red-Blue) Dominating Set. Our hardness claims beat existing results. The FPT algorithms use a new parameter "treebreadth", associated to a weighted tree partition, as well as DP and ILP.

Keywords: Parameterized complexity · Graph algorithms · Network flow · Graph orientation · Capacitated dominating set · Tree partitions · Stable gonality

1 Introduction

The Parameterization Paradigm. Problems on finite (multi-)graphs that are NP-hard may become polynomial by restricting a specific graph parameter k. If there exists an algorithm that solves the problem in time bounded by a computable function of the parameter k times a power of the input size, we say that the problem becomes *fixed parameter tractable* (FPT) for the parameter k [15, 1.1]. Despite the fact that computing the parameter itself can often be shown to be NP-hard or NP-complete, the FPT-paradigm, originating in the work of Downey and Fellows [18], has shown to be very fruitful in both theory and practice.

One successful approach is to consider graph parameters that measure how far a given graph is from being acyclic; e.g. how the graph may be decomposed into "small" pieces, such that the interrelation of the pieces is described by a tree-like structure. A prime example of such a parameter is the *treewidth* $\mathrm{tw}(G)$ of a graph G ([15, Ch. 7]).

Other parameters have been considered (see, e.g., [21,27], [15, 7.9]), but for some famous graph orientation and graph flow problems, as well as capacitated

M. A. Bekos and M. Kaufmann (Eds.): WG 2022, LNCS 13453, pp. 84–97, 2022.
https://doi.org/10.1007/978-3-031-15914-5_7

version of classical problems, many of these parameters did not succumb to the FPT paradigm. As shown by Ganian et al. [22], the parameter "tree-cut width" of Wolan [32] is successful in dealing with several such problems. We propose a new parameter, based on *mapping* the graph to a tree, rather than decomposing the graph, that gives FPT-algorithms for a larger collection of graphs.

A Novel Parameter: Stable Gonality. The new multigraph parameter, based on "tree-likeness", is the so-called *stable gonality* $\mathrm{sgon}(G)$ of a multigraph G, introduced in [13, §3], and originating in algebraic geometry, where a similar construction has been used since the 19th century. One replaces tree *decompositions* of a graph G by graph *morphisms* from G to trees, and the "width" of the decomposition by the "degree" of the morphism, where lower degree maps correspond to less complex graphs. For example, connected graphs of stable gonality 1 are trees [7, Example 2.13], those of stable gonality 2 are so-called hyperelliptic graphs, i.e., graphs that admit, after refinement, a graph automorphism of order two such that the quotient graph is a tree (decidable in quasilinear time [7, Thm. 6.1]). The formal definition, given in Sect. 2.2, requires taking care of two technicalities, related to harmonicity of the map and refinement of the graph.

It has been shown that $\mathrm{tw}(G) \leq \mathrm{sgon}(G)$ [16, §6], that $\mathrm{sgon}(G)$ is computable, and NP-complete [24,26]. One attractive point of stable gonality as parameter for weighted problems stems from the fact that it is sensitive to multigraph properties, whereas the treewidth is not. Given an undirected weighted graph $G = (V, E, w)$ where $w\colon E \to \mathbb{Z}_{>0}$ denotes the edge weights, we have an associated (unweighted) multigraph $\tilde{G}$, with the same vertex set, but where each simple edge $e = uv$ in G is replaced by $w(e)$ parallel edges between the vertices u and v. The stable gonality of the weighted graph G is then by definition $\mathrm{sgon}(G) := \mathrm{sgon}(\tilde{G})$.

Three Sample Problems. We now introduce three problems that are exemplary for our work. We later discuss a few additional variants of these problems. Throughout, we assume that all integers are given in unary.

A typical orientation problem is the following.

Minimum Maximum Outdegree (cf. Szeider [31])
Given: Undirected weighted graph $G = (V, E, w)$ with a weight function $w\colon E \to \mathbb{Z}_{>0}$; integer r
Question: Is there an orientation of G such that for each $v \in V$, the total weight of all edges directed out of v is at most r?

A *flow network* (see, e.g., [1]) is a *directed* graph $D = (N, A)$, given with two nodes s (source) and t (target) in N, and a capacity $c(e) \in \mathbb{Z}_{>0}$ for each arc $e \in A$. Given a function $f\colon A \to \mathbb{Z}_{\geq 0}$ and a node v, we call $\sum_{wv \in A} f(wv)$ the *flow to* v and $\sum_{vw \in A} f(vw)$ the *flow out of* v. We say f is an *s-t-flow* if for each arc $a \in A$, the flow over the arc is nonnegative and at most its capacity (i.e., $0 \leq f(a) \leq c(a)$), and for each node $v \in N \backslash \{s, t\}$, the flow conservation law holds: the flow to v equals the flow out of v. The *value* $\mathrm{val}(f)$ of a flow is the flow out of s minus the flow to s.

Undirected Flow with Lower Bounds ([23, Problem ND37])[1]
Given: Undirected graph $G = (V, E)$, for each edge $e \in E$ a capacity $c(e) \in \mathbb{Z}_{>0}$ and a lower bound $\ell(e) \in \mathbb{Z}_{\geq 0}$, vertices s (source) and t (target), a value $R \in \mathbb{Z}_{>0}$
Question: Is there an orientation of G such that the resulting directed graph D allows an s-t-flow f that meets capacities and lower bounds (i.e., $\ell(a) \leq f(a) \leq c(a)$ for all arcs a in D), with value R?

Capacitated versions of classical graph problems impose a limitation on the available "resources", placing them closer to real-world situations. The following is a well-studied such graph problem, that can be viewed as an abstract form of facility location questions.

Capacitated Dominating Set
Given: Undirected graph $G = (V, E)$, for each vertex $v \in V$ a positive integer capacity $c(v) \in \mathbb{Z}_{>0}$, integer k
Question: Are there a set $D \subset V$ of size $|D| \leq k$ and a function $f : V \backslash D \to D$ such that $vf(v) \in E$ for all $v \in V \backslash D$ and $|f^{-1}(v)| \leq c(v)$ for all $v \in D$?

Main Results: Hard Problems for Treewidth but Easy for Stable Gonality. To specify the (parameterized) hardness of problems, we use the parameterized complexity class XNLP from Elberfeld et al. [20]: problems that can be solved non-deterministically in time $O(f(k)n^c)$ $(c \geq 0)$ and space $O(f(k)\log(n))$ where n is the input size, k the parameter, and f is a computable function. We note that, in terms of the more familiar W-hierarchy of Downey and Fellows [15, 13.3], XNLP-hardness implies W[t]-hardness for all t [11, Lemma 2.2].

Theorem 1. Minimum Maximum Outdegree (MMO), Undirected Flow with Lower Bounds (UFLB) *and* Capacitated Dominating Set (CDS) *are* XNLP*-complete for pathwidth, and* XNLP*-hard for treewidth* (*given a path or tree decomposition realising the path- or treewidth*)*, but are* FPT *for stable gonality* (*given a refinement and graph morphism from the associated multigraph to a tree realising the stable gonality*)*.*

Our proof that UFLB is XNLP-hard for pathwidth is by reduction from Accepting Non-deterministic Checking Counter Machine from [11]. XNLP-completeness of CDS for pathwidth was shown in [10, Thm. 8]. Hardness for the other problems follows by easy transformations from UFLB. Membership in XNLP follows each time by observing that a known dynamic programming algorithm can be transformed to a non-deterministic algorithm with bounded space. Details are given in the full paper [8]. The condition that a path decomposition realising the pathwidth is given as part of the input may be removed when an FPT algorithm is known that finds such decompositions and uses logarithmic space (see [11,19,20]).

[1] In [23] it is required that $\text{val}(f) \geq R$ rather than $\text{val}(f) = R$, but the problems are of the same complexity, cf. [28].

Itai [28, Thm. 4.1] showed that UFLB is strongly NP-complete. DOMINATING SET is $W[2]$-complete for the size of the dominating set [15, Thm. 13.28], and FPT for treewidth [15, Thm. 7.7]. CDS was shown to be W[1]-hard for treewidth (more precisely, for treedepth plus the size of the dominating set) by Dom et al. [17]. Szeider [31] showed that CDS is W[1]-hard for treewidth, which was improved to W[1]-hardness for vertex cover by Gima et al. [25].

For proving FPT under stable gonality, we revive an older idea of Seese on tree-partite graphs and their widths [30]; in contrast to the tree decompositions used in defining treewidth, we partition the original graph vertices into *disjoint* sets ('bags') labelled by vertices of a tree, such that adjacent vertices are in the same bag or in bags labelled by adjacent vertices in the tree. Seese introduced *tree partition width* to be the maximal size of a bag in such a partition. We consider weighted graphs and define a new parameter, *breadth*, given as the maximum of the bag size *and* the sum of the weights of edges between adjacent bags. The *treebreadth* of a graph G is the minimum breadth of a tree partition of G. This allows us to divide the proof in two parts: (a) show that, given a graph morphism from the associated multigraph to a tree, one can compute in polynomial time a tree partition of the weighted graph of breadth upper bounded by the stable gonality of the associated multigraph; (b) provide an FPT-algorithm, given a tree partition of bounded breadth. By reductions, the two algorithms we specify are the following, for the indicated parameters.

Theorem 2. MMO *is* FPT *for treebreadth* (*given a tree partition realising the treebreadth*)*, and* CDS *is* FPT *for tree partition width* (*given a tree partition with bounded width*)*.*

The technique to obtain Theorem 2 is similar to one used by Ganian et al. [22] who obtained FPT algorithms for a number of problems with *tree-cut width* as parameter, including CDS. Our results show that the technique from [22] can be extended to a wider class of graphs: with an upper bound on the weight of all edges, tree partition width and treebreadth are bounded by a polynomial in the tree-cut width, while stable tree partition width and tree partition width are polynomially related; see the discussion in [9, §5]. The second half of Theorem 1 is obtained from Theorem 2 by transforming the data required for sgon (a refinement with to a harmonic morphism to a tree) into a tree partition. We also prove that MMO and UFLB are W[1]-hard for vertex cover number by reduction from BIN PACKING [29]. In Sect. 3, we list some related problems for which algorithmic and hardness results hold as well. Due to space considerations, several details and all hardness proofs are omitted and can be found in the full version [8].

2 Preliminaries

2.1 Conventions and Notations

We will consider *multigraphs* $G = (V, E)$ that consist of a finite set V of vertices, as well as a finite multiset E of unoriented (unweighted) edges, i.e., a set of pairs

of (possibly equal) vertices, with finite multiplicity on each such pair. We denote such an edge between vertices $u, v \in V$ as uv. For $v \in V$, E_v denotes the edges incident with v, and for two disjoint subsets $X, Y \subset V$, $E(X, Y)$ is the collection of edges from any vertex in X to any vertex in Y. We also consider *weighted simple graphs*, where edges have positive integer weights. We will make repeated use of the correspondence between integer weighted simple graphs and multigraphs given by replacing every edge with weight k by k parallel edges. All graphs we consider are connected. (If a graph is not connected, we can solve the problem at hand separately on each connected component.) For convenience, we use the terminology "vertex" and "edge" for undirected graphs, and "arc" and "node" for either directed graphs, or for trees that occur in graph morphisms or tree partitions. We write $\mathbb{Z}$ for all integers, with unique subscripts indicating ranges (so $\mathbb{Z}_{>0}$ is the positive integers and $\mathbb{Z}_{\geq 0}$ the non-negative integers). We use interval notation for sets of integers, e.g., $[2, 5] = \{2, 3, 4, 5\}$.

2.2 Stable Gonality and Treebreadth

Stable Gonality. A *graph homomorphism* between two multigraphs G and H, denoted $\phi\colon G \to H$ consists of two (not necessarily surjective) maps $\phi\colon V(G) \to V(H)$ and $\phi\colon E(G) \to E(H)$ such that $\phi(uv) = \phi(u)\phi(v) \in E(H)$ for all $uv \in E(G)$. One would like to define the "degree" of such a graph homomorphism as the number of pre-images of any vertex or edge, but in general, this obviously depends on the chosen vertex or edge. However, by introducing certain weights on the edges via an additional index function, we get a large collection of "indexed" maps for which the degree can be defined as the sum of the indices of the pre-image of a given edge, as long as the indices satisfy a certain condition of "harmonicity" above every vertex in the target. We make this precise.

Definition 1. *A* finite morphism ϕ *between two loopless multigraphs G and H consists of a graph homomorphism $\phi\colon G \to H$ (denoted by the same letter), and an index function $r\colon E(G) \to \mathbb{Z}_{>0}$ (hidden from notation). The* index of $v \in V(G)$ *in the direction of* $e \in E(H)$*, where e is incident to $\phi(v)$, is defined by*

$$m_e(v) := \sum_{\substack{e' \in E_v, \\ \phi(e') = e}} r(e').$$

We call ϕ harmonic *if this index is independent of $e \in E(H)$ for any given vertex $v \in V(G)$. We call this simply the* index of v*, and denote it by $m(v)$. The* degree *of a finite harmonic morphism ϕ is*

$$\deg(\phi) = \sum_{\substack{e' \in E(G), \\ \phi(e') = e}} r(e') = \sum_{\substack{v' \in V(G), \\ \phi(v') = v}} m(v').$$

where $e \in E(H)$ is any edge and $v \in V(H)$ is any vertex. Since ϕ is harmonic, this number does not depend on the choice of e or v, and both expressions are indeed equal.

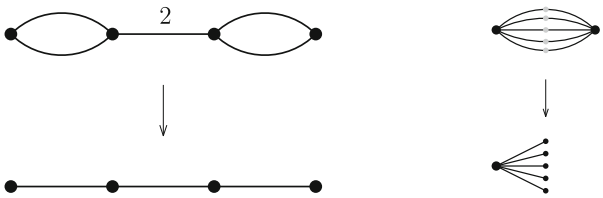

Fig. 1. Two examples of a finite harmonic morphisms of degree 2. The edges without label have index 1. The small grey vertices represent refinements of the graph. (Color figure online)

The second ingredient in defining stable gonality is the notion *refinement*.

Definition 2. *Let G be a multigraph. A* refinement *of G is a graph obtained using the following two operations iteratively finitely often: (a) add a leaf (i.e., a vertex of degree one), (b) subdivide an edge.*

Definition 3. *Let G be a multigraph. The* stable gonality *of G is the minimum degree of a finite harmonic morphism from a refinement of G to a tree.*

Two examples are found in Fig. 1. The left-hand side illustrates the need for an index function (the middle edge needs label 2), and the right hand side shows the effect of subdivision. Stable gonality is well-defined, as each graph $G = (V, E)$ has a refinement that maps to $K_{1,|E|}$: refine each edge once, map each original vertex to the center, and each refinement vertex to a unique leaf.

Tree Partitions and Their Breadth. The existence of a harmonic morphism to a tree imposes a special structure on the graph that we can exploit in designing algorithms. To capture this structure, we define the "breadth" of *tree partitions* of weighted graphs. The notion resembles that of "tree-partite graphs" from [30].

Definition 4. *A* tree partition $\mathcal{T}$ *of a weighted graph $G = (V, E, w)$ is a pair*

$$\mathcal{T} = (\{X_i \mid i \in I\},\ T = (I, F))$$

where each X_i is a (possibly empty) subset of the vertex set V and $T = (I, F)$ is a tree, such that $\{X_i \mid i \in I\}$ forms a partition of V (i.e., for each $v \in V$, there is exactly one $i \in I$ with $v \in X_i$); and adjacent vertices are in the same set X_i or in sets corresponding to adjacent nodes (i.e., for each $uv \in E$, there exists an $i \in I$ such that $\{u, v\} \subseteq X_i$ or there exists $ij \in F$ with $\{u, v\} \subseteq X_i \cup X_j$). The breadth *of a tree partition $\mathcal{T}$ of G is defined as*

$$\mathrm{b}(\mathcal{T}) := \max\left\{\max_{i \in I} |X_i|, \max_{jk \in F} w(X_j, X_k)\right\},$$

with $w(X_j, X_k) = \sum_{e \in E(X_j, X_k)} w(e)$ the total weight of the edges connecting vertices in X_j to vertices in X_k. The treebreadth $\mathrm{tb}(G)$ *of a weighted graph G is the minimum breadth of a tree partition of G.*

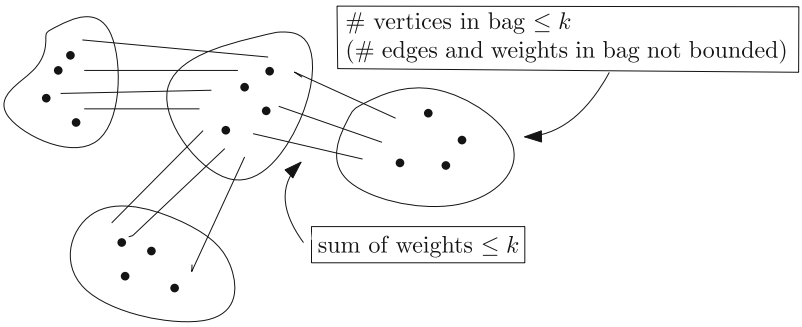

Fig. 2. Schematic representation of a tree partition of a graph of breadth $\leq k$

We refer to Fig. 2 for a schematic view of a tree partition with weights and bounded breadth. If we have a tree partition of a weighted graph G using a tree T, for convenience we will call the vertices of T *nodes* and the edges of T *arcs*. We call the sets X_i *bags*. Observe that in a tree partition of breadth k, if the total weight of edges between two vertices u and v is more than k, then u and v will be in the same bag.

Remark 1. In Seese's work [30], the structure/weights of edges between bags does not contribute to the total width; Seese's tree-partition-width $\text{tpw}(G)$ of a simple graph G, defined as the minimum over all tree partitions of G of the maximum bag size in the tree partition, is thus a lower bound for the treebreadth $\text{tb}(G)$ (in particular, for sgon, see below). For any G, $\text{tpw}(G)$ is lower bounded in terms of $\text{tw}(G)$, but also upper bounded in terms of $\text{tw}(G)$ and the maximal degree in G, cf. [33].

From Morphisms to Tree Partitions. The existence of a finite harmonic morphism ϕ of some degree k from a multigraph to a tree implies the existence of a tree partition of breadth k for the associated weighted simple graph. The basic idea is to use the pre-images of vertices in T as partitioning sets.

Theorem 3. *Suppose G is a weighted simple graph, and $\phi\colon H \to T$ is a finite harmonic morphism of degree $\deg(\phi) = k$, where H is a loopless refinement of the multigraph corresponding to G and T is a tree. Then one can construct in time $O(k \cdot |V(T)|)$ a tree partition $\mathcal{T} = (X, T')$ for a subdivision of G such that $\text{b}(\mathcal{T}) \leq k$, and $|V(T')| \leq 2|V(G)|$.*

For the proof, construct a tree partition $\mathcal{T} = (X, T)$ as follows. For every node $t \in V(T)$, define $X_t = \phi^{-1}(t) \cap V(G)$. For every edge $uv \in E(G)$, do the following. Let $i \in V(T)$ be such that $u \in X_i$ and let $j \in V(T)$ be such that $v \in X_j$. Let $i, t_1, t_2, \ldots, t_l, j$ be the path between i and j in T. Subdivide the edge uv into a path $u, s_1, s_2, \ldots, s_l, v$ and add the vertex s_r to the set X_{t_r} for each r. To get a bound on the size of T, remove all vertices t from T for which

$X_t = \emptyset$, and, for every degree 2 vertex t of T' for which X_t does not contain a vertex of $V(G)$, contract t with one of its neighbours, and contract all vertices in X_t with a neighbour as well.

Example 1. For the multigraphs in Fig. 1, the constructed tree partitions have breadth two, equal to the stable gonality: for (a), each vertex forms an individual bag, and bags are connected by edges of weight 2; for (b), there is one bag containing both non-subdivision vertices, and no edges.

Thus, to prove that a graph problem is FPT for sgon (given a morphism of a refinement of the corresponding multigraph to a tree of the correct degree), it suffices to prove that it is FPT for the breadth of a given tree-partition of a subdivision of the weighted graph.

3 Related Problems and Reductions

We consider variations of the problems MMO and UFLB.

- Circulating Orientation (CO): given an undirected weighted graph (V, E, w), is there an orientation such that for all vertices, the total weight of outgoing edges equals that of incoming edges?
- Outdegree Restricted Orientation (ORO): given an interval for each vertex, is there an orientation such that for every vertex, the total weight of outgoing edges belongs to the given interval?
- Target Outdegree Orientation (TOO): given an integer m_v for each vertex v, is there an orientation such that for every vertex v, the total weight of outgoing edges equals m_v?
- Chosen Maximum Outdegree (CMO): given an integer m_v for each vertex v, is there an orientation such that for every vertex v, the total weight of outgoing edges is at most m_v?
- All or Nothing Flow (AoNF): Given a directed graph with a positive capacity for each arc, two nodes s, t and a value R, is there a flow with value R whose value on each arc is either zero or the given capacity? (Cf. [2].)

We claim that these problems can be transformed into one another according to the diagrams in Fig. 3 preserving parameterized complexity for the indicated parameters. All complexity statements are then reduced to the following claims: (a) ORO is FPT for treebreadth; (b) AoNF is XNLP-complete for pathwidth; (c) TOO is W[1]-hard for vertex cover number.

The proof of (a) is outlined in Sect. 4.1; a full proof of (a) and the hardness proofs in (b) and (c) are given in the full version [8].

4 Algorithms for ORO and CDS for Graphs with Bounded Treebreadth

4.1 Outdegree Restricted Orientations

We give the main ideas for an algorithm for ORO, when we are given a tree partition of a subdivision of G of bounded breadth. Subdivisions can be handled

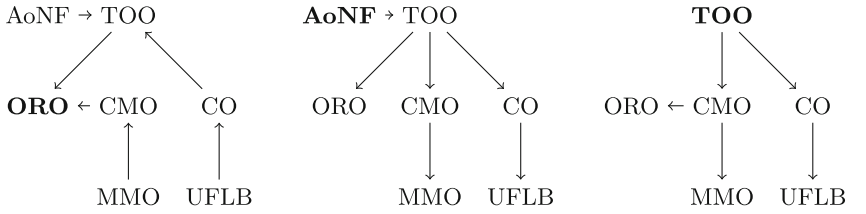

Fig. 3. Transformation between different problems with respect to parameter (a) sgon and treebreadth, for which ORO is FPT, (b) pathwidth, for which AoNF is XNLP-complete, and (c) vertex cover number, for which TOO is W[1]-hard. AonF with vertex cover number has a separate $W[1]$-hardness proof (see [8], based on [2].)

by replacing each subdivision of an edge e by a vertex x_e with $D_{x_e} = [w(e), w(e)]$. This gives an equivalent instance with a corresponding tree partition of the same breadth.

We can now assume that we have a tree partition $\mathcal{T}$ of G, i.e., adjacent vertices are in the same or neighbouring bags. Let k be the breadth of $\mathcal{T}$.

We add a new root node r to $\mathcal{T}$, and set $X_r = \emptyset$. For each node i, we let V_i be the union of all bags j with $j = i$ or j is a descendant of i. For an arc $a = ii'$ in $\mathcal{T}$ with i the parent of i', we let E_a be the set of all edges with either both endpoints in $V_{i'}$, or with one endpoint in X_i and one endpoint in $X_{i'}$.

A *partial solution* for the arc $a = ii'$ (with again i the parent of i') is an orientation of all edges in E_a such that for all vertices $v \in V_{i'}$, its total outdegree in this orientation is in D_v. Vertices in X_i can have oriented incident edges (namely to neighbours in $X_{i'}$) and incident edges that are not yet oriented (namely to neighbours in other bags than $X_{i'}$). The *fingerprint* of a partial solution is the function $\delta : X_i \to [0, k]$, that maps each vertex in X_i to its total outdegree in this partial solution, i.e., for $v \in X_i$, the sum of the weights of the edges vw with $w \in X_{i'}$ and vw is oriented from v to w. These sums are bounded by the breadth, and thus, the total number of possible fingerprints for an arc is bounded by a function of k.

The algorithm to solve ORO uses dynamic programming: for each arc in $\mathcal{T}$, we compute the table A_a of all fingerprints of partial solutions for that arc. This is done bottom-up in the tree.

It is straightforward to compute the table A_a for an arc a to a leaf of $\mathcal{T}$, by enumerating all orientations of edges in E_a.

Now, suppose we have a node i' with children $j_1, \ldots, j_q$ and we have already computed the tables $A_{i'j_1}, \ldots, A_{i'j_q}$. To compute the table $A_{ii'}$, for the arc ii' from i' to its parent i, we express in an Integer Linear Program (with number of variables bounded by a function of k), the property that we can extend an orientation of the edges between X_i and $X_{i'}$ and between pairs of vertices in $X_{i'}$ to a partial solution. Some details are given below.

Define an equivalence relation $\sim$ on the children of i', with two children equivalent if they have precisely the same set of fingerprints. As the number of

different fingerprints is a function of k, the number of equivalence classes of $\sim$ is a (double exponential) function of k.

We write Γ for the set of equivalence classes of $\sim$, and Δ for the set of all possible fingerprints of partial solutions for arcs from i' to a child.

We then enumerate all orientations ρ of the edges in $X_i \times X_{i'}$ and in $X_{i'} \times X_{i'}$. Each such orientation would fix a fingerprint for ii'—what needs to be done is checking whether there is actually a partial solution for ii' that extends ρ.

To do this, we introduce yet another concept: the *blueprint* of a partial solution for ii'. The blueprint is a function that maps a pair (γ, f) of an equivalence class $\gamma \in \Gamma$ and a fingerprint $f \in \Delta$ to the number of children j_α of i' with the following two properties: (1) the restriction of the partial solution to $E_{i'j_\alpha}$ has fingerprint f, and (2) j_α is in equivalence class γ.

Note that ρ and the blueprint contain all that is needed to compute the outdegrees of vertices in $X_{i'}$: from it, we can see, for each $v \in X_{i'}$ and for each weight in $[0, k]$, how many edges with that weight are directed from v to a vertex in a child bag of i'.

This allows us to formulate an ILP that expresses the property that there exists a blueprint of a partial solution that extends ρ. We have a non-negative integer variable $x_{\gamma,f}$ for each pair $\gamma \in \Gamma$ and $f \in \Delta$ that should give the value of this pair in the blueprint.

The ILP has no objective function, and the following constraints:

- For each γ, $\sum_f x_{\gamma,f}$ equals the number of children in equivalence class γ.
- If f is not a fingerprint for children in equivalence class γ, then $x_{\gamma,f} = 0$.
- For all $v \in X_{i'}$, we have a condition that checks that the outdegree of v in the orientation belongs to D_v. Let $D_v = [d_{\min,v}, d_{\max,v}]$. Let α be the total weight of all edges in ρ that have v as endpoint and are directed out of v. Now, add the inequalities:

$$d_{\min,v} \leq \alpha + \sum_{\gamma,f} f(v) \cdot x_{\gamma,f} \leq d_{\max,v}$$

We sum over all $\gamma \in \Gamma$, and $f \in \gamma$.

The first two conditions guarantee that we can choose for each child a fingerprint, such that for each equivalence class γ and each fingerprint f we have $x_{\gamma,f}$ children in the class γ with fingerprint f; the first condition ensures that we have the right amount of fingerprints per class, and the second that we do not assign fingerprints to children that have no corresponding partial solution in that subtree.

The third condition ensures that each vertex in $X_{i'}$ has an outdegree in its interval: we have $x_{\gamma,f}$ children in the equivalence class γ from which we take fingerprint f, and here v gets outdegree $f(v)$ for the edges between v and a vertex in such a child bag.

Note that the number of variables of the ILP is bounded by a function of k. Thus, the ILPs can be solved by an FPT algorithm, see [15, Theorem 6.4].

Once we have the table A_{a_r} for the arc a_r to the root, we can decide: the instance of ORO has a solution if and only if this table A_{a_r} is non-empty, as any partial solution for this arc is actually an orientation that fulfils the requirements for G. Thus, by processing the bags of $\mathcal{T}$ in bottom-up order, we finally obtain the table for the root and can decide the problem.

4.2 Capacitated Dominating Set

We now also sketch some main ideas for the FPT algorithm for Capacitated Dominating Set. The algorithm again uses dynamic programming, with an ILP that determines the number of children with a partial solution having a fingerprint of a certain type (compare with [22]).[2]

We define *a partial solution* for an arc ii' with i the parent of i' as a set S of vertices in $V_{i'}$ together with a mapping that maps all vertices in $V_{i'} \setminus X_{i'}$ and a subset $D \subseteq X_i \cup X_{i'}$ to a neighbour in S, such that no vertex in S has more than its capacity number of neighbours mapped to it. At this point, vertices in $X_{i'}$ do not need to be dominated yet, and they can be used to dominate vertices in the parent bag X_i. All vertices in bags that are a descendant of i' must be dominated. The *fingerprint* of a partial solution is the set D: the dominated vertices in $X_i \cup X_{i'}$.

In a dynamic programming algorithm, we compute for each arc ii' and for each fingerprint $D \subseteq X_i \cup X_{i'}$ the minimum size of a set S that gives a partial solution with this fingerprint. Let $B_{ii'}(D) \in \mathbb{Z}_{>0} \cup \{\infty\}$ be the minimum such size for a fingerprint D. Using the classical theory of matchings in graphs and inspiration from [12], we find the following.

Lemma 1. *If the instance of* CDS *has a solution, then* $B_{ii'}(\emptyset) \in \mathbb{Z}_{>0}$. *If there is a partial solution with fingerprint* $D \subseteq X_i \cup X_{i'}$, *then* $B_{ii'}(\emptyset) \leq B_{ii'}(D) \leq B_{ii'}(\emptyset) + 2k$.

In the step where we attempt to compute the table $B_{ii'}$ given such tables for the children of i', we add up all values $B_{i'j_\alpha}(\emptyset)$ and create tables $B'_{i'j_\alpha}$ by setting

$$B'_{i'j_\alpha}(D) = B_{i'j_\alpha}(D) - B_{i'j_\alpha}(\emptyset)$$

Now, $B'_{i'j_\alpha}$ is a function that maps subsets of $X_i \cup X_{i'}$ to values in $[0, 2k]$, and thus, the number of possible such functions is bounded by a function of k. This is, however, not sufficient to build an equivalence relation on the children of i', as the non-dominated vertices in such children still must be dominated by vertices in X_i. Instead, we look to extensions of partial solutions, where we also dominate vertices in $X_{i'}$ by vertices in X_i, and prescribe how much capacity each vertex in X_i uses to dominate vertices in $X_{i'}$. This gives a number of equivalence classes that is bounded by a function of k. Once we built an

[2] In the full version [8], we in fact give a detailed, slightly different, algorithm for Capacitated Red-Blue Dominating Set and then deduce the result for Capacitated Dominating Set.

equivalence relation on the children, the algorithm proceeds in a similar fashion as for ORO: an ILP is constructed that expresses for each class in the equivalence relation and fingerprint of a partial solution how many children in that class have that fingerprint. The ILP has an objective function which gives the size of the partial solution built (which is a sum of B'-values).

5 Conclusion

We showed that various classical instances of flow, orientation and capacitated graph problems are XNLP-hard when parameterized by treewidth (and even pathwidth), but FPT for a novel graph parameter, stable gonality. Following Goethe's motto "Das Schwierige leicht behandelt zu sehen, gibt uns das Anschauen des Unmöglichen", we venture into stating some open problems.

Is stable gonality fixed parameter tractable? Can multigraphs of fixed stable gonality be recognized efficiently (this holds for treewidth; for sgon $= 2$ this can be done in quasilinear time [7])? Given the stable gonality of a graph, can a refinement and morphism of that degree to a tree be constructed in reasonable time (the analogous problem for treewidth can be done in linear time)? Can we find a tree partition of a subdivision with bounded treebreadth? The same question can be asked in the approximate sense.

Find a multigraph version of Courcelle's theorem (that provides a logical characterisation of problems that are FPT for treewidth, see [14]), using stable gonality instead of treewidth: give a logical description of the class of multigraph problems that are FPT for stable gonality.

Stable gonality and (stable) treebreadth seem useful parameters for more edge-weighted or multigraph problems that are hard for treewidth. Find other problems that become FPT for such a parameter. Here, our proof technique of combining tree partitions with ILP with a bounded number of variables becomes relevant.

Conversely, find problems that are hard for treewidth and remain hard for stable gonality or (stable) treebreadth. We believe candidates to consider are in the realm of problems concerning "many" neighbours of given vertices (where our use of ILP seems to break down), such as Defensive Aliance and Secure Set, proven to be W[1]-hard for treewidth (but FPT for solution size) [5,6]. For such problems, it is also interesting to upgrade known W[1]-hardness to XNLP.

Other flavours of graph gonality (untied to stable gonality) exist, based on the theory of divisors on graphs (cf. [3,4].) Investigate whether such 'divisorial' gonality is a useful parameter for hard graph problems.

Acknowledgements. We thank Carla Groenland and Hugo Jacob for various discussions, and in particular for suggestions related to the capacitated dominating set problems, and the relations between tree cut-width, tree partition width, and stable tree-partition width.

References

1. Ahuja, R.K., Magnanti, T.L., Orlin, J.B.: Network Flows – Theory, Algorithms and Applications. Prentice Hall (1993)
2. Alexandersson, P.: NP-complete variants of some classical graph problems. CoRR, abs/2001.04120 (2020). arXiv:2001.04120
3. Baker, M.: Specialization of linear systems from curves to graphs. Algebra Number Theory **2**(6), 613–653 (2008). https://doi.org/10.2140/ant.2008.2.613. With an appendix by Brian Conrad
4. Baker, M., Norine, S.: Harmonic morphisms and hyperelliptic graphs. Int. Math. Res. Not. IMRN **15**, 2914–2955 (2009). https://doi.org/10.1093/imrn/rnp037
5. Bliem, B., Woltran, S.: Complexity of secure sets. Algorithmica **80**(10), 2909–2940 (2017). https://doi.org/10.1007/s00453-017-0358-5
6. Bliem, B., Woltran, S.: Defensive alliances in graphs of bounded treewidth. Discrete Appl. Math. **251**, 334–339 (2018). https://doi.org/10.1016/j.dam.2018.04.001
7. Bodewes, J.M., Bodlaender, H.L., Cornelissen, G., van der Wegen, M.: Recognizing hyperelliptic graphs in polynomial time. Theoret. Comput. Sci. **815**, 121–146 (2020). https://doi.org/10.1016/j.tcs.2020.02.013
8. Bodlaender, H.L., Cornelissen, G., van der Wegen, M.: Problems hard for treewidth but easy for stable gonality. CoRR, abs/2202.06838 (2022). arXiv:2202.06838
9. Bodlaender, H.L., Groenland, C., Jacob, H.: On the parameterized complexity of computing tree-partitions. CoRR, abs/2206.11832 (2022). arXiv:2206.11832
10. Bodlaender, H.L., Groenland, C., Jacob, H.: XNLP-completeness for parameterized problems on graphs with a linear structure. CoRR, abs/2201.13119 (2022). arXiv:2201.13119
11. Bodlaender, H.L., Groenland, C., Nederlof, J., Swennenhuis, C.M.F.: Parameterized problems complete for nondeterministic FPT time and logarithmic space. In: Proceedings 62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, pp. 193–204 (2021). https://doi.org/10.1109/FOCS52979.2021.00027
12. Bodlaender, H.L., van Antwerpen-de Fluiter, B.: Reduction algorithms for graphs of small treewidth. Inf. Comput. **167**(2), 86–119 (2001). https://doi.org/10.1006/inco.2000.2958
13. Cornelissen, G., Kato, F., Kool, J.: A combinatorial Li-Yau inequality and rational points on curves. Math. Ann. (10), 211–258 (2014). https://doi.org/10.1007/s00208-014-1067-x
14. Courcelle, B.: The monadic second-order logic of graphs. I. Recognizable sets of finite graphs. Inform. and Comput. **85**(1), 12–75 (1990). https://doi.org/10.1016/0890-5401(90)90043-H
15. Cygan, M., et al.: Parameterized Algorithms. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-21275-3
16. van Dobben de Bruyn, J., Gijswijt, D.: Treewidth is a lower bound on graph gonality. Algebr. Comb. **3**(4), 941–953 (2020). https://doi.org/10.5802/alco.124
17. Dom, M., Lokshtanov, D., Saurabh, S., Villanger, Y.: Capacitated domination and covering: a parameterized perspective. In: Grohe, M., Niedermeier, R. (eds.) IWPEC 2008. LNCS, vol. 5018, pp. 78–90. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-79723-4_9
18. Downey, R.G., Fellows, M.R.: Parameterized Complexity. Monographs in Computer Science, Springer, New York (1999). https://doi.org/10.1007/978-1-4612-0515-9

19. Elberfeld, M., Jakoby, A., Tantau, T.: Logspace versions of the theorems of Bodlaender and Courcelle. In: Proceedings 51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, pp. 143–152. IEEE Computer Society (2010). https://doi.org/10.1109/FOCS.2010.21
20. Elberfeld, M., Stockhusen, C., Tantau, T.: On the Space and Circuit Complexity of Parameterized Problems: classes and Completeness. Algorithmica **71**(3), 661–701 (2014). https://doi.org/10.1007/s00453-014-9944-y
21. Fiala, J., Golovach, P.A., Kratochvíl, J.: Parameterized complexity of coloring problems: treewidth versus vertex cover. Theoret. Comput. Sci. **412**(23), 2513–2523 (2011). https://doi.org/10.1016/j.tcs.2010.10.043
22. Ganian, R., Kim, E.J., Szeider, S.: Algorithmic applications of tree-cut width. In: Italiano, G.F., Pighizzini, G., Sannella, D.T. (eds.) MFCS 2015. LNCS, vol. 9235, pp. 348–360. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-48054-0_29
23. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W. H. Freeman (1979)
24. Gijswijt, D., Smit, H., van der Wegen, M.: Computing graph gonality is hard. Discret. Appl. Math. **287**, 134–149 (2020). https://doi.org/10.1016/j.dam.2020.08.013
25. Gima, T., Hanaka, T., Kiyomi, M., Kobayashi, Y., Otachi, Y.: Exploring the gap between treedepth and vertex cover through vertex integrity. Theoret. Comput. Sci. **918**, 60–76 (2022). https://doi.org/10.1016/j.tcs.2022.03.021
26. Koerkamp, R.G., van der Wegen, M.: Stable gonality is computable. Discrete Math. Theor. Comput. Sci. **21**(1), 14 (2019). https://doi.org/10.23638/DMTCS-21-1-10. Paper No. 10
27. Hliněný, P., Oum, S., Seese, D., Gottlob, G.: Width parameters beyond tree-width and their applications. Comput. J. **51**(3), 326–362 (2007). https://doi.org/10.1093/comjnl/bxm052
28. Itai, A.: Two-commodity flow. J. ACM **25**(4), 596–611 (1978). https://doi.org/10.1145/322092.322100
29. Jansen, K., Kratsch, S., Marx, D., Schlotter, I.: Bin packing with fixed number of bins revisited. J. Comput. Syst. Sci. **79**(1), 39–49 (2013). https://doi.org/10.1016/j.jcss.2012.04.004
30. Seese, D.: Tree-partite graphs and the complexity of algorithms. In: Budach, L. (ed.) FCT 1985. LNCS, vol. 199, pp. 412–421. Springer, Heidelberg (1985). https://doi.org/10.1007/BFb0028825
31. Szeider, S.: Not so easy problems for tree decomposable graphs. In: Advances in Discrete Mathematics and Applications: Mysore, 2008. Ramanujan Mathematical Society Lecture Note Series, vol. 13, pp. 179–190. Ramanujan Mathematical Society, Mysore (2010). arXiv:1107.1177
32. Wollan, P.: The structure of graphs not admitting a fixed immersion. J. Comb. Theory Ser. B **110**, 47–66 (2015). https://doi.org/10.1016/j.jctb.2014.07.003
33. Wood, D.R.: On tree-partition-width. Eur. J. Combin. **30**(5), 1245–1253 (2009). https://doi.org/10.1016/j.ejc.2008.11.010