Stable gonality is computable

Ragnar Groot Koerkamp[1]   Marieke van der Wegen[1,2]

[1] *Mathematical Institute, Utrecht University, The Netherlands*
[2] *Department of Information and Computing Sciences, Utrecht University, The Netherlands*

Stable gonality is a multigraph parameter that measures the complexity of a graph. It is defined using maps to trees. Those maps, in some sense, divide the edges equally over the edges of the tree; stable gonality asks for the map with the minimum number of edges mapped to each edge of the tree. This parameter is related to treewidth, but unlike treewidth, it distinguishes multigraphs from their underlying simple graphs. Stable gonality is relevant for problems in number theory. In this paper, we show that deciding whether the stable gonality of a given graph is at most a given integer k belongs to the class NP, and we give an algorithm that computes the stable gonality of a graph in $O((1.33n)^n m^m \text{poly}(n, m))$ time.

Keywords: algorithm, gonality, graph parameter

1 Introduction

The gonality of an algebraic curve X is the minimal degree of a non-constant morphism to the projective line $\mathbf{P}^1$. Algorithms are known that, given equations for X, compute its gonality, see for example [13]. Based on analogies between algebraic curves and graphs, various analogues of gonality have been defined in graph theory, see [2, 5, 6]. In this paper, we are concerned with the computation of the so-called stable gonality $\text{sgon}(G)$ of a multigraph G. Stable gonality is defined as the direct analogue of the above geometric definition: $\text{sgon}(G)$ is the minimal degree of a finite harmonic morphism of a refinement of G to a tree. Here, a refinement of G is given by iteratively subdividing edges or adding leaves (see 2.1-2.6 infra). Our main result says:

Theorem A. *There is an algorithm that, given a graph G, computes its stable gonality $\text{sgon}(G)$ in time $O((1.33n)^n m^m \, \text{poly}(n, m))$. Furthermore, deciding whether a graph has stable gonality at most a given integer is in NP.*

It is not immediately clear that stable gonality of a graph is computable, since its definition involves three quantifiers over infinite sets. In this paper we bound the number of refinements, trees and morphisms that we have to consider and give an algorithm to compute the stable gonality of a graph, which shows that stable gonality is computable.

There are similar notions of gonality for tropical curves and graphs, see [1, 5]. Also, on algebraic curves there is an equivalent definition of gonality, using linear systems of divisors. Another notion of gonality for graphs, divisorial gonality, has been introduced using the analogue of linear systems [3, 2].

Interestingly, divisorial gonality turns out to be different from stable gonality, as shown in [5, Example 4] and by an example of Luo in [1, Example 5.13].

Some results about the computational complexity of gonality for graphs are known. For example, both notions of gonality for graphs are NP-hard to compute [10]. Deciding whether a graph has gonality 2, on the other hand, can be done in quasilinear time for both notions of gonality [4]. An algorithm is known that computes the divisorial gonality of a graph in $O(n^k \operatorname{poly}(n, m))$ time, where n is the number of vertices, m the number of edges and k the divisorial gonality of the graph [7]. Hence computing the divisorial gonality is in XP. On the other hand, it is not known whether stable or divisorial gonality can be used for fixed parameter tractable algorithms: it would be interesting to see examples of problems that are tractable on graphs of bounded gonality (either stable or divisorial gonality), but are not tractable on graphs of bounded treewidth.

Computing stable gonality is relevant in the theory of diophantine equations. More specifically, if X is a smooth projective curve defined over a global field K with stable reduction graph G at some non-archimedean place, it is known that $\operatorname{gon}(X) \geq \operatorname{sgon}(G)$ [6, §4]. The following "uniform boundedness result" follows: X has only finitely many points in the union of all field extensions of K of degree at most $(\operatorname{sgon}(G) - 1)/2$ [6, §11].

This paper is structured as follows. In Section 2 we introduce the definition of stable gonality. We give an algorithm to compute the stable gonality of a graph in Section 3, and in Section 4, 5 and 6 we prove that this algorithm is correct. This work implies that the stable gonality problem belongs to the complexity class NP, see Section 7.

2 Preliminaries

Stable gonality is a multigraph parameter, so in this paper, we consider multigraphs; whenever we write graph, we mean finite undirected connected multigraph. A multigraph G consists of a set $V(G)$ of vertices and a multiset $E(G)$ of edges. By E_v we denote the set of edges incident to a vertex v.

In this section, we will define stable gonality as in [6, Definition 3.6], using finite harmonic morphisms.

Definition 2.1. Let G and H be loopless graphs. A *finite morphism* is a map $\phi \colon G \to H$, such that

- vertices are mapped to vertices: $\phi(v) \in V(H)$ for all $v \in V(G)$,

- edges are preserved: $\phi(e) = \phi(u)\phi(v) \in E(H)$ for all $e = uv \in E(G)$.

together with, for every edge $e \in E(G)$, an *index* $r_\phi(e) \in \mathbb{N}$.

Remark. Let $e = uv$ be an edge. Notice that e is mapped to an edge connecting the images of its endpoints: $\phi(e) = e'$ with $e' = \phi(u)\phi(v)$. Moreover, this also means that there has to be such an edge e', and more specifically it holds that $\phi(u) \neq \phi(v)$.

Definition 2.2. Let $\phi \colon G \to H$ be a finite morphism. Let $v \in V(G)$ be a vertex of G and $e \in E_{\phi(v)}$ an edge in H. The *index of v in the direction of e* is

$$\sum_{d \in E_v,\, \phi(d) = e} r_\phi(d).$$

We write $m_{\phi,e}(v)$ for this number.

We can think of these indices as follows: a vertex v has a certain weight, namely the sum of the indices of the edges incident to v. The index in the direction of an edge e incident to $\phi(v)$ indicates how much of this weight is send to the edge e. In order for a morphism to be harmonic, we want that every vertex distributes its weight equally over all edges incident to $\phi(v)$.

Definition 2.3. A finite morphism $\phi\colon G \to H$ is *harmonic* if, for all $v \in V(G)$ and for all $e, e' \in E_{\phi(v)}$,

$$\sum_{d \in E_v, \phi(d)=e} r_\phi(d) = \sum_{d' \in E_v, \phi(d')=e'} r_\phi(d').$$

In other words, ϕ is harmonic if for every vertex the index in each direction is the same. We call this number the *index* of v and denote it by $m_\phi(v)$.

A consequence of a finite morphism being harmonic is that the total weight that is mapped to each edge is equal. We call this amount the *degree* of the finite harmonic morphism.

Definition 2.4. The *degree* of a finite harmonic morphism $\phi\colon G \to H$ is

$$\deg(\phi) = \sum_{d \in \phi^{-1}(e)} r_\phi(d) = \sum_{u \in \phi^{-1}(v)} m_\phi(u),$$

for any choice of $e \in E(H)$ or $v \in V(H)$. This number is independent of the choice [3, Lemma 2.4].

Definition 2.5. Let G be a graph. A *refinement* of G is a graph H that can be obtained by applying the following operations finitely many times:

- add a new leaf, *i.e.* a vertex of degree 1;

- subdivide an edge, *i.e.* replace an edge by a vertex of degree 2.

We call a vertex of $H \backslash G$ from which there are two disjoint paths to vertices of G, *internal added vertex*, we call the other vertices of $H \backslash G$ *external added vertices*.

Definition 2.6. Let G be a graph. The *stable gonality* of G is

$$\mathrm{sgon}(G) = \min\{\deg(\phi) \mid \phi\colon H \to T \text{ a finite harmonic morphism},$$
$$H \text{ a refinement of } G, \text{ and } T \text{ a tree}\}.$$

Example 2.7. Consider the graph in Figure 1. We can map this graph to the path graph on five vertices as follows: $\phi(v_1) = p_1$, $\phi(v_2) = p_2$, $\phi(v_3) = \phi(v_4) = p_3$, $\phi(v_5) = p_4$ and $\phi(v_6) = p_5$, see Figure 1 for an illustration. Give the edge $v_5 v_6$ index 2, and all other edges index 1. This is a finite morphism.

We can check that ϕ is harmonic. Consider, for example, vertex v_5. There are two edges incident to $\phi(v_5)$, namely $p_3 p_4$ and $p_4 p_5$. We can compute

$$m_{\phi, p_3 p_4}(v_5) = \sum_{e \in E_{v_5}, \phi(e)=p_3 p_4} r_\phi(e) = r_\phi(v_3 v_5) + r_\phi(v_4 v_5) = 2,$$

$$m_{\phi, p_4 p_5}(v_5) = \sum_{e \in E_{v_5}, \phi(e)=p_4 p_5} r_\phi(e) = r_\phi(v_5 v_6) = 2.$$

We see that these sums are indeed equal, and $m_\phi(v_5) = 2$. Analogously, we can check that $m_\phi(v_1) = m_\phi(v_2) = m_\phi(v_6) = 2$ and $m_\phi(v_3) = m_\phi(v_4) = 1$.

The degree of ϕ is $\sum_{v \in V(G), \phi(v)=p_4} m_\phi(v) = m_\phi(v_5) = 2$. So we conclude that $\mathrm{sgon}(G) \leq 2$.
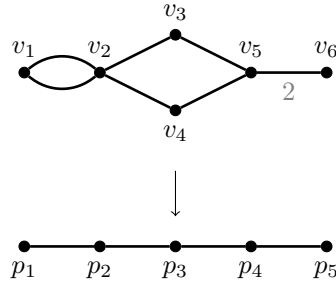
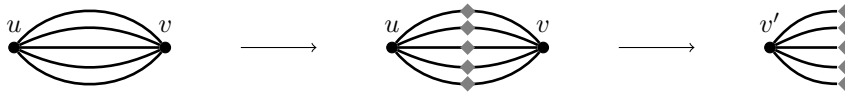**Fig. 1:** The graph G of Example 2.7 and a finite harmonic morphism of degree 2.

Fig. 2: The banana graph admits, after refining, a finite harmonic morphism of degree 2.

Example 2.8 ([6, Example 3.9]). The banana graph B_m is a graph with 2 vertices u and v and $m \geq 2$ edges, see Figure 2. Let $\phi\colon B_m \to T$ be a finite harmonic morphism to a tree T. It follows that $\phi(u) \neq \phi(v)$, otherwise the edges uv are not send to an edge $\phi(u)\phi(v)$. It follows that all m edges are mapped to the edge $\phi(u)\phi(v)$, thus $\deg(\phi) \geq m$.

By refining B_m first, we can obtain finite harmonic morphisms with lower degree. Consider the refinement G' where every edge is subdivided once. Let T be a tree with a vertex v' and m leaves, see Figure 2. Let $\phi : G' \to T$ be the map such that $\phi(u) = \phi(v) = v'$ and all other vertices are mapped to a unique leaf. Assign index 1 to every edge of G'. Now we see that ϕ is a finite harmonic morphism of degree 2.

Stable gonality is related to other graph parameters, like treewidth and the first Betti number. The first Betti number of a graph equals $m - n + 1$, where m is the number of edges and n the number of vertices. Treewidth only depends on the underlying simple graph of a multigraph, but stable gonality distinguishes multigraphs and their underlying simple graphs. The following inequalities hold for all graphs G with m edges and n vertices: $\mathrm{tw}(G) \leq \mathrm{sgon}(G) \leq \lfloor \frac{m-n+4}{2} \rfloor$ [6, 8].

Lastly, we introduce notation for a part of a graph from a given vertex v in the direction of a given vertex u.

Definition 2.9. Let G be a graph and u, v vertices. Let U be the connected component of $G - v$ containing u. By $G_v(u)$ we denote the induced subgraph on $U \cup \{v\}$. By G_{uv} we denote the graph $(G_u(v))_v(u)$.

Intuitively, G_{uv} is the part of G 'between' u and v. See Figure 3 for an example.

We say we *add $G_v(u)$ to a vertex w in a graph H*, when we add a copy of $G_v(u)$ to H and identify v with w.

We say we *refine edge uv of a graph H as $G_{u'v'}$*, when we remove uv from H, add a copy of $G_{u'v'}$ and identify u with u' and v with v'. This can be done with any graph G, but in the rest of the paper, when we refine an edge as $G_{u'v'}$, the graph G will always be a tree.
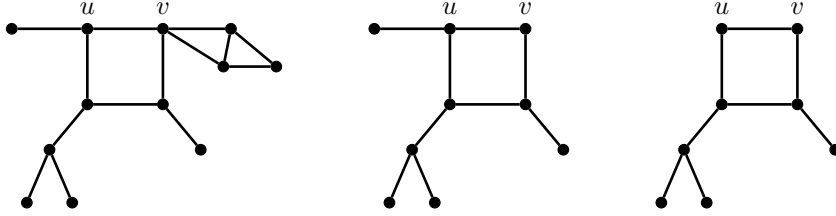
Fig. 3: A graph G, the graph $G_v(u)$ and the graph G_{uv}

3 Algorithm overview

In this paper we will give an algorithm to compute the stable gonality of a graph. Notice that this is not initially trivial: there are infinitely many refinements of a graph, there are infinitely many trees and there are infinitely many finite harmonic morphisms, since there are infinitely many assignments of indices to the edges. We bound the trees and maps that we have to consider; our algorithm enumerates all those trees and maps.

The algorithm considers all tuples $\alpha = (T, f, r)$, where

- T is a tree with at most $n = |V(G)|$ vertices,

- $f : V(G) \to V(T)$ is a surjective map,

- $r : E(G) \to [\lfloor \frac{m-n+4}{2} \rfloor]$, where we denote $[k]$ for the set $\{1, 2, 3, \ldots, k\}$, is a map assigning indices to the edges of G.

Given such a tuple, we construct a finite harmonic morphism ϕ_α from a refinement of G to a tree constructed from T by optionally adding at most m leaves. We compute the degree of ϕ_α, and output the minimum degree over all tuples α. The remainder of this section covers the construction of ϕ_α and analyses the runtime of the algorithm. The remainder of the paper proves that the assumptions made by the algorithm are valid. Sections 4 and 5 show that it suffices to only consider trees T of size at most n and indices $r_\phi(e)$ at most $\lfloor (m - n + 4)/2 \rfloor$. Section 6 proves that there exists a finite harmonic morphism of the form ϕ_α that attains the minimal degree $\mathrm{sgon}(G)$ indeed.

3.1 Construction of ϕ_α

We now explain how to construct a refinement H and a finite harmonic morphism ϕ_α from a tuple $\alpha = (T, f, r)$.

First we set $\phi(v) = f(v)$ for every vertex v of G. For each edge $uv \in E(G)$ with $f(u) = f(v)$, we add a vertex e_{uv} to uv. Besides, we add a leaf e'_{uv} to $f(u)$. We assign index 1 to those new edges and set $\phi(e_{uv}) = e'_{uv}$. This is depicted in the second column of Figure 4. Write T' for the tree we constructed from T by adding these leaves. Now, for every edge $uv \in E(G)$ with $f(u) \neq f(v)$, we refine uv as $T'_{f(u)f(v)}$. Assign index $r(uv)$ to all those new edges and use the identity map to map this part of the refinement to $T'_{f(u)f(v)}$. This is the third column in the figure. In the last column, we ensure that our map is harmonic: for every vertex $v \in V(G)$, let $e \in E_{f(v)}$ be the edge such that $m_{\phi,e}(v)$ is maximal. Now, for every edge $e' = f(v)u' \in E_{f(v)}$ with $m_{\phi,e'}(v) < m_{\phi,e}(v)$, we add $T'_{f(v)}(u')$ to v, assign index
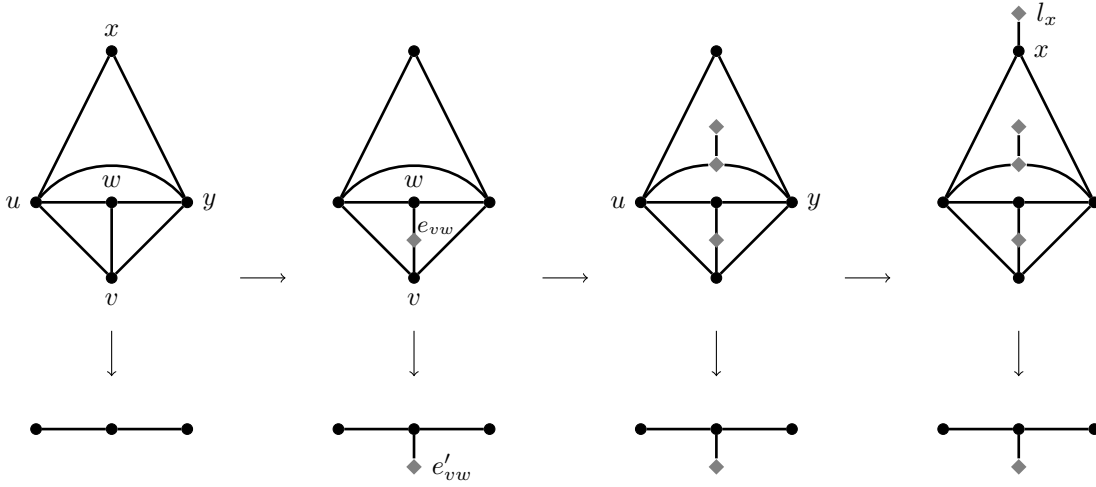
Fig. 4: Consider the map where every vertex is mapped to the vertex of the tree below it. When both ends of an edge are mapped to the same vertex, for example edge vw, we add a vertex e_{vw} and map it to a new vertex e'_{vw}. Then, we refine edges for which the ends are not mapped to the same vertex, like edge uy, as the part of the tree they correspond with. Lastly, we add copies of part of the tree, like l_x, to make sure the morphism is harmonic at every vertex. In this example, all edges have index 1.

$m_{\phi,e}(v) - m_{\phi,e'}(v)$ to these new edges and use the identity map to map this part of the refinement to $T'_{f(v)}(u')$.

Let H_α be the refinement constructed in this way, and set $\phi_\alpha = \phi$. Now ϕ_α is a finite harmonic morphism from H_α to T'.

3.2 Runtime analysis

We now analyse the runtime of the algorithm. We first count the number of pairs (T, f) where T is a tree and f is a surjective map from G to T.

By Cayley's formula, there are k^{k-2} labelled trees of size k [12, Section 2.3.4.4]. The number of unlabelled partitions of the n vertices of G into k non-empty sets is $\left\{{n \atop k}\right\}$, the Stirling number of the second kind [12, Section 1.2.6]. By assigning one label to each set, we see that the number of surjective maps to a fixed tree of size $1 \leq k \leq n$ is $\left\{{n \atop k}\right\}$ indeed. Summing over all trees of size at most n, the total number of pairs (T, f) is

$$\sum_{k=1}^{n} k^{k-2} \left\{{n \atop k}\right\}.$$

We will bound this quantity. When we first choose one vertex for each set, and then distribute the remaining $n - k$ vertices amongst those sets, we overcount the number of unlabelled partitions. Hence, $\left\{{n \atop k}\right\} \leq \binom{n}{k} k^{n-k}$. This implies

$$\sum_{k=1}^{n} k^{k-2} \left\{{n \atop k}\right\} \leq \sum_{k=1}^{n} k^{k-2} \binom{n}{k} k^{n-k} \leq \sum_{k=1}^{n} k^{n} \binom{n}{k}.$$

By Stirling's approximation we have $\binom{n}{k} \leq (\frac{n}{k})^k (\frac{n}{n-k})^{n-k}$ [12, Section 1.2.6]. We infer that

$$\sum_{k=1}^{n} k^{k-2} \left\{ {n \atop k} \right\} \leq \sum_{k=1}^{n} k^n \cdot n^n \cdot k^{-k} \cdot (n-k)^{-(n-k)} = \sum_{k=1}^{n} n^n \cdot \left(\frac{k}{n-k} \right)^{n-k}.$$

Now write x for k/n. We have

$$\left(\frac{k}{n-k} \right)^{(n-k)/n} = \left(\frac{x}{1-x} \right)^{1-x}.$$

Since this has a finite limit in both $x \to 0$ and $x \to 1$, we may consider the maximum on the $x \in (0,1)$ interval. A calculation shows that this maximum is less than 1.33. Substituting this in the bound on $\sum_{k=1}^{n} k^{k-2} \left\{ {n \atop k} \right\}$ yields

$$\sum_{k=1}^{n} k^{k-2} \left\{ {n \atop k} \right\} \leq \sum_{k=1}^{n} n^n \cdot 1.33^n \leq n \cdot (1.33n)^n \leq (1.33n)^{n+1}.$$

The number of functions $r : E(G) \to \left[\left\lfloor \frac{m-n+4}{2} \right\rfloor \right]$ assigning indices to the edges is $\left\lfloor \frac{m-n+4}{2} \right\rfloor^m$.

When we are given a tuple $\alpha = (T, f, r)$, the construction of H and ϕ_α can be done in polynomial time in n and m: we have to consider every edge once to refine it and we have to consider every vertex once to make the map harmonic. The calculation of the degree can also be done in polynomial time, by picking an edge e in the tree, and check for every edge of H whether it is mapped to e. It follows that the runtime of our algorithm is bounded by

$$O \left(\mathrm{poly}(n, m) \cdot (1.33n)^n \left(\frac{m-n+4}{2} \right)^m \right).$$

Remark. In practice, it can be useful to do some pre-processing first, although this does not change the runtime in general. Before trying all tuples α, we can contract all vertices of degree one or two. The graph obtained in this way is called a *stable graph*. It is known that this stable graph has the same stable gonality as the graph we started with [6, Lemma 5.4].

Remark. A C++ implementation of this algorithm is made. Unfortunately, this algorithm is too time consuming to use for graphs with more than 5 vertices. The implementation is available from the first author upon request.

4   Bounding the size of the tree

In this section we will show that we only have to consider finite harmonic morphisms to trees with at most $|V(G)|$ internal (*i.e.* non leaf) vertices. In particular, we show that any finite harmonic morphism can be transformed in such a way that every internal vertex of T is covered by at least one vertex of G.

Definition 4.1. Let a graph G, a refinement H of G, a tree T, and a finite harmonic morphism $\phi \colon H \to T$ be given. A *transformation* of ϕ is a new finite harmonic morphism $\phi' \colon H' \to T'$ where H' is again a refinement of G and T' a tree, such that $\deg(\phi') \leq \deg(\phi)$.
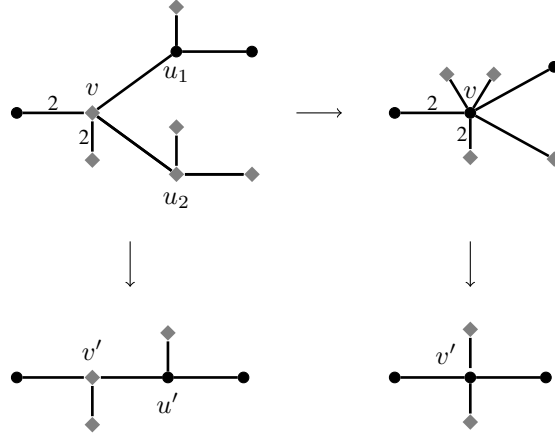
Fig. 5: In the first case of the proof of Lemma 4.4, we contract vertex v to all its neighbours u_j mapping to u'.

Remark. From here on H and H' will always be refinements of G while T and T' will always be trees. The phrase *let $\phi\colon H \to T$ be given* will implicitly assume this.

Definition 4.2. Let G be a graph with a refinement H. A vertex $v \in V(H)$ is *alien* if it is not a vertex in G. Let $\phi\colon H \to T$ be a finite harmonic morphism. A vertex $v' \in V(T)$ is *alien* if all vertices in $\phi^{-1}(v')$ are alien.

The main lemma of this section is that we can remove all internal alien vertices from T.

Lemma 4.3. *Let $\phi\colon H \to T$ be given. There exists a transformation $\phi'\colon H' \to T'$ such that T' has no internal alien vertices.*

To prove this we contract all internal alien vertices in T to a neighbour, while simultaneously contracting all edges in the preimages of these edges. By G/e we denote the graph obtained by contracting edge e in graph G.

Lemma 4.4. *Let $\phi\colon H \to T$ be given, and let v' be an internal alien vertex of T with a neighbour u'. There exists a transformation $\phi'\colon H' \to T/u'v'$.*

Proof: Let $v_1, \ldots, v_k$ be the vertices of H that are mapped to v' and let $u_1, \ldots, u_l$ be the vertices that are mapped to u'. We will construct a refinement H' of G as follows. Consider a vertex $v := v_i$. Notice that v is an alien vertex of H, so there are at most two edges vw_1 and vw_2 such that $H_v(w_1)$ and $H_v(w_2)$ contain vertices of G.

$\underline{\text{Case 1:}}$ If there is at most one vertex u_j neighbouring v such that $H_v(u_j)$ contains vertices of G, then contract all edges vu_h, and keep all indices. This process is shown in Figure 5.

$\underline{\text{Case 2:}}$ Now suppose that there are two vertices u_{j_1} and u_{j_2} that are neighbours of v such that $H_v(u_{j_1})$ and $H_v(u_{j_2})$ contain vertices of G. Since v' is internal, there is a neighbour x' of v' not equal to u'. Contract v to all its neighbours that map to x'. Then, for each neighbour y of v that maps to a vertex different from x' and u', remove the tree $G_v(y)$ from H and add a copy of $T_{v'}(\phi(y))$ to each of the u_j. We assign index $r_\phi(vu_j)$ to the trees that we just added to u_j, while all other edges keep their current index. This is demonstrated in Figure 6.
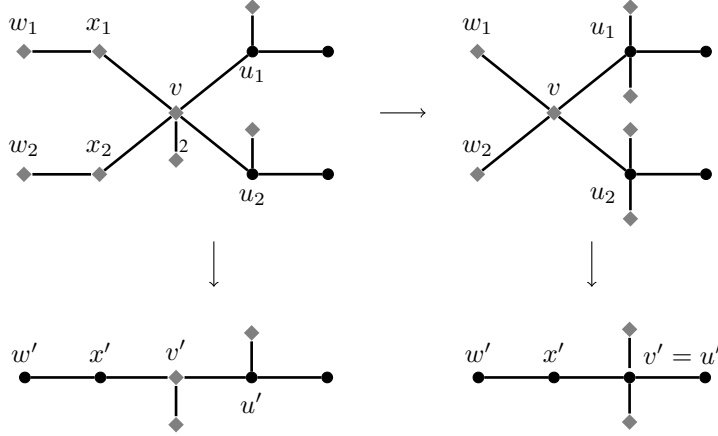
Fig. 6: In the second case of the proof of Lemma 4.4, we contract vertex v to all its neighbours x_i mapping to $x' \neq u'$. We also remove all other trees from v, and copy them to all u_j.

Repeat this for all vertices v_i. Write H' for the resulting graph.

In the construction of H', we never contract two vertices of G to the same vertex, so H' is a refinement of G. Now consider the map $\phi' \colon H' \to T/u'v'$ we constructed. This map is a finite harmonic morphism with degree at most the degree of ϕ. □

Proof Proof of Lemma 4.3: Repeatedly apply Lemma 4.4. □

It follows that, for any given morphism $\phi \colon H \to T$, there is a transformation $\phi' \colon H' \to T'$ such that $\phi'(V(G))$ is a connected subtree of T with at most n vertices. So, for our algorithm it suffices to only consider trees of size at most n and surjective maps to those trees.

5 Bounding the indices

Cornelissen et al. [6, Theorem 5.7] gave an upper bound on the stable gonality of a graph G:

$$\mathrm{sgon}(G) \leq \left\lfloor \frac{m - n + 4}{2} \right\rfloor .$$

From this it follows that all finite harmonic morphisms $\phi \colon H \to T$, from a refinement H of G to a tree T with $\deg(\phi) = \mathrm{sgon}(G)$, assign index at most $\lfloor \frac{m-n+4}{2} \rfloor$ to the edges of H. Hence it is sufficient for our algorithm to only consider functions $r \colon E(G) \to [\lfloor \frac{m-n+4}{2} \rfloor]$.

6 Reduction to ϕ_α

In this section, we will prove that our algorithm will find a finite harmonic morphism of minimal degree. Let G be a graph. We know that there exists a refinement H of G, a tree T and a finite harmonic morphism ϕ of degree $\mathrm{sgon}(G)$. We will show that we can transform ϕ to a morphism ϕ_α for some tuple α. In all

lemmas we assume G to be the given graph, H a refinement, T a tree and $\phi\colon H \to T$ a finite harmonic morphism of degree $\mathrm{sgon}(G)$.

First, we will prove that, when two vertices u, v are mapped to the same vertex, we can refine the edges uv by adding just one vertex. For ease of notation we give the following two graphs names: we write P_3 for the path on three vertices and C_2 for the cycle of length two.

Lemma 6.1. *Let $\phi\colon H \to T$ be given. Let $uv \in E(G)$ be an edge such that $\phi(u) = \phi(v)$. There exists a transformation $\phi'\colon H' \to T'$ such that $R'_{uv} \in \{P_3, C_2\}$, where R'_{uv} is the refinement of uv in H'.*

Proof: Let R_{uv} be the refinement of the edge uv in H. Let x be the neighbour of u in R_{uv} and let y be the neighbour of v. Replace R_{uv} by P_3 when $u \neq v$, and by C_2 when $u = v$. Write w for the new vertex. Assign index 1 to the two new edges. If $m_\phi(u) > 1$, add a leaf w_1 to u and assign index $m_\phi(u) - 1$ to it. Do the same for v. For all vertices $z \in \phi^{-1}(\phi(v))\backslash R_{uv}$, add a leaf w_z to z and assign index $m_\phi(z)$ to the edge zw_z. Add a copy of $T_{\phi(u)}(\phi(x))$ to u and assign index $r_\phi(ux)$ to all edges in this tree. Add a copy of $T_{\phi(v)}(\phi(y))$ to v and assign index $r_\phi(vy)$ to the edges. Write H' for this refinement of G. Add a vertex w' to $\phi(v)$ in T and write T' for this new tree. Let $\phi'\colon H' \to T'$ be the map that sends all vertices of $z \in V(H)$ to $\phi(z)$, that sends w, w_1, w_2 and all vertices w_z to w' and uses the identity map to send the vertices of the trees we added. Notice that ϕ' is a finite harmonic morphism with $\deg(\phi') \leq \deg(\phi)$. $\qquad\square$

Second, we will prove that for two vertices u, v that are not mapped to the same vertex, we can refine the edges uv as $T_{\phi(u)\phi(v)}$.

Lemma 6.2. *Let $\phi\colon H \to T$ be given. Let $uv \in E(G)$ be an edge such that $\phi(u) \neq \phi(v)$. There exists a transformation $\phi'\colon H' \to T$ such that $R'_{uv} = T_{\phi(u)\phi(v)}$, where R'_{uv} is the refinement of uv in H'.*

Proof: Let R_{uv} be the refinement of uv in H. Let x be the neighbour of u in R_{uv} and let y be the neighbour of v. We distinguish three cases.

First suppose that $\phi(x) \in T_{\phi(u)\phi(v)}$ and $\phi(y) \in T_{\phi(u)\phi(v)}$. Replace R_{uv} by $T_{\phi(u)\phi(v)}$ in H. Assign index $\min\{r_\phi(ux), r_\phi(vy)\}$ to the new edges. Assume, without loss of generality, that $r_\phi(ux) \leq r_\phi(vy)$. If $r_\phi(ux) < r_\phi(vy)$, add a copy of the tree $T_{\phi(u)}(\phi(x))$ to u and assign index $r_\phi(vy) - r_\phi(ux)$ to the new edges. Write H' for this new refinement of G. Let $\phi'\colon H' \to T$ be the map that sends all vertices of $w \in V(H)$ to $\phi(w)$ and uses the identity map to send the new vertices to $T_{\phi(u)\phi(v)}$ and $T_{\phi(u)}(\phi(x))$. Notice that ϕ' is a finite harmonic morphism with $\deg(\phi') \leq \deg(\phi)$.

Now suppose that $\phi(x) \notin T_{\phi(u)\phi(v)}$ and $\phi(y) \notin T_{\phi(u)\phi(v)}$. Replace R_{uv} by $T_{\phi(u)\phi(v)}$ in H. Assign index 1 to the new edges. Add a copy of $T_{\phi(u)}(\phi(x))$ to u and assign index $r_\phi(ux) + 1$ to its edges. For all neighbours z of $\phi(u)$ such that $z \neq \phi(x)$ and $z \notin T_{\phi(u)\phi(v)}$, add a copy of $T_{\phi(u)}(z)$ to u and assign index 1 to the new edges. Do the same for vertex v. Write H' for this refinement of G. Let $\phi'\colon H' \to T$ be the map that sends all vertices $w \in V(H)$ to $\phi(w)$ and uses the identity map for all new vertices. Notice that by the choice of the indices ϕ' is a finite harmonic morphism with $\deg(\phi') \leq \deg(\phi)$.

In the last case, suppose, without loss of generality, that $\phi(x) \in T_{\phi(u)\phi(v)}$ and $\phi(y) \notin T_{\phi(u)\phi(v)}$. Replace R_{uv} by $T_{\phi(u)\phi(v)}$ in H. Assign index 1 to the new edges. Add a copy of $T_{\phi(v)}(\phi(y))$ to v and assign index $r_\phi(vy) + 1$ to its edges. For all neighbours z of $\phi(v)$ such that $z \neq \phi(y)$ and $z \notin T_{\phi(u)\phi(v)}$, add a copy of $T_{\phi(v)}(z)$ to v and assign index 1 to the new edges. If $r_\phi(ux) > 1$, add a copy of $T_{\phi(u)}(\phi(x))$ to u and assign index $r_\phi(ux) - 1$ to its edges. Write H' for this refinement of G. Let $\phi'\colon H' \to T$ be the map that sends all vertices $w \in V(H)$ to $\phi(w)$ and uses the identity map for all new vertices. Notice that by the choice of the indices ϕ' is a finite harmonic morphism with $\deg(\phi') \leq \deg(\phi)$. $\qquad\square$

Lastly, we prove two lemmas that ensure that there are not more external added vertices than necessary.

Lemma 6.3. *Let $\phi\colon H \to T$ be given. Let $v' \in V(T)$ be a leaf such that all vertices $v \in V(H)$ with $\phi(v) = v'$ are external added vertices. Define $T' = T\backslash\{v'\}$. There exists a transformation $\phi'\colon H' \to T'$.*

Proof: Remove all vertices that are mapped to v' from H. If H becomes disconnected, remove all connected components that do not contain a vertex of G. Since all removed vertices were external added vertices, there is only one remaining component. Write H' for this graph. Define $\phi'\colon H' \to T'$ as the restriction of ϕ to H'. Notice that ϕ' is a finite harmonic morphism, and $\deg(\phi') \leq \deg(\phi)$. □

Lemma 6.4. *Let $\phi\colon H \to T$ be given. Let v be a vertex of G and u' a neighbour of $\phi(v)$. There exists a transformation $\phi'\colon H' \to T$ such that v has at most one external added neighbour u such that $\phi'(u) = u'$, and moreover, $G_v(u) = T_{\phi(v)}(u')$.*

Proof: Let $u_1, \ldots, u_l \in$ be the external added neighbours of v such that $\phi(u_i) = u'$. Remove all trees $G_v(u_i)$ from G and add one copy of $T_{\phi(v)}(u')$ to v. Assign index $\sum_{i=1}^{l} r_\phi(vu_i)$ to all new edges. Notice that ϕ' is a finite harmonic morphism, and $\deg(\phi') \leq \deg(\phi)$. □

Lemma 6.5. *Let G be a graph with $\mathrm{sgon}(G) = k$. Then there is a refinement H of G, a tree T and a finite harmonic morphism $\phi\colon H \to T$ of degree k, that satisfy the following properties:*

- *T does not contain internal alien vertices.*

- *For every edge $uv \in E(G)$ such that $\phi(u) = \phi(v)$, it holds that $R_{uv} \in \{P_3, C_2\}$, where R_{uv} is the refinement of uv in H; moreover, the two edges of R_{uv} have index 1.*

- *For every edge $uv \in E(G)$ such that $\phi(u) \neq \phi(v)$, it holds that $R_{uv} = T_{\phi(u)\phi(v)}$, where R_{uv} is the refinement of uv in H; moreover, every edge in R_{uv} is assigned the same index.*

- *For every vertex $v' \in V(T)$, let $v_1, \ldots, v_l$ be the vertices of H such that $\phi(v_i) = v'$, then, not all of $v_1, \ldots, v_l$ are external added vertices.*

- *For every vertex $v \in V(G)$ and for every neighbour u' of $\phi(v)$, v has at most one external added neighbour u such that $\phi'(u) = u'$, and moreover, $G_v(u) = T_{\phi(v)}(u')$.*

Proof: This follows from Lemmas 4.3, 6.1, 6.2, 6.3, and 6.4. □

Now we are ready to prove that our algorithm will find a morphism of minimal degree.

Theorem 6.6. *Let G be a graph with $\mathrm{sgon}(G) = k$. There is a tuple α such that ϕ_α has degree k.*

Proof: By Lemma 6.5 we know that there is a refinement H of G, a tree T and a finite harmonic morphism $\phi\colon H \to T$ of degree k which satisfy the properties in Lemma 6.5. By Section 5, we know that all indices are at most $\lfloor \frac{m-n+4}{2} \rfloor$.

Define T' as the subtree of T that consists of the vertices $\phi(V(G))$. Notice that by Section 4, T' is connected and has at most n vertices. Define $f\colon G \to T'$ as ϕ restricted to $V(G)$. Now let $r(uv)$ be the index that is assigned to every edge in R_{uv}. Let α be the tuple (T', f, r).

By the properties of ϕ and the construction of ϕ_α, it follows that $\phi_\alpha = \phi$. Thus ϕ_α has degree $\mathrm{sgon}(G)$. □

7 Stable gonality is in NP

In this section we consider the decision problem "*stable gonality problem*": Given a graph G and an integer k, does it hold that $\operatorname{sgon}(G) \le k$?

Theorem 7.1. *The stable gonality problem belongs to the class NP.*

Proof: Let (G, k) be an instance of the stable gonality problem. By Theorem 6.6 we know that there is a tuple $\alpha = (T, f, r)$ as in Section 3 such that ϕ_α has degree $\operatorname{sgon}(G)$. This tuple has polynomial size. Given a tuple α, we can construct ϕ_α in polynomial time, and we can compute its degree in polynomial time.

So for a yes-instance (G, k), there is a tuple $\alpha = (T, f, r)$ with polynomial size and $\deg(\phi_\alpha) \le k$, and we can check in polynomial time whether a tuple is a certificate for (G, k). For a no-instance no such tuple exists. $\qquad\square$

8 NP-hard subproblem

As described in Section 3, for a given graph G our algorithm considers all tuples $\alpha = (T, f, r)$. The tree T' that is constructed from this tuple and the vertices that will be internally added to G do not depend on r. The question arises whether we can construct an r such that $\phi_{(T,f,r)}$ has minimal degree when we are already given the pair (T, f). This problem turns out to be NP-hard.

For the proof we will use a reduction from the three-dimensional matching problem: Let A, B, C be finite, disjoint sets, let $S \subseteq A \times B \times C$ and let k be a natural number. Does there exist a set $M \subseteq S$ with $|M| \ge k$ such that every element of $A \cup B \cup C$ is contained in at most one tuple in M? We call such a set M a *matching*. This problem is known to be NP-hard, even when restricted to cases where $|A| = |B| = |C| = k$ and for every element of $A \cup B \cup C$ there are at least two tuple in S containing this element [11].

Theorem 8.1. *Given a graph G, a pair (T, f) and an integer k. The following problem is NP-hard: Does there exist a function $r : E(G) \to [\lfloor (m - n + 4)/2 \rfloor]$ such that the morphism $\phi_{(T,f,r)}$ has degree at most k?*

Proof: Let (A_1, A_2, A_3, S, k) be an instance of the three-dimensional matching problem, where $|A_1| = |A_2| = |A_3| = k$ and for every element of $A_1 \cup A_2 \cup A_3$ there are at least two tuples in S containing this element. Define $A = A_1 \cup A_2 \cup A_3$.

Construct the following graph G. Add two vertices u_a and v_a for every element $a \in A$ and add a vertex w_s for every $s \in S$. Add an edge between u_a and v_a for every $a \in A$. For every tuple $s = \{x, y, z\} \in S$, we add edges $v_x w_s$, $v_y w_s$, $v_z w_s$ and edges $u_x v_x$, $u_y v_y$, and $u_z v_z$. See Figure 7 for an illustration.

Let T be the tree that consists of vertex w, vertices u_i, v_i for $1 \le i \le 3$, and edges $u_i v_i$ and $v_i w$ for $1 \le i \le 3$. Define $f : V(G) \to V(T)$ as follows:

$$
f(x) = \begin{cases}
w & \text{if } x = w_s \text{ for } s \in S, \\
v_i & \text{if } x = v_a \text{ with } a \in A_i, \\
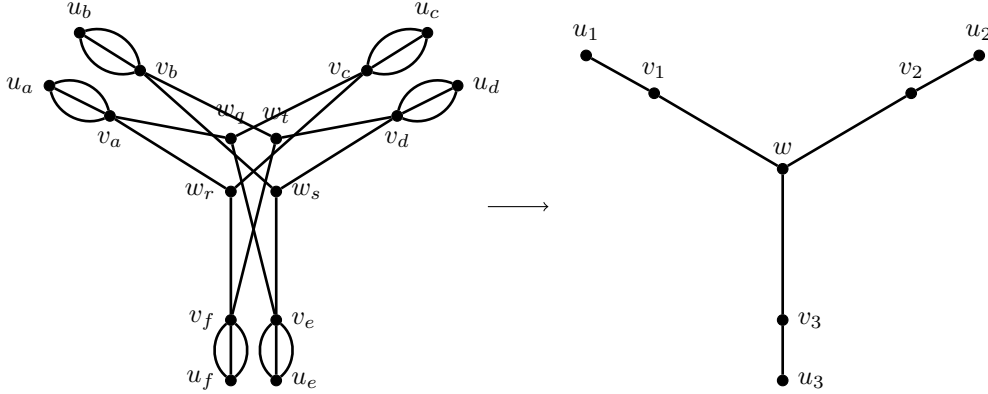u_i & \text{if } x = u_a \text{ with } a \in A_i.
\end{cases}
$$

Fig. 7: An example of the graph G and tree T constructed in the proof of Theorem 8.1 for the sets $A_1 = \{a, b\}$, $A_2 = \{c, d\}$, $A_3 = \{e, f\}$ and $S = \{(a, c, e), (a, c, f), (b, d, e), (b, d, f)\}$.

Now $(G, T, f, |S| + k)$ is an instance of our problem. Notice that the degree of $\phi_{(T,f,r)}$ will be at least $|S| + k$ for any r, since at least $|S| + k$ edges map to each edge $u_i v_i$. We will now prove that there is an r such that $\phi_{(T,f,r)}$ has degree $|S| + k$ if and only if there is a matching $M \subseteq S$ with $|M| \geq k$.

Given a perfect matching $M \subseteq S$, that is, a matching with $|M| = k$, we can find a ϕ of degree $|S| + k$ by putting index 2 on all edges $w_m v_a$ with $m \in M$ and $a \in m$.

Now suppose that ϕ has degree $|S| + k$. This implies that each edge $u_a v_a$ has index 1 and that there are no externally added vertices mapped u_i for all i. Given a vertex v_a. Either one of the edges $v_a w_s$ has index 2, or v_a has an external added neighbour that is mapped to w. In the last case, there is an external added vertex mapped to one of the vertices u_i, which yields a contradiction. So, exactly one of the edges $v_a w_s$ must have index 2; the others will have index 1. At each vertex w_s, all three edges will have the same index. This index must be either 1 or 2, since otherwise there will be an external added vertex that is mapped to a vertex u_i. This implies that there are exactly k vertices $s \in S$ with index 2, while all other vertices s have index 1. These k tuples (a_1, a_2, a_3) will form a matching, and hence we can solve the three-dimensional matching problem using the problem of finding an optimal r. $\qquad\square$

9 Conclusion

Stable gonality is defined using three infinite loops: there are infinitely many refinements of a graph, there are infinitely many trees, and there are infinitely many finite harmonic morphisms from a refinement to a tree. In this paper we bounded the refinements, trees and morphism which we have to consider. This yields an algorithm to compute the stable gonality of a graph in $O\big(\text{poly}(n, m) \cdot (1.33n)^n \big(\frac{m-n+4}{2}\big)^m\big)$ time. From these bounds and the algorithm it also follows that the stable gonality problem is in NP.

Some interesting questions remain open. Firstly: is there a faster algorithm to compute the stable gonality of a graph? Secondly, we do not know whether computing stable gonality is in XP or FPT, or whether it is W[1]-hard. Thirdly, are there problems which are untractable with treewidth as parameter, that are tractable with stable gonality as parameter? Lastly, is there a variant of Courcelle's theorem [9, Chapter 13] for graphs of bounded stable gonality?

References

[1] Omid Amini, Matthew Baker, Erwan Brugallé, and Joseph Rabinoff. Lifting harmonic morphisms II: Tropical curves and metrized complexes. *Algebra & Number Theory*, 9(2):267–315, 2015. `doi:10.2140/ant.2015.9.267`.

[2] Matthew Baker. Specialization of linear systems from curves to graphs. *Algebra & Number Theory*, 2(6):613–653, 2008. With an appendix by Brian Conrad. `doi:10.2140/ant.2008.2.613`.

[3] Matthew Baker and Serguei Norine. Riemann–Roch and Abel–Jacobi theory on a finite graph. *Advances in Mathematics*, 215(2):766 – 788, 2007. `doi:10.1016/j.aim.2007.04.012`.

[4] Jelco M. Bodewes, Hans L. Bodlaender, Gunther Cornelissen, and Marieke van der Wegen. Recognizing hyperelliptic graphs in polynomial time. In Andreas Brandstädt, Ekkehard Köhler, and Klaus Meer, editors, *Graph-Theoretic Concepts in Computer Science*, pages 52–64, 2018. (extended abstract of arXiv:1706.05670).

[5] Lucia Caporaso. Gonality of algebraic curves and graphs. In *Algebraic and Complex Geometry, In Honour of Klaus Hulek's 60th Birthday*, volume 71 of *Springer Proceedings in Mathematics & Statistics*, pages 77–108. Springer, 2014. `doi:10.1007/978-3-319-05404-9`.

[6] Gunther Cornelissen, Fumiharu Kato, and Janne Kool. A combinatorial Li–Yau inequality and rational points on curves. *Mathematische Annalen*, 361(1):211–258, 2015. `doi:10.1007/s00208-014-1067-x`.

[7] Josse van Dobben de Bruyn. Reduced divisors and gonality in finite graphs. Bachelor's thesis, Leiden University, 2012. URL: `https://www.universiteitleiden.nl/binaries/content/assets/science/mi/scripties/bachvandobbendebruyn.pdf`.

[8] Josse van Dobben de Bruyn and Dion Gijswijt. Treewidth is a lower bound on graph gonality. Preprint, arXiv:1407.7055v2, 2014.

[9] Rodney G. Downey and Michael R. Fellows. *Fundamentals of Parameterized Complexity*. Springer, 2013. `doi:10.1007/978-1-4471-5559-1_13`.

[10] Dion Gijswijt, Harry Smit, and Marieke van der Wegen. Computing graph gonality is hard. To appear (Extended version of arXiv:1504.06713), 2018.

[11] Richard M. Karp. Reducibility among combinatorial problems. In Raymond E. Miller, James W. Thatcher, and Jean D. Bohlinger, editors, *Complexity of Computer Computations*, pages 85–103. Springer, 1972. `doi:10.1007/978-1-4684-2001-2_9`.

[12] Donald E. Knuth. *The Art of Computer Programming, Volume 1: Fundamental Algorithms*. Addison-Wesley Publishing Company, 1968.

[13] J. Schicho, F.-O. Schreyer, and M. Weimann. Computational aspects of gonal maps and radical parametrization of curves. *Applicable Algebra in Engineering, Communication and Computing*, 24(5):313–341, 2013. `doi:10.1007/s00200-013-0205-0`.