Stable gonality is computable

Ragnar Groot Koerkamp — **Mathematical Institute, Utrecht University, PO Box 80.010, 3508 TA Utrecht, The Netherlands,** `ragnar.grootkoerkamp@gmail.com` **(now at Google)**
Marieke van der Wegen — **Department of Information and Computing Sciences, Utrecht University, PO Box 80.089, 3508 TB Utrecht, The Netherlands,**
`m.vanderwegen@uu.nl`

Abstract

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 give an algorithm that computes the stable gonality of a graph in $O((1.33n)^n m^m \operatorname{poly}(n,m))$ time.

1 Introduction

The gonality of an algebraic curve X is the minimal degree of a morphism to the projective line $\mathbf{P}^1$. Algorithms are known that, given equations for X, compute the gonality (using Gröbner basis algorithms, e.g. [7]). Various analogues of gonality have been defined in graph theory, see [1, 3, 4]. In this paper, we are concerned with the computation of the so-called stable gonality $\operatorname{sgon}(G)$ of a multigraph G, which is defined as the direct analogue of the above geometric definition: $\operatorname{sgon}(G)$ is the minimal degree of a finite harmonic morphism of a refinement of G to a tree, where a refinement of G is given by iteratively subdividing edges or adding leaves (cf. 1-6 infra). Our main result says:

Theorem 1. *There is an algorithm that, given a graph G, computes its stable gonality* $\operatorname{sgon}(G)$ *in time $O((1.33n)^n m^m \operatorname{poly}(n,m))$. 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 in general, 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.

Computing stable gonality is relevant in the theory of diophantine equations. More specifically, if X is a smooth projective curve defined over a number field K with stable reduction graph G at some non-archimedean place, it is known that $\operatorname{gon}(X) \geq \operatorname{sgon}(G)$ [4, §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 $(\operatorname{sgon}(G) - 1)/2$ [4, §11].

2 Preliminaries

By 'graph' we mean finite undirected connected multigraph. In this section, we will define stable gonality as in [4, Definition 3.6], using finite harmonic morphisms.

Definition 1. Let G and H be loopless graphs. A *finite morphism* is a map $\phi\colon G \to H$, such that

- $\phi(v) \in V(H)$ for all $v \in V(G)$,
- $\phi(e) = \phi(u)\phi(v) \in E(H)$ for all $e = uv \in E(G)$,

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

Definition 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*, denoted by $m_{\phi,e}(v)$ is

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

In order for a morphism ϕ to be harmonic, we want that for every vertex v its incident edges are distributed equally (counted with their index) over the edges incident to $\phi(v)$:

Definition 3. A finite morphism $\phi\colon G \to H$ is *harmonic* if, for all $v \in V(G)$ and for all $e, e' \in E_{\phi(v)}$,

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

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 number of edges (counted with their index) that are mapped to each edge is equal. We call this amount the *degree* of the finite harmonic morphism.

Definition 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 [2, Lemma 2.4].

Definition 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 by adding a vertex of degree 2.

Definition 6. Let G be a graph. The *stable gonality* of G is

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

In Figure 1(d) we see an example of a refinement of the graph in Figure 1(a) and a finite harmonic morphism of degree 4.

The stable gonality of a graph G with n vertices and m edges is related to other graph parameters like treewidth [5, Section 5] and the first Betti number $g := m - n + 1$ [4, Theorem 5.7] by the inequalities $\operatorname{tw}(G) \leq \operatorname{sgon}(G) \leq \lfloor \frac{g+3}{2} \rfloor$. Whereas treewidth only depends on the underlying simple graph of a multigraph, stable gonality distinguishes multigraphs and their underlying simple graphs.

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

Definition 7. 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 $(G_u(v))_v(u)$.

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* G *as* $T_{u'v'}$, when we remove uv from G, add a copy of $T_{u'v'}$ and identify u with u' and v with v'.

3 Algorithm overview

In this paper we will give an algorithm to compute the stable gonality of a graph. Notice that the computability of stable gonality is not immediate from the definition: 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, after which 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,
- $r : E(G) \to \{1, 2, \ldots, \lfloor \frac{m-n+4}{2} \rfloor\}$ 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.

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 1. 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 $m_{\phi,e}(v) - m_{\phi,e'}(v)$ 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'.

4 Correctness

To show that our algorithm is correct, we need to prove that our algorithm will find a finite harmonic morphism of minimal degree. For this we use several lemmas of the form: 'There exists a finite harmonic morphism $\phi\colon H \to T$ for a refinement H of G and a tree T such that ϕ has a certain

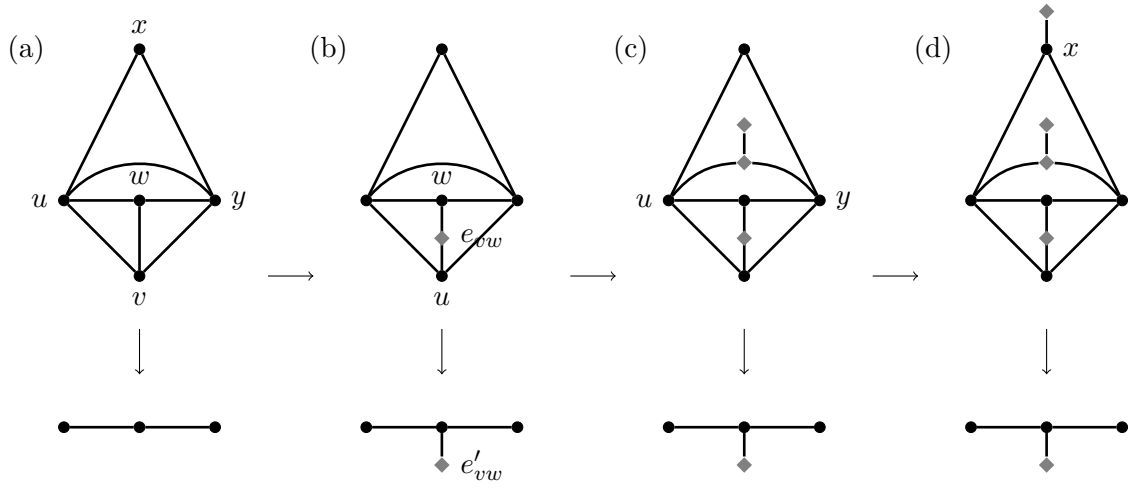


Figure 1: Consider the map (in (a)) where every vertex is mapped to the vertex of the tree below it. When both ends of an edge are mapped to the same vertex, for example edge vw, we add a vertex e_{vw} and map it to a new vertex e'_{vw} (see (b)). Then, we refine edges for which the ends are not mapped to the same vertex, like edge uy, as the part of the tree they correspond with (see (c)). Lastly, we add copies of parts of the tree, to make sure the morphism is harmonic at every vertex (see (d)). In this example, all edges have index 1.

property.'. To prove these lemmas we make some local changes to a finite harmonic morphism without changing the degree. Using these lemmas we prove that there exists a finite harmonic morphism with minimum degree and having these properties, from which it will follow that there exists a finite harmonic morphism of the from ϕ_α with minimum degree for some α. For all proofs, see [6].

References

[1] 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.

[2] 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.

[3] 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.

[4] 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.

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

[6] Ragnar Groot Koerkamp and Marieke van der Wegen. Stable gonality is computable. preprint, arXiv:1801.07553, 2018.

[7] Josef Schicho, Frank-Olaf Schreyer, and Martin 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.