An algorithm for checking whether the toric ideal of an affine monomial curve is a complete intersection

Ignacio García-Marco
University of La Laguna, Spain

Based on the paper

I. Bermejo, I. García-Marco, J.J. Salazar-González, An algorithm for checking whether the toric ideal of an affine monomial curve is a complete intersection, *J. Symbolic Comput.* **42** (2007), 971–991.

Implementation

I. Bermejo, I. García-Marco, J.J. Salazar-González, cimonom.lib, Singular 3.0.3, 2007.

Main references

I. Bermejo, Ph. Gimenez, E. Reyes, R. H. Villarreal, Complete intersections in affine monomial curves, *Bol. Soc. Mat. Mexicana (3a) Serie* **11 (2)** (2005), 191–204.

M. Clausen, A. Fortenbacher, Efficient Solution of Linear Diophantine Equations, *J. Symbolic Comput.* **8** (1989), 201–216.

Ch. Delorme, Sous-monoïdes d'intersection complète de $\mathbb{N}$, *Ann. Sci. École Norm. Sup.* **9** (1976), 145–154.

J. Herzog, Generators and relations of abelian semigroups and semigroup rings, *Manuscripta Math.* **3** (1970), 175–193.

Definitions and basic results I

Let $R = K[x_1, \ldots, x_n]$ be the polynomial ring in n variables over an arbitrary field K.

Denote by x^a the monomial $x_1^{a_1} \cdots x_n^{a_n}$, where $a = (a_1, \ldots, a_n) \in \mathbb{N}^n$.

A **binomial** f in R is a difference of two monomials, i.e., $f = x^a - x^b$ for some $a, b \in \mathbb{N}^n$. An ideal of R generated by binomials is called a **binomial ideal**.

Let $\{d_1, \ldots, d_n\}$ be a set of all-different positive integers and consider the affine monomial curve:

$$\Gamma = \{(t^{d_1}, \ldots, t^{d_n}) \in \mathbb{A}_K^n \mid t \in K\}$$

The kernel of the homomorphism of K-algebras $\phi : R \longrightarrow K[t]$ induced by $x_i \longmapsto t^{d_i}$ is called the **toric ideal of** Γ and will be denoted by

$$I(d_1, \ldots, d_n)$$

Definitions and basic results II

- $I(d_1, \ldots, d_n)$ is a 1-dimensional binomial ideal
- $I(d_1, \ldots, d_n)$ is generated by **quasi-homogeneous** binomials, i.e., homogeneous binomials when one gives degree d_i to variable x_i for all $i \in \{1, \ldots, n\}$
- If either $\gcd\{d_1, \ldots, d_n\} = 1$ or $K = \overline{K}$, we get $\Gamma = V(I(d_1, \ldots, d_n))$, i.e., Γ is a toric variety
- If K is an infinite field, $I(\Gamma) = I(d_1, \ldots, d_n)$

$I(d_1, \ldots, d_n)$ is a **complete intersection** if there exists a system of quasi-homogeneous binomials $g_1, \ldots, g_{n-1}$ such that

$$I(d_1, \ldots, d_n) = (g_1, \ldots, g_{n-1})$$

The definition **coincides with the usual one**.

Aim of the work

To obtain an **efficient algorithm** for checking whether or not $I(d_1, \ldots, d_n)$ is a complete intersection.

For all $i \in \{1, \ldots, n\}$ let us define

$$c_i := \min\left(\mathbb{Z}^+ d_i \cap \sum_{j \in \{1,\ldots,n\}\setminus\{i\}} \mathbb{N}\, d_j\right)$$

Herzog gives the following result when $n = 3$:

$I(d_1, d_2, d_3)$ is a complete intersection $\Leftrightarrow$ $\exists\, r, s : 1 \leq r < s \leq 3$, such that $c_r = c_s$.

This result does not hold for $n > 3$.

The aim of this work is to design an efficient algorithm for checking whether $I(d_1, \ldots, d_n)$ is a complete intersection which is mainly **based on the computation of some c_i**.

First attempt I

Proposition. Let $I(d_1, \ldots, d_n)$ be a **complete intersection**. Then the following two conditions hold:

1. $\exists\, r, s : 1 \leq r < s \leq n$ such that $c_r = c_s$;
2. whenever $c_r = c_s$ for $r, s : 1 \leq r < s \leq n$, one has

(a) $I(d_1, ..., \widehat{d_r}, ..., \widehat{d_s}, ..., d_{n+1})$ is a **complete intersection**, where $d_{n+1} := \gcd\{d_r, d_s\}$;

(b) if one sets

$$c_{n+1} := \min\left(\mathbb{Z}^+ d_{n+1} \cap \sum_{j \in \{1,\ldots,n\} \setminus \{r,s\}} \mathbb{N}\, d_j\right)$$

then $c_{n+1} \in \mathbb{N}\, d_r + \mathbb{N}\, d_s$;

(c) if for all $i \in \{1, \ldots, n\} \setminus \{r, s\}$ one sets

$$c_i' := \min\left(\mathbb{Z}^+ d_i \cap \sum_{j \in \{1,\ldots,n+1\} \setminus \{i,r,s\}} \mathbb{N}\, d_j\right)$$

then $c_i' = c_i$.

First attempt II

The $\mathbf{2}$ necessary conditions for $I(d_1, \ldots, d_n)$ to be a complete intersection in **Proposition** also turn out to be sufficient when $n = 4$.

Therefore, they provide an algorithm for determining whether or not $I(d_1, d_2, d_3, d_4)$ is a complete intersection that requires the design of procedures to solve the following problems:

- **To compute the smallest positive multiple of an integer that belongs to a semigroup.**

- **To check whether or not a positive integer belongs to a semigroup.**

First attempt III

This characterization does not hold for $n \geq 5$

Example. For $d_1 = 45, d_2 = 70, d_3 = 75, d_4 = 98$ and $d_5 = 147$, the toric ideal $I(d_1, d_2, d_3, d_4, d_5)$ **is not a complete intersection**.

Nevertheless, $c_1 = c_3 = 225$ and setting $d_6 := \gcd\{d_1, d_3\} = 15$, one has that

- $I(d_2, d_4, d_5, d_6)$ is a complete intersection
- $c_6 = \min(\mathbb{Z}^+ d_6 \cap \langle d_2, d_4, d_5 \rangle) = 210 \in \langle d_1, d_3 \rangle$
- $c_2' = c_2 = 210$, $c_4' = c_4 = c_5' = c_5 = 294$

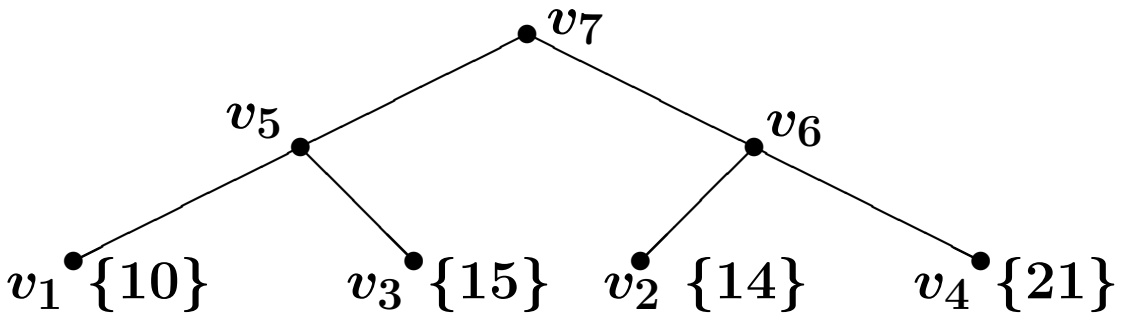
In spite of this, **Proposition** is **essential to describing our algorithm**.

Binary trees labeled by $\{d_1, \ldots, d_n\}$

A **binary tree** is a directed tree in which every node has either two children or zero. Nodes with no children are called **terminal nodes** and the only node with no parent is called the **root**.

A binary tree with n terminal nodes $v_1, \ldots, v_n$ and non-terminal nodes $v_{n+1}, \ldots, v_{2n-1}$ is said to be **labeled by** $\{d_1, \ldots, d_n\}$ if v_i is labeled by $\{d_i\}$ for all $i \in \{1, \ldots, n\}$, and for all $i \in \{n+1, \ldots, 2n-1\}$, v_i has children v_j, v_k with $j, k < i$.

This is a **binary tree labeled by** $\{10, 14, 15, 21\}$:



Main result I

Let $\mathcal{T}$ be a binary tree labeled by $\{d_1, \ldots, d_n\}$ and v_i a node of $\mathcal{T}$ **different from the root node**. Denote by Δ_{v_i} the subset of $\{d_1, \ldots, d_n\}$ such that the subtree of $\mathcal{T}$ whose root node is v_i is labeled by Δ_{v_i}.

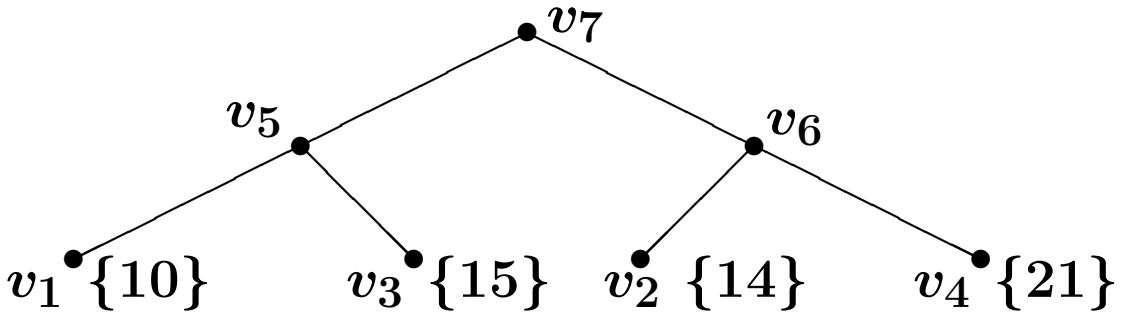
Moreover if v_i is a non-terminal node define $d_i := \gcd(\Delta_{v_i})$ and c_i as

$$\min\left(\mathbb{Z}^+ d_i \bigcap \mathbb{N}\{d_s \mid v_s \text{ child of } v_t,\ s < i < t\}\right).$$

Theorem - Algorithm. $I(d_1, \ldots, d_n)$ is a **complete intersection** $\Longleftrightarrow$ one can construct a binary tree $\mathcal{T}$ labeled by $\{d_1, \ldots, d_n\}$ such that for each $i \in \{n+1, \ldots, 2n-2\}$, the node v_i of $\mathcal{T}$ with children v_j and v_k has $c_j = c_k$ and $c_i \in \sum_{d_r \in \Delta_{v_i}} \mathbb{N}\, d_r$.

Main result II

$I(10, 14, 15, 21)$ is a complete intersection since the following binary tree labeled by $\{10, 14, 15, 21\}$ satisfies the arithmetical conditions stated in **Theorem - Algorithm**:



Indeed, $c_1 = c_3 = 30$.

Setting $d_5 := \gcd\{d_1, d_3\} = 5$ and

$$c_5 := \min\left(\mathbb{Z}^+ d_5 \cap \mathbb{N}\{d_2, d_4\}\right) = 35\,,$$

one finds that $c_5 \in \mathbb{N}\{d_1, d_3\} \longleftrightarrow c_5 = 2d_1 + d_3$

Moreover, $c_2 = c_4 = 42$.

Setting $d_6 := \gcd\{d_2, d_4\} = 7$ and

$$c_6 := \min\left(\mathbb{Z}^+ d_5 \cap \mathbb{N}\, d_6\right) = 35\,,$$

one finds that $c_6 \in \mathbb{N}\{d_2, d_4\} \longleftrightarrow c_6 = d_2 + d_4$.

Algorithm CI

Require: $\{d_1, \ldots, d_n\}$

Ensure: **TRUE** or **FALSE**

$G_1 := \{d_1, \ldots, d_n\}$

for $i = 1$ to n **do**

$V_i := \{d_i\}$, $c_i := \min\left(\mathbb{Z}^+ d_i \cap \sum_{j \in \{1,\ldots,n\}\setminus\{i\}} \mathbb{N}\, d_j\right)$

end for

for $i = 1$ to $n-2$ **do**

if $c_j \neq c_k$ for all $j, k : j \neq k$ and $d_j, d_k \in G_i$ **then**

return **FALSE**

end if

Let $j, k : j \neq k$ such that $d_j, d_k \in G_i$ and $c_j = c_k$

$d_{n+i} := \gcd\{d_j, d_k\}$, $V_{n+i} := V_j \cup V_k$

$G_{i+1} := G_i \setminus \{d_j, d_k\} \cup \{d_{n+i}\}$

$c_{n+i} := \min\left(\mathbb{Z}^+ d_{n+i} \cap \sum_{d_s \in G_{i+1}\setminus\{d_{n+i}\}} \mathbb{N}\, d_s\right)$

if $c_{n+i} \notin \sum_{d_j \in V_{n+i}} \mathbb{N}\, d_j$ **then**

return **FALSE**

end if

end for

return **TRUE**

Algorithm CI

Given $\{d_1, \ldots, d_n\}$ such that $I(d_1, \ldots, d_n)$ is a complete intersection, **Algorithm CI** returns, with no additional effort, **a system of $n-1$ quasi-homogeneous generators for the toric ideal $I(d_1, \ldots, d_n)$**.

In the previous example, one gets:

$$x_1^3 - x_3^2 \ \ldots\ldots\ldots\ldots\ldots\ c_1 = c_3 = \underline{3}d_1 = \underline{2}d_3$$
$$x_2^3 - x_4^2 \ \ldots\ldots\ldots\ldots\ldots\ c_2 = c_4 = \underline{3}d_2 = \underline{2}d_4$$
$$x_1^2x_3 - x_2x_4 \ldots\ c_5 = c_6 = \underline{2}d_1 + \underline{1}d_3 = \underline{1}d_2 + \underline{1}d_4$$

Moreover, when $\gcd\{d_1, \ldots, d_n\} = 1$, **Algorithm CI** also returns the **Frobenius number $\mathrm{g}(\mathcal{S})$ of the semigroup $\mathcal{S} := \sum_{i=1}^{n} \mathbb{N}\, d_i$**, i.e., the largest integer not in $\mathcal{S}$, using the formula:

$$\mathrm{g}(\mathcal{S}) = \frac{1}{2} \sum_{i=1}^{2n-2} c_i - \sum_{i=1}^{n} d_i \,.$$

In the previous example, $\mathrm{g}(\mathcal{S}) = 47$.

Computational aspects I

A direct implementation of **Algorithm CI** requires an **efficient procedure** to compute the values c_i.

Given $\{d_1, \ldots, d_n\}$, the optimization problem of computing

$$c_1 = \min \left(\mathbb{Z}^+ d_1 \cap \sum_{j \in \{2,\ldots,n\}} \mathbb{N}\, d_j \right)$$

can be formulated by the following Integer Linear Programming (ILP) model:

$$\begin{aligned} x_1^* := \min \quad & x_1 && (1)\\ & d_1 x_1 = d_2 x_2 + \cdots + d_n x_n && (2)\\ & x_1 \geq 1;\ x_2, \ldots, x_n \geq 0 && (3)\\ & x_1, x_2, \ldots, x_n \in \mathbb{Z} && (4) \end{aligned}$$

Then $c_1 = d_1 x_1^*$.

The computation of c_1 is a $\mathcal{NP}$**-hard** problem.

Computational aspects II

To compute x_1^*, we use a **Graph Theory** representation of the problem.

The approach is similar in spirit to that of **Clausen & Fortenbacher** to solve linear diophantine equations.

The idea is to represent each solution

$$(x_1, x_2, \ldots, x_n)$$

of (2)–(4) as a **walk** in a **graph**, where the **weight** of the walk is x_1.

Then the combinatorial problem modeled in (1)–(4) is equivalent to **finding a shortest path in the graph**.

Computational aspects III

Consider the following directed weighted graph $\mathcal{G} = (V, A)$, where the **node set** is

$$V := \{0, 1, \ldots, d_1 - 1\},$$

the **arc set** is

$$A := \bigcup_{i=2}^{n} \{(v, (v + d_i) \bmod d_1) \mid v \in V\},$$

and for all $v \in V$ and $i \in \{2, \ldots, n\}$, the **weight of the arc** $(v, (v + d_i) \bmod d_1)$ is defined equal to

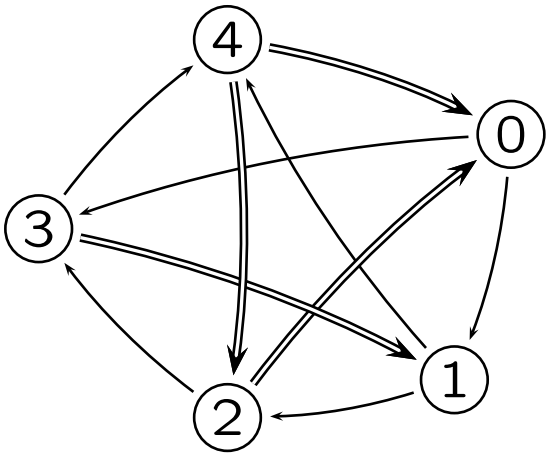
$$w_{(v,i)} := \left\lfloor \frac{v + d_i}{d_1} \right\rfloor.$$

Lemma. There is an onto map of the set of closed walks in $\mathcal{G}$ starting at 0 with weight x_1 into the set of solutions $(x_1, x_2, \ldots, x_n)$ of (2)–(4).

Proof. $(v + d_i) \bmod d_1 = v + d_i - w_{(v,i)} d_1$

Computational aspects IV

The graph $\mathcal{G}$ for the instance $d_1 = 5\,, d_2 = 6\,, d_3 = 8$ is the following:



where arcs represented by **single lines** have a **weight** equal to **one**, and arcs represented by **double lines** have a **weight** equal to **two**.

The path $\textcircled{0} \rightarrow \textcircled{3} \rightarrow \textcircled{4} \rightarrow \textcircled{0}$

corresponds to the solution $(4, 2, 1)$ of (2)–(4):

$$\left.\begin{array}{l} 0 + d_3 - 1 \cdot d_1 = 3 \\ 3 + d_2 - 1 \cdot d_1 = 4 \\ 4 + d_2 - 2 \cdot d_1 = 0 \end{array}\right| \Rightarrow 4d_1 = 2d_2 + d_3$$

Computational aspects V

To compute x_1^* one can apply **Dijkstra's algorithm** to find a shortest cycle starting from 0 in $\mathcal{G}$.

The **complexity** of the Dijkstra algorithm depends on the number of nodes and arcs in the graph. Since the number of nodes in $\mathcal{G}$ is d_1 and the number of arcs is $(n-1)d_1$, the optimization problem (1)–(4) can be solved in $\mathcal{O}(n \cdot d_1 + d_1 \cdot \log(d_1))$.

Corollary. Each c_i in **Algorithm CI** can be computed in **pseudo-polynomial time**.

Computational aspects VI

Our computational experiments show that **Algorithm CI** is able to solve **large-size instances**.

For instance, our implementation takes **less that one second** on a personal computer with Intel Pentium IV 3Ghz. to prove that the toric ideal $I(d_1, \ldots, d_{13})$ is a **complete intersection**, where

$$d_1 = 304920 \quad d_2 = 381150 \quad d_3 = 457380$$

$$d_4 = 571725 \quad d_5 = 97911 \quad d_6 = 223146$$

$$d_7 = 239085 \quad d_8 = 159390 \quad d_9 = 334719$$

$$d_{10} = 224112 \quad d_{11} = 238119 \quad d_{12} = 252126$$

$$d_{13} = 334949$$

Computational aspects VII

Additionally, the implementation also gives a **minimal set of quasi-homogeneous generators of the toric ideal**:

$$x_3^2 - x_1^3 \qquad x_8^3 - x_7^2$$

$$x_9^2 - x_6^3 \qquad x_{12}^8 - x_{10}^9$$

$$x_1\,x_3 - x_2^2 \qquad x_6\,x_9 - x_7\,x_8^2$$

$$x_{10}\,x_{12} - x_{11}^2 \qquad x_2^3 - x_4^2$$

$$x_{10}\,x_{11} - x_6\,x_7 \qquad x_6\,x_{10}\,x_{11} - x_5^7$$

$$x_7^3\,x_8 - x_1\,x_4 \qquad x_5^2\,x_8^2\,x_{11}\,x_{12} - x_{13}^3$$

and shows that the **Frobenius number** of the semigroup $\sum_{i=1}^{13} \mathbb{N}\,d_i$ is $\mathbf{6229597}$.