Dividing sum of cycles in the semiring of functional digraphs *

Florian Bridoux[1], Christophe Crespelle[1], Thi Ha Duong Phan[2], and Adrien Richard[1]

[1]Université Côte d'Azur, CNRS, I3S, Sophia Antipolis, France
{florian.bridoux,christophe.crespelle,adrien.richard}@univ-cotedazur.fr
[2]ICRTM - Institute of Mathematics, Vietnam Academy of Science and Technology, Vietnam
phanhaduong@math.ac.vn

April 9, 2025

Abstract

Functional digraphs are unlabelled finite digraphs where each vertex has exactly one out-neighbor. They are isomorphic classes of finite discrete-time dynamical systems. Endowed with the direct sum and product, functional digraphs form a semiring with an interesting multiplicative structure. For instance, we do not know if the following division problem can be solved in polynomial time: given two functional digraphs A and B, does A divide B? That A divides B means that there exists a functional digraph X such that AX is isomorphic to B, and many such X can exist. We can thus ask for the number of solutions X. In this paper, we focus on the case where B is a sum of cycles (a disjoint union of cycles, corresponding to the limit behavior of finite discrete-time dynamical systems). There is then a naïve sub-exponential algorithm to compute the non-isomorphic solutions X, and our main result is an improvement of this algorithm which has the property to be polynomial when A is fixed. It uses a divide-and-conquer technique that should be useful for further developments on the division problem.

Keywords: Finite Dynamical Systems, Functional digraphs, Graph direct product, Graph factorization.

1 Introduction

A deterministic, finite, discrete-time dynamical system is a function from a finite set (of states or configurations) to itself. Equivalently, this is a *functional digraph*, that is, a finite directed graph where each vertex has a unique out-neighbor. In addition to being ubiquitous objects in discrete mathematics, such systems have many real-life applications: they are usual models for gene networks [26, 36, 38, 25], neural networks [28, 24, 19], reaction systems [14], social interactions [31, 21] and more [37, 20].

A functional digraph A basically contains two parts: the *cyclic part*, which is the collection of the cycles of A (which are vertex disjoint), and the *transient part*, which is obtained by deleting arcs in cycles, and which is a disjoint union of out-trees; see Figure 1. From a dynamical point of view, the cyclic part is fundamental since it describes the asymptotic behavior of the system, and this is the part we focus on in this paper.

Furthermore, we consider functional digraphs up to isomorphism. This is motivated by the fact that many studied dynamical properties are invariant by isomorphism: number of fixed

*This paper is an extended version of [2]

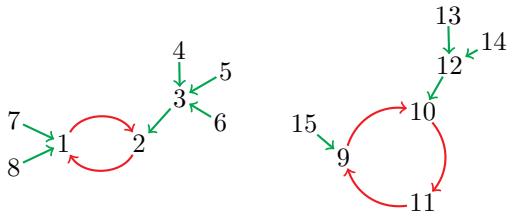

Figure 1: A functional digraph, with cyclic part in red, and transient part in green.

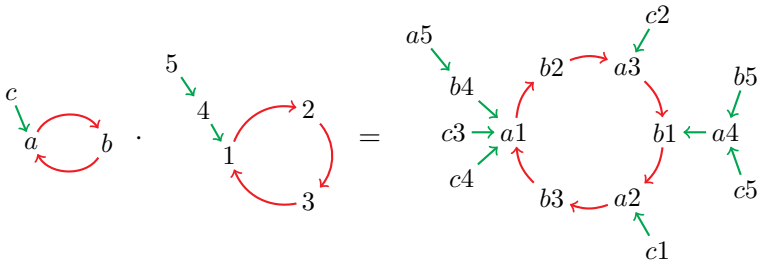

Figure 2: Product of two functional digraphs.

points, periodic points, limit cycles, lengths of limits cycles and so on; see e.g. [16] in the context of random functional digraphs, and [32, 17] in the context of automata networks. An isomorphism class then corresponds to an unlabelled functional digraph, and we write $A = B$ to mean that A is isomorphic to B. Since a planar embedding of a functional digraph can be obtained in linear time, testing if $A = B$ can be done in linear time [23].

There are two natural algebraic operations to obtain larger systems from smaller ones. Given two functional digraphs A and B, the *addition* $A + B$ is the disjoint union of A and B, while the *multiplication* $A \cdot B$ (or simply AB) is the *direct product* of A and B: the vertex set of AB is the Cartesian product of the vertex set of A and the vertex set of B, and the out-neighbor of (a, b) in AB is (a', b') where a' is the out-neighbor of a in A and b' is the out-neighbor of vertex b in B; see Figure 2. Hence AB describes the parallel evolution of the dynamics described by A and B. Endowed with these two operations, the set of functional digraphs forms a semiring $\mathbb{F}$, introduced as such in [5]. The identity element for the addition is the empty functional digraph, while for the product it is the cycle of length one, denoted C_1.

The semiring $\mathbb{F}$ contains an isomorphic copy of $\mathbb{N}$ (by identifying each integer n with the sum of n copies of C_1). The set of sums of cycles $\mathbb{S}$, which describes the cyclic parts of functional digraphs, is also a sub-semiring of $\mathbb{F}$. Because the multiplication of cycles involves the least common multiple operation, the multiplicative structure of $\mathbb{F}$ and $\mathbb{S}$ has interesting properties, which differ from those of $\mathbb{N}$ (or the semiring of polynomials), and have been recently studied in [6, 7, 18, 33, 8, 12, 11, 29, 9], leading to many interesting open complexity problems.

To emphase such differences we need some definitions. Let us say that $X \in \mathbb{F}$ is *irreducible* if, for all $A, B \in \mathbb{F}$, $X = AB$ implies $A = C_1$ or $B = C_1$. Let say that A *divides* B, $A \mid B$ in notation, if there is X such that $AX = B$. Finally, let us say that X is *prime* if $X \neq C_1$ and, for all $A, B \in \mathbb{F}$, $X \mid AB$ implies $X \mid A$ or $X \mid B$. In $\mathbb{N}$, irreducibility and primeness are a same concept, but this is far from being true for $\mathbb{F}$: almost all functional digraphs are irreducible [13], while Seifert [35] proved, with technical arguments, that there is no prime functional digraph. Using only very simple arguments, we will prove that $\mathbb{S}$ share the same two properties. In practice, asking for irreducibility is very natural: given an observed dynamical system, does it correspond (up to isomorphism) to the parallel evolution of two smaller systems? Testing

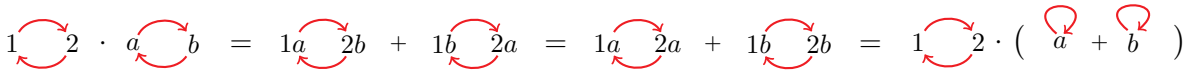

Figure 3: The two irreducible factorizations of $C_2 + C_2$.

irreducibility is obviously in coNP, but this problem does not seem to have been studied in depth.

Another key point is that functional digraphs do not have a unique factorization into irreducibles (distinct from C_1). This is even true for sums of cycles. For instance, denoting C_ℓ the cycle of length ℓ, the sum of cycles $C_2 + C_2$ has two distinct irreducible factorizations, which are $C_2 \cdot C_2$ and $C_2 \cdot (C_1 + C_1)$; see Figure 3. Actually, we will prove that, for every $\epsilon > 0$ and infinitely many n, there exists a sum of cycles with n^2 vertices which has at least

$$n^{2^{\frac{\ln n}{(1+\epsilon)\ln\ln n}}} \tag{1}$$

irreducible factorizations, which is super-polynomial in n. Using a very rough argument, we will also show that every sum of cycles with n vertices has at most $e^{O(\sqrt{n})}$ irreducible factorizations, which is sub-exponential, and we think that this upper-bound is far from being accurate. Here again, there is a lack of theoretical results concerning the complexity of finding an irreducible factorization.

As last general observation concerning $\mathbb{S}$, the division is not unique: given two sums of cycles A, B, the equation $AX = B$ can have several solutions X. For instance, Figure 3 shows that there exists two X such that $C_2 \cdot X = 2C_2$, which are $X = C_2$ and $X = C_1 + C_1$. Actually, we will prove that, for every $\epsilon > 0$ and infinitely many n, there exists a sum of cycles A with n vertices and a sum of cycles B with n^2 vertices such that the number of solutions X to $AX = B$ is at least the expression given in (1), super-polynomial in n.

From an algorithmic point of view, the *division problem*, which consists, given A, B, to decide if A divides B, is at the basis of many other problems [33, 8, 29, 9]. The main purpose of this paper is to study the complexity of this problem for sums of cycles. This restriction has already been considered in [8] as an important step for solving polynomial equations over functional digraphs. On the other side, [29] gives a polynomial algorithm to decide if A divides B when B is a *dendron*, that is, B contains a unique cycle, of length 1, so that B consists of an out-tree plus a loop on the root. This result should be very useful to treat the transient part in the division problem. One may hope that efficient algorithms for the cyclic and transient parts could be combined to obtain an efficient algorithm for the general case.

Given two sum of cycles A, B, there is a simple sub-exponential algorithm to compute all the solutions X of $AX = B$, that we call *brute force approach*. It works as follows. Let $|A|$ and $|B|$ be the number of vertices in A and B, respectively; $|A| + |B|$ is the size of the instance. If $AX = B$ then X has $n = |B|/|A|$ vertices. Now remark that sums of cycles with n vertices are in bijection with partitions of n: a sum of cycles $C_{\ell_1} + \cdots + C_{\ell_k}$ with n vertices is completely described by the sequence $\ell_1, \ldots, \ell_k$ of the length of its cycles, which form a partition of n; and conversely, the parts of a partition of n describe the lengths of the cycles of a sum of cycles with n vertices. For instance, $C_1 + C_3$ corresponds to the partition $1 + 3$ of 4. So to compute the solutions, we can enumerate the partitions of n, and check for each if the corresponding sum of cycles X satisfies $AX = B$. This gives a sub-exponential algorithm. Indeed, the number of partitions of n can be enumerated with linear-time delay and there are at most $e^{O(\sqrt{n})}$ such partitions, giving a total running time in $|B|e^{O(\sqrt{n})}$; details will be given in Section 3.

Frustratingly, we were not able to find a faster algorithm, say running in $|B|^d e^{O(n^\epsilon)}$ for some constants d and $\epsilon < 1/2$, to decide if A divides B. That a polynomial algorithm exists is open, and [9] gives a positive answer under the strong condition that, in A or B, all the cycles have the same length.

Here we design a general algorithm to compute the *number* of solutions which has the interesting property to be polynomial when A is fixed. The precise statement, Theorem 1 below, involves some definitions. The *support* of a sum of cycles A is the set $L(A)$ of positive integers ℓ such that A contains C_ℓ. Given $N \subseteq \mathbb{N}$, $\operatorname{lcm} N$ is the least common multiple of the integers in N, and $\operatorname{div}(n)$ is the number of divisors of n.

Theorem 1. *There is an algorithm that, given two non-empty sums of cycles A, B, computes the number of sums of cycles X satisfying $AX = B$ with time complexity in*

$$O\left(|B|^3 \left(\frac{|B|}{|A|}\right)^{\operatorname{div}(\operatorname{lcm} L(A))}\right). \tag{2}$$

For the proof, we introduce two operations on an instance (A, B) that we hope to be useful, or at least an inspiration, for further progress on the division problem. These are the *split* and *reduction* operations. The first splits B into $B = B_1 + B_2$ so that any solution of (A, B) is obtained by adding a solution to (A, B_1) with a solution to (A, B_2). The second reduces (A, B) into a smaller instance (A', B') so that any solution of (A, B) is obtained by multiplying the length of the cycles of a solution of (A', B') by some constant d. Repeating as much as possible these operations, we obtain a *decomposition* of (A, B) into few smaller instances, which can be quickly solved with an easy improvement of the brute force approach, that we call *brute force approach on the support*. The solutions of (A, B) are then obtained with a simple combination of the solutions of the instances of its decomposition. In particular, the number of solutions of (A, B) is simply the product of the number of solutions of the instances of the decomposition.

Organization of the paper Section 2 gives the notations and basic results used throughout the paper. Section 3 presents in detail the brute force approach sketched above and its improvement, the brute force approach on the support, used in the algorithm of Theorem 1. Section 4 presents the decomposition technique, involving the split and reduction operations. This decomposition is actually the main contribution, Theorem 1 being a simple application. Section 5 discusses an improvement of the brute force approach on the support, which should be a potential useful tool for further developments. To illustrate this, we easily prove that it gives, in polynomial time, all the solutions of instances (A, B) such that $L(A) \setminus \{1\}$ only contains primes, or such that $1 \in L(B)$. Concluding remarks are given in Section 6. The appendix gives the proofs of all the general observations made on $\mathbb{S}$ in this introduction; it is independent of the rest of the paper in that it only refers on the material given in Section 2. In Appendix A, we prove that almost all sums of cycles are irreducible and that $\mathbb{S}$ has no prime element. In Appendix B we prove that, for every $\epsilon > 0$ and infinitely many n, there exists an instance (A, B) with $|A| = n$ and $|B| = n^2$ such that the number of solutions of (A, B) and the number of irreducible factorizations of B is at least (1), hence super-polynomial. Finally, in Appendix C, we prove that a sum of cycles with n vertices has at most $e^{O(\sqrt{n})}$ irreducible factorizations.

2 Preliminaries

Integers Given $N \subseteq \mathbb{N}$, we denote by $\operatorname{lcm} N$ and $\gcd N$ the least common multiple and the greatest common divisor of the integers in N, respectively. For $n, m \in \mathbb{N}$, we set $n \vee m = \operatorname{lcm}(n, m)$ and $n \wedge m = \gcd(n, m)$, and for $N, M \subseteq \mathbb{N}$ we set $N \vee M = \{n \vee m \mid n \in N, m \in M\}$.

We denote by $\mathrm{Div}(n)$ and $\mathrm{Mult}(n)$ the set of divisors and positive multiples of n, respectively, and set $\mathrm{div}(n) = |\mathrm{Div}(n)|$. We set $\mathrm{Div}(N) = \bigcup_{n\in N} \mathrm{Div}(n)$, $\mathrm{Mult}(N) = \bigcup_{n\in N} \mathrm{Mult}(n)$ and $\mathrm{div}(N) = |\mathrm{Div}(N)|$. For a positive integer p, we write $p \mid N$ to means that $p \mid n$ for all $n \in N$. We set $pN = \{pn \mid n \in N\}$ and $N/p = \{n/p \mid n \in N, p \mid n\}$. For a positive integer n, and a prime p, we use the usual notation $\nu_p(n)$ to denote the greatest integer α such that $p^\alpha \mid n$.

Sums of cycles The cycle of length ℓ is denoted C_ℓ, and the sum of cycles that consists of n cycles of length ℓ is denoted by nC_ℓ. Let A, B, X be sums of cycles. The number of vertices in A is denoted $|A|$ and is the *size* of A. The number of cycles of length ℓ in A is denoted by $A(\ell)$. Thus $A = \sum_{\ell\geq 1} A(\ell)C_\ell$ and

$$|A| = \sum_{\ell\geq 1} \ell A(\ell). \tag{3}$$

The *support* of A is the set of ℓ such that $A(\ell) > 0$, denoted $L(A)$. We write $A \subseteq B$ to mean that A is a subgraph of B, equivalently, $A(\ell) \leq B(\ell)$ for all $\ell \geq 1$. If $A \subseteq B$ then $B - A$ is the sum of cycles A' such that $A + A' = B$. We easily check (see e.g [5]) that the product between C_a and C_b is

$$C_a C_b = \left(\frac{ab}{a \vee b}\right) C_{a\vee b} = (a \wedge b)C_{a\vee b}.$$

By distributivity, the product AX satisfies, for all integers ℓ,

$$(AX)(\ell) = \frac{1}{\ell} \sum_{\substack{a,x\in\mathbb{N}\\ a\vee x=\ell}} aA(a)xX(x). \tag{4}$$

Note that since $A(a)X(x) = 0$ if $a \notin L(A)$ or $x \notin L(X)$, we can restrict the sum to the couples $(a, b) \in L(A) \times L(X)$ satisfying $a \vee x = \ell$ to get a finite expression. It is important to note that, by (4), we have

$$L(AX) = L(A) \vee L(X) \tag{5}$$

and thus

$$L(X) \subseteq \mathrm{Div}(L(AX)). \tag{6}$$

If A, B are non-empty, then (A, B) is an *instance*, and its *size* is $|A| + |B|$. A *solution* of (A, B) is a sum of cycles X such that $AX = B$. Note that $|X| = |B|/|A|$ for every solution X. We denote by $\mathrm{Sol}(A, B)$ the set of solutions, and $\mathrm{sol}(A, B) = |\mathrm{Sol}(A, B)|$. If a solution exists, we say that A *divides* B and we write $A \mid B$.

Integer partitions Let n be a positive integer. A *partition* of n is a non-decreasing sequence of positive integers which sum to n. We denote by $p(n)$ the numbers of partition of n. As mentioned in the introduction, $p(n)$ is te number of sum of cycles with n vertices; by (3), we can regard a sum of cycles A with n vertices as a partition of n where the number of occurences of each integer ℓ is $A(\ell)$. Let $c_0 = \pi\sqrt{2/3}$. Hardy and Ramanujan [22] famously proved that $p(n) \sim (e^{c_0\sqrt{n}})/(4\sqrt{3}n)$. Here, we will only use the following bounds, proved by Erdős (using only elementary methods) [15]: for every $\epsilon > 0$, and n large enough

$$(c_0 - \epsilon)\sqrt{n} \leq \ln p(n) \leq c_0\sqrt{n}. \tag{7}$$

A partition of n with parts in L is non-decreasing sequence of positive integers, all in L, which sum to n. We denote by $p_L(n)$ the number of partitions of n with parts in L. Nathanson [30] gives an asymptotic formula for $p_L(n)$ when L is fixed, but we won't need it.

3 Brute force approach on the support

We saw in the introduction that there is a natural bijection between sums of cycles with n vertices and partitions of n. This gives the following brute force approach to find all the solutions of an instance (A, B):

> **Brute force approach.** If $n = |B|/|A|$ is not an integer, return $\emptyset$. Otherwise, enumerate all the partitions of n, select those which correspond to solutions, and return them.

Let us analyse the complexity of this simple algorithm. By (7), there are $e^{O(\sqrt{n})}$ partitions of n to enumerate, and there are simple linear-time delay algorithms for this enumeration [27] [1]. Hence the enumeration can be done in $e^{O(\sqrt{n})}$. To check if a partition corresponds to a solution, we take the sum of cycles X corresponding to the partition, we compute the product AX in $O(|B|)$, and we check if $AX = B$ in $O(|B|)$ [23]. The total running time is thus

$$|B|e^{O(\sqrt{n})}. \tag{8}$$

Even if the only information the algorithm extracts from the instance is the size $|B|/|A|$ of the solutions, it's actually not so easy to improve it significantly. Our approach to extract more information is to consider what we call the *support* of the instance, which depends on the supports of A and B and the size $|B|/|A|$ of solutions.

Definition 1. *The support of an instance (A, B), denoted $L(A, B)$, is the set of positive integers $x \leq |B|/|A|$ such that $L(A) \vee x \subseteq L(B)$.*

Example 1.

$$\begin{aligned} L(C_6, C_6 + C_{12}) &= \{1, 2, 3\} \\ L(C_6, 3C_6 + 8C_{12}) &= \{1, 2, 3, 4, 6, 12\} \\ L(C_6, 6C_5 + C_6) &= \{1, 2, 3, 6\} \end{aligned}$$

Note that $6, 12 \notin L(C_6, C_6 + C_{12})$ since $6, 12 \geq |C_6 + C_{12}|/|C_6| = 18/6 = 3$.

The main property of $L(A, B)$ is that it bounds the support of any solution:

$$\forall X \in \mathrm{Sol}(A, B), \qquad L(X) \subseteq L(A, B). \tag{9}$$

Indeed, if $AX = B$ then, by (5), we have $L(A) \vee L(X) = L(B)$. Since $\max L(X) \leq |X| = |B|/|A|$ we have $L(X) \subseteq L(A, B)$. In other words solutions are partitions of $n = |B|/|A|$ with parts in $L(A, B)$. This gives the following improvement to the previous algorithm:

> **Brute force approach on the support.** If $n = |B|/|A|$ is not an integer, return $\emptyset$. Otherwise, compute $L(A, B)$, enumerate all the partitions of n with parts in $L(A, B)$, select those which correspond to solutions, and return them.

Since $L(A, B)$ only contains integers between 1 and $n = |B|/|A|$, we can compute $L(A, B)$ with time complexity $O(|B|^2)$ by enumerating the integers $1 \leq x \leq n$ and selecting, in $O(|A||B|)$, those that satisfy $L(A) \vee x \subseteq L(B)$. Next, to enumerate the partitions of n with parts in $L = L(A, B)$, we proceed as follows. In such a partition, the number of occurrences of $x \in L$ is between 0 and $\lfloor n/x \rfloor$. Hence we can enumerate all the functions $f : L \to \mathbb{N}$ such that $f(x) \leq 1 + \lfloor n/x \rfloor$ for all $x \in L$, and select those which correspond to partitions, that is, such that $\sum_{x \in L} x f(x) = n$. It is easy to show, by induction on $|L|$, that the number of such functions

[1] This is much more difficult to enumerate all the functional digraphs with n vertices, and the recent paper [4] provides quadratic-time delay algorithm for this task.

is at most $n^{|L|-1}$. Since the delay for the enumeration of these functions is linear, this gives an enumeration of the partitions of n with parts in L in $O(n^{|L|})$. Then, as before, we check if a partition corresponds to a solution in $O(|B|)$. The total complexity is thus $O(|B|^2 + |B|n^{|L|})$, that we simplify in $O(|B|^2 n^{|L|})$. Hence we have proved the following.

Lemma 1. *The brute force approach on the support computes the set of solutions of* (A, B) *in*

$$O\left(|B|^2 \left(\frac{|B|}{|A|}\right)^{|L(A,B)|}\right). \tag{10}$$

4 Decomposition lemma

In this section, we state and prove our main result, which is the polynomial-time decomposition of an instance (A, B) into smaller instances $(A_1, B_1), \ldots, (A_k, B_k)$, so that the number of solutions (A, B) can be reconstructed from the number of solutions of the smaller instances (A_i, B_i). These smaller instances always exhibit a particular property: the size of their support is bounded according to A_i. Using the brute force approach on the support to compute the number of solutions of each (A_i, B_i), we obtain the number of solutions for (A, B) with a complexity similar to (10), but where the critical term $|L(A, B)|$ in exponent is, as for the smaller instances, bounded according to A. Hence the whole algorithm becomes polynomial when A is fixed.

For the detail we need some definitions. Let us say that an instance (A, B) is *consistent* if

$$L(A) \vee L(A, B) = L(B).$$

Then non-consistent instances have no solution: if $AX = B$ then (A, B) is consistent since

$$L(B) = L(AX) \overset{(5)}{=} L(A) \vee L(X) \overset{(9)}{\subseteq} L(A) \vee L(A, B) \subseteq L(B).$$

Example 2. *We take again the three instances of Example 1.*

$$\begin{array}{rcll} L(C_6, C_6 + C_{12}) & = & \{1,2,3\} & \textit{non-consistent}: \quad \{6\} \vee \{1,2,3\} = \{6\} \neq \{6,12\} \\ L(C_6, 3C_6 + 8C_{12}) & = & \{1,2,3,4,6,12\} & \textit{consistent}: \quad \{6\} \vee \{1,2,3,4,6,12\} = \{6,12\} \\ L(C_6, 6C_5 + C_6) & = & \{1,2,3,6\} & \textit{non-consistent}: \quad \{6\} \vee \{1,2,3,6\} = \{6\} \neq \{5,6\} \end{array}$$

Let us say that an instance (A, B) is *basic* if

$$L(B) \subseteq \mathrm{Div}(\mathrm{lcm} L(A)).$$

This is equivalent to say that for any prime p and $b \in L(B)$, there exists $a \in L(A)$ such that $\nu_p(b) \leq \nu_p(a)$. Note that the support of a basic instance (A, B) is bounded according to A only:

$$L(A, B) \subseteq \mathrm{Div}(L(B)) \subseteq \mathrm{Div}(\mathrm{lcm} L(A)).$$

Example 3. *The two instances below are consistent, but only one is basic:*

$$\begin{array}{lll} (C_2 + C_6, C_{12}) & \textit{non-basic}: & L(C_{12}) = \{12\} \not\subseteq \mathrm{Div}(\mathrm{lcm}\{2,6\}) = \mathrm{Div}(6). \\ (C_5 + C_6, C_6 + C_{15}) & \textit{basic}: & L(C_6 + C_{15}) = \{6,15\} \subseteq \mathrm{Div}(\mathrm{lcm}\{5,6\}) = \mathrm{Div}(30). \end{array}$$

The *cycle length multiplication* of A by p, denoted $A \otimes p$, is the sum of cycles obtained from A by multiplying by p the length of every cycle in A. In other words: for all $a \geq 1$, we have

$$(A \otimes p)(a) = \begin{cases} A(a/p) & \text{if } p \mid a \\ 0 & \text{otherwise.} \end{cases}$$

Example 4.

$$(2C_1 + 3C_2 + 5C_3) \otimes 3 = 2C_3 + 3C_6 + 5C_9.$$

Given two sets of sums of cycles $\boldsymbol{A}$ and $\boldsymbol{B}$ we set

$$\boldsymbol{A} + \boldsymbol{B} = \{A + B \mid A \in \boldsymbol{A}, B \in \boldsymbol{B}\}, \qquad \boldsymbol{A} \otimes p = \{A \otimes p \mid A \in \boldsymbol{A}\}.$$

We are now ready to state the decomposition sketched at the beginning of the section.

Lemma 2 (Decomposition lemma). *There is an algorithm that, given a consistent instance (A, B), computes in $O(|B|^3)$ a list of $k \leq |B|$ basic instances $(A_1, B_2), \ldots, (A_k, B_k)$ and positive integers $p_1, \ldots, p_k$ such that: for all $1 \leq i \leq k$,*

- $|A_i| = |A|$,
- $\mathrm{lcm}L(A_i) \mid \mathrm{lcm}L(A)$,
- $|B_1| + \cdots + |B_k| \leq |B|$,
- $\mathrm{Sol}(A, B) = (\mathrm{Sol}(A_1, B_1) \otimes p_1) + \cdots + (\mathrm{Sol}(A_k, B_k) \otimes p_k)$.

Before proving this lemma, let us show that it easily implies Theorem 1 given in the introduction, restated below. It shows that there is a general algorithm for computing the solutions of an instance (A, B) with property to be polynomial when A is fixed.

Theorem 1. *There is an algorithm that, given two non-empty sums of cycles A, B, computes the number of sums of cycles X satisfying $AX = B$ with time complexity in*

$$O\left(|B|^3 \left(\frac{|B|}{|A|}\right)^{\mathrm{div}(\mathrm{lcm}L(A))}\right).$$

Proof of Theorem 1 assuming Lemma 2. The algorithm is as follows. First we check if (A, B) is consistent. If (A, B) is non-consistent, then (A, B) has no solution and we output 0. Otherwise, we compute the $k \leq |B|$ basic instances $(A_1, B_1), \ldots, (A_k, B_k)$ with the algorithm of Lemma 2. Then, for all $1 \leq i \leq k$, we use the brute force approach on the support to compute number s_i of solutions of (A_i, B_i), and we output the product $s_1 \cdots s_k$, which is correct by the fourth bullet in Lemma 2.

Let us analyse the complexity. First, since $L(A, B)$ can be computed in $O(|B|^2)$ (see Section 3), we can check if (A, B) is consistent with the same complexity. Then, by Lemma 2, the basic instances $(A_1, B_1), \ldots, (A_k, B_k)$ are computed in $O(|B|^3)$. Let $1 \leq i \leq k$. Using Lemma 1 and the fact that (A_i, B_i) is basic, s_i is computed in

$$O\left(|B_i|^2 \left(\frac{|B_i|}{|A_i|}\right)^{\mathrm{div}(\mathrm{lcm}L(A_i))}\right).$$

By Lemma 2, we have $|A_i| = |A|$, $|B_i| \leq |B|$ and $\mathrm{lcm}L(A_i)$ divides $\mathrm{lcm}L(A)$. We can thus simplify the expression, with a loss of precision, as follows: s_i is computed in

$$O\left(|B|^2 \left(\frac{|B|}{|A|}\right)^{\mathrm{div}(\mathrm{lcm}L(A))}\right).$$

Since $k \leq |B|$ we obtain the global time-complexity of the statement. □

The rest of the section is devoted to the proof of Lemma 2.

4.1 Split

The decomposition lemma involves two operations. The first is the split, the second is the reduction. In this subsection, we treat the split, the reduction is the subject of the next subsection.

A *split* of an instance (A, B) is a couple of instances $(A, B_1), (A, B_2)$ such that $B = B_1 + B_2$; it is *consistent* if the two instances are. A key observation is that if $(A, B_1), (A, B_2)$ is a split of (A, B) then

$$\mathrm{Sol}(A, B_1) + \mathrm{Sol}(A, B_2) \subseteq \mathrm{Sol}(A, B) \tag{11}$$

Indeed, if $AX_1 = B_1$ and $AX_2 = B_2$ then $A(X_1 + X_2) = AX_1 + AX_2 = B_1 + B_2 = B$. If the inclusion above is an equality, we say that the split is *perfect*. We can then reconstruct the solutions of (A, B) from that of (A, B_1) and (A, B_2), which is obviously interesting from an algorithmic point of view.

Here is a simple sufficient condition for an instance to admit a consistent perfect split.

Lemma 3. *Let $(A, B_1), (A, B_2)$ be a split of a consistent instance (A, B). Suppose that $L(B_1)$ and $L(B_2)$ are disjoint, and suppose that $L(A, B_1), L(A, B_2)$ is a partition of $L(A, B)$. Then $(A, B_1), (A, B_2)$ is a consistent perfect split of (A, B).*

Proof. For $i = 1, 2$, let $L_i = L(A, B_i)$. By definition, $L(A) \vee L_i \subseteq L(B_i)$. Since (A, B) is consistent, and $L(A, B) = L_1 \cup L_2$ we have

$$L(B) = L(A) \vee L(A, B) = (L(A) \vee L_1) \cup (L(A) \vee L_2) \subseteq L(B_1) \cup L(B_2) = L(B).$$

Thus the above inclusion is actually an equality. Since $L(A) \vee L_i \subseteq L(B_i)$ and $L(B_1) \cap L(B_2) = \emptyset$ we deduce that $L(A) \vee L_i = L(B_i)$. Hence the split $(A, B_1), (A, B_2)$ is consistent. It remains to prove that it is perfect.

As already mentioned, if X_1, X_2 are solutions of $(A, B_1), (A, B_2)$ then

$$A(X_1 + X_2) = AX_1 + AX_2 = B_1 + B_2 = B,$$

thus $X = X_1 + X_2$ is a solution of (A, B).

Conversely, let X be a solution of (A, B) and let us prove that $X = X_1 + X_2$ for some solutions X_1, X_2 of $(A, B_1), (A, B_2)$. Let X_i be the sum of cycles which contains exactly all the cycles of X whose length is in L_i. Since $L_1 \cap L_2 = \emptyset$, we have $L(X_1) \cap L(X_2) = \emptyset$. By (9) we have $L(X) \subseteq L(A, B)$ and since L_1, L_2 is a partition of $L(A, B)$ we have $X = X_1 + X_2$. Using (5) we obtain

$$\begin{aligned} L(B) = L(AX) &= L(A) \vee L(X) \\ &= (L(A) \vee L(X_1)) \cup (L(A) \vee L(X_2)) = L(AX_1) \cup L(AX_2). \end{aligned}$$

For $i = 1, 2$, we have $L(X_i) \subseteq L_i$ and thus, using (5),

$$L(AX_i) = L(A) \vee L(X_i) \subseteq L(A) \vee L_i = L(B_i).$$

Since $L(B_1), L(B_2)$ is a partition of $L(B)$, we deduce that $L(AX_i) = L(B_i)$ for $i = 1, 2$. Hence, to prove that X_i is a solution of (A, B_i), it is sufficient to prove that $AX_i(b) = B_i(b)$ for all $b \in L(B_i)$. So let $b \in L(B_i)$. Since $L(B_1) \cap L(B_2) = \emptyset$, for every $a \in L(A)$ and $x \in L(X)$ with $a \vee x = b$ we have $x \in L_i$ and thus $x \in L(X_i)$. Furthermore, since $L(X_1) \cap L(X_2) = \emptyset$, we have

$X_i(x) = X(x)$. Consequently,

$$\begin{aligned}(AX_i)(b) = \frac{1}{b}\sum_{\substack{a\in L(A)\\ x\in L(X_i)\\ a\vee x=b}} aA(a)xX_i(x) &= \frac{1}{b}\sum_{\substack{a\in L(A)\\ x\in L(X_i)\\ a\vee x=b}} aA(a)xX(x)\\ &= \frac{1}{b}\sum_{\substack{a\in L(A)\\ x\in L(X)\\ a\vee x=b}} aA(a)xX(x) = B(b) = B_i(b).\end{aligned}$$

□

We now prove that a non-basic consistent instance (A, B) with $\gcd L(A, B) = 1$ admits a consistent perfect split; we will then prove, with the reduction operation (next subsection), that, in some sense, the condition on the gcd can be suppressed, leading to a consistent prefect split of every non-basic consistent instance.

Lemma 4. *Let (A, B) be a non-basic consistent instance with $\gcd L(A, B) = 1$. Let $b \in L(B)$ and a prime p such that $\nu_p(b) > \nu_p(a)$ for all $a \in L(A)$; b and p exist since (A, B) is not basic. Let B_1 be the sum of cycles which contains exactly all the cycles of B whose length ℓ satisfies $\nu_p(\ell) = \nu_p(b)$, and let $B_2 = B - B_1$. Then $(A, B_1), (A, B_2)$ is a consistent perfect split of (A, B).*

Proof. By construction we have $L(B_1) \cap L(B_2) = \emptyset$. Let $x \in L(A, B)$. If $\nu_p(x) = \nu_p(b)$ then, for all $\ell \in (L(A) \vee x)$ we have $\nu_p(\ell) = \nu_p(b)$. Since $\ell \in L(B)$ we deduce that $\ell \in L(B_1)$. Consequently, $x \in L(A, B_1)$. If $\nu_p(x) \neq \nu_p(b)$ then, for all $\ell \in (L(A) \vee x)$ we have $\nu_p(\ell) \neq \nu_p(b)$. Since $\ell \in L(B)$ we deduce that $\ell \in L(B_2)$. Consequently, $x \in L(A, B_2)$. This proves that $L(A, B_1), L(A, B_2)$ is a partition of $L(A, B)$. This also proves that $p \mid \gcd L(A, B_1)$, and since $\gcd L(A, B) = 1$ we deduce that $L(A, B_2) \neq \emptyset$, and thus B_2 is non-empty. So $(A, B_1), (A, B_2)$ is a split of (A, B), which is consistent and perfect by Lemma 3. □

The example below illustrates this lemma.

Example 5. *Let $A = C_6$ and $B = 3C_6 + 8C_{12}$. Then (A, B) is consistent, but obviously not basic, and $\gcd L(A, B) = 1$ (see Ex. 2). Applying Lemma 4 with $b = 12$ and $p = 2$ we obtain the consistent prefect split $(A, B_1), (A, B_2)$ with $B_1 = 8C_{12}$ and $B_2 = 3C_6$. Since $L(A, B_1) = \{4, 12\}$ we have*

$$\begin{aligned}AX_1 = B_1 &\iff C_6\big(X_1(4)C_4 + X_1(12)C_{12}\big) = 8C_{12}\\ &\iff 2X_1(4)C_{12} + 6X_1(12)C_{12} = 8C_{12}\\ &\iff 2X_1(4) + 6X_1(12) = 8.\end{aligned}$$

Thus each solution X_1 to (A, B_1) corresponds to a partition of 8 with parts in $\{2, 6\}$: these are $2 + 2 + 2 + 2$ and $2 + 6$, giving the following two solutions:

$$\begin{aligned}&4C_4\\ &C_4 + C_{12}.\end{aligned}$$

Since $L(A, B_2) = \{1, 2, 3, 6\}$ we have

$$\begin{aligned}AX_2 = B_2 &\iff C_6\big(X_2(1)C_1 + X_2(2)C_2 + X_2(3)C_3 + X_2(6)C_6\big) = 3C_6\\ &\iff X_2(1)C_6 + 2X_2(2)C_6 + 3X_2(3)C_6 + 6X_2(6)C_6 = 3C_6\\ &\iff X_2(1) + 2X_2(2) + 3X_2(3) + 6X_2(6) = 3.\end{aligned}$$

Thus each solution X_2 to (A, B_2) corresponds to a partition of 3 with parts in $\{1, 2, 3, 6\}$: these are $1+1+1$, $1+2$, and 3, giving the following three solutions:

$$3C_1$$
$$C_1 + C_2$$
$$C_3.$$

Since the split is perfect, we have $\mathrm{Sol}(A, B) = \mathrm{Sol}(A, B_1) + \mathrm{Sol}(A, B_2)$ and thus (A, B) has the following 6 solutions:

$$3C_1 + 4C_4$$
$$3C_1 + C_4 + C_{12}$$
$$C_1 + C_2 + 4C_4$$
$$C_1 + C_2 + C_4 + C_{12}$$
$$C_3 + 4C_4$$
$$C_3 + C_4 + C_{12}.$$

4.2 Reduction

We say that an instance (A, B) is *compact* if $\gcd L(A, B) = 1$. This condition is used in Lemma 4 to split non-basic instances, but in this section we show that every instance can be reduced to an "equivalent" compact instance, which can then be splitted. For this reduction, we need two operations.

The first operation is the cycle length division. The *cycle length division* of a sum of cycles A by a positive integer p, denoted $A \oslash p$, is the sum of cycles obtained from A by deleting every cycle whose length is not a multiple of p, and by dividing by p the length of the remaining cycles; in other words: for all $a \geq 1$,

$$(A \oslash p)(a) = A(pa).$$

Note that $L(A \oslash p) = L(A)/p$ and if $p \mid L(A)$ then $L(A) = pL(A \oslash p)$.

Example 6.

$$(2C_1 + 3C_3 + 5C_4 + 7C_6) \oslash 3 = 3C_1 + 7C_2.$$

The second operation is the contraction. The *contraction* of a sum of cycles A by a positive integer p, denoted $A \boxslash p$, is inductively defined as follows. Firstly, $A \boxslash 1 = A$. Secondly, if p is prime then, for all $a \geq 1$,

$$(A \boxslash p)(a) = \begin{cases} A(a) + pA(pa) & \text{if } p \nmid a \\ pA(pa) & \text{otherwise.} \end{cases} \tag{12}$$

This operation transforms each cycle of length pa into p cycles of length a (and thus keeps the number of vertices unchanged). Note that $L(A \boxslash p)$ is the set of integers a such that either $a \in L(A)$ and $p \nmid a$ or $pa \in L(A)$. Finally, if p is composite, we take the smallest prime q that divides p and set

$$A \boxslash p = (A \boxslash p/q) \boxslash q.$$

Example 7.

$$\begin{aligned} (2C_1 + 3C_3 + 5C_4 + 7C_6) \boxslash 3 &= 2C_1 + 9C_1 + 5C_4 + 21C_2 \\ &= 11C_1 + 5C_4 + 21C_2. \end{aligned}$$

$$\begin{aligned} (2C_1 + 3C_3 + 5C_4 + 7C_6) \boxslash 6 &= ((2C_1 + 3C_3 + 5C_4 + 7C_6) \boxslash 3) \boxslash 2 \\ &= (11C_1 + 5C_4 + 21C_2) \boxslash 2 \\ &= (11C_1 + 10C_2 + 42C_1) \\ &= (53C_1 + 10C_2). \end{aligned}$$

Note that, for every positive integers p, q, we have

$$(A \otimes p) \otimes q = A \otimes pq, \quad (A \oslash p) \oslash q = A \oslash pq, \quad (A \boxslash p) \boxslash q = A \boxslash pq. \tag{13}$$

The first two equalities are obvious. The third results from the following easy to check commutativity property: $(A \boxslash p) \boxslash q = (A \boxslash q) \boxslash p$ when p and q are primes.

We can finally define the reduction operation. The *reduction* of a consistent instance (A, B) is

$$(A \boxslash \gcd L(A, B), B \oslash \gcd L(A, B)).$$

The main property of this reduction, stated below, is that it preserves consistency and results in compact instances without loss of information concerning the solutions.

Lemma 5. *Let (A, B) be a consistent instance. Its reduction (A', B') is consistent and satisfies*

$$L(A', B') = L(A, B)/\gcd L(A, B) \quad \textit{and} \quad \mathrm{Sol}(A, B) = \mathrm{Sol}(A', B') \otimes \gcd L(A, B).$$

Before proving this lemma, we illustrate it with an example.

Example 8. *The support of $(C_6, 8C_{12})$ is $\{4, 12\}$ and $\mathrm{lcm}\{4, 12\} = 4$. We have*

$$C_6 \boxslash 4 = (C_6 \boxslash 2) \boxslash 2 = 2C_3 \boxslash 2 = 2C_3$$

and

$$8C_{12} \oslash 4 = 8C_3.$$

Thus the reduction of $(C_6, 8C_{12})$ is $(2C_3, 8C_3)$. Hence the support of the reduction is $\{1, 3\} = \{4, 12\}/4$, as predicted by Lemma 5, so the reduction is compact, and we have

$$\begin{aligned} 2C_3X = 8C_3 &\iff 2C_3\big(X(1)C_1 + X(3)C_3\big) = 8C_3 \\ &\iff 2X(1)C_3 + 6X(3)C_3 = 8C_3 \\ &\iff 2X(1) + 6X(3) = 8. \end{aligned}$$

Thus each solution X corresponds to a partition of 8 with parts in $\{2, 6\}$: these are $2{+}2{+}2{+}2$ and $2 + 6$, giving the following two solutions:

$$\begin{array}{l} 4C_1 \\ C_1 + C_3. \end{array}$$

By Lemma 5, we have $\mathrm{Sol}(C_6, 8C_{12}) = \mathrm{Sol}(2C_3, 8C_3) \otimes 4$. Hence the solutions of $(C_6, 8C_{12})$ are

$$\begin{array}{lcl} (4C_1) \otimes 4 & = & 4C_4 \\ (C_1 + C_3) \otimes 4 & = & C_4 + C_{12} \end{array}$$

which is consistent with the direct computation given in Ex. 5.

The rest of this subsection is devoted to the proof of Lemma 5. We first show that the cycle length division $\oslash$ is the inverse of the cycle length multiplication $\otimes$.

Lemma 6. *Let A be a sum of cycles and let p be a positive integer. Then $(A \otimes p) \oslash p = A$, and if $p \mid L(A)$ then $(A \oslash p) \otimes p = A$.*

Proof. For all $a \geq 1$, we have $((A \otimes p) \oslash p)(a) = (A \otimes p)(pa) = A(a)$. Suppose that $p \mid L(A)$ and let $a \geq 1$. If $p \nmid a$ then $A(a) = 0$ and $((A \oslash p) \otimes p)(a) = 0$ (since $p \mid L((A \oslash p) \otimes p)$). If $p \mid a$ then $((A \oslash p) \otimes p)(a) = (A \oslash p)(a/p) = A(a)$. □

The key argument follows.

Lemma 7. *Let A, X be sums of cycles, and suppose that $p \mid L(X)$ for some prime p. Then*

$$(A \boxslash p)(X \oslash p) = (AX) \oslash p.$$

Proof. Let $A' = A \boxslash p$ and $X' = X \oslash p$. We have to prove that $A'X' = AX \oslash p$, that is, for all $\ell \geq 1$, $A'X'(\ell) = (AX \oslash p)(\ell) = AX(p\ell)$. Let us fix $\ell \geq 1$. We have

$$p\ell A'X'(\ell) = \sum_{\substack{a,x \\ a \vee x = \ell}} paA'(a)xX'(x) = \sum_{\substack{a,x \\ a \vee x = \ell}} aA'(a)pxX(px).$$

Denoting by Ω the set of couples $(a,x) \in \mathbb{N}^2$ with $p \mid x$ and $a \vee \frac{x}{p} = \ell$, we obtain

$$p\ell A'X'(\ell) = \sum_{(a,x)\in\Omega} aA'(a)xX(x).$$

By splinting the sum according to the definition of A' we obtain

$$\begin{aligned} p\ell A'X'(\ell) &= \sum_{\substack{(a,x)\in\Omega \\ p\nmid a}} \big(aA(a) + paA(pa)\big)xX(x) + \sum_{\substack{(a,x)\in\Omega \\ p\mid a}} paA(pa)xX(x) \\ &= \sum_{\substack{(a,x)\in\Omega \\ p\nmid a}} aA(a)xX(x) + \sum_{\substack{(a,x)\in\Omega \\ p\nmid a}} paA(pa)xX(x) + \sum_{\substack{(a,x)\in\Omega \\ p\mid a}} paA(pa)xX(x). \end{aligned}$$

Denoting by Ω' the set of $(a,x) \in \mathbb{N}^2$ with $p \mid x$, $p \mid a$ and $\frac{a}{p} \vee \frac{x}{p} = \ell$, we obtain

$$p\ell A'X'(\ell) = \sum_{\substack{(a,x)\in\Omega \\ p\nmid a}} aA(a)xX(x) + \sum_{\substack{(a,x)\in\Omega' \\ p\nmid \frac{a}{p}}} aA(a)xX(x) + \sum_{\substack{(a,x)\in\Omega' \\ p\mid \frac{a}{p}}} aA(a)xX(x).$$

If $p \nmid a$ then $a \vee \frac{x}{p} = \ell$ iff $a \vee x = p\ell$; and $\frac{a}{p} \vee \frac{x}{p} = \ell$ iff $a \vee x = p\ell$. Consequently

$$p\ell A'X'(\ell) = \sum_{\substack{a,x \\ p\mid x \\ a\vee x = p\ell}} aA(a)xX(x).$$

Since $p \mid L(X)$, if $p \nmid x$ then $X(x) = 0$, so

$$p\ell A'X'(\ell) = \sum_{\substack{a,x \\ a\vee x = p\ell}} aA(a)xX(x) = p\ell AX(p\ell).$$

Thus $A'X'(\ell) = AX(p\ell)$ for all $\ell \geq 0$, as desired. □

We obtain that every non compact instance can be partially reduced by a prime.

Lemma 8. *Let (A,B) be a consistent instance, and suppose $p \mid L(A,B)$ for some prime p. Then $(A \boxslash p, B \oslash p)$ is consistent with support $L(A,B)/p$, and*

$$\mathrm{Sol}(A,B) = \mathrm{Sol}(A \boxslash p, B \oslash p) \otimes p. \tag{14}$$

Proof. Let $A' = A \boxslash p$ and $B' = B \oslash p$. We first prove (14). Let X be a solution of (A, B). By (9) we have $L(X) \subseteq L(A, B)$ and since $p \mid L(A, B)$ we have $p \mid L(X)$. Hence, by Lemma 7, $A'(X \oslash p) = AX \oslash p = B \oslash p = B'$, that is, $X \oslash p$ is a solution of (A', B'). Since $p \mid L(X)$, by Lemma 6 we have $(X \oslash p) \otimes p = X$ and thus $X \in \mathrm{Sol}(A'\, B') \otimes p$.

We now prove the converse direction. Let X' be a solution of (A', B'), and let $X = X' \otimes p$. We have to prove that X is a solution of (A, B). By Lemma 6 we have $X \oslash p = X'$ thus $A'(X \oslash p) = B' = B \oslash p$. Since $p \mid L(X)$, by Lemma 7, we have $A'(X \oslash p) = (AX) \oslash p$. Thus $(AX) \oslash p = B \oslash p$. Since $p \mid L(A, B)$ and (A, B) is consistent, we have $p \mid L(B)$. Since $X = X' \otimes p$ we obviously have $p \mid L(X)$. Thus p divides $L(A) \vee L(X) = L(AX)$. Using Lemma 6 we obtain $AX = ((AX) \oslash p) \otimes p = (B \oslash p) \otimes p = B$. Thus X is a solution of (A, B). This proves (14).

We now prove that $L(A, B)/p \subseteq L(A', B')$. For that, we fix $x \in L(A, B)/p$, and we prove that $a \vee x$ is in $L(B')$ for any $a \in L(A')$. Indeed, if $pa \in L(A)$ then $pa \vee px = b$ for some $b \in L(B)$ and we deduce that $a \vee x = b/p \in L(B')$. If $pa \notin L(A)$, then $p \nmid a$ and $a \in L(A)$. Thus $a \vee px = b$ for some $b \in L(B)$ and since $p \nmid a$ we deduce that $a \vee x = b/p \in L(B')$.

We now prove the converse inclusion. For that, we fix $x \in L(A', B')$, and we prove that $a \vee px$ is in $L(B)$ for any $a \in L(A)$. Indeed, if $p \mid a$ then $a/p \in L(A')$ and thus $(a/p) \vee x = b$ for some $b \in L(B')$ so that $a \vee px = pb \in L(B)$. If $p \nmid a$ then $a \in L(A')$ and thus $a \vee x = b$ for some $b \in L(B')$, and since $p \nmid a$ we have $a \vee px = pb \in L(B)$.

We finally prove that (A', B') is consistent. Since $L(A') \vee L(A', B') \subseteq L(B')$, we only have to prove the converse inclusion. Let $b \in L(B')$. Then $pb \in L(B)$ and since (A, B) is consistent, there is $a \in L(A)$ and $x \in L(A, B)$ with $a \vee x = pb$. Hence $x/p \in L(A', B')$. If $p \mid a$ then $(a/p) \vee (x/p) = b$ and we are done since $a/p \in L(A')$. If $p \nmid a$ then $a \vee (x/p) = b$ and we are done since $a \in L(A')$. □

The proof of Lemma 5 is then obtained by applying several times the previous lemma.

Proof of Lemma 5. Let $d = \gcd L(A, B)$. Suppose that $d > 1$ since otherwise there is nothing to prove. Let us write d as the product of $k \geq 1$ primes, not necessarily distinct, say $d = p_1 p_2 \dots p_k$ with $p_1 \leq p_2 \leq \dots \leq p_k$. Let $A_0 = A$, $B_0 = B$ and, for $1 \leq \ell \leq k$, let $A_\ell = A_{\ell-1} \boxslash p_\ell$ and $B_\ell = B_{\ell-1} \oslash p_\ell$. By Lemma 8, (A_ℓ, B_ℓ) is a consistent instance and $\mathrm{Sol}(A_{\ell-1}, B_{\ell-1}) = \mathrm{Sol}(A_\ell, B_\ell) \otimes p_\ell$. By (13) we have $A_k = A \boxslash d$, $B_k = B \boxslash d$, $L(A_k, B_k) = L(A, B)/d$ and $\mathrm{Sol}(A, B) = \mathrm{Sol}(A_k, B_k) \otimes d$. □

4.3 Proof of the decomposition lemma

In this subsection, we finally prove Lemma 2. It is actually more convenient to prove something stronger. We start with a definition. Let (A, B) be a consistent instance. A *decomposition* of (A, B) is a list $\mathcal{L}$ of triples (A_i, B_i, p_i), $1 \leq i \leq k$, such that

- (A_i, B_i) is a compact and consistent instance, and p_i is a positive integer,
- $|A_i| = |A|$ and $\mathrm{lcm} L(A_i) \mid \mathrm{lcm} L(A)$,
- $|B_1| + \dots + |B_k| \leq |B|$,
- $\mathrm{Sol}(A, B) = (\mathrm{Sol}(A_1, B_1) \otimes p_1) + \dots + (\mathrm{Sol}(A_k, B_k) \otimes p_k)$.

We call k the *length* of $\mathcal{L}$; note that by the third point, $k \leq |B|$. Furthermore, we say that $\mathcal{L}$ is *basic* if (A_i, B_i) is basic for all $1 \leq i \leq k$. We will prove that we can compute in $O(|A||B|^2)$ a basic decomposition, which clearly implies Lemma 2. For that we first prove that if (A, B) has

a non-basic decomposition, we can obtain a longer decomposition by partitioning a non-basic instance (Lemma 4) and then reducing its parts (Lemma 5).

Lemma 9. *There is an algorithm that, given a consistent instance (A, B) and a non-basic decomposition $\mathcal{L}$ of (A, B) of length k, computes in $O(|B|^2)$ a decomposition $\mathcal{L}'$ of (A, B) of length $k + 1$.*

Proof. The algorithm is as follows. Let $(A_1, B_1, p_1), \ldots, (A_k, B_k, p_k)$ be the triples of $\mathcal{L}$. We can test if (A_i, B_i, p_i) is basic or not in $O(|A_i| + |B_i|) = O(|B_i|)$. Since $k \leq |B|$ (by the third point of the definition of a decomposition), and since $\mathcal{L}$ is not basic, we can thus find a non-basic instance (A_i, B_i) in $O(|B|^2)$. Since (A_i, B_i) is compact and consistent, by Lemma 4, it has a consistent perfect split, say $(A_i, B_{i1}), (A_i, B_{i2})$. For $j = 1, 2$, we compute in $O(|B|^2)$ the integer $p_{ij} = \gcd L(A_i, B_{ij})$ and the reduction $(A_{ij}, B'_{ij}) = (A_i \boxslash p_{ij}, B_{ij} \oslash p_{ij})$ of (A_i, B_{ij}). Finally, we output the list $\mathcal{L}'$ of length $k+1$ obtained from $\mathcal{L}$ by deleting (A_i, B_i, p_i) and adding $(A_{i1}, B'_{i1}, p_i p_{i1})$ and $(A_{i2}, B'_{i2}, p_i p_{i2})$. So the running time is $O(|B|^2)$.

Let us prove that $\mathcal{L}'$ is a decomposition. By Lemma 5, (A_{ij}, B'_{ij}) is compact and consistent. Furthermore, $|A_{ij}| = |A_i| = |A|$ and since A_{ij} is a contraction of A_i, each member of $L(A_{ij})$ divides some member of $L(A_i)$, thus $\mathrm{lcm} L(A_{ij})$ divides $\mathrm{lcm} L(A_i)$, which divides $\mathrm{lcm} L(A)$. Thus $\mathrm{lcm} L(A_{ij})$ divides $\mathrm{lcm} L(A)$. Next, since $|B'_{i1}| + |B'_{i2}| \leq |B_{i1}| + |B_{i2}| = |B_i|$, the third point of the definition of a decomposition is preserved. Finally, by Lemma 5, $\mathrm{Sol}(A_i, B_{ij}) = \mathrm{Sol}(A_{ij}, B'_{ij}) \otimes p_{ij}$. Since the split $(A_i, B_{i1}), (A_i, B_{i2})$ is perfect, we obtain

$$\begin{aligned} \mathrm{Sol}(A_i, B_i) \otimes p_i &= (\mathrm{Sol}(A_i, B_{i1}) + \mathrm{Sol}(A_i, B_{i2})) \otimes p_i \\ &= (\mathrm{Sol}(A_i, B_{i1}) \otimes p_i) + (\mathrm{Sol}(A_i, B_{i2}) \otimes p_i) \\ &= (\mathrm{Sol}(A_{i1}, B'_{i1}) \otimes p_i p_{i1}) + (\mathrm{Sol}(A_{i2}, B'_{i2}) \otimes p_i p_{i2}) \end{aligned}$$

and this proves that the last point of the definition of a decomposition is preserved. So $\mathcal{L}'$ is indeed a decomposition of (A, B). □

Iterating the previous lemma, we get the following, which implies Lemma 2.

Lemma 10. *There is an algorithm that, given a consistent instance (A, B), computes in $O(|B|^3)$ a basic decomposition of (A, B).*

Proof. The algorithm constructs recursively a list $\mathcal{L}_1, \ldots, \mathcal{L}_{|B|}$ of decompositions of (A, B), where the length of $\mathcal{L}_r$ is at most r, and output $\mathcal{L}_{|B|}$. First we compute $\mathcal{L}_1 = \{(A \boxslash d, B \oslash d, d)\}$ where $d = \gcd L(A, B)$ in $O(|A||B|)$; by Lemma 5, $\mathcal{L}_1$ is a decomposition of length one. Now, suppose that the decomposition $\mathcal{L}_r$ of length $k \leq r < |B|$ has already been computed. If $\mathcal{L}_r$ is basic, we set $\mathcal{L}_{r+1} = \mathcal{L}_r$. Otherwise, using Lemma 9, we compute in $O(|B|^2)$ a decomposition $\mathcal{L}_{r+1}$ of length $k + 1$. Hence the running time is $O(|B|^3)$. It remains to prove that $\mathcal{L}_{|B|}$ is basic. If $\mathcal{L}_r = \mathcal{L}_{r+1}$ for some $r < |B|$ then $\mathcal{L}_r$ is basic and $\mathcal{L}_s = \mathcal{L}_r$ for all $r < s \leq |B|$ thus $\mathcal{L}_{|B|}$ is basic. Otherwise, $\mathcal{L}_1, \ldots, \mathcal{L}_{|B|}$ are all distinct thus the length of $\mathcal{L}_{|B|}$ is $|B|$. If $\mathcal{L}_{|B|}$ is not basic, by Lemma 9, (A, B) has a decomposition of length $|B| + 1$, a contradiction. Thus $\mathcal{L}_{|B|}$ is basic. □

Example 9. *Combining Ex. 5 and 8, we get that the basic decomposition of*

$$(C_6, 3C_6 + 8C12)$$

is

$$(2C_3, 8C_3, 4), (C_6, 3C_6, 1)$$

and so

$$\mathrm{Sol}(C_6, 3C_6 + 8C12) = (\mathrm{Sol}(2C_3, 8C_3) \otimes 4) + \mathrm{Sol}(C_6, 3C_6).$$

This can be easily checked from Ex. 5 and 8.

5 Principal support

The *principal support* of an instance (A, B) is

$$L'(A, B) = L(A, B) \setminus L(B).$$

If X is a solution of (A, B), we have $L(X) \subseteq L(A, B)$. But actually, all the information on X is contained on the principal support: if X' is a solution of (A, B) with $X'(x) = X(x)$ for all $x \in L'(A, B)$, then $X' = X$. This is a consequence of the following lemma.

Lemma 11. *Let (A, B) be an instance, and let X be a sum of cycles with $L(X) \subseteq L'(A, B)$. There exists at most one sum of cycles Y with $L(Y) \subseteq L(B)$ such that $A(X + Y) = B$, and there exists an algorithm, with running time in $O(|B|^3)$, that decides if Y exists, and outputs it if it exists.*

Proof. If AX is not contained in B, there is certainly no solution containing X, and if $AX = B$ we must take $Y = \emptyset$. So suppose that AX is strictly contained in B. Let us compute a sequence $(B_0, Y_0, b_0), (B_1, Y_1, b_1), \ldots, (B_k, Y_k, b_k)$ as follows. First we set $B_0 = B - AX$ and $Y_0 = \emptyset$ and $b_0 = 0$. Then, (B_i, Y_i, b_i) having been already computed, we do the following:

- If $B_i = \emptyset$, then we stop the algorithm.
- Let b be the minimum of $L(B_i)$.
- If AC_b is not contained in B_i, then we stop the algorithm.
- Otherwise, we set $B_{i+1} = B_i - AC_b$ and $Y_{i+1} = Y_i + C_b$ and $b_{i+1} = b$.

We output Y_k if $B_k = \emptyset$, and "there is no Y with $L(Y) \subseteq L(B)$ such that $A(X + Y) = B$" otherwise. Since each step can be done in $O(|B|^2)$, and since $k \leq |B|$, the total running time is in $O(|B|^3)$. It is easy to check, by induction on i, that

$$B_{i+1} = B_0 - AY_{i+1}. \tag{15}$$

So $B_k = \emptyset$ is equivalent to $AY_k = B_0$, which is equivalent to $A(X + Y_k) = B$. Hence it remains to prove the uniqueness of Y_k. So let Y be a sum of cycles with $L(Y) \subseteq L(B)$ such that $A(X + Y) = B$, which is equivalent to $AY = B_0$, and let us prove that $Y = Y_k$. It is sufficient to prove that $Y_i \subseteq Y$ for all $0 \leq i \leq k$, and that a strict inclusion implies $i < k$. The base case $Y_0 \subseteq Y$ is obvious. Suppose that $Y_i \subseteq Y$, with $0 \leq i \leq k$. By (15) we have

$$B_i = B_0 - AY_i = AY - AY_i = A(Y - Y_i).$$

If $B_i = \emptyset$ then $i = k$ and $Y_k = Y_i = Y$. Otherwise, since $B_i = A(Y - Y_i)$, the minimum b of $L(B_i)$ has a divisor $y \in L(Y - Y_i)$. Since $y \in L(B)$ and (A, B) has a solution, y has a divisor $a \in L(A)$. Then $y = a \vee y \in L(B_i)$, and since $y \mid b$ we have $y = b$ by the choice of b. Hence $b \in L(Y - Y_i)$ and since $A(Y - Y_i) = B_i$ we have $AC_b \subseteq B_i$. Consequently, $i < k$ and $Y_{i+1} = Y_i + C_b \subseteq Y$, completing the induction. □

Consequently, to compute all the solutions of an instance (A, B), it is enough to enumerate the sum of cycles X with $L(X) \subseteq L(A, B) \setminus L(B)$ with at most $|B|/|A|$ vertices, and to check if X can be extended to a global solution with the algorithm of Lemma 11. This gives the following refinement of the brute force approach on the support.

Brute force approach on the principal support. If $n = |B|/|A|$ is not an integer, return $\emptyset$. Otherwise, compute $L'(A, B)$. For all $1 \leq m \leq n$, enumerate all the partitions of m with parts in $L'(A, B)$, take the corresponding sum of cycles X, select those for which there exists a sum of cycles Y with $Y \subseteq L(B)$ such that $X + Y$ is a solution, and return the selected sums $X + Y$.

Since $L(A, B)$ can be computed in $O(|B|^2)$, $L'(A, B)$ can be computed with the same complexity. Next, to enumerate the partitions of $1 \leq m \leq n$ with parts in $L = L'(A, B)$, we proceed almost exactly as in Section 3. In such a partition, the number of occurrences of $x \in L$ is between 0 and $\lfloor n/x \rfloor$. Hence we can enumerate all the functions $f : L \to \mathbb{N}$ such that $f(x) \leq 1 + \lfloor n/x \rfloor$ for all $x \in L$, and select those which correspond to partitions such that $1 \leq \sum_{x \in L} xf(x) \leq n$. It is easy to show, by induction on $|L|$, that the number of such functions is at most $n^{|L|-1}$. As explained in Section 3, this enumeration can be done in $O(n^{|L|})$. Then, for each partition we take the corresponding sums of cycles X, and we use the algorithm of Lemma 11 to check in $O(|B|^3)$ if there exists a sum of cycles Y with $Y \subseteq L(B)$ such that $X + Y$ is a solution. The algorithm is correct thanks to this lemma, and the total complexity is

$$O\left(|B|^3 \left(\frac{|B|}{|A|}\right)^{|L(A,B)\setminus L(B)|}\right).$$

The interesting fact is that the size of the principal support is sometimes significantly smaller than the support, and may lead to polynomial algorithms for some particular classes of instances. We illustrate this with two simple examples.

Suppose first that (A, B) is a basic instance and that $L(A) \setminus \{1\}$ only contains primes. Then the set of solutions can be computed in $O(|B|^4)$, and there are at most $|B|/|A|$ solutions. Indeed, let $x \in L(A, B)$, $x > 1$. Let p be a prime that divides x. Since the instance is basic, x divides $\mathrm{lcm} L(A)$, which is the product of the primes contained in $L(A)$. We deduce that $p \in L(A)$, and thus $x = p \vee x \in L(B)$. This proves that $L(A, B) \setminus L(B) \subseteq \{1\}$ and thus the brute force approach on the principal support runs in $O(|B|^4)$, and the number of solutions is at most $|B|/|A|$.

Frustratingly, we were not able to find a polynomial algorithm to decide if $A \mid B$ when $L(A)$ is a primitive set of prime powers, that is, $L(A) = \{p_1^{\alpha_1}, \ldots, p_k^{\alpha_k}\}$ for distinct primes $p_1, \ldots, p_k$.

Suppose now that (A, B) is a consistent instance and that $1 \in L(B)$. By the easy lemma below, $L'(A, B) = \emptyset$, hence there is a unique solution which can be computed in $O(|B|^3)$ with the brute force approach on the principal support.

Lemma 12. *If (A, B) is a consistent instance, then*

$$L'(A, B) = L(A, B) \setminus \mathrm{Mult}(L(B)).$$

Proof. Suppose that there exists $x \in L(A, B)$ and $b \in L(B)$ with $b \mid x$. Since the instance is consistent, there exists $a \in L(A)$ such that $a \mid b$, and again because the instance is consistent we have $a \vee x \in L(B)$. But since $a \mid x$, we have $a \vee x = x$ and thus $x \in L(B)$. This proves that $L'(A, B) \subseteq L(A, B) \setminus \mathrm{Mult}(L(B))$, and the converse inclusion is obvious since $L(B) \subseteq \mathrm{Mult}(L(B))$. □

6 Concluding remarks

- Our main contribution is to show that, up to polynomial transformations, we can restrict the division problem for sums of cycles to basic compact and consistent instances (A, B)

(Lemma 10). By the basic property, the exponent term $|L(A,B)|$ in the complexity of the brute force approach on the support is bounded as a function of A only, namely $|L(A,B)| \leq \mathrm{div}(\mathrm{lcm}L(A))$, leading to a polynomial algorithm when A is fixed. But this bound does not show that the complexity is significantly better than the (very naïve) brute force approach, running in $|B|e^{O(\sqrt{n})}$ where $n = |B|/|A|$. The problem is that it is rather difficult to bound $|L(A,B)|$ according to n, and a real improvement would be obtained by showing that $|L(A,B)| = o(\sqrt{n})$. Actually, considering instead the brute force approach on the principal support, showing $|L'(A,B)| = o(\sqrt{n})$ would be sufficient. Anyway, the existence of an algorithm for the division problem for sums of cycles running in $\mathrm{poly}(|B|)e^{o(\sqrt{n})}$ remains an open problem.

- One way to better understand what makes the problem apparently difficult is to find polynomial algorithms for particular classes of instances. For instance, it is not difficult to prove that the problem is polynomial when $L(B)$ is a *chain*, that is, when $b \mid b'$ or $b' \mid b$ for all $b, b' \in L(B)$. This leads to consider the dual situation, where $L(B)$ is an *antichain* (called primitive set in number theory), that is, there is no distinct $b, b' \in L(B)$ such that $b \mid b'$. This case seems much more challenging. In that direction, we show that there is a polynomial algorithm when $L(A)$ is a set of primes (Section 5), but, frustratingly, we were not able to prove the same property when $L(A)$ is an antichain of prime powers. This case is particularly interesting since it is a typical situation where the bound $|L(A,B)| \leq \mathrm{div}(\mathrm{lcm}L(A))$ gives few information on the real size of $L(A,B)$.

- A principal drawback of our method is that, before using a brute force approach, the only information used to analyze an instance (A,B) is the support of A and B and the size $n = |B|/|A|$ of potential solutions: we do not consider the multiplicities of cycles in A or B, that is, the quantities $A(a)$ and $B(b)$ for $a \in L(A)$ and $b \in L(B)$. It is very likely that these quantities will have to be analyzed in order to make progress. To illustrate this, suppose that (A,B) is a consistent instance and that all the cycles in B have distinct lengths, that is, $B(b) = 1$ for all $b \in L(B)$. One easily checks that the decomposition preserves this property, thus we can also suppose that (A,B) is basic. Then the problem is very simple: either C_1 is the unique solution, or there is no solution. Indeed, suppose that $AX = B$. Let $x \in L(X)$, $a \in L(A)$ and $b = a \vee x$. If a and x are not coprime, that is $ax > b$, then, by (4), we have $B(b) \geq \frac{1}{b}aX(a)xX(x) > 1$, a contradiction. Thus any $x \in L(X)$ is coprime with any $a \in L(A)$. Suppose now that there exists $x \in L(X)$ with $x > 1$ and let p be a prime that divides x. Then $L(B)$ contains a multiple of p, and since the instance is basic, we deduce that p divides some $a \in L(A)$. But then a and x are not coprime, a contradiction. We deduce that $X = nC_1$ with $n = |B|/|A|$. But if $n > 1$ then, for any $a \in L(A)$, we have $B(a) \geq \frac{1}{a}aX(a)X(1) = X(a)X(1) = X(a)n > 1$, a contradiction. Thus $X = C_1$.

- Beyond sums of cycles, that there is a polynomial algorithm for the division problem inside the whole set of functional digraphs seems a challenging problem. In the spirit of this paper, it would be at least interesting to prove that, for every fixed functional digraph A, there is a polynomial algorithm to decide, given any functional digraph B, if A divides B. As a small step in this direction, using [29, Corollary 14], it seems rather easy to adapt our arguments to prove that, for every sum of cycles A and every functional digraph B, there is an algorithm to decide if A divides B which is polynomial when A is fixed and with a complexity very similar to that given in (2).

- We proved that, for every $\epsilon > 0$ and infinitely many n, there exists an instance (A,B) with $|A| = n$ and $|B| = n^2$ such that the number of (irreducible) solutions of (A,B) and the number of irreducible factorizations of B is at least (1), hence super-polynomial

in n. One may ask whether these lower bounds are far from the truth, and thus ask for non-trivial upper bounds. The only upper bound we give concerns the number of irreducible factorizations of a sum of cycles X with n vertices, which is at most $ne^{c_0\sqrt{n}}$. But this bound is certainly very loose since it is actually a bound on $p^\times(n)$, the number of ways of producing sums of cycles with n vertices from products of sums of cycles, each distinct from C_1 (the order of the factors in the product being irrelevant). It therefore seems that much more precise upper bounds could be obtained. Note that, from the proof that $p^\times(n) \leq ne^{c_0\sqrt{n}}$ (Appendix C), we easily deduce that we can enumerate in $e^{O(\sqrt{n})}$ the products of sums of cycles, each distinct from C_1, resulting in sums of cycles with n vertices. Consequently, the irreducibility decision problem for sums of cycles is sub-exponential. The arguments being very rough, huge progress seems possible for this decision problem, which is perhaps even more natural than the division problem.

Acknowledgments This work has been funded by the project ANR-24-CE48-7504 "ALARICE" and the HORIZON-MSCA-2022-SE-01 project 101131549 "Application-driven Challenges for Automata Networks and Complex Systems (ACANCOS)".

A Irreducible and prime sums of cycles

Recall that a sum of cycles X is *irreducible* if, for all sums of cycles A, B, $X = AB$ implies $A = C_1$ or $B = C_1$. For instance, it is an easy exercise to prove that a single cycle C_n is irreducible if and only if n is a prime power, and this property will be used many times in the following. Let $\mathrm{irred}(n)$ be the number of irreducible sums of cycles with n vertices, and let $\mathrm{red}(n) = p(n) - \mathrm{irred}(n)$ be the number of reducible sums of cycles with n vertices. We will prove that almost all sums of cycles are irreducible using a basic counting argument.

Proposition 1. $\mathrm{irred}(n)/p(n) \to 1$ *as* $n \to \infty$.

Proof. Let X be a reducible sum of cycles with n vertices. Then there exists two sums of cycles $A, B \neq C_1$ such that $AB = X$. Let $d = |A|$ and suppose, without loss, that $|A| \leq |B|$. Then $2 \leq d \leq \sqrt{n}$. Consequently,

$$\mathrm{red}(n) \leq \sum_{\substack{d|n \\ 2\leq d\leq\sqrt{n}}} p(d)p(n/d) \leq \sum_{\substack{d|n \\ 2\leq d\leq\sqrt{n}}} p(\lceil\sqrt{n}\rceil)p(\lceil n/2\rceil) \leq \sqrt{n}p(\lceil\sqrt{n}\rceil)p(\lceil n/2\rceil).$$

Let $\delta = \sqrt{1/2}$ and $0 < \epsilon < \frac{1-\delta}{1+\delta}c_0$. Using (7), for n large enough we have

$$\sqrt{n}p(\lceil\sqrt{n}\rceil)p(\lceil n/2\rceil) \quad \leq \quad \sqrt{n}e^{c_0\left(\sqrt{\lceil\sqrt{n}\rceil}+\sqrt{\lceil n/2\rceil}\right)} \leq e^{(c_0+\epsilon)\sqrt{n/2}} = e^{(c_0+\epsilon)\delta\sqrt{n}}.$$

Using again (7), we deduce that, for n large enough,

$$\frac{\mathrm{red}(n)}{p(n)} \leq \frac{e^{(c_0+\epsilon)\delta\sqrt{n}}}{e^{(c_0-\epsilon)\sqrt{n}}} = \frac{1}{e^{((c_0-\epsilon)-\delta(c_0+\epsilon))\sqrt{n}}}$$

By the choice of ϵ, we have $c_0 - \epsilon > (c_0+\epsilon)\delta$ so $\mathrm{red}(n)/p(n) \to 0$ as $n \to \infty$, and we are done. □

Recall that a functional digraph X is *prime* if $X \neq C_1$ and, for all functional digraphs A, B, $X \mid AB$ implies $X \mid A$ or $X \mid B$. While proving that the semiring of functional digraphs has no prime element is difficult [35], proving that the same is true for the semiring of sums of cycles is an easy exercise.

Proposition 2. *There is no prime element in the semiring of sums of cycles: there is no sum of cycles $X \neq C_1$ such that, for all sums of cycles A, B, $X \mid AB$ implies $X \mid A$ or $X \mid B$.*

Proof. There are four cases:

- Suppose that $X = xC_1$ for some integer $x \geq 2$. If x is not prime then X is reducible, and thus X is not prime. So suppose that $X = pC_1$ for some prime p. Then

$$pC_1 \cdot C_p = pC_p = C_p \cdot C_p.$$

  Since C_p is irreducible, we deduce that X is not prime.

- Suppose that $X = C_{p^\alpha}$ for some prime p and $\alpha \geq 1$. Then

$$C_{p^\alpha} \cdot pC_{p^{\alpha+1}} = p^{\alpha+1} C_{p^{\alpha+1}} = C_{p^{\alpha+1}} \cdot C_{p^{\alpha+1}}.$$

  Since $C_{p^{\alpha+1}}$ is irreducible, we deduce that X is not prime.

- Suppose now that $\mathrm{lcm} L(X)$ is a prime power p^α and $|X| > p^\alpha$. Then

$$X \cdot C_{p^\alpha} = |X| C_{p^\alpha} = C_{p^\alpha} \cdot |X| C_1.$$

  Since $|X| > p^\alpha$ and X does not divide $|X|C_1$, we deduce that X is not prime.

- In the other cases, $\ell = \mathrm{lcm} L(X) > 1$ and is not a prime power. Let $\ell = p_1^{\alpha_1} p_2^{\alpha_2} \cdots p_k^{\alpha_k}$ be the prime decomposition of ℓ; by hypothesis $k \geq 2$. We have

$$X \cdot C_\ell = |X| C_\ell = |X| C_{p_1^{\alpha_1}} \cdot C_{\ell / p_1^{\alpha_1}}.$$

  By (6), X does not divide $|X| C_{p_1^{\alpha_1}}$ since $L(X)$ contains a multiple of $p_2^{\alpha_2}$, and it does not divide $C_{\ell / p_1^{\alpha_1}}$ since $L(X)$ contains a multiple of $p_1^{\alpha_1}$.

□

B Instance with many irreducible solutions

Let $f(n)$ be the number of partitions of n with parts in $\mathrm{Div}(n)$. In this section, we show that the instance (C_n, nC_n) has exactly $f(n)$ solutions (Lemma 13). Hopefully, Bowman, Erdős and Odlyzko [1] obtained an accurate estimate of $\ln f(n)$ depending on n and the number of divisors of n: for every $\epsilon > 0$ and n large enough,

$$(1+\epsilon)\left(\frac{\mathrm{div}(n)}{2} - 1\right) \ln n \leq \ln f(n) \leq (1+\epsilon) \frac{\mathrm{div}(n)}{2} \ln n. \tag{16}$$

By choosing n with many divisors, we deduce from this that f grows faster than any polynomial: for every d, it exists infinitely many n such that $f(n) > n^d$ (Lemma 14). The number of solutions to the instance (C_n, nC_n) is, in that sense, super-polynomial. Actually, we show something stronger: (C_n, nC_n) has a super-polynomial number of irreducible solutions (Theorem 2) and, as an immediate consequence, nC_n has a super-polynomial number of irreducible factorizations (Corollary 1).

Lemma 13. *For all $n \geq 1$, we have $X \in \mathrm{Sol}(C_n, nCn)$ iff $|X| = n$ and $L(X) \subseteq \mathrm{Div}(n)$, so that*

$$\mathrm{sol}(C_n, nC_n) = f(n).$$

Proof. Suppose that X is a sum of cycles with n vertices and $L(X) \subseteq \mathrm{Div}(n)$. Since $C_d C_n = dC_n$ whenever $d \mid n$, we have $C_n \cdot X = |X| C_n = nC_n$. Conversely, if $C_n \cdot X = nC_n$ then X has n vertices and, by (6), we have $L(X) \subseteq \mathrm{Div}(L(nC_n)) = \mathrm{Div}(n)$. □

We now show that f is super-polynomial. Let us defined the *kth primorial number* as the product of the first k primes. If n is the kth primorial number, then it has 2^k divisors and it is well known (see e.g. [3]) that, for every $\epsilon > 0$ and k large enough,

$$k \ln k \leq \ln n \leq (1+\epsilon) k \ln k. \tag{17}$$

Lemma 14. *For every $\epsilon > 0$, if n is a sufficiently large primorial number, then*

$$f(n) \geq n^{2^{\frac{\ln n}{(1+\epsilon)\ln\ln n}}}.$$

Proof. Let $0 < \delta < \epsilon$, and let n be kth primorial number. By (17), for k large enough we have

$$k \geq \frac{\ln n}{(1+\delta)\ln k} \geq \frac{\ln n}{(1+\delta)\ln\ln n}.$$

Using (16), we obtain, for k large enough,

$$\frac{\ln f(n)}{\ln n} \geq 2^{k-1} \geq 2^{\frac{\ln n}{(1+\delta)\ln\ln n} - 1} \geq 2^{\frac{\ln n}{(1+\epsilon)\ln\ln n}}.$$

□

Theorem 2. *For every $\epsilon > 0$, if n is a sufficiently large primorial number, then the number of irreducible solutions to the instance (C_n, nC_n) is at least*

$$n^{2^{\frac{\ln n}{(1+\epsilon)\ln\ln n}}}.$$

Proof. Let $n = p_1 p_2 \cdots p_k$ be the kth primorial number. Let I be the set of sums of cycles X with n vertices such that $L(X) \subseteq \mathrm{Div}(n)$ and such that X has a unique cycle whose length is a multiple of p_k, and this cycle is precisely C_{p_k}. In other words, $X = C_{p_k} + Y$ for some sum of cycles Y with $n - p_k$ vertices satisfying $L(Y) \subseteq \mathrm{Div}(n/p_k)$. By Lemma 13,

$$I \subseteq \mathrm{Sol}(C_n, nC_n).$$

Let $X \in I$, and let us prove that it is irreducible. Let A, B be sums of cycles such that $X = AB$. Since X contains C_{p_k}, which is irreducible, at least one of A, B contains this cycle, say B without loss. If A contains $C_{p_k \ell}$ for some $\ell \geq 1$, then $X = AB$ contains $C_{p_k \ell} C_{p_k} = p_k C_{p_k \ell}$, so X contains at least p_k cycles whose length is a multiple of p_k, a contradiction. If A contains $C_{p_i \ell}$ for some $1 \leq i < k$ and $\ell \geq 1$, then $X = AB$ contains $C_{p_i \ell} C_{p_k} = C_{p_i p_k \ell}$ and this contradicts the fact that C_{p_k} is the unique cycle of X whose length is a multiple of p_k. Hence $L(A)$ has no multiple of p_i for $1 \leq i \leq k$. By (6), we have $L(A) \subseteq \mathrm{Div}(L(X)) \subseteq \mathrm{Div}(n)$, and we deduce that $L(A) = \{1\}$. So $A = aC_1$ for some $a \geq 1$. If $a \geq 2$ then $X = AB$ contains $aC_1 C_{p_k} = aC_{p_k}$ and this contradicts the fact that X has a unique cycle of length p_k. Consequently, $A = C_1$ and thus X is irreducible.

We now give a lower bound on $|I|$, which is exactly the number of partitions of $n - p_k$ with parts in $\mathrm{Div}(n/p_k)$. For $k \geq 2$ we have $n - p_k \geq n/p_k$ and thus

$$|I| = p_{\mathrm{Div}(n/p_k)}(n - p_k) \geq p_{\mathrm{Div}(n/p_k)}(n/p_k) = f(n/p_k).$$

Let $0 < \delta < \epsilon$. Since $p_k \le 2k \ln k$ (see e.g. [34]), using (17) we get $p_k \le 2 \ln n$ and thus $\ln p_k \le \ln \ln n + 1$. Since n/p_k is the $(k-1)$th primorial number, we deduce from Lemma 14 that, for k large enough,

$$\ln |I| \ge \ln f(n/p_k) \ge 2^{\frac{\ln n - \ln \ln n - 1}{(1+\delta) \ln \ln n}} (\ln n - \ln \ln n - 1) \ge 2^{\frac{\ln n}{(1+\epsilon) \ln \ln n}} \ln n.$$

□

The number of *irreducible factorizations* of a sum of cycles X is the number of ways of expressing X as a product of irreducible sums of cycles, the order of the factors in the product being irrelevant.

Corollary 1. *For every $\epsilon > 0$, if n is a sufficiently large primorial number, then the number of irreducible factorizations of nC_n is at least*

$$n^{2^{\frac{\ln n}{(1+\epsilon) \ln \ln n}}}.$$

Proof. Let $n = p_1 p_2 \cdots p_k$ be the kth primorial number. Then C_n has a unique irreducible factorization, which is $C_{p_1} C_{p_2} \cdots C_{p_k}$. Let X be an irreducible sum of cycles such that $C_n \cdot X = nC_n$. Then $C_{p_1} C_{p_2} \cdots C_{p_k} \cdot X$ is an irreducible factorization of nC_n, and we are done using Theorem 2. □

C Products of sums of cycles resulting in n vertices

For $n \ge 2$, let $P^*(n)$ be the set of sequences of integers $(a_1, a_2, \ldots, a_k)$ such that $2 \le a_1 \le a_2 \le \cdots \le a_k$ and $a_1 a_2 \cdots a_k = n$. Thus $p^*(n) = |P^*(n)|$ is the number of ways of expressing n as the product of positive integers, each distinct from 1, the order of the factors in the product being irrelevant. Dodd and Mattics [10] proved that

$$p^*(n) \le n. \tag{18}$$

We will consider the same concept for sums of cycles. For $n \ge 2$, let $P^\times(n)$ be the set of sequences of sums of cycles $(A_1, A_2, \ldots, A_k)$ such that $2 \le |A_1| \le |A_2| \le \cdots \le |A_k|$ and $|A_1 A_2 \cdots A_k| = n$. Thus $p^\times(n) = |P^\times(n)|$ is the number of ways of producing sums of cycles with n vertices from products of sums of cycles, each distinct from C_1, the order of the factors in the product being irrelevant.

Proposition 3. *For all $n \ge 2$,*

$$p^\times(n) \le n e^{c_0 \sqrt{n}}.$$

Proof. Let $(A_1, A_2, \ldots, A_k)$ be a sequence in $P^\times(n)$. Let us call $(|A_1|, \ldots, |A_k|)$ the profile sequence; it belongs to $P^*(n)$. Let $(a_1, \ldots, a_k)$ be a sequence in $P^*(n)$, and let σ be the number of sequences in $P^\times(n)$ with this profile. Hence $\sigma = p(a_1) p(a_2) \cdots p(a_k)$. We will prove that $\sigma \le e^{c_0 \sqrt{n}}$. Since, by (18), there are at most n possible profiles, this proves the theorem.

Using (7), we have

$$\sigma \le e^{c_0 \sum_{i=1}^n \sqrt{a_i}},$$

If $a_1 \ge 4$ then

$$\sigma \le e^{c_0 \sum_{i=1}^n \sqrt{a_i}} \le e^{c_0 \prod_{i=1}^n \sqrt{a_i}} = e^{c_0 \sqrt{n}}.$$

If $a_k \le 3$ then, since $p(2) = 2$ and $p(3) = 3$, we have

$$\sigma = a_1 a_2 \cdots a_k = n \le e^{c_0 \sqrt{n}}.$$

Otherwise, there is $1 \leq \ell < k$ with $a_\ell < 4 \leq a_{\ell+1}$. Let $n_1 = a_1 a_2 \dots a_\ell$ and $n_2 = n/n_1$. With the previous arguments, we get

$$\sigma \leq n_1 \cdot e^{c_0\sqrt{n_2}}.$$

If $n_1 \leq 3$ we easily check that $\sigma \leq e^{c_0\sqrt{n}}$, and if $n_1 \geq 4$ then

$$\sigma \leq n_1 \cdot e^{c_0\sqrt{n_2}} \leq e^{\sqrt{n_1}} e^{c_0\sqrt{n_2}} \leq e^{c_0(\sqrt{n_1}+\sqrt{n_2})} \leq e^{c_0\sqrt{n}}.$$

Thus $\sigma \leq e^{c_0\sqrt{n}}$ in every case. □

Let X be a sum of cycles with n vertices, and let $\mathrm{fact}(X)$ be the number of irreducible factorizations of X. We obviously have

$$\mathrm{fact}(X) \leq p^{\times}(n) \leq n e^{c_0\sqrt{n}}.$$

This proves, with very rough arguments, the assertion given in the introduction: the number of irreducible factorizations of a sum of cycles with n vertices is at most $e^{O(\sqrt{n})}$.

References

[1] Douglas Bowman, Paul Erdos, and Andrew M Odlyzko. 6640. *The American Mathematical Monthly*, 99(3):276–277, 1992.

[2] Florian Bridoux, Christophe Crespelle, Thi Ha Duong Phan, and Adrien Richard. Dividing permutations in the semiring of functional digraphs. In *International Workshop on Cellular Automata and Discrete Complex Systems*, pages 95–107. Springer, 2024.

[3] Borislav Crstici, Dragoslav S Mitrinovic, and József Sándor. *Handbook of Number Theory I*. Springer, 2006.

[4] Oscar Defrain, Antonio E Porreca, and Ekaterina Timofeeva. Polynomial-delay generation of functional digraphs up to isomorphism. *Discrete Applied Mathematics*, 357:24–33, 2024.

[5] Alberto Dennunzio, Valentina Dorigatti, Enrico Formenti, Luca Manzoni, and Antonio E Porreca. Polynomial equations over finite, discrete-time dynamical systems. In *Cellular Automata: 13th International Conference on Cellular Automata for Research and Industry, ACRI 2018, Como, Italy, September 17–21, 2018, Proceedings 13*, pages 298–306. Springer, 2018.

[6] Alberto Dennunzio, Valentina Dorigatti, Enrico Formenti, Luca Manzoni, and Antonio E Porreca. Polynomial equations over finite, discrete-time dynamical systems. In *Cellular Automata: 13th International Conference on Cellular Automata for Research and Industry, ACRI 2018, Como, Italy, September 17–21, 2018, Proceedings 13*, pages 298–306. Springer, 2018.

[7] Alberto Dennunzio, Enrico Formenti, Luciano Margara, Valentin Montmirail, and Sara Riva. Solving equations on discrete dynamical systems. In *International Meeting on Computational Intelligence Methods for Bioinformatics and Biostatistics*, pages 119–132. Springer, 2019.

[8] Alberto Dennunzio, Enrico Formenti, Luciano Margara, and Sara Riva. An algorithmic pipeline for solving equations over discrete dynamical systems modelling hypothesis on real phenomena. *Journal of Computational Science*, 66:101932, 2023.

[9] Alberto Dennunzio, Enrico Formenti, Luciano Margara, and Sara Riva. A note on solving basic equations over the semiring of functional digraphs. *arXiv preprint arXiv:2402.16923*, 2024.

[10] FW Dodd and LE Mattics. Estimating the number of multiplicative partitions. *The Rocky Mountain journal of mathematics*, 17(4):797–813, 1987.

[11] François Doré, Enrico Formenti, Antonio E Porreca, and Sara Riva. Decomposition and factorisation of transients in functional graphs. *Theoretical Computer Science*, 999:114514, 2024.

[12] François Doré, Kévin Perrot, Antonio E Porreca, Sara Riva, and Marius Rolland. Roots in the semiring of finite deterministic dynamical systems. In *International Workshop on Cellular Automata and Discrete Complex Systems*, pages 120–132. Springer, 2024.

[13] Valentina Dorigatti. Algorithms and complexity of the algebraic analysis of finite discrete dynamical systems. Master's thesis, Universitá degli Studi di Milano Bicocca, 2017.

[14] Andrzej Ehrenfeucht and Grzegorz Rozenberg. Reaction systems. *Fundamenta informaticae*, 75(1-4):263–280, 2007.

[15] Paul Erdös. On an elementary proof of some asymptotic formulas in the theory of partitions. *Annals of Mathematics*, 43(3):437–450, 1942.

[16] Philippe Flajolet and Andrew M Odlyzko. Random mapping statistics. In *Workshop on the Theory and Application of of Cryptographic Techniques*, pages 329–354. Springer, 1989.

[17] Maximilien Gadouleau. On the influence of the interaction graph on a finite dynamical system. *Natural Computing*, 19(1):15–28, 2020.

[18] Caroline Gaze-Maillot and Antonio E Porreca. Profiles of dynamical systems and their algebra. *arXiv preprint arXiv:2008.00843*, 2020.

[19] E. Goles. Dynamics of positive automata networks. *Theoretical Computer Science*, 41:19–32, 1985.

[20] E. Goles and S. Martínez. *Neural and Automata Networks: Dynamical Behavior and Applications*. Kluwer Academic Publishers, 1990.

[21] E. Goles and M. Tchuente. Iterative behaviour of generalized majority functions. *Mathematical Social Sciences*, 4:197–204, 1982.

[22] Godfrey H Hardy and Srinivasa Ramanujan. Asymptotic formulaæ in combinatory analysis. *Proceedings of the London Mathematical Society*, 2(1):75–115, 1918.

[23] John E Hopcroft and Jin-Kue Wong. Linear time algorithm for isomorphism of planar graphs (preliminary report). In *Proceedings of the sixth annual ACM symposium on Theory of computing*, pages 172–184, 1974.

[24] J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proc. Nat. Acad. Sc. U.S.A.*, 79:2554–2558, 1982.

[25] Hidde De Jong. Modeling and simulation of genetic regulatory systems: A literature review. *Journal of Computational Biology*, 9:67–103, 2002.

[26] S. A. Kauffman. Metabolic stability and epigenesis in randomly connected nets. *Journal of Theoretical Biology*, 22:437–467, 1969.

[27] Donald E Knuth. *The Art of Computer Programming, Volume 4, Fascicle 3: Generating All Combinations and Partitions*. Addison-Wesley Professional, 2005.

[28] W. S. Mac Culloch and W. S. Pitts. A logical calculus of the ideas immanent in nervous activity. *Bull. Math Bio. Phys.*, 5:113–115, 1943.

[29] Émile Naquin and Maximilien Gadouleau. Factorisation in the semiring of finite dynamical systems. *Theoretical Computer Science*, 998:114509, 2024.

[30] Melvyn B Nathanson. *Elementary methods in number theory*, volume 195. Springer Science & Business Media, 2000.

[31] S. Poljak and M. Sura. On periodical behaviour in societies with symmetric influences. *Combinatorica*, 3:119–121, 1982.

[32] A. Richard. Positive and negative cycles in boolean networks. *Journal of theoretical biology*, 463:67–76, 2019.

[33] Sara Riva. *Factorisation of discrete dynamical systems*. PhD thesis, Université Côte d'Azur; Università degli studi di Milano-Bicocca, 2022.

[34] J Barkley Rosser and Lowell Schoenfeld. Approximate formulas for some functions of prime numbers. *Illinois Journal of Mathematics*, 6(1):64–94, 1962.

[35] Ralph Seifert. On prime binary relational structures. *Fundamenta Mathematicae*, 70(2):187–203, 1971.

[36] R. Thomas. Boolean formalization of genetic control circuits. *Journal of Theoretical Biology*, 42(3):563–585, 1973.

[37] R. Thomas and R. d'Ari. *Biological Feedback*. CRC Press, 1990.

[38] R. Thomas and M. Kaufman. Multistationarity, the basis of cell differentiation and memory. II. Logical analysis of regulatory networks in terms of feedback circuits. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 11(1):180–195, 2001.