BEYOND BOOLEAN NETWORKS, A MULTI-VALUED APPROACH

J. GARCÍA GALOFRE[1], M. PÉREZ MILLÁN[2,3], A. GALARZA RIAL[4], R. LAUBENBACHER[5], A. DICKENSTEIN[2,3]

ABSTRACT. Boolean networks can be viewed as functions on the set of binary strings of a given length, described via logical rules. They were introduced as dynamic models into biology, in particular as logical models of intracellular regulatory networks involving genes, proteins, and metabolites. Since genes can have several modes of action, depending on their expression levels, binary variables are often not sufficiently rich, requiring the use of multi-valued networks instead. The steady state analysis of Boolean networks is computationally complex, and increasing the number of variable values beyond 2 adds substantially to this complexity, and no general methods are available beyond simulation. The main contribution of this paper is to give an algorithm to compute the steady states of a multi-valued network that has a complexity that, in many cases, is essentially the same as that for the case of binary values. Our approach is based on a representation of multi-valued networks using multi-valued logic functions, providing a biologically intuitive representation of the network. Furthermore, it uses tools to compute lattice points in rational polytopes, tapping a rich area of algebraic combinatorics as a source for combinatorial algorithms for Boolean network analysis. An implementation of the algorithm is provided.

1. INTRODUCTION

Boolean networks have been used in computer science, engineering, and physics extensively. They were introduced into biology by Stuart Kauffman [13], as a model class for intracellular regulatory networks, viewed as logical switching networks rather than as biochemical reaction networks. This has proven very useful, in part because their specification is intuitive from a biological point of view, and there are now hundreds of published Boolean network models of this type. A major drawback of this model type is the absence of mathematical tools for their analysis, in particular the efficient computation of steady states. For models with a small number of variables, exhaustive model simulation is feasible, while for larger models, a common method is sampling of the state space of the model.

For applications of this modeling framework to biology, it is often useful to be able to assume that certain variables have more than two possible values. For gene regulation, for instance, a given gene may have several models of action, depending on its expression levels (several examples are given below). In order to capture this complexity, a variable might need to take on several different values, such as "0 (off), 1 (medium expression), 2 (high expression)." The functions governing such networks can be expressed in terms of multi-valued logic, and we refer to them as *multi-valued networks*. The steady state analysis becomes significantly more complex.

This paper presents an algorithm to carry out such an analysis in a computationally high efficient way. We first make the problem more precise. Given n interacting (biological) species whose values can be described by $m+1$ values, we can establish a bijection between the set of values and the set

$$X_m = \Big\{0, \frac{1}{m}, \frac{2}{m}, \ldots, \frac{m-1}{m}, 1\Big\}. \tag{1}$$

In particular, if $m = 1$, we have a Boolean network. Our aim is to study the dynamics of the iteration of a function $f : X_m^n \to X_m^n$, with synchronous variable update. (Note that the order in which individual variables are updated, whether synchronous or asynchronous, does not change the steady states of the system.)

Any function on n variables defined over a finite set X of cardinality a power of a prime number can be thought as a function over a finite field. Moreover, it can be expressed as a polynomial function with coefficients in this field, and this opens the use of tools from computational algebraic geometry as Gröbner bases [20]. The advantage is that it is not necessary to express the function via a whole (big)

MPM and AD were partially supported by UBACYT 20020220200166BA and CONICET PIP 11220200100182CO, Argentina. RL was partially supported by NIH Grants 1 R01 HL169974-01 and 1U01EB024501-01.

table. For instance, given a function $f : X^n \to X^n$, we can find the *fixed points* $\{x \in X^n : f(x) = x\}$ of f (also known as *steady states*) solving the square polynomial system $f(x) - x = 0$. In [2] the authors show that in the Boolean case $m = 1$, it is possible to compute the fixed points with complexity $O(2^{0.841n})$ (under some hypotheses), while the bound for exhaustive search is $4\log_2(n)\,2^n$.

In [14] the authors study multilinear functions representing network functions $f : X_m^n \to X_m$ on a multivalued network. We propose instead to use in the multivariate setting the operations $\odot$, $\oplus$ and neg introduced in Section 2 that come from multivalued logic [6, 8]. These operations are more intuitive and closer to biological interpretations than the usual operations of polynomials over a finite field (and there is no restriction on the number of values). See for instance Figure 1, that features an example of two interacting nodes, with three values each, where the second variable acts as a repressor of the first one. The update function of the first node, f_1, is represented both as a polynomial over the field with 3 elements, which gives no clue of its behaviour, and in terms of the logic operations, whose interpretation is transparent.

$x_1 \setminus x_2$	0	1	2
0	0	0	0
1	1	0	0
2	2	1	0

(a) $f_1 : \mathbb{F}_3^2 \to \mathbb{F}_3$
$f_1(x_1, x_2) = x_1 + 2x_1x_2 + x_1^2x_2 + x_1x_2^2 + x_1^2x_2^2$

$x_1 \setminus x_2$	0	$\frac{1}{2}$	1
0	0	0	0
$\frac{1}{2}$	$\frac{1}{2}$	0	0
1	1	$\frac{1}{2}$	0

(b) $f_1 : \{0, 1/2, 1\}^2 \to \{0, 1/2, 1\}$
$f_1(x_1, x_2) = x_1 \ominus x_2$.

FIGURE 1. (a) The nodes take the values $\{0, 1, 2\}$ and the polynomial $f_1 \in \mathbb{F}_3[x_1, x_2]$ summarizes the values in the table. (b) The nodes take the values $\{0, 1/2, 1\}$ and the table is the translation of the first one; the function $f_1 = x_1 \ominus x_2$ accounts for the repression shown in the table.

Why model the qualitative behaviour of biological networks with more than two values? The setting of Boolean networks owes much to the work of the Belgian scientist René Thomas. His research included DNA biochemistry, genetics, biophysics, mathematical biology, and dynamical systems. His purpose was to decipher the key logical principles at the basis of the behaviour of biological systems and how complex dynamical behaviour could emerge. He realized that the researcher's intuition was not enough to understand the intrincated biological regulatory networks and he then proposed to formalize the models using the operations in Boolean algebra, where variables take only two values (0 or OFF and 1 or ON) and the logical operators are AND, OR and NOT, starting with [15]. In [16] he argues that even if one is interested in a logical description, the binary case could be too simple in many cases of interest. He adds that for instance, when there is a variable with more than a single action, different levels might be required to produce different effects. However, there are still few articles dealing with multivalued models.

Even if we work with a synchronous update of the nodes, we present in Example 25 a small network extracted from the foundational book [17], where the synchronous multivalued setting can model the features of an asynchronous model, as the networks become more *expressive*. Boolean networks became an effective mathematical tool to model many different phenomena in science and engineering and their development has attracted the interest of too many researchers to be cited here. One interesting feature is to understand well how the dynamics of a Boolean network could have high complexity, even if it is composed of simple elementary units [12].

We show that via the proposed operations of multivalued logic, which are indeed expressed in terms of linear inequalities, we can recover most of the good properties of Boolean networks. Moreover, we show that the computation of fixed points can be done algorithmically for any m using tools to find lattice points in rational polytopes and that in many cases, the complexity to compute the fixed points is essentially the same as in the Boolean setting. This is an alternative to translating a multivalued model into a partial Boolean one, but with mn nodes (see for instance [18] and the references therein).

In Section 2 we introduce the logic operations $\oplus, \odot, \mathrm{neg}$ together with their main properties and we give, in Theorem 10, a constructive way of turning data from a table into an expression of the function in terms of $\odot, \mathrm{neg}$ and constant functions. When modeling biological networks, it is useful to have in mind the behaviour of different small networks that appear frequently as subnetworks of bigger ones, called *motifs*. We also show in this section the behavior of several small motifs.

In § 3, we further show how to algorithmically translate any network into a $\odot$-neg network and we show in Theorem 24 how to recover the fixed points. In § 4 we propose several possible reductions of the number of variables (inspired by [21] in the Boolean setting) without increasing the indegree, that is, without increasing the maximum number of variables on which each variable depends (see Proposition 28). We present in § 4.2 an example which is the translation to our setting of an interesting network extracted from `www.cellcolective.org`, where the number of nodes eventually decreases. In the most interesting cases, the computation is too heavy to be done by hand and we used our publicly available implementation [10] that we explain in the next section.

In § 5, we address the computation of fixed points for $\odot$-neg networks. As we mentioned, this gives in turn the fixed points of any network function $F : X_m^n \to X_m^n$ by Theorem 24. One crucial algorithmic fact for $\odot$-neg networks is Theorem 29 that allows us to automatize the computations without simulations, which would be too costly in general. We present the basic setting for our implementation in Algorithm 2 and we discuss the cost of these computations. When the hypotheses of Theorem 33 do not hold, one needs to count the number of lattice points in a rational polytope, that is, all the integer solutions of a system of inequalities given by linear forms with integer coefficients defining a bounded region. Barvinok proposed in [3] an algorithm that counts these lattice points in polynomial time when the dimension of the polytope is fixed. The theoretical basis was given by Brion in the beautiful paper [4], where he uses the relation of rational polytopes with the theory of toric varieties and results in equivariant K-theory. We exemplify the use of our free app for the computation of fixed points in the network introduced in § 4.2.

In Section 6 we present a multi-valued model for the denitrification network in *Pseudomonas aeruginosa* based on the time-discrete and multi-valued deterministic model introduced in [1]. *P. aeruginosa* can perform (complete) denitrification, a respiratory process that eventually produces atmospheric N_2. The external parameters in the model and some variables are treated as Boolean (low or high), and other variables are considered ternary (low, medium or high). The update rules are formulated in [1] in terms of MIN, MAX and NOT (which correspond to AND, OR and NOT in a Boolean setting). However, the regulations of a few variables cannot be expressed in terms of these operators (marked with a * in the column “Update Rules” in their Table 3). As stated in Theorem 10, any function can be written in terms of constants and the logical operations $\oplus, \odot, \mathrm{neg}$ we propose to use in this paper, so we exactly represent their Update Rules and find the fixed points using results along the paper. Some details are provided in an Appendix.

2. Our setting

This section introduces the basic operations in multivalued logic and how we use them to represent the functions in biological models.

We fix a natural number m and consider networks with $m+1$ values in the set

$$X_m = \left\{0, \frac{1}{m}, \frac{2}{m}, \ldots, \frac{m-1}{m}, 1\right\} \subset [0,1]. \tag{2}$$

Boolean networks are a particular case when $m = 1$. As X_m is a subset of $\mathbb{R}$, we can also consider the standard operations of addition/substraction $(+/-)$ and multiplication $(.)$ on the real line. However, we introduce in the multivariate setting the operations $\odot$, $\oplus$ that come from multivalued logic [6, 8], which are more intuitive and closer to biological interpretations.

2.1. The MV operations.

The three basic operations in multivalued logic that we use to build all the functions that we consider are introduced in the following definition:

Definition 1. We consider the following operations/maps on X_m:

$$\text{neg} : X_m \to X_m, \text{ where } \text{neg}(x) = 1 - x. \tag{3}$$

$$\oplus : X_m \times X_m \to X, \text{ where } x \oplus y = min\{1, x + y\}. \tag{4}$$

$$\odot : X_m \times X_m \to X_m, \text{ where } x \odot y = max\{0, x + y - 1\}. \tag{5}$$

Example 2. When $m = 2$, the operations (4) and (5) in chart presentation are as follows:

$\oplus$	0	$\frac{1}{2}$	1
0	0	$\frac{1}{2}$	1
$\frac{1}{2}$	$\frac{1}{2}$	1	1
1	1	1	1

$\odot$	0	$\frac{1}{2}$	1
0	0	0	0
$\frac{1}{2}$	0	0	$\frac{1}{2}$
1	0	$\frac{1}{2}$	1

Remark 3. The following relations between $\oplus$ and $\odot$, which are formally equal to the De Morgan's law between AND and OR in the Boolean case, hold:

$$x \odot y = \text{neg}(\text{neg}(x) \oplus \text{neg}(y)), \tag{6}$$

$$x \oplus y = \text{neg}(\text{neg}(x) \odot \text{neg}(y)). \tag{7}$$

We list in the following proposition other useful properties which are straightforward to prove.

Proposition 4. *Given m and X_m as in* (2)*, consider the operations defined in Definition 1. The following properties hold:*

(i) $\text{neg}(0) = 1$ *and* $\text{neg}(1) = 0$.

(ii) The product of several variables can be expressed as

$$x_1 \odot \cdots \odot x_n = \max\{0, x_1 + \cdots + x_n - (n - 1)\}.$$

(iii) Analogously,

$$x_1 \oplus \cdots \oplus x_n = \min\{1, x_1 + \cdots + x_n\}.$$

(iv) The product $\odot$ is associative and commutative.

(v) $0 \odot x = 0$ *and* $1 \odot x = x$ *for any* $x \in X$.

(vi) For any finite index set I, $\text{neg}(\bigoplus_{i\in I} a_i) = \bigodot_{i\in I} \text{neg}(a_i)$.

(vii) $\max\{x, y\} = (x \odot \text{neg}(y)) \oplus y$.

(viii) $\min\{x, y\} = (x \oplus \text{neg}(y)) \odot y$.

(ix) $x \leq y$ *if and only if* $x \odot \text{neg}(y) = 0$.

(x) $x \odot y = 0$ *if and only if* $x + y \leq 1$. *More in general,* $x_1 \odot \cdots \odot x_n = 0$ *if and only if* $x_1 + \cdots + x_n \leq (n - 1)$.

Remark 5. Associativity will allow us to avoid writing parentheses in some cases. More precisely, we will write $s_1(x_1) \odot s_2(x_2) \odot s_3(x_3)$ instead of $\big(s_1(x_1) \odot s_2(x_2)\big) \odot s_3(x_3)$ or $s_1(x_1) \odot \big(s_2(x_2) \odot s_3(x_3)\big)$ for $s_i(x_i) = x_i$ or $s_i(x_i) = \text{neg}(x_i)$.

On the other hand, the operations we defined do not satisfy distributivity. In general, $(x \oplus y) \odot z \neq (x \odot z) \oplus (y \odot z)$. Consider for example $m = 5, x = \frac{3}{5}, y = \frac{2}{5}, z = \frac{1}{5}$ then $(x \oplus y) \odot z = \frac{1}{5}$ while $(x \odot z) \oplus (y \odot z) = 0$.

Definition 6. We also define the substraction, exponentiation and multiplication by a positive integer $k \in \mathbb{N}$ as:

$$x \ominus y := x \odot \text{neg}(y) = \max\{0, x - y\} = \begin{cases} x - y & if \quad x - y \geq 0 \\ 0 & otherwise \end{cases}, \tag{8}$$

$$x^k := \underbrace{x \odot \cdots \odot x}_{k \text{ times}} = \max\{0, kx - (k - 1)\}, \tag{9}$$

$$kx := \underbrace{x \oplus \cdots \oplus x}_{k \text{ times}} = \min\{1, kx\}. \tag{10}$$

Remark 7. Note that $x \odot y \leq x$ for any $x, y \in X_m$, and equality holds only when $y = 1$ or $x = 0$. In particular, $x^k \leq x$ for any $x \in X_m$ and any $k \in \mathbb{N}$, with equality only when $x = 0, 1$ or $k = 1$.

Remark 8. It is interesting to note that $\mathrm{neg}(kx) = \mathrm{neg}(x)^k$, indeed:

$$\mathrm{neg}(kx) = \mathrm{neg}(\underbrace{x \oplus \cdots \oplus x}_{k \text{ times}}) = \mathrm{neg}(\mathrm{neg}(\underbrace{\mathrm{neg}(x) \odot \cdots \odot \mathrm{neg}(x)}_{k \text{ times}})) = \mathrm{neg}(x)^k.$$

2.2. **Representability of functions.** We show next how to represent a function $f : X_m^n \to X_m$, in terms of $\odot$ and neg operators in a constructive way. We start with an example.

Example 9. Let $m = 2$, so that $X_m = X_2 = \{0, \frac{1}{2}, 1\}$. Define the following functions

$$f_0(x) = \mathrm{neg}(x)^2,\ f_1(x) = x^2, \text{ and } f_{\frac{1}{2}}(x) = \mathrm{neg}(f_0(x)) \odot \mathrm{neg}(f_1(x)).$$

Then,

x	$f_0(x)$	$f_{\frac{1}{2}}(x)$	$f_1(x)$
0	1	0	0
$\frac{1}{2}$	0	1	0
1	0	0	1

This means that $f_0, f_{\frac{1}{2}}, f_1$ are interpolators and this allows us to write any $f : X_2 \to X_2$ in terms of $\oplus, \odot, \mathrm{neg}$ and constants. Moreover, we deduce from the identities in Remark 3 that we could further translate any $\oplus$ operator into operations involving only $\odot$ and neg operations.

The following result is well known for experts in the area, but for the convenience of the reader, we give simple proof based on the interpolating functions as in the previous example.

Theorem 10. *Given $X_m = \{0, \frac{1}{m}, \ldots, \frac{m-1}{m}, 1\}$ and $n \in \mathbb{N}$. Every function $f : X_m^n \longrightarrow X_m$ with variables $x_1, \ldots, x_n$, is expressible in terms of $\odot, \mathrm{neg}$ and constant functions.*

Proof. Any function $f : X_m^n \to X_m$ can be represented as

$$f(x_1, \ldots, x_n) = \bigoplus_{(i_1, i_2, \ldots, i_n) \in X^n} f(i_1, i_2, \ldots, i_n) \odot \ell_{i_1}(x_1) \odot \cdots \odot \ell_{i_n}(x_n),$$

where the functions $\ell_{\frac{j}{m}}$ are defined as follows.

Set $\ell_0(x) = (\mathrm{neg}(x))^m$ and $\ell_1(x) = x^m$. Moreover, for $1 \leq j \leq m-1$, we define

$$g_j(x) := (x \oplus \frac{m-j}{m})^m = \begin{cases} 0 & \text{if } x < \frac{j}{m} \\ 1 & \text{if } x \geq \frac{j}{m} \end{cases}, \quad h_j(x) := (\mathrm{neg}(x) \oplus \frac{j}{m})^m = \begin{cases} 0 & \text{if } x > \frac{j}{m} \\ 1 & \text{if } x \leq \frac{j}{m} \end{cases}$$

Then, set $\ell_{\frac{j}{m}}(x) = g_j(x) \odot h_j(x) = \begin{cases} 0 & \text{if } x \neq \frac{j}{m} \\ 1 & \text{if } x = \frac{j}{m} \end{cases}$. It is straightforward to check that $\ell_{i_1}(x_1) \odot \cdots \odot \ell_{i_n}(x_n) = 0$ unless $(x_1, \ldots, x_n) = (i_1, \ldots, i_n)$, in which case it equals 1.

It is now enough to iteratively apply the equalities in Remark 3 to get rid of the $\oplus$ operation. $\square$

In fact, we can also express any function in terms of $\oplus, \mathrm{neg}$ and constant functions. For instance, the interpolators in Example 9 equal

$$f_0(x) = \mathrm{neg}(x \oplus x),\ f_1(x) = \mathrm{neg}(\mathrm{neg}(x) \oplus \mathrm{neg}(x)), \text{ and } f_{\frac{1}{2}}(x) = \mathrm{neg}(f_0(x) \oplus f_1(x)).$$

But we will use in general the version without $\oplus$ to mimic the standard choice in the Boolean case. On the other side,

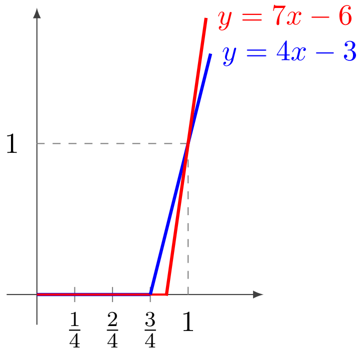
it is important to note that if we do not allow constant functions in Theorem 10, the result is not true. For instance, when $m = 2$, the constant function $\frac{1}{2}$ cannot be expressed in terms of $\oplus, \odot$ and neg.

Remark 11. Different functions over $\mathbb{R}$ can coincide on X_m and there may exist more than one $\odot$-neg expression for the same function.

(i) For instance,

$$x^s = \max\{0, sx - (s-1)\}$$

has the same value on X_m as x^m for any $s \geq m$. Indeed $x^s = 0$ unless $x > \frac{s-1}{s} \geq \frac{m-1}{m}$, that is, unless $x = 1$ and in this case, its value is 1 (see Figure2).



x	0	$\frac{1}{4}$	$\frac{2}{4}$	$\frac{3}{4}$	1
x^4	0	0	0	0	1
x^7	0	0	0	0	1

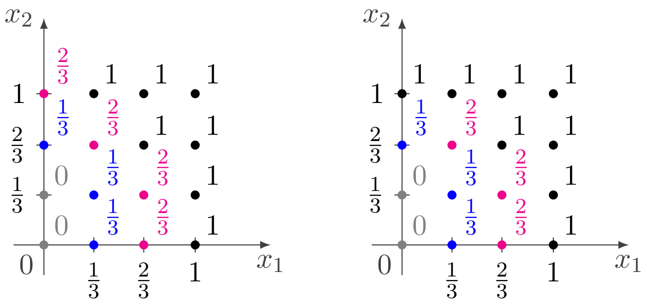
FIGURE 2. Comparing x^4 and x^7 when $m = 4$. As real-valued functions, $max\{0, 7x - 6\}$ and $max\{0, 4x - 3\}$ are different but they coincide in $X_m = \{0, \frac{1}{4}, \frac{2}{4}, \frac{3}{4}, 1\}$.

(ii) For $m = 3$, the equality $x^2 \odot \frac{1}{3} = x \odot \frac{1}{3}$ holds for every $x \in X$. This function takes the value 1 at 1 and 0 otherwise. More generally, given $m \in \mathbb{N}$, for every $2 \leq p \leq m$ the equality $x \odot \frac{1}{m} = x^p \odot \frac{1}{m}$ holds for every $x \in X$.

2.3. **Motifs.** We present here four simple *motifs*, that is, small networks that appear frequently as subnetworks of bigger ones.They are built using the multivalued logic operations for a network with $n \geq 3$ interacting biological species whose values can be described by $m + 1$ values in $X_m = \{0, \frac{1}{m}, \dots, \frac{m-1}{m}, 1\}$.

Motif 1: We start by describing two simple mechanisms, where node 1 is moderately activated by node 2. We show the resulting functions that describe this situation, when $m = 3$.

Motif 1a: $f_1 = x_1 \oplus (x_2 \ominus \frac{1}{m})$ Motif 1b:$f_1 = x_1 \oplus x_2^2$



Motif 2: In this motif, node 1 depends on its previous state, node 2 acts as an activator and node 3 acts as a weaker activator.

$$\begin{aligned} f_1 &= x_1 \oplus x_2 \oplus x_3^2 \\ &= \min\{1, x_1 + x_2 + \max\{0, 2x_3 - 1\}\}. \end{aligned}$$

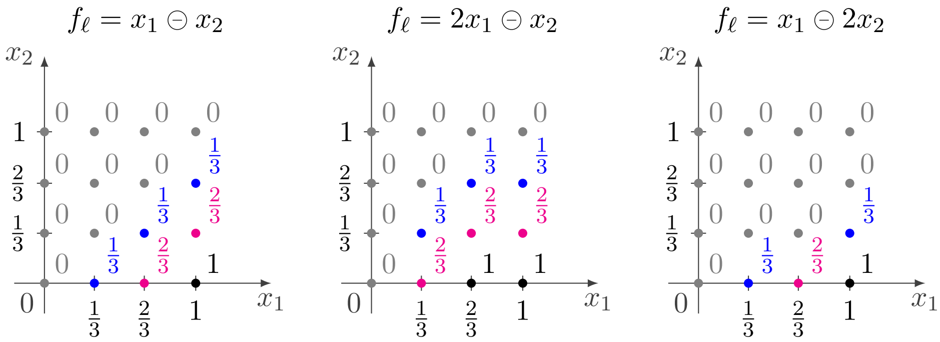
Motif 3: In this last motif, the update function of node 1 depends on itself, an activator node 2 and a repressor node 3.

$$\begin{aligned} f_1 &= x_2 \oplus (x_1 \ominus x_3) \\ &= x_2 \oplus (x_1 \odot \operatorname{neg}(x_3)) \\ &= \min\{1, x_2 + \max\{0, x_1 - x_3\}\}. \end{aligned}$$

Motif 4: We finally introduce a mechanism that considers the outcome of several activators and inhibitors acting simultaneously on a single node. We propose the following motif that describes the total effect on node ℓ produced by activators $i \in A$ and inhibitors $j \in B$, with $A, B \subseteq \{1, \dots, n\}$, $A \cap B = \emptyset$: $f_\ell = \oplus_{i \in A} x_i \ominus (\oplus_{j \in B} x_j)$, or more generally, given positive weights $w_k \in \mathbb{N}$:

$$f_\ell \;=\; \bigoplus_{i\in A} w_i x_i \ominus \left(\bigoplus_{j\in B} w_j x_j\right) \;=\; \max\{0, \min\{1, \sum_{i\in A} w_i x_i\} - \min\{1, \sum_{j\in B} w_j x_j\}\}.$$

Note that if $\sum_{j\in B} w_j x_j \geq 1$ (i.e. when the effect of the inhibitors is big enough), then $f_\ell = 0$. And if $\sum_{i\in A} w_i x_i \leq 1$ and $\sum_{j\in B} w_j x_j \leq 1$, then $f_\ell > 0$ if and only if $\sum_{i\in A} w_i x_i > \sum_{j\in B} w_j x_j$. This mimics Boolean threshold functions. For instance, we show three of these functions for $m = 3$:



2.4. **Fixed points.** When studying the dynamics of biological systems one important aspect to be considered is the long-term behavior of the system. Especially the equilibria of the dynamical system. In the case of continuous-time biological systems, it is common to study the *steady states* of the system. In the discrete-time context, the steady states (or attractors) are called *fixed points* of the system. If moreover there is a finite number of states for every node and the dynamical system is given by $F : X_m^n \to X_m^n$, the fixed points are the points $x \in X_m^n$ such that $F(x) = x$ (i.e. $f_i(x) = x_i$ for all $i \in \{1, \ldots, n\}$). These points will be our main focus in the subsequent sections.

It is actually a sensible question to look for the fixed points of a function $F : X_m^n \to X_m^n$ with X_m a finite set since most of these functions have at least one. In fact, by counting the number of these functions without fixed points the following result is immediate:

Proposition 12. *Set $q = (m+1)^n$, $\#X = m+1$. The proportion of $F : X_m^n \to X_m^n$ with at least one fixed point equals*

$$\frac{q^q - (q-1)^q}{q^q} = 1 - \left(\frac{q-1}{q}\right)^q$$

So, when $n \to +\infty$ or $m \to +\infty$ this proportion is close to $1 - \frac{1}{e} \sim \frac{2}{3}$

We describe below how we can make a function *smoother* without altering its fixed points. Following the ideas in [5], we introduce a function to *preserve continuity*, which means that each node will change at most $\frac{1}{m}$ in one time step. For this purpose, we take into account the previous state of the corresponding node (we add a self-regulation loop to each network node). The future value of the regulated variable under continuity is computed as follows:

Definition 13. Given $X_m = \{0, \frac{1}{m}, \ldots, 1\}$ we define the function $h : X_m^2 \to X_m$ as

$$h(x,y) = \begin{cases} x \oplus \frac{1}{m} & \text{if } y > x \\ x & \text{if } y = x \\ x \ominus \frac{1}{m} & \text{if } y < x \end{cases} \tag{11}$$

Given $F = (f_1, \ldots, f_n) : X_m^n \to X_m^n$, the continuous version of each coordinate function $f_i : X_m^n \to X_m$ is denoted by $f_{i,cont}$ and is defined as

$$f_{i,cont}(x) = h(x_i, f_i(x)), \tag{12}$$

for $i = 1, \ldots, n$. We denote $F_{cont} = (f_{1,cont}, \ldots, f_{n,cont})$.

The following lemma is an immediate consequence of the previous definition:

Lemma 14. *A function $F : X_m^n \to X_m^n$ and its associated function F_{cont} in Definition 13 have the same fixed points.*

For any m, the continuous version of all the powers x^k is the same function, independently of k.

Lemma 15. *Given $k \in \mathbb{N}_{\geq 2}$, then*

$$x^k_{cont} = \begin{cases} x \ominus \frac{1}{m} & \text{if } x < 1 \\ 1 & \text{if } x = 1 \end{cases}$$

Proof. It is straightforward to check that $x \geq x^k$ for any $x \in X$ and that the equality holds only if $x = 0$ or $x = 1$. □

3. From any vector function to a $\odot$-neg network function

In the subsequent sections we will focus on dynamical systems over the set X_m. Consider $n \in \mathbb{N}$ and a function $F : X_m^n \to X_m^n$, $F = (f_1, \ldots, f_n)$ that describes the synchronous update of a biological system with n nodes with values in the finite set X_m as in (2).

Inspired by the framework in [22], we now propose our own method to compute the fixed points of the network. The equations can be previously simplified using the considerations in Section 4.

We will now mimic a procedure which has been proved useful in Boolean networks. Recall that every Boolean function f can be written in conjunctive normal form as

$$f = w_1 \wedge \cdots \wedge w_r$$

where

$$w_j = s_1 x_{i_1} \vee \cdots \vee s_{n_j} x_{i_{n_j}}$$

for $j \in \{1, \ldots, r\}$, $\{i_1, \ldots, i_{n_j}\} \subseteq \{1, \ldots, n\}$ and $s_k \in \{id, \mathrm{neg}\}$. This gives the idea of AND-NOT functions: a vector function $F = (f_1, \ldots, f_n) : X_m^n \to X_m^n$ such that every $f_i : X_m \to X_m$ is an AND-NOT function, has the same information as its wiring diagram [22, 21]. In order to obtain similar properties in our context, we define what we call $\odot$-neg functions.

Definition 16. A function $f : X_m^n \longrightarrow X_m$ is called a $\odot$-neg function if it can be written as

$$f(x_1, \ldots, x_n) = \bigodot_{j \in A} x_j^{p_j} \odot \bigodot_{j \in B} \mathrm{neg}(x_j)^{q_j} \bigodot c$$

with $c \in X_m$, $A \cup B \subseteq \{1, \ldots, n\}$ and $\{p_j, q_j\} \subseteq \mathbb{N}$.

A vector function $F = (f_1, \ldots, f_n) : X_m^n \longrightarrow X_m^n$ is called an $\odot$-neg network function if f_i is a $\odot$-neg function for all $i \in \{1, \ldots, n\}$.

The most important feature of $\odot$-neg networks is that there is a direct correspondence between their wiring diagram and the network functions. All the information about the network dynamics can be read from the wiring diagram (and vice versa), which is defined by a labeled graph $G = (V_G, E_G)$ with vertices

$$V_G = \{1, \ldots, n, C\}$$

($\{1, \ldots, n\}$ may also be denoted by $\{x_1, \ldots, x_n\}$) and edges E_G given as follows. If $F = (f_1, \ldots, f_n)$ and

$$f_l(x_1, \ldots, x_n) = \bigodot_{i \in A_l} x_i^{p_{l_i}} \odot \bigodot_{j \in B_l} \mathrm{neg}(x_j)^{q_{l_j}} \bigodot c_l,$$

then there exist labeled edges of the form

$$i \xrightarrow{p_{l_i}} l \text{ if } p_{l_i} > 0; \qquad j \xrightarrow{q_{l_j}} \!\!\dashv l \text{ if } q_{l_j} > 0; \qquad C \xrightarrow{c_l} l \ .$$

Not every function is a $\odot$-neg function. Consider for instance $f_1 = \mathrm{neg}(x_i \odot x_j)$ for any i, j.

Example 17. Given $F(x_1, x_2, x_3) = (x_2^2 \odot \operatorname{neg}(x_3)^q, x_1, \frac{1}{2} \odot x_1^p)$, the associated wiring diagram is as follows:

$$\begin{tikzcd}
& C \arrow[dl, "\frac{1}{2}"'] \arrow[d, dashed, "1"] \arrow[dr, dashed, "1"] & \\
x_3 \arrow[r, -|, "q"] & x_1 \arrow[l, bend left, "p"] \arrow[r, bend right, "1"'] & x_2 \arrow[l, "2"']
\end{tikzcd}$$

The dotted arrows can be skipped.

Proposition 18. *Given $f \not\equiv 0$ a $\odot$-neg function written as in Definition 16, the sets A, B can be chosen such that $A \cap B = \emptyset$, $c \neq 0$ and $1 \leq p_j, q_j \leq m$.*

Proof. First note that if $A \cap B \neq \emptyset$ then $f = 0$ since $x \odot x = 0$ for any x by (8). The last part of the statement follows from Remark 11. $\square$

Before we show that any network function $F : X_m^n \to X_m^n$ can be transformed into a $\odot$-neg network function where we can trace the fixed points of F, we introduce a complexity measure that will estimate the size of the new $\odot$-neg network.

Definition 19. We define by Algorithm 1 below a complexity measure of a function $f : X_m^n \to X_m$ called *depth* denoted by $\mu(f)$. We also define the associated function $\overline{f}$ depending on $\mu(f)$ new variables.

Algorithm 1 How to build μ and $\overline{f}$

Input: $f : X_m^n \to X_m$.
Output: $\mu(f)$ and a network function $(\overline{f}, \overline{f}_{u_1}, \ldots, \overline{f}_{u_{\mu(f)}}) : X_m^{n+\mu(f)} \to X_m^{1+\mu(f)}$.

$\mu(f) \leftarrow 0$
$\overline{f}_0 \leftarrow f$
$k \leftarrow 1$
while there is an expression $\operatorname{neg}(h)$, with h a $\odot$-neg function with two or more factors, in f_{k-1} **do**
  Add a new variable u_k
  $\overline{f}_k \leftarrow$ the resulting function after replacing h in $\overline{f}_{k-1}$ with the new variable u_k
  $\overline{f}_{u_k} \leftarrow h$
  $\mu(f) \leftarrow \mu(f) + 1$
  $k \leftarrow k + 1$
end while
$\overline{f} \leftarrow \overline{f}_k$

Given a network function $F = (f_1, \ldots, f_n) : X_m^n \to X_m^n$, apply the algorithm to every f_i, and we define $\mu(F) = \sum_{i=1}^n \mu(f_i)$. $\mu(F)$ gives an upper bound to the total amount of new variables one needs to add in order to write each F_i as a $\odot$-neg function as in Definition 16.

Remark 20. Algorithm 1 can be optimised, for example, by making a list of substitutions. A new variable is added only if the new substitution is not on the list.

Remark 21. For $f : X_m^n \to X_m$ and $1 \leq k \leq \mu(f)$, note that $\overline{f}_{u_k}$ only depends on previously defined variables: $\overline{f}_{u_1} = \overline{f}_{u_1}(x_1, \ldots, x_n)$, and $\overline{f}_{u_k} = \overline{f}_{u_k}(x_1, \ldots, x_n, u_1, \ldots, u_{k-1})$, for $2 \leq k$.

Example 22. If we go back to Motifs 2, 3 and 4 from § 2.3, we can transform each function into a $\odot$-neg function, by reproducing the steps in Algorithm 1.

<u>**Motif 2:**</u>

$$\begin{aligned} f_1 &= x_1 \oplus x_2 \oplus x_3^2 \\ &= \min\{1, x_1 + x_2 + \max\{0, 2x_3 - 1\}\} \\ &= \operatorname{neg}(\operatorname{neg}(x_1) \odot \operatorname{neg}(x_2) \odot \operatorname{neg}(x_3 \odot x_3)) \end{aligned} \quad \Rightarrow \quad \begin{aligned} \overline{f}_1 &= \operatorname{neg}(u_2) \\ \overline{f}_{u_1} &= x_3 \odot x_3 \\ \overline{f}_{u_2} &= \operatorname{neg}(x_1) \odot \operatorname{neg}(x_2) \odot \operatorname{neg}(u_1) \end{aligned}$$

<u>**Motif 3:**</u>

$$\begin{aligned} f_1 &= x_2 \oplus (x_1 \ominus x_3) \\ &= \min\{1, x_2 + \max\{0, x_1 - x_3\}\} \\ &= \operatorname{neg}(\operatorname{neg}(x_2) \odot \operatorname{neg}(x_1 \odot \operatorname{neg}(x_3))) \end{aligned} \quad \Rightarrow \quad \begin{aligned} \overline{f}_1 &= \operatorname{neg}(u_2) \\ \overline{f}_{u_1} &= x_1 \odot \operatorname{neg}(x_3) \\ \overline{f}_{u_2} &= \operatorname{neg}(x_2) \odot \operatorname{neg}(u_1) \end{aligned}$$

Motif 4:

$$\begin{aligned} f_\ell &= \bigoplus\nolimits_{i\in A} w_i x_i \ominus \left(\bigoplus\nolimits_{j\in B} w_j x_j\right) \\ &= \max\{0, \min\{1, \sum_{i\in A} w_i x_i\} - \min\{1, \sum_{j\in B} w_j x_j\}\} \\ &= \operatorname{neg}(\bigodot\nolimits_{i\in A} \operatorname{neg}(x_i)^{w_i}) \odot (\bigodot\nolimits_{j\in B} \operatorname{neg}(x_j)^{w_j}) \end{aligned} \quad \Rightarrow \quad \begin{aligned} \overline{f}_\ell &= \operatorname{neg}(u_1) \odot u_2 \\ \overline{f}_{u_1} &= \bigodot\nolimits_{i\in A} \operatorname{neg}(x_i)^{w_i} \\ \overline{f}_{u_2} &= \bigodot\nolimits_{j\in B} \operatorname{neg}(x_j)^{w_j} \end{aligned}$$

In the three cases, the corresponding value of μ equals 2 and we can see that this is related to the number of alternations between maximum and minimum occurring in the functions.

Remark 23. Note that by items (ii) and (iii) in Proposition 4, consecutive $\oplus$ (resp. $\odot$) operations can be replaced by a single minimum (resp. maximum). Thus, given any function $f : X_m^n \to X_m$, $\mu(f)$ *measures* the alternation of $\oplus$ and $\odot$ operations in the expression of f.

We could have written any function in terms of $\oplus$, neg and constant functions and proposed a similar algorithm to add a new variable for every negation of a sum. We chose $\odot$-neg functions instead of $\oplus$-neg ones to generalize [22].

Wiring diagrams of $\odot$-neg networks encode all the information that the network carries. The fixed points of the dynamical system obtained by iteration of F are in bijection with the fixed points of $\overline{F}$. Given $F : X_m^{n_1} \to X_m^{n_1}$ and the corresponding $\overline{F} : X_m^{n_2} \to X_m^{n_2}$ constructed as in Definition 19, call $x = (x_1, \dots, x_{n_1})$, we define the function $U : X_m^{n_1} \to X_m^{n_2-n_1}$ in a recursive way (see Remark 21):

$$\begin{cases} U_1(x) = \overline{f}_{u_1}(x), \\ U_k(x) = \overline{f}_{u_k}(x, U_1(x), \dots, U_{k-1}(x)), \text{ for } 2 \le k \le n_2 - n_1. \end{cases} \tag{13}$$

Theorem 24. *Given $F : X_m^{n_1} \to X_m^{n_1}$, the algorithm in Definition 19 constructs $\overline{F} : X_m^{n_2} \to X_m^{n_2}$ which consists of $\odot$-neg functions and the function $U : X_m^{n_1} \to X_m^{n_2-n_1}$ as in* (13), *where $n_2 - n_1 \le \mu(F)$. Moreover, the following diagram commutes:*

$$\begin{tikzcd} X_m^{n_1} \arrow[r, "(Id,U)"] \arrow[d, "F"'] & X_m^{n_2} \arrow[d, "\overline{F}"] \\ X_m^{n_1} & X_m^{n_2} \arrow[l, "\pi(x,y)=x"] \end{tikzcd} \tag{14}$$

and the fixed points of F are in bijection with the fixed points of $\overline{F}$ via projection onto the first n_1 coordinates.

Before the proof we present a clarifying example.

Example 25. This example is based on Thomas and D'Ari [17, Chapter 4,§II]. Consider three genes $\mathcal{X}, \mathcal{Y}, \mathcal{Z}$ with respective products denoted by x, y, z. Gene $\mathcal{X}$ is expressed constitutively (not regulated), gene $\mathcal{Y}$ is expressed only in the absence of product x, and gene $\mathcal{Z}$ is expressed provided product y or product z is present.

The Boolean logical relations proposed in [17] are

$$(x, y, z) \mapsto (1, \neg x, y \vee z),$$

with two steady states: $(1,0,0)$ and $(1,0,1)$. The authors argue that starting from the initial state $(0,0,0)$ if gene $\mathcal{X}$ is turned on (e.g. immediately after infection by a virus) then gene $\mathcal{Y}$ will be expressed until product x appears, switching it off. Gene $\mathcal{Z}$ will be turned on only if y appears before x switches gene Y off. This model is asynchronous with three time delays t_x, t_y, t_z. If $t_x < t_y$, then $(0,0,0) \mapsto (1,0,0)$ and the system remains there as in the synchronous Boolean case.

When $t_x > t_y$ then $(0,0,0) \mapsto (0,1,0)$. If moreover $t_x > t_y + t_z$, then $(0,1,0) \mapsto (1,1,1) \mapsto (1,0,1)$ and the system remains at $(1,0,1)$. We propose instead the following *synchronous* modeling with $m = 3$. We assume that $\mathcal{X}$ switches $\mathcal{Y}$ off completely only if x is at level 1 (fully activated). We then consider the following network function $F = (f_x, f_y, f_z)$:

$$F : X_m^3 \to X_m^3, \quad F(x, y, z) = \left(x \oplus \frac{1}{3}, \mathrm{neg}(x), y \oplus z\right).$$

The orbit of $(0,0,0)$ under F equals: $(0,0,0) \mapsto (\frac{1}{3}, 1, 0) \mapsto (\frac{2}{3}, \frac{2}{3}, 1) \mapsto (1, \frac{1}{3}, 1) \mapsto (1, 0, 1) \mapsto (1, 0, 1)$. So, $(1, 0, 1)$ is a fixed point and as $\mathcal{X}$ is activated in three steps, $\mathcal{Z}$ can be turned on even if $\mathcal{Y}$ is eventually turned off. Note that $(1, 0, z_0)$ is a fixed point for any value z_0 of z.

To translate F to an $\odot - \mathrm{neg}$ function $\overline{F}$ we need to add two variables u_1, u_2. Thus, we have that $\overline{F} = (\overline{f}_x, \overline{f}_y, \overline{f}_z, \overline{f}_{u_1}, \overline{f}_{u_2}) : X_m^5 \to X_m^5$ is defined by

$$\begin{aligned}
\overline{f}_x &= \mathrm{neg}(u_1) \\
\overline{f}_y &= \mathrm{neg}(x) \\
\overline{f}_z &= \mathrm{neg}(u_2) \\
\overline{f}_{u_1} &= \mathrm{neg}(x) \odot \tfrac{2}{3} \\
\overline{f}_{u_2} &= \mathrm{neg}(y) \odot \mathrm{neg}(z).
\end{aligned}$$

Its fixed points are in bijection with those of F by projecting into the first 3 coordinates. Indeed, if $(x^*, y^*, z^*, u_1^*, u_2^*)$ is a fixed point of $\overline{F}$ then

$$\begin{aligned}
&x^* = \mathrm{neg}(u_1^*) = \mathrm{neg}(\mathrm{neg}(x^*) \odot \frac{2}{3}) = x^* \oplus \frac{1}{3} = f_x(x^*, y^*, z^*), \\
&y^* = \mathrm{neg}(x^*) = f_y(x^*, y^*, z^*), \\
&z^* = \mathrm{neg}(u_2^*) = \mathrm{neg}(\mathrm{neg}(y^*) \odot \mathrm{neg}(z^*)) = y^* \oplus z^* = f_z(x^*, y^*, z^*),
\end{aligned}$$

and (x^*, y^*, z^*) is a fixed point of F. Reciprocally, if (x^*, y^*, z^*) is a fixed point of F, define $u_1^* = \mathrm{neg}(x^*) \odot \frac{2}{3}$ and $u_2^* = \mathrm{neg}(y^*) \odot \mathrm{neg}(z^*)$ and it is straightforward to check that $(x^*, y^*, z^*, u_1^*, u_2^*)$ is a fixed point of $\overline{F}$.

Proof of Theorem 24. From Algorithm 1 and Definition 19 it is straightforward to check that $n_2 - n_1 \leq \mu(F)$. It is also immediate from the definition of $\overline{F}$ and U that $\pi(\overline{F}(x, U(x))) = F(x)$ for any $x \in X^{n_1}$ and the diagram in (14) commutes.

Consider a fixed point x^* of F and define $u^* = U(x^*)$. Then, $\overline{f}_i(x^*, u^*) = \overline{f}_i(x^*, U(x^*)) = f_i(x^*) = x_i^*$ for $1 \leq i \leq n_1$, and $\overline{f}_i(x^*, u^*) = u_i^*$ for $1 \leq i \leq n_2 - n_1$ by definition of u^*. Thus, (x^*, u^*) is a fixed point of $\overline{F}$.

Conversely, if (x^*, u^*) is a fixed point of $\overline{F}$, by the recursive definition (13) of U we have that $u^* = U(x^*)$. Then, $F(x^*) = \pi(\overline{F}(x^*, U(x^*))) = \pi(\overline{F}(x^*, u^*)) = \pi(x^*, u^*) = x^*$. We conclude that x^* is a fixed point of F. □

4. Simplifying the fixed point equations

We propose in this section several reductions that can be applied to a multivalued network $F : X_m^n \to X_m^n$, and its $\odot$-neg corresponding network $\overline{F}$, that eventually produce a $\odot$-neg network $G : X_m^{n'} \to X_m^{n'}$ whose fixed points allow us to reconstruct immediately the fixed points of F. Usually $n' << n$, see for instance § 4.2 ($n = 19$, $n' = 4$) or § 6 ($n = 15$, $n' = 5$ in the worst scenario).

We first simplify the equations with the aid of some reductions that can be easily read from the algebraic expressions. We then transform the reduced network into a $\odot$-neg network, in which new variables are added to gain simplicity when dealing with minima and maxima imposed by the logical frameworks. We can then apply more network reductions to the $\odot$-neg network before dealing with the (usually small number of) fixed point equations.

4.1. **Network reductions.** Many network reductions can be done and they are proposed in [21] for AND-NOT Boolean networks. In our multivalued context, some of these reductions are still valid. Here we provide a list of reductions with the objective of reducing the complexity of the network. When using these reductions, the network will change but the fixed points will remain the same.

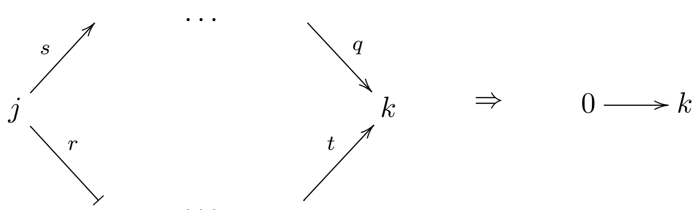
The first reductions we introduce are the following. In what follows w, w', w'' will denote (any/suitable) multivalued functions

(i) If $f_i = c$ with $c \in X_m$, then x_i can be *pulled apart*.
(ii) If $f_i = x_k$ or $f_i = \mathrm{neg}(x_k)$ then x_i can be *pulled apart* and we can replace $x_i = x_k$ or $x_i = \mathrm{neg}(x_k)$, respectively, in each other function.
(iii) If $f_i = f_j$, then we can set $f_i = x_j$ and *pull apart* x_i.
(iv) If x_i does not does not appear in any f_j then, x_i can be *pulled apart*.
(v) For all x we have the identity $x \odot \mathrm{neg}(x) = 0$ by (8). More precisely, if $f_i = x_k \odot \mathrm{neg}(x_k) \odot w$, then we can replace $x_i = 0$, and x_i can be *pulled apart*.
(vi) If $f_i = x_j \odot x_k \odot w$, and $f_j = \mathrm{neg}(x_k) \odot w'$, then, at a fixed point, $f_i = x_k \odot \mathrm{neg}(x_k) \odot w'' = 0$. Then, at a fixed point, $x_i = 0$ and x_i can be *pulled apart*.
(vii) By Remark 11, we can replace any power $s > m$ by m.

We say a variable is *pulled apart* when we replace x_i by the corresponding expression of f_i in every function f_j where x_i appears (for example, if $f_i = x_k$, $i \neq k$, we replace $x_i = x_k$ in every f_j for $j \neq i$) and both, x_i and f_i are no longer involved in the system of equations to be considered. We keep aside the equation $x_i = f_i$ to obtain the value of x_i at fixed point after the values of the other variables involved in f_i are obtained (see, for instance, § 4.2).

Recall that there is a direct correspondence between the wiring diagram of a $\odot$-neg network and the network functions. We present in the following proposition a new reduction that can be easily detected from the wiring diagram of such a network. The proof is straightforward.

Proposition 26. *Let $F : X_m^n \to X_m^n$ be a $\odot$-*neg *network. If x_k is at the end of two chains that start at the same node x_j, both chains have only positive arrows except for only one negative arrow leaving from x_j, then $f_k = x_j \odot \mathrm{neg}(x_j) \odot w = 0$.*



Examples of the reductions above can be seen in the following sections.

Definition 27. We define the indegree $indeg(F)$ of an $\odot$-neg network function $F : X_m^n \to X_m$ as follows. F is represented by a digraph with (at most $n+1$) nodes. Nodes are elements in $\{x_1, \dots, x_n, C\}$, where $f_j(x)$ may contain x_i^p or $\mathrm{neg}(x_i)^q$ as a factor: $C \xrightarrow{c} x_j$, $x_i \xrightarrow{p} x_j$, $x_i \overset{q}{\relbar\!\!\relbar\!\!\dashv} x_j$, for some $p, q \in \mathbb{N}$. We define $indeg(f_i) = \#\{\textit{arrows arriving to } x_i\}$ and $indeg(F) = \max\{indeg(f_i)\}$.

Thus, the indegree measures the maximum number of variables on which any coordinate function f_i of F depend. In the Boolean case, most of the reductions described in [21] do not increase the indegree. Applying any sequence of the reductions we introduced before in this section, we could reduce the number of variables and the computation of the fixed points of a $\odot$-neg network function F to the computation of the fixed points of any reduced $\odot$-neg function, that we denote by F_{red}.

Proposition 28. *Any sequence of the described reductions from a $\odot$-*neg *network function F to a network function F_{red} does not increase the indegree.*

The proof of Proposition 28 is straightforward.

4.2. **Example from Cell Collective: Mammalian Cell Cycle (1607).** Here we generalize a Boolean network https://cellcollective.org//#module/1607:1/mammalian-cell-cycle/1 . It models the transmembrane tyrosine kinase ERBB2 interaction network. The overexpression of this protein is an adverse prognostic marker in breast cancer, and occurs in almost 30% of the patients.

Using our reductions, we can reduce from 19 original variables to 4 variables and a constant. Yet, in one of the cases, the computation is too long to be done by hand and we used our implementation [10], that gives an immediate answer in this case

We call c = EGF, x_1 = ErbB1, x_2 = ERa, x_3 = CycE1, x_4 = CycD1, x_5 = CDK4, x_6 = ErbB3, x_7 = cMYC, x_8 = Akt1, x_9 = ErbB2, x_{10} = p21, x_{11} = ErbB1_2, x_{12} = ErbB1_3, x_{13} = IGF1R, x_{14} = p27, x_{15} = pRB, x_{16} = ErbB2_3, x_{17} = CDK6, x_{18} = CDK2, and x_{19} = MEK1. We translate the Boolean network to our setting by changing $\vee \rightsquigarrow \oplus$, $\wedge \rightsquigarrow \odot$ and $\neg \rightsquigarrow \mathrm{neg}$ for any value of m:

$$
\begin{array}{rl@{\qquad}rl}
f_1 = & \mathrm{c} & f_{11} = & x_1 \odot x_9 \\
f_2 = & x_8 \oplus x_{19} & f_{12} = & x_1 \odot x_6 \\
f_3 = & x_7 & f_{13} = & (x_8 \odot \mathrm{neg}(x_{16})) \oplus (x_2 \odot \mathrm{neg}(x_{16})) \\
f_4 = & (x_{19} \odot x_2 \odot x_7) \oplus (x_8 \odot x_2 \odot x_7) & f_{14} = & x_2 \odot \mathrm{neg}(x_{18}) \odot \mathrm{neg}(x_5) \odot \mathrm{neg}(x_7) \odot \mathrm{neg}(x_8) \\
f_5 = & x_4 \odot \mathrm{neg}(x_{10}) \odot \mathrm{neg}(x_{14}) & f_{15} = & (x_{18} \odot x_5 \odot x_{17}) \oplus (x_5 \odot x_{17}) \\
f_6 = & c & f_{16} = & x_9 \odot x_6 \\
f_7 = & x_{19} \oplus x_2 \oplus x_8 & f_{17} = & x_4 \\
f_8 = & x_{12} \oplus x_{13} \oplus x_1 \oplus x_{11} \oplus x_{16} & f_{18} = & x_3 \odot \mathrm{neg}(x_{10}) \odot \mathrm{neg}(x_{14}) \\
f_9 = & \mathrm{c} & f_{19} = & x_1 \oplus x_{13} \oplus x_{12} \oplus x_{16} \oplus x_{11} \\
f_{10} = & x_2 \odot \mathrm{neg}(x_8) \odot \mathrm{neg}(x_7) \odot \mathrm{neg}(x_5) & &
\end{array}
$$

Following the reductions in § 4.1, we start by replacing $x_1 = x_6 = x_9 = c$, $x_3 = x_7$, $x_{17} = x_4$ and removing f_1, f_3, f_6, f_9 and f_{17}. As x_{15} is not involved in any equation, we set $x_{15} = (x_{18} \odot x_5 \odot x_{17}) \oplus (x_5 \odot x_{17})$ and remove f_{15}. We then go further by replacing $x_{11} = x_{12} = x_{16} = c^2$, $x_{19} = x_8$ and removing f_{11}, f_{12}, f_{16} and f_{19}. Moreover, as $x_{10} = x_2 \odot \mathrm{neg}(x_8) \odot \mathrm{neg}(x_7) \odot \mathrm{neg}(x_5)$ we can replace this expression in f_{14}.

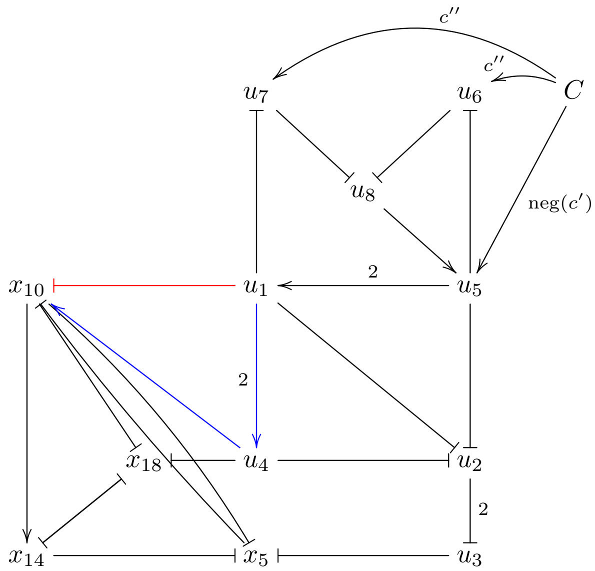
$$
\begin{array}{rl c rl}
f_2 = & x_8 \oplus x_{19} & & f_2 = & x_8 \oplus x_8 \\
f_4 = & (x_{19} \odot x_2 \odot x_7) \oplus (x_8 \odot x_2 \odot x_7) & & f_4 = & (x_2 \odot x_7 \odot x_8) \oplus (x_2 \odot x_7 \odot x_8) \\
f_5 = & x_4 \odot \mathrm{neg}(x_{10}) \odot \mathrm{neg}(x_{14}) & & f_5 = & x_4 \odot \mathrm{neg}(x_{10}) \odot \mathrm{neg}(x_{14}) \\
f_7 = & x_{19} \oplus x_2 \oplus x_8 & & f_7 = & x_2 \oplus x_8 \oplus x_8 \\
f_8 = & x_{12} \oplus x_{13} \oplus c \oplus x_{11} \oplus x_{16} & \rightsquigarrow & f_8 = & x_{13} \oplus c \oplus c^2 \oplus c^2 \oplus c^2 \\
f_{10} = & x_2 \odot \mathrm{neg}(x_8) \odot \mathrm{neg}(x_7) \odot \mathrm{neg}(x_5) & & f_{10} = & x_2 \odot \mathrm{neg}(x_8) \odot \mathrm{neg}(x_7) \odot \mathrm{neg}(x_5) \\
f_{11} = & c^2 & & f_{13} = & (x_8 \odot \mathrm{neg}(c^2)) \oplus (x_2 \odot \mathrm{neg}(c^2)) \\
f_{12} = & c^2 & & f_{14} = & x_{10} \odot \mathrm{neg}(x_{18}) \\
f_{13} = & (x_8 \odot \mathrm{neg}(x_{16})) \oplus (x_2 \odot \mathrm{neg}(x_{16})) & & f_{18} = & x_7 \odot \mathrm{neg}(x_{10}) \odot \mathrm{neg}(x_{14}) \\
f_{14} = & x_{10} \odot \mathrm{neg}(x_{18}) & & & \\
f_{16} = & c^2 & & & \\
f_{18} = & x_7 \odot \mathrm{neg}(x_{10}) \odot \mathrm{neg}(x_{14}) & & & \\
f_{19} = & c \oplus x_{13} \oplus x_{12} \oplus x_{16} \oplus x_{11} & & &
\end{array}
$$

We get the following $\odot$-neg network function, where we call $c' = c \oplus c^2 \oplus c^2 \oplus c^2$, $c'' = \mathrm{neg}(c^2)$:

$$
\begin{array}{rl@{\qquad}rl}
\overline{f}_2 = & \mathrm{neg}(u_1) & \overline{f}_{u_1} = & \mathrm{neg}(x_8)^2 \\
\overline{f}_4 = & \mathrm{neg}(u_3) & \overline{f}_{u_2} = & x_2 \odot x_7 \odot x_8 \\
\overline{f}_5 = & x_4 \odot \mathrm{neg}(x_{10}) \odot \mathrm{neg}(x_{14}) & \overline{f}_{u_3} = & \mathrm{neg}(u_2)^2 \\
\overline{f}_7 = & \mathrm{neg}(u_4) & \overline{f}_{u_4} = & u_1 \odot \mathrm{neg}(x_2) \\
\overline{f}_8 = & \mathrm{neg}(u_5) & \overline{f}_{u_5} = & \mathrm{neg}(x_{13}) \odot \mathrm{neg}(c') \\
\overline{f}_{10} = & x_2 \odot \mathrm{neg}(x_8) \odot \mathrm{neg}(x_7) \odot \mathrm{neg}(x_5) & \overline{f}_{u_6} = & x_8 \odot c'' \\
\overline{f}_{13} = & \mathrm{neg}(u_8) & \overline{f}_{u_7} = & x_2 \odot c'' \\
\overline{f}_{14} = & x_{10} \odot \mathrm{neg}(x_{18}) & \overline{f}_{u_8} = & \mathrm{neg}(u_6) \odot \mathrm{neg}(u_7) \\
\overline{f}_{18} = & x_7 \odot \mathrm{neg}(x_{10}) \odot \mathrm{neg}(x_{14}) & &
\end{array}
$$

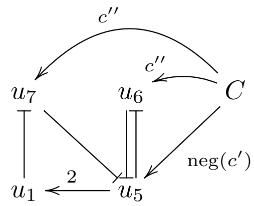
Again following the reductions in § 4.1, we replace $x_2 = \mathrm{neg}(u_1)$, $x_4 = \mathrm{neg}(u_3)$, $x_7 = \mathrm{neg}(u_4)$, $x_8 = \mathrm{neg}(u_5)$, $x_{13} = \mathrm{neg}(u_8)$, and remove $\overline{f_2}$, $\overline{f_4}$, $\overline{f_7}$, $\overline{f_8}$, $\overline{f_{13}}$. We call g_i the functions obtained after applying the reductions to $\overline{f_i}$.

$$
\begin{aligned}
g_5 &= \mathrm{neg}(x_{10}) \odot \mathrm{neg}(x_{14}) \odot \mathrm{neg}(u_3)\\
g_{10} &= \mathrm{neg}(x_5) \odot \mathrm{neg}(u_1) \odot u_4 \odot u_5\\
g_{14} &= x_{10} \odot \mathrm{neg}(x_{18})\\
g_{18} &= \mathrm{neg}(x_{10}) \odot \mathrm{neg}(x_{14}) \odot \mathrm{neg}(u_4)\\
g_{u_1} &= u_5^2\\
g_{u_2} &= \mathrm{neg}(u_1) \odot \mathrm{neg}(u_4) \odot \mathrm{neg}(u_5)\\
g_{u_3} &= \mathrm{neg}(u_2)^2\\
g_{u_4} &= u_1^2\\
g_{u_5} &= u_8 \odot \mathrm{neg}(c')\\
g_{u_6} &= \mathrm{neg}(u_5) \odot\ c''\\
g_{u_7} &= \mathrm{neg}(u_1) \odot\ c''\\
g_{u_8} &= \mathrm{neg}(u_6) \odot \mathrm{neg}(u_7)
\end{aligned}
$$



In order to further reduce the fixed point equations, we now apply the reduction in Proposition 26 to node 10 with the aid of the arrows depicted with red and blue in the network digraph on the right. We then get $x_{10} = 0$ and by replacing this information in g_{14} we also get $x_{14} = 0$. We make the corresponding replacements and then remove g_{10} and g_{14}. We further replace $x_5 = \mathrm{neg}(u_3)$ and $x_{18} = \mathrm{neg}(u_4)$, and remove g_5 and g_{18}. As u_3 is not involved in any of the remaining equations, we set $u_3 = \mathrm{neg}(u_2)^2$ and remove g_{u_3}. With the same reasoning, we replace $u_2 = \mathrm{neg}(u_1) \odot \mathrm{neg}(u_4) \odot \mathrm{neg}(u_5)$ and remove g_{u_2}, and $u_4 = u_1^2$ and remove g_{u_4}. We can moreover replace $u_8 = \mathrm{neg}(u_6) \odot \mathrm{neg}(u_7)$, remove g_{u_8}. Call h_i the functions obtained after applying the proposed reductions:

$$
(15)\qquad
\begin{aligned}
h_{u_1} &= u_5^2\\
h_{u_5} &= \mathrm{neg}(u_6) \odot \mathrm{neg}(u_7) \odot \mathrm{neg}(c')\\
h_{u_6} &= \mathrm{neg}(u_5) \odot\ c''\\
h_{u_7} &= \mathrm{neg}(u_1) \odot\ c''
\end{aligned}
$$



We will refer to system (15) as the *reduced system*.

So far we have $x_1 = x_6 = x_9 = c$, $x_2 = \mathrm{neg}(u_1)$, $x_4 = x_5 = x_{17} = \mathrm{neg}(u_3)$, $x_3 = x_7 = x_{18} = \mathrm{neg}(u_4)$, $x_8 = x_{19} = \mathrm{neg}(u_5)$, $x_{10} = x_{14} = 0$, $x_{11} = x_{12} = x_{16} = c^2$, $x_{13} = \mathrm{neg}(u_8)$, $x_{15} = (x_{18} \odot x_5 \odot x_{17}) \oplus (x_5 \odot x_{17})$, and $u_2 = \mathrm{neg}(u_1) \odot \mathrm{neg}(u_4) \odot \mathrm{neg}(u_5)$, $u_3 = \mathrm{neg}(u_2)^2$, $u_4 = u_1^2$, and $u_8 = \mathrm{neg}(u_6) \odot \mathrm{neg}(u_7)$.

Note that $c^2 = \max\{0, 2c-1\}$. Then,

- If $c \leq \frac{1}{2}$, then $c^2 = 0$ and $c' = c$, $c'' = 1$. We obtain then $u_6 = \mathrm{neg}(u_5)$, $u_7 = \mathrm{neg}(u_1)$ and we can remove h_{u_6} and h_{u_7} and get $h_{u_1} = u_5^2$, $h_{u_5} = u_5 \odot u_1 \odot \mathrm{neg}(c)$.

  By going one step further and replacing $u_1 = u_5^2$ we are left with only one fixed point equation:
  $$u_5 = u_5^3 \odot \mathrm{neg}(c).$$
  - If $u_5 \leq \frac{2}{3}$, then $u_5^3 = 0$ an then necessarily $u_5 = 0$. This gives the following fixed point:
    $$x_1 = x_6 = x_9 = c,\ x_{10} = x_{11} = x_{12} = x_{14} = x_{16} = 0,$$
    $$x_2 = x_3 = x_4 = x_5 = x_7 = x_8 = x_{13} = x_{15} = x_{17} = x_{18} = x_{19} = 1.$$
  - If $u_5 > \frac{2}{3}$ then $u_5 = \max\{0, 3u_5 + (1-c) - 3\} = 3u_5 + (1-c) - 3$ $\Leftrightarrow\ 2u_5 = 2 + c\ \Leftrightarrow\ u_5 = 1 + \frac{c}{2}$. This can only happen if $u_5 = 1$ and $c = 0$, and then, $c' = 0$, $c'' = 1$, and by substitution we obtain $x_i = 0$ for $i \in \{1, \cdots, 19\}$.

- If $c > \frac{1}{2}$, then $c^2 = 2c - 1$ and $c' = \min\{1, 7c - 3\}$.
  - If moreover $c \geq \frac{4}{7}$, then $c' = 1$, $\mathrm{neg}(c') = 0$ and this gives $u_5 = 0$. We deduce then $u_1 = 0$, $u_6 = u_7 = c'' = \mathrm{neg}(c^2) = 2 - 2c$ and $u_8 = c^4 = \max\{0, 4c - 3\}$. By replacing these values we obtain the unique fixed point:

$$x_1 = x_6 = x_9 = c,\ x_{11} = x_{12} = x_{16} = 2c - 1,\ x_{13} = \mathrm{neg}(c^4),\ x_{10} = x_{14} = 0,$$

$$x_2 = x_3 = x_4 = x_5 = x_7 = x_8 = x_{15} = x_{17} = x_{18} = x_{19} = 1.$$

  - If $c \in (\frac{1}{2}, \frac{4}{7})$, then $0 < c^2 = 2c - 1 < \frac{1}{7}$ and $c'' = \mathrm{neg}(c^2) = 2(1 - c)$. It is easy to check that $u_1 = u_5 = 0$, $u_6 = u_7 = 2(1 - c)$ is a fixed point of the reduced system and, by replacing, we obtain:

$$x_1 = x_6 = x_9 = c,\ x_{11} = x_{12} = x_{16} = 2c - 1,\ x_{10} = x_{14} = 0,$$

$$x_2 = x_3 = x_4 = x_5 = x_7 = x_8 = x_{13} = x_{15} = x_{17} = x_{18} = x_{19} = 1.$$

Indeed, we checked that is the unique fixed point in two cases. With $m = 9$ and $c = 5/9$ (and so $c' = c'' = 8/9$), and with $m = 13$ and $c = 7/13$ (and $c' = 10/13, c'' = 12/13$), using the implementation by A. Galarza Rial [10]. In this not so big example, these computations are heavy to be made by hand.

We will explain in the next section how to systematically compute the fixed points.

5. Computing fixed points

Given $X_m = \{0, \frac{1}{m}, \ldots, \frac{m-1}{m}, 1\}$ with $m \geq 1$ arbitrary, and a multivalued network F over X_m. We will now focus on effectively finding the fixed points of the system.

Theorem 29. *Given a ($\odot - \mathrm{neg}$)-network function $F = (f_1, \ldots, f_n)$, every function*

$$f_i = \bigodot_{j \in A_i} x_j^{p_{ij}} \odot \bigodot_{j \in B_i} (\mathrm{neg}(x_j))^{q_{ij}} \odot c_i,\ A_i, B_i \subseteq \{1, \ldots, n\},\ A_i \cap B_i = \emptyset,$$

with $p_{ij}, q_{i,j}$ integers in $[1, m]$, can be written as

$$f_i = max\{0, \ell_i(x)\}, \tag{16}$$

where

$$\begin{aligned} \ell_i(x) &= \ell_i'(x) + c_i^*, \quad c_i^* \in (\mathbb{Z}\frac{1}{m}) \cap \mathbb{Z}_{\leq 1}, \\ \ell_i'(x) &= \sum_{j \in A_i} p_{ij} x_j - \sum_{j \in B_i} q_{ij} x_j, \quad \textit{and} \quad c_i^* = c_i - \sum_{j \in A_i} p_{ij}. \end{aligned} \tag{17}$$

Proof. From item (ii) in Proposition 4 we deduce Equation (16) with

$$\begin{aligned} \ell_i(x) &= \sum_{j \in A_i} p_{ij} x_j + \sum_{j \in B_i} q_{ij}(1 - x_j) + c_i - (\sum_{j \in A_i} p_{ij} + \sum_{j \in B_i} q_{ij} + 1 - 1) \\ &= \sum_{j \in A_i} p_{ij} x_j - \sum_{j \in B_i} q_{ij} x_j + c_i - \sum_{j \in A_i} p_{ij}. \end{aligned}$$

As $c_i \in [0, 1] \cap \mathbb{Z}.\frac{1}{m}$ and $\sum_{j \in A_i} p_{ij} \in \mathbb{Z}_{\geq 0}$ we deduce $c_i^* \in (\mathbb{Z}\frac{1}{m}) \cap \mathbb{Z}_{\leq 1}$, which is what we wanted to prove. □

Lemma 30. *Any linear function of the form $\ell = \sum_{j \in A} p_j x_j - \sum_{j \in B} q_j x_j + (c - \sum_{j \in A} p_j)$, where $A, B \subseteq \mathbb{N}$ are disjoint finite sets, $p_j, q_j \leq m$, $p_j, q_j \in \mathbb{N}$ and $c \in X$, defines a $\odot - \mathrm{neg}$ function f.*

Proof. The function defined by $f = \bigodot_{j \in A} x_j^{p_j} \odot \bigodot_{j \in B} (\mathrm{neg}(x_j))^{q_j} \odot c$ verifies every hypothesis. Any other function verifying every hypothesis, would coincide with f on every point. □

Remark 31. We have already seen in Remark 11 that the expression of a $\odot$-neg function might not be unique in general. However, there are some cases where the expression of a $\odot$-neg function is indeed unique. Given $m \in \mathbb{N}$ and two expressions defining the same function on X_m:

$$\bigodot_{j\in A} x_j^{p_j} \odot \bigodot_{j\in B} \operatorname{neg}(x_j)^{q_j} = \bigodot_{j\in A'} x_j^{p'_j} \odot \bigodot_{j\in B'} \operatorname{neg}(x_j)^{q'_j}$$

with $A\cap B = A'\cap B' = \emptyset$, $1 \leq p_j, q_j, p_{j'}, q_{j'} \leq m$, it is straightforward to prove that $A = A'$, $B = B'$, $p_j = p'_j$ and $q_j = q'_j$.

We deduce from Theorem 29 the validity of Algorithm 2 below to compute the fixed points of any $(\odot - \operatorname{neg})$-network function $F : X_m^n \to X_m^n$.

First, let J be the set of indices i for which $f_i = \ell_i$ consists of a constant, a single variable or its negation. Of course for each of these linear functions we can write them as $\max\{0, \ell_i(x)\}$, but this would introduce unnecessary branchings in the computations. When J is non-empty, we can apply reductions (i) and (ii) in § 4.1 and pull apart the variables with indices in J. Note that if we start with an $\odot$-neg function, we obtain a reduced function $f_{red} : X_m^{n-|J|} \to X^{n-|J|}$ that is also of this form. Then, the number of regions to consider in Step 1 of the following Algorithm 2 is $2^{n-|J|}$. The linear systems to be solved in Step 2 of the Algorithm have also at most $n - |J|$ variables. In Step 3, we also add the values of the variables with indices in J to output the fixed points. To alleviate the notation, we assume that $J = \emptyset$ but the input of Algorithm 2 should be the reduced function f_{red} and not f.

Algorithm 2 We compute the fixed points of a $\odot$-neg network with at least one $\odot$ operation in each f_i.

Input: A $\odot$-neg network $F = (f_1, \ldots, f_n) : X_m^n \to X_m^n$, with $f_i = max\{0, \ell_i(x)\}$ as in (16).
Output: The fixed points of F.

Step 1: For each $I \subseteq \{1, \ldots, n\}$ consider the region $\mathcal{R}_I = \left\{ \begin{array}{l} \ell_i(x) \geq 0 \; i \in I \\ \ell_j(x) \leq 0 \; j \notin I \end{array} \right\}$.

Step 2: Solve for the linear system $\left\{ \begin{array}{rl} \ell_i(x) = x_i & i \in I \\ 0 = x_j & j \notin I \end{array} \right.$ In case there is a single (rational) solution, check if $x \in X_m^n$. In case the linear system has infinitely many rational solutions, change the variables to $y_i = mx_i$, translate the linear forms $\ell_i - x_i$ to integer linear forms $\bar{\ell}_i(y) - y_i$ and find the lattice points in the rational polytope

$$P_I = \{\bar{\ell}_i(y) - y_i = 0, 0 \leq y_i \leq m, i \in I, \text{ and } y_j = 0, j \notin I\}.$$

Translate back the solutions to solutions $x \in X_m^n$.

Step 3: For each I, and a solution $x \in X_m^n$ of the linear system, check if $\ell_j(x) \leq 0$ for all $j \notin I$. If this is satisfied, then x is a fixed point of F in $\mathcal{R}_I$.

Now, how does one algorithmically find all the lattice points in rational polytopes? A good implementation of Barvinok's algorithm mentioned in the Introductin was provided in the free software Latte [11], explained in the paper [7] (now available in SageMath). We used the free implementation in [9], explained in the report [24] that counts the lattice points and also provides them explicitly.

Note that for any $m \geq 1$ there are at most $2^{n-|J|}$ regions $\mathcal{R}_I$ (independently of m), while the exhaustive search of fixed points requires $(m+1)^n$ initial states. Moreover, the computations are parallelizable for different I and further simplifications and improvements in the tree of computations might be certainly done in particular instances.

The best possible case is when the square linear systems in Step 2 have a unique solution (necessarily rational). In this case, solving each system has complexity of order $O(|I|^3)$ and we then check if its solution lies in X_m^n. We give in Theorem 33 below an easy general condition under which this happens. We remark however that when the rank is not maximum, the complexity of Step 2 might raise (up to the order of $(m+1)^n$).

Definition 32. Consider an $\odot$-neg network function $F = (f_1, \ldots, f_n)$ with at least one $\odot$ operation in each f_i. Given a $I \subseteq \{1, \ldots, n\}$ of cardinality s we denote by M_I the $s \times s$ matrix with both rows and columns indicated by I and with entries:

$$(M_I)_{ij} = \begin{cases} \ell_{ii} - 1 & \text{if } i = j, \\ \ell_{ij} & \text{if } i \neq j \end{cases} \tag{18}$$

Theorem 33. *Given a* ($\odot - \mathrm{neg}$)*-network defined by* ℓ_i $i \in \{1, \dots, n\}$*. Let* $J \subset \{1, \dots, n\}$ *denote the set of indices* i *for which* f_i *has no* $\odot$ *operations. If for* $I \subseteq \{1, \dots, n\}$ *with* $I \subset J^c$*, the matrix* M_I *defined in* (18) *satisfies* $\det(M_I) \neq 0$*, then there is at most one fixed point in the region* $\mathcal{R}_I$*.*

The proof of Theorem 33 is immediate as M_I is the matrix of the linear system $\begin{cases} \ell_i(x) = x_i & i \in I \\ 0 = x_j & j \notin I \end{cases}$. The following corollary is also immediate.

Corollary 34. *Let* $L' \in \mathbb{Z}^{n \times n}$ *be the matrix of coefficients of the linear forms* $\ell'_1, \dots, \ell'_n$ *as in Theorem 29 and* Id_n *be the* $n \times n$ *identity matrix. If* $\det(L' - Id_n) \neq 0$ *then there exists at most a single fixed point with all nonzero coordinates.*

We end this section showing how to compute the fixed points using our app [10].

Example 35. We consider again the biological example in § 4.2, in case $m = 9, c = 5/9, c' = c'' = 8/9$. The reduced system (15) has four variables (ordered u_1, u_5, u_6, u_7). It can be written as:

$$\begin{aligned} h_{u_1} &= \max\{0, 2u_5 - 1\} \\ h_{u_5} &= \max\{0, -u_6 - u_7 + 1/9\} \\ h_{u_6} &= \max\{0, -u_5 + 8/9\} \\ h_{u_7} &= \max\{0, -u_1 + 8/9\} \end{aligned}$$

After installing our app *Fixed points of MV networks* from the github repository [10], input the coefficients of the linear forms in each function, multiplying *only* the constant terms by m, to get the output as in Figure 3. Of course, there is a lot of room to improve this implementation.

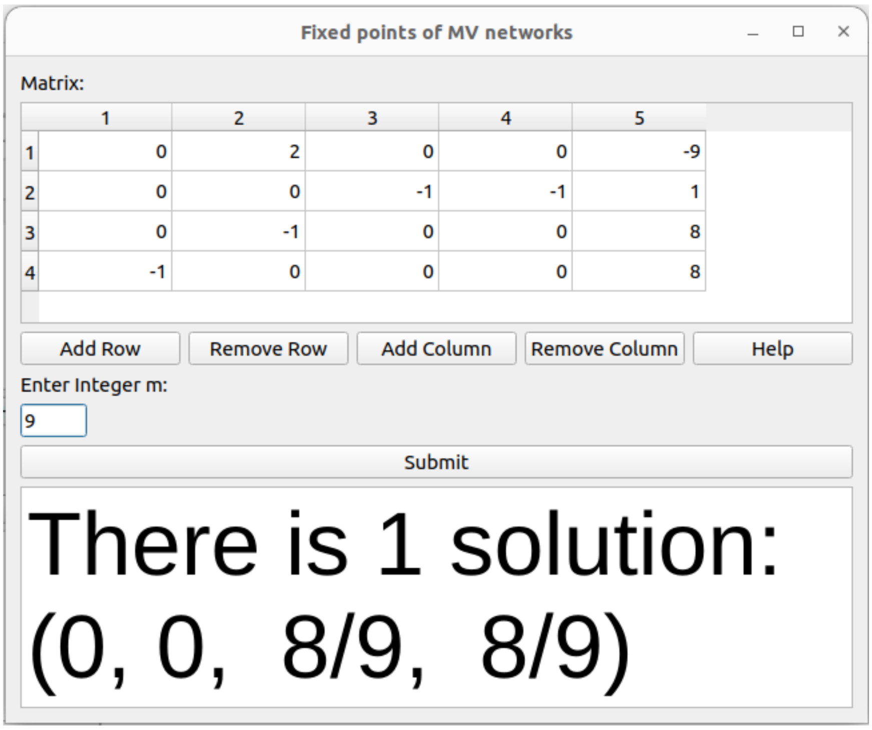


FIGURE 3. Screenshot of the app [10]

6. Application to the Model for Denitrification in Pseudomonas aeruginosa

In [1] the authors try to unveil the environmental factors that contribute to greenhouse gas N_2O accumulation, especially in Lake Erie (Laurentian Great Lakes, USA). Their model predicts the long-term behavior of the denitrification pathway, it has three Boolean external parameters (O_2, PO_4, NO_3), four Boolean variables ($PhoRB$, $PhoPQ$, $PmrA$,Anr), and eleven ternary variables ($NarXL$, Dnr, $NirQ$, nar, nir, nor, nos, NO_2, NO, N_2O, N_2). In our setting, we assume that they take values in X_2. The corresponding update rules in [1] are:

$$
\begin{array}{lll}
f_{PhoRB} = \mathrm{neg}(PO_4), & f_{NarXL} = \min\{Anr, NO_3\}, & f_{NO} = \min\{NO_2, nir\}, \\
f_{PhoPQ} = PhoRB, & f_{nir} = \max\{NirQ, \min\{Dnr, NO\}\}, & f_{N_2O} = \min\{NO, nor\}, \\
f_{PmrA} = \mathrm{neg}(PhoPQ), & f_{nor} = \max\{NirQ, \min\{Dnr, NO\}\}, & f_{N_2} = \min\{N_2O, nos\}, \\
f_{Anr} = \mathrm{neg}(O_2), & f_{nos} = \min\{Dnr, NO\}. &
\end{array}
$$

As we pointed out in our Introduction, they cannot accurately express four variables (Dnr, $NirQ$, nar and NO_2) using *only* the operators MAX, MIN and NOT (indicated with a $*$ in [1, Table 3]), and are then forced to show all the possible values in supplementary tables. From the transition tables of each of the 15 variables, they build a polynomial dynamical system and compute the fixed points of the system (with a software based on Gröbner basis computations) under the environmental conditions of interest that emerge after fixing the values of the external parameters.

To further show the strength of the tools we developed in this paper, we implemented their model in our setting. We can translate the functions shown above to operations with $\odot$ and $\oplus$ using Proposition 4. For the remaining functions, f_{Dnr}, f_{NirQ}, f_{nar} and f_{NO_2}, we considered the supplementary tables in [1] without the smoothing process since the fixed points are not changed (see Lemma 14). For instance, the values for $NirQ$ are shown in Figure 4. We can see that the fourth transition $(NarXL = 1, Dnr = 0) \mapsto NirQ = \frac{1}{2}$ does not agree with the function $\max\{NarXL, Dnr\}$ in [1]. However, we can represent any multivalued function in our framework. There is always a standard alternative to do this by building the interpolation function in Theorem 10. In this case, there is a much better alternative: find a more intuitive and simple expression where Dnr is an activator and $NarXL$ is a mild activator (see Motifs 1 and 4 in § 2.3). See Figure 4.

NarXL	Dnr	NirQ
0	0	0
0	1/2	1/2
0	1	1
1	0	1/2
1	1/2	1
1	1	1

Update rule in [1]: $*\max\{NarXL, Dnr\}$. The value $1/2$ at $(0, 1)$ in the fourth row is not obtained with this expression.
Interpolation expression:

$$
\begin{aligned}
f_{NirQ} = {} & \frac{1}{2} \odot neg(NarXL)^2 \odot neg(neg(Dnr) \odot \frac{1}{2})^2 \odot neg(Dnr \odot \frac{1}{2})^2 \\
& \oplus neg(NarXL)^2 \odot Dnr^2 \\
& \oplus \frac{1}{2} \odot NarXL^2 \odot neg(Dnr)^2 \\
& \oplus NarXL^2 \odot neg(neg(Dnr) \odot \frac{1}{2})^2 \odot neg(Dnr \odot \frac{1}{2})^2 \\
& \oplus NarXL^2 \odot Dnr^2.
\end{aligned}
$$

Simpler expression: $f_{NirQ} = (NarXL \ominus \frac{1}{2}) \oplus Dnr$.

Figure 4. $NirQ$ update rule. The table on the left is the multivalued non-smooth version of S3 Table in [1]. On the right: the update rule approximation shown in [1]; the update rule obtained by interpolation from the table according to Theorem 10; and the expression obtained by considering Dnr as an activator and $NarXL$ as a mild activator. The last two expressions coincide with the values in the table.

We can represent $f_{NirQ} = (NarXL \ominus \frac{1}{2}) \oplus Dnr$, $f_{Dnr} = (Anr \ominus \frac{1}{2}) \oplus (\mathrm{neg}(PmrA) \odot Anr \odot NarXL)$, and $f_{NO_2} = NO_3 \odot nar$ which have intuitive interpretations.

In our setting, we fix the value of m for all functions in the network. In this model, most variables take three values, corresponding to the choice $m = 2$. Their Boolean variables can be reduced to three.

Instead of assigning them arbitrarily values in $X_m = \{0, 1/2, 1\}$, for instance, the values 0 or 1, it is more reasonable in our setting to consider the 8 different resulting models in the reminder 3-valued variables, for each of the possible values of the Boolean variables.

We then set the values of each Boolean external parameter (O_2, PO_4, NO_3) and study the corresponding fixed points of each of the eight possible systems separately. We expand on the details in the Appendix. The two relevant conditions for denitrification with low O_2 are: $O_2 = PO_4 = 0$, $NO_3 = 1$ (considered "the perfect condition for denitrification") and $O_2 = 0$, $PO_4 = NO_3 = 1$ (the denitrification condition disrupted by PO_4 availability). There are other two conditions with low O_2 that the authors in [1] disregard because they are less likely in fresh-waters: $O_2 = PO_4 = NO_3 = 0$, and $O_2 = NO_3 = 0$, $PO_4 = 1$, but nonetheless we computed the fixed points in our setting since they are easy to find. We also address the remaining conditions, with high O_2 (the "aerobic conditions"). The fixed points we computed in all cases coincide with those found in [1].

Also, as remarked before, it is not necessary to implement the continuous versions of the update functions since the fixed points are not changed. We give more details about our implementation of this model in the Appendix.

7. Discussion

Time- and valued-discrete models, such as the multi-valued Boolean networks studied here, are broadly applicable in biology and beyond, such as engineering, computer science, and physics. Mathematically, they are simply set functions on strings of a given finite length over a finite alphabet. As such, there is no mathematical structure that could be used to classify or analyze them. Their dynamics is captured by a directed graph, exponential in the number of variables of the model. Traditionally, exhaustive simulation or sampling the state space, for larger models, was the only approach to characterize model dynamics. One strategy to make mathematical tools available for this purpose is based on the observation that any function $f : k^n \longrightarrow k$ can be expressed as a polynomial function over a finite field k. This allows one to use tools from computer algebra. For instance, primary decomposition of monomial ideals can be used for reverse-engineering of gene regulatory networks from time courses of experimental observations [19, 23].

Here, we propose a different approach to bring mathematical tools to bear on the problem of analysis of a broad class of discrete models. Using multi-valued logic, we connect this class of models to problems in combinatorics and enumerative geometry, in particular the calculation of rational points in polytopes. This is combined with a model reduction method that generally results in substantially smaller models with steady states corresponding one-to-one with the steady states of the original model. The reduction steps are ad hoc, and there is no theory behind them that would allow the derivation of a minimal reduced model or even a result about whether the order in which the reductions are applied matters in terms of the final reduced model. We believe that there is further interesting mathematics to be discovered in this context. The next important step is to use the tools we present to understand the full dynamics of the models.

Generally, we believe that multi-valued networks and related models are rich and fascinating mathematical objects at the cross roads of combinatorics, algebra, and dynamical systems. They can be studied from the point of view of applications, or for purely mathematical reasons, or both, as in [12].

Acknowledgements

We are grateful to Alejandro Petrovich and to Patricio Díaz Varela for the references and explanations about MV-algebras in logic.

References

[1] Arat S., Bullerjahn G.S., Laubenbacher R. (2015) *A network biology approach to denitrification in Pseudomonas aeruginosa.* PLoS One. 10(2):e0118235.

[2] Bardet M., Faugère J.C., Salvy B., Spaenlehauer P.J. (2013) *On the complexity of solving quadratic boolean systems.* J. Complex., 29(1), 53–75.

[3] Barvinok A. (1993). *A polynomial time algorithm for counting integral points in polyhedra when the dimension is fixed.* In: 34th Annual Symposium on Foundations of Computer Science, 566–572. IEEE.

[4] Brion M. (1988) *Points entiers dans les polyèdres convexes.* Ann. Sci. École Norm. Sup. (4)21, no.4, 653–663.
[5] Chifman J., Arat S., Deng Z., Lemler E., Pino J. C., Harris L. A., Kochen M., Lopez C. F., Akman S.A., Torti F.M., Torti S.V., Laubenbacher R. (2017) *Activated Oncogenic Pathway Modifies Iron Network in Breast Epithelial Cells: A Dynamic Modeling Perspective.* PLoS Comput. Biol. 13(2): e1005352.
[6] Cignoli R. L. O., D'Ottaviano I. M. L., Mundici D. (2000) *Algebraic foundations of many-valued reasoning*. Trends in Logic—Studia Logica Library, Vol. 7, Kluwer Academic Publishers, Dordrecht.
[7] De Loera J. A., Hemmecke R., Tauzer J., Yoshida R. (2004) *Effective Lattice Point Counting in Rational Convex Polytopes*. J. Symb. Comput. 38(4), 1273–1302.
[8] Epstein G. (1960) *The lattice theory of Post algebras.* Trans. Am. Math. Soc. 95(2), 300–317.
[9] Free software *Barvinok*, by Verdoolaege S. and collaborators. Available at: `https://barvinok.sourceforge.io/`
[10] Free software *Fixed points of MV networks*, by Galarza Rial A., available at `https://github.com/ayegari89/multivalued_fixed_points`
[11] Free software *Latte*, by de Loera J., Köppe M. and collaborators, available at: `https://www.math.ucdavis.edu/~latte/`.
[12] Kadelka C., Wheeler M., Veliz-Cuba A., Murrugarra D., Laubenbacher R. (2023) *Modularity of biological systems: a link between structure and function*. J. R. Soc. Interface 20: 20230505.
[13] Kauffman, S. (1969) *Homeostasis and Differentiation in Random Genetic Control Networks*. Nature 224, 177–178.
[14] Li, Z., Cheng, D. (2010) *Algebraic approach to dynamics of multivalued networks*. International Journal of Bifurcation and Chaos, 20(03), 561–582.
[15] Thomas R., (1973) *Boolean formalisation of genetic control circuits*. J. Theor. Biol. 42, 565–583.
[16] Thomas R. (1991) *Regulatory networks seen as asynchronous automata: a logical description*, J. Theor. Biol. 153, 1–23.
[17] Thomas R., d'Ari R. (1990) *Biological feedback*. CRC press.
[18] Tonello E. (2019) *On the conversion of multivalued to Boolean dynamics*. Discrete Appl. Math. 259, 193–204
[19] Veliz-Cuba A. (2012) *An Algebraic Approach to Reverse Engineering Finite Dynamical Systems Arising from Biology*. SIAM J. Appl. Dyn. Syst. 11(1), 31–48.
[20] Veliz-Cuba A., Jarrah A., Laubenbacher R. (2010) *Polynomial algebra of discrete models in systems biology*. Bioinformatics, 26, 1637–1643.
[21] Veliz-Cuba A, Aguilar B., Laubenbacher R. (2015) *Dimension reduction of AND-NOT network model*. Electronic Notes in Theoretical Computer Science 316 , 83–95.
[22] Veliz-Cuba A., Buschur K., Hamershock R., Kniss A., Wolff E., Laubenbacher, R. (2013) *AND-NOT logic framework for steady state analysis of Boolean network models*. Appl. Math. Inf. Sci. 7(4): 1263–1274.
[23] Veliz-Cuba A, Newsome-Slade V., Dimitrova E. (2024) *A Unified Approach to Reverse Engineering and Data Selection for Unique Network Identification*. SIAM J. Appl. Dyn. Syst. 23(1), 592–615.
[24] Verdoolaege S., Woods K., Bruynooghe M., Cools R. (2005) *Computation and Manipulation of Enumerators of Integer Projections of Parametric Polytopes.* Report CW 392, March. Katholieke Universiteit Leuven Department of Computer Science.

Appendix A. Multivalued model for *P. Aeruginosa*

As we mentioned in Section 6, we translated the MIN/MAX functions in [1, Table 3] using Proposition 4. For those variables the authors could not explain with MIN/MAX expressions, we represent exactly the update function presented in the supplementary tables in [1] (without the smoothing process) with the following expressions over X_2:

$$\begin{aligned}
f_{NirQ} &= (NarXL \ominus \frac{1}{2}) \oplus Dnr,\\
f_{Dnr} &= (Anr \ominus \frac{1}{2}) \oplus (\mathrm{neg}(PmrA) \odot Anr \odot NarXL),\\
f_{NO_2} &= NO_3 \odot nar,\\
f_{nar} &= \left[\left(NarXL \oplus \frac{1}{2}\right)^2 \odot \left(Dnr \oplus \frac{1}{2}\right)^2 \odot \left(NO \oplus \frac{1}{2}\right)^2\right] \oplus\\
&\quad \oplus \left[\frac{1}{2} \odot \left(NarXL \oplus \left(Dnr^2 \odot NO^2\right)\right)\right].
\end{aligned}$$

We then set the values of each Boolean external parameter (O_2, PO_4, NO_3) and obtain eight different systems. For each of them we compute the fixed points. The condition with low O_2, low PO_4 and high NO_3 (O_2 =PO_4 $= 0$, NO_3 $= 1$) is considered in [1] "the perfect condition for denitrification". The condition with low O_2, high PO_4 and high NO_3 (i.e. $O_2 = 0$, PO_4 =NO_3 $= 1$) corresponds to the denitrification condition disrupted by PO_4 availability. There are two conditions that they disregard because they are less likely in fresh-waters: low O_2, low PO_4 and low NO_3 (O_2 =PO_4 =NO_3 $= 0$), and low O_2, high PO_4 and low NO_3 (O_2=NO_3 $= 0$, PO_4 $= 1$), but as the fixed points can be computed easily in our setting (and are unique in each case) we show them in Table 1. The remaining conditions are considered as "aerobic conditions" and their fixed points (which are unique for each case) are also immediate to obtain and are shown in Table 1.

External parameters			Steady State														
O_2	PO_4	NO_3	PhoRB	PhoPQ	PmrA	Anr	NarXL	Dnr	NirQ	nar	nir	nor	nos	NO_2	NO	N_2O	N_2
0	0	0	1	1	0	1	0	$\frac{1}{2}$	$\frac{1}{2}$	0	$\frac{1}{2}$	$\frac{1}{2}$	0	0	0	0	0
0	1	0	0	0	1	1	0	$\frac{1}{2}$	$\frac{1}{2}$	0	$\frac{1}{2}$	$\frac{1}{2}$	0	0	0	0	0
1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

TABLE 1. Steady states under the environmental conditions determined by the external parameters. The first two conditions are disregarded in [1] because they are not biologically relevant. The last four cases (the "aerobic conditions") coincide with [1, Fig. 2].

For the two cases of interest $O_2 = 0$, $PO_4 = 0$, $NO_3 = 1$ ("the perfect condition for denitrification") and $O_2 = 0$, $PO_4 = 1$, $NO_3 = 1$ ("the denitrification condition disrupted by PO_4 availability") we built the $\odot - \mathrm{neg}$ system in Theorem 24 by adding 20 new variables. We apply the reductions in § 4.1 and obtain, in each case, reduced systems for the fixed points.

The case $O_2 = 0$, $PO_4 = 1$, $NO_3 = 1$ renders a system of three equations in three variables which has a unique solution and determines the fixed point:

PhoRB	PhoPQ	PmrA	Anr	NarXL	Dnr	NirQ	nar	nir	nor	nos	NO_2	NO	N_2O	N_2
1	1	0	1	1	1	1	1	1	1	1	1	1	1	1

The case $O_2 = 0$, $PO_4 = 0$, $NO_3 = 1$ gives rise to a system of five equations in five variables which has a unique solution and determines the fixed point:

PhoRB	PhoPQ	PmrA	Anr	NarXL	Dnr	NirQ	nar	nir	nor	nos	NO_2	NO	N_2O	N_2
0	0	1	1	1	$\frac{1}{2}$	1	1	1	1	$\frac{1}{2}$	1	1	1	$\frac{1}{2}$

All the fixed points we obtain coincide with the fixed points obtained in [1]. We have also implemented their model using the interpolating functions in Theorem 10 for f_{Dnr}, f_{NirQ}, f_{nar} and f_{NO_2}, and computed the fixed points of each of the eight systems determined by fixing the values of the external parameters. The steady states in Table 1 were obtained straightforwardly. We used the implementation [10] (without considering any network reductions) for the remaining two cases.

[1] DEPARTAMENTO DE MATEMÁTICA Y CIENCIAS, UNIVERSIDAD DE SAN ANDRES, (B1644BID) VICTORIA, PROV. B. AIRES, ARGENTINA.

[2] INSTITUTO DE INVESTIGACIONES MATEMÁTICAS "LUIS A. SANTALÓ" (UBA-CONICET), C1428EGA BUENOS AIRES, ARGENTINA.

[3] DEPARTAMENTO DE MATEMÁTICA, FCEN, UNIVERSIDAD DE BUENOS AIRES, CIUDAD UNIVERSITARIA, PAB. I, C1428EGA BUENOS AIRES, ARGENTINA.

[4] DEPARTAMENTO DE CIENCIAS EXACTAS, CICLO BÁSICO COMÚN, UNIVERSIDAD DE BUENOS AIRES, C1405CAE BUENOS AIRES, ARGENTINA

[5] LABORATORY FOR SYSTEMS MEDICINE, DEPARTMENT OF MEDICINE, UNIVERSITY OF FLORIDA, GAINESVILLE, FL, USA.

Email address: jgarciagalofre@udesa.edu.ar, mpmillan@dm.uba.ar, ayegari89@gmail.com, laubenbacher@medicine.ufl.edu, alidick@dm.uba.ar