DFG-Research Center MATHEON
Mathematics for Key Technologies

Computing Symbolic Steady States of Boolean Networks

Hannes Klarner **Alexander Bockmayr** **Heike Siebert**

MATHEON **preprint**
http://opus4.kobv.de/opus4-matheon

Preprint **April 2014**

Computing Symbolic Steady States of Boolean Networks

Hannes Klarner, Alexander Bockmayr and Heike Siebert

Freie Universität Berlin, FB Mathematik und Informatik
Arnimallee 6, 14195 Berlin, Germany
Hannes.Klarner@FU-Berlin.de

Abstract. Asymptotic behavior is often of particular interest when analyzing asynchronous Boolean networks representing biological systems such as signal transduction or gene regulatory networks. Methods based on a generalization of the steady state notion, the so-called symbolic steady states, can be exploited to investigate attractor properties as well as for model reduction techniques conserving attractors. In this paper, we propose a novel optimization-based method for computing all maximal symbolic steady states and motivate their use. In particular, we add a new result yielding a lower bound for the number of cyclic attractors and illustrate the methods with a short study of a MAPK pathway model.

1 Introduction

Boolean network models have long since proved their worth in the context of modeling complex biological systems [1]. Of particular interest are number and size of attractors as well as their location in state space since these properties often relate well to important biological behavior. Different approaches are available to solve several variations of this problem (see e. g. [2] and references therein).

In this paper, we utilize the notion of symbolic steady state [3] and a generalization called seeds which represent dynamically closed subspaces of state space, i.e., subspaces that no trajectory can leave. This property can clearly be exploited for model reduction and attractor analysis as has been illustrated in [3]. However, related methods can only be useful in practice if identification of such symbolic steady states is not based on comprehensive state space analysis.

After providing the relevant terminology, we motivate our research by presenting some theoretical results concerning model reduction and number of attractors. We then introduce the *prime implicant graph* as an object capturing the essential dynamical information of a network and a related optimization-based approach allowing for efficient computation of maximal symbolic steady states. Lastly, we provide a short illustration using a MAPK pathway model.

1.1 Background

We consider variables from the Boolean domain $\mathbb{B} = \{0, 1\}$ where 1 and 0 represent the truth values *true* and *false*. A Boolean *expression* f over the variables

$V = \{v_1, \ldots, v_n\}$ is defined by a formula over the grammar

$$f ::= 0 \mid 1 \mid v \mid \overline{f} \mid f_1 \cdot f_2 \mid f_1 + f_2$$

where $v \in V$ signifies a variable, $\overline{f}$ the negation, $f_1 \cdot f_2$ the conjunction and $f_1 + f_2$ the (inclusive) disjunction of the expressions f, f_1 and f_2. Given an *assignment* $s : V \to \mathbb{B}$, an expression f can be evaluated to a value $f(s) \in \mathbb{B}$ by substituting the values $s(v)$ for the variables $v \in V$. If $f(s) = f(t)$ for all assignments $s, t : V \to \mathbb{B}$, we say f is *constant* and write $f = c$, with $c \in \mathbb{B}$ being the constant value. A *Boolean network* (V, F) consists of n variables $V = \{v_1, \ldots, v_n\}$ and n corresponding Boolean expressions $F = \{f_1, \ldots, f_n\}$ over V. In this context, an assignment $s : V \to \mathbb{B}$ is also called a *state* of the network and the *state space* $S = S(V)$ consists of all possible 2^n states. The expressions F can be thought of as a function $F : S \to S$ governing the network behavior. The *image* $F(s)$ of a state s under F is defined to be the state t that satisfies $t(v_i) = f_i(s)$. The *steady states* or *fixpoints* $\mathcal{S} = \mathcal{S}(V, F)$ of a Boolean network (V, F) are all states $s \in \mathcal{S}$ that satisfy $F(s) = s$. To illustrate these concepts we introduce a running example in Fig. 1.

$$\begin{array}{lll} f_1 = v_1 + v_2 & f_1(1101) = 1 + 1 = 1 & F(1101) = 1101 \\ f_2 = v_1 \cdot v_4 & f_2(1101) = 1 \cdot 1 = 1 & F(0000) = 0001 \\ f_3 = \overline{v_1} \cdot v_4 & f_3(1101) = \overline{1} \cdot 1 = 0 & \mathcal{S} = \{1101\} \\ f_4 = \overline{v_3} & f_4(1101) = \overline{0} = 1 & \end{array}$$

Fig. 1. (left) An example Boolean network with 4 variables. (middle) An example for how the Boolean expressions are evaluated for a given state $s = 1101$. We specify states by a sequence of n values that correspond to the variables in the order given in V, i.e., $s = 1101$ should be read as $s(v_1) = 1, s(v_2) = 1, s(v_3) = 0$ and $s(v_4) = 1$. (right) An example of the image $F(s)$ of two states 1101 and 0000 where the first is a steady state and the second is not.

2 Methods

2.1 Symbolic Steady States and Seeds

A *partial* or *symbolic state* of the network (V, F) is an assignment $p : U \to \mathbb{B}$ where $U \subseteq V$ is a subset of variables. The set $PS = PS(V, F)$ denotes all possible 3^n partial states of V. We denote the *domain* U of a partial state p by U_p. The partial state $p \in PS$ with $U_p = \varnothing$ is called the *empty partial state* and denoted by $p = \varnothing$. A partial state p references the *subspace* $S[p] := \{s \in S \mid \forall v \in U_p : s(v) = p(v)\}$ of S. The *size* $|p|$ of a partial state p is defined to be $|p| := |U_p|$. Two partial states $p, q \in PS$ are said to be *consistent* if for all $v \in U_p \cap U_q : p(v) = q(v)$. With the *ordering* $p \leq q$ iff p, q are consistent and

$U_p \subseteq U_q$, PS becomes a partially ordered set. We define the *union* $p \sqcup q$ of two consistent partial states p, q by $U_{p \sqcup q} := U_p \cup U_q$ and $p \sqcup q(v) := p(v)$, if $v \in U_p$, and $p \sqcup q(v) := q(v)$, otherwise. Every $\varnothing \neq p \in PS$ has a unique decomposition into $|p|$ partial states of size 1. For example, $0_2 1_3 1_4 = 0_2 \sqcup 1_3 \sqcup 1_4$.

Analogous to the evaluation $f(s)$ of an expression f at a state s we define the *expression* $f[p]$ obtained by substituting the values $p(v)$ for the variables $v \in U_p$ in f. The *image* $F[p]$ of a partial state under $F = \{f_1, \ldots, f_n\}$ is the partial state $q : U_q \to \mathbb{B}$ defined by $U_q := \{v_i \in V \mid f_i[p] \text{ is constant}\}$ and $q(v_i) := f_i[p]$, for all $v_i \in U_q$. If $U_q = \varnothing$, we have $F[p] = \varnothing$.

Like in [3], a *symbolic steady state* is a partial state p that satisfies $F[p] = p$, and a *seed* is a partial state p that satisfies $F[p] \geq p$. As we will see, these conditions ensure that the network components that belong to U_p stay fixed when regarding the network dynamics in the subspace $S[p]$. We write $Seeds = Seeds(V, F) := \{p \in PS \mid F[p] \geq p\}$ and $Sym\mathcal{S} = Sym\mathcal{S}(V, F) := \{p \in PS \mid F[p] = p\}$. For our running example, these concepts are illustrated in Fig. 2. Note that seeds are a relaxation of symbolic steady states and $\mathcal{S} \subseteq Sym\mathcal{S} \subseteq Seeds$ holds, i.e., every steady state is a symbolic steady state and every symbolic steady state is a seed. The motivation for this terminology is that for every seed p there is a unique corresponding symbolic steady state q obtained by repeatedly applying $F[\cdot]$ until $F^k[p] = F^{k+1}[p] =: q$. This process is also called *percolation*. Note that if p is a seed, then $F[p]$ is also a seed (since for any $r \in PS$ with $r \geq p$ and any expression f, if $f[p] = c$ then $f[r] = c$).

$p := 1_1$	$q := 0_1 0_2$	$r := 1_3 1_4$	$s := 0_2 1_4$
$f_1[p] = 1$	$f_1[q] = 0$	$f_1[r] = v_1 + v_2$	$f_1[s] = v_1 + 0$
$f_2[p] = 1 \cdot v_4$	$f_2[q] = 0$	$f_2[r] = v_1 \cdot 1$	$f_2[s] = v_1 \cdot 1$
$f_3[p] = 0$	$f_3[q] = \overline{0} \cdot v_4$	$f_3[r] = \overline{v_1} \cdot 1$	$f_3[s] = \overline{v_1} \cdot 1$
$f_4[p] = \overline{v_3}$	$f_4[q] = \overline{v_3}$	$f_4[r] = 0$	$f_4[s] = \overline{v_3}$
$F[p] = 1_1 0_3 > p$	$F[q] = 0_1 0_2 = q$	$F[r] = 0_4 \not\geq r$	$F[s] = \varnothing$

Fig. 2. We specify partial states by a sequence of $|p|$ values whose subscript corresponds to the index of the variable, i.e., $1_1 0_3$ means $U_p = \{v_1, v_3\}$ and $p(v_1) = 1, p(v_3) = 0$. Here, p is a seed but not a symbolic steady state because $F[p] > p$, whereas q is a symbolic steady state because $F[q] = q$ and r and s are neither seeds nor symbolic steady states. The example network has 4 seeds and 3 symbolic steady states, namely $Seeds = \{\varnothing, 1_1, 0_1 0_2, 1101\}$ and $Sym\mathcal{S} = \{\varnothing, 0_1 0_2, 1101\}$.

2.2 Applications

The *state transition graph* of a Boolean network (V, F) is the directed graph $(S, \to)$ where S is the state space of (V, F) and the transitions $\to \subseteq S \times S$ are obtained from F via a given update rule. We mention two update rules here, the *synchronous rule* and its transition relation $\twoheadrightarrow_F \subseteq S \times S$, and the *asynchronous*

rule and its transition relation $\hookrightarrow_F \subseteq S \times S$. The former is defined by $s \twoheadrightarrow_F t$ iff $F(s) = t$. To define $\hookrightarrow_F$ we need the Hamming distance $\Delta : S \times S \to \{1, \ldots, n\}$ between states which is given by $\Delta(s,t) := |\{v \in V \mid s(v) \neq t(v)\}|$. We define $s \hookrightarrow_F t$ iff either $s = t$ and $F(s) = s$ or $\Delta(s,t) = 1$ and $\Delta(t, F(s)) < \Delta(s, F(s))$.

Since $(S, \to)$ is a directed graph, the standard digraph terminology applies. A path in $(S, \to)$ is a sequence of states $(s_1, \ldots, s_{l+1})$ with $s_i \to s_{i+1}$, for $1 \leq i \leq l$. When Boolean networks are used to model the dynamics of a system, paths are also called *trajectories*. An non-empty set $R \subseteq S$ is a *trap set* if for every $r \in R$ and $s \in S$ with $r \to s$ it holds that $s \in R$. An inclusion-wise minimal trap set is also called an *attractor* of (V, F). Note that every trap set contains at least one minimal trap set and therefore at least one attractor. We distinguish two types of attractors depending on their size. If $X \subseteq S$ is an attractor and $|X| = 1$ then X is a steady state and if $|X| > 1$ we call it a *cyclic attractor*. As mentioned before, a symbolic state p references the subspace $S[p]$. A central idea for various applications is given in the next result, which immediately follows from the properties of p and the definition of $\hookrightarrow_F$ and $\twoheadrightarrow_F$.

Theorem 1. *If $p \in Seeds$, then $S[p]$ is a trap set in $(S, \hookrightarrow_F)$ and in $(S, \twoheadrightarrow_F)$.*

Proof. Assume $p \in Seeds$ but $S[p]$ is not a trap set. Then there is a $r \to s$ such that $r \in S[p]$ and $s \notin S[p]$. But then $\exists v_i \in U_p : f_i(r) \neq p(v_i)$ which contradicts $p \leq r$ and $f_i[p] = p(v_i)$.

Application 1: Model Reduction Let $R \subseteq S$ be a trap set. The partial state $p := stab(R) \in PS$ obtained from R by $U_p := \{v \in V \mid \forall s, t \in R : s(v) = t(v)\}$ and $p(v) := s(v)$, for $v \in U_p$ and $s \in R$ arbitrary, is called the *induced stable state* of R. Note that, in general, the induced stable state of a trap set references a superset of R, i.e., $S[stab(R)] \supseteq R$. A natural model reduction technique is based on the observation that for any trap set $R \subseteq S$, the transitions of any trajectory with an initial state $s_1 \in R$ are governed by the reduced system (V_p, F_p) with $p := stab(R)$ and

$$V_p := \{v \in V \mid v \notin U_p\}, \quad F_p := \{f_i[p] \mid f_i \in F, v_i \in V_p\}\,.$$

Intuitively speaking, the network (V_p, F_p) is obtained by "dividing out" the partial state p that describes the steady variables in R, see [3] for more details and Fig. 3 for an application to the running example.

Since seeds reference trap sets and since $p \in Seeds$ implies that $stab(S[p]) = p$, they naturally lend themselves for the above mentioned model reduction technique. The largest reduction in terms of state space cardinality is then obtained by considering those symbolic steady states that are maximal w. r. t. the partial order $\leq$, since they yield the smallest subspaces $S[p]$.

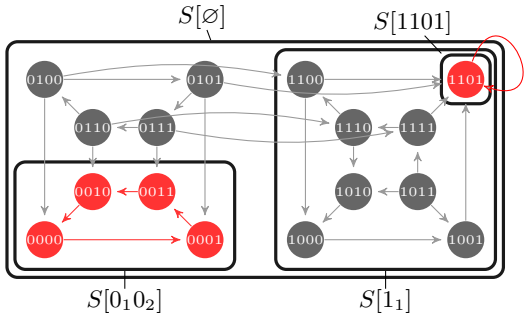
Application 2: Cyclic Attractors The set $\max(Sym\mathcal{S})$ of maximal symbolic steady states w.r.t. $\leq$ has the property that every $p \in \max(Sym\mathcal{S})$ with $|p| < n$ is such that $S[p]$ contains only cyclic attractors.

$f_3 = v_4$
$f_4 = \overline{v_3}$

(a) Reduction by $p = 0_1 0_2$

$f_2 = v_4$
$f_3 = 0$
$f_4 = \overline{v_3}$

(b) Reduction by $q = 1_1$



(c) The state transition graph $(S, \hookrightarrow_F)$

Fig. 3. (a,b) The *trap set reduction* applied to the 2 seeds $p = 0_1 0_2$ and $q = 1_1$ of the running example. (c) The asynchronous state transition graph $(S, \hookrightarrow_F)$. The 2 attractors are highlighted with red states and transitions. Rounded rectangles indicate the 4 trap sets that correspond to the 4 seeds of (V, F). Note that each trap set contains an attractor and that $p = 0_1 0_2$, which is maximal w.r.t. $\leq$ and satisfies $|p| = 2 < n$, is such that $S[p]$ contains only cyclic attractors. $q = 1_1$ on the other hand is not maximal and so $S[q]$ may, as is the case here, not contain a cyclic attractor.

Theorem 2. $|\{p \in \max(Sym\mathcal{S}) \mid |p| < n\}|$ *is a lower bound on the number of cyclic attractors of* (V, F).

Proof. Let $p \in \max(Sym\mathcal{S})$ and $|p| < n$. By Theorem 1, $S[p]$ is a trap set and therefore contains an attractor X. If $X = \{x\}$ then x is a steady state such that $p \leq x$, which contradicts the maximality of p.

Furthermore, since $stab(S[p]) = p$ for $p \in Sym\mathcal{S}$, we may conclude that some $v \in V \setminus U_{stab(S[p])}$ must be involved in the cyclic behavior. In our running example, the partial state $p = 0_1 0_2$ is a maximal symbolic steady state and therefore contains only cyclic attractors, see Fig. 3.

Note that $\mathcal{S} \subseteq \max(Seeds) = \max(Sym\mathcal{S}) \subseteq Sym\mathcal{S}$. Calculating all maximal symbolic steady states thus yields, in addition to the information on cyclic attractors, also all steady states.

2.3 The Prime Implicant Graph

In this section we propose a method for computing the seeds of (V, F). The idea is to translate the task into a hypergraph problem in which every seed is represented by a set of arcs that satisfy certain constraints. As we will show, those arc sets can be computed with existing solvers for integer linear programs.

We consider a directed hypergraph in which each arc corresponds to a minimal size implicant of f_i or $\overline{f_i}$, for some $v_i \in V$. Minimal size implicants were named prime implicants by Quine in [4]. We define the following slight variation: For $c \in \mathbb{B}$, a *c-prime implicant* of a non-constant f is a partial state $p \in PS$ satisfying $f[p] = c$, and $f[q] \neq c$ for all $q < p$. For a constant $f_i = c$ we define

that p with $U_p := \{v_i\}$ and $p(v_i) = c$ is its single prime implicant. In the running example, $1_1 1_2$ satisfies $f_1[1_1 1_2] = 1$. But it is not a 1-prime implicant of f_1, because $1_1 < 1_1 1_2$ and $f_1[1_1] = 1$. The set of all prime implicants of a Boolean network is denoted by

$$PI = PI(V, F) := \{(p, c, v_i) \in PS \times \mathbb{B} \times V \mid p \text{ is a } c\text{-prime implicant of } f_i\}.$$

The *prime implicant graph* is the directed hypergraph $(\mathcal{N}, \mathcal{A})$ where $\mathcal{N} = \mathcal{N}(V) := \{p \in PS \mid |p| = 1\}$ consists of all size 1 partial states (corresponding to literals in propositional logic). The arcs $\mathcal{A} = \mathcal{A}(V, F) \subset 2^{\mathcal{N}} \times 2^{\mathcal{N}}$ are defined by the mapping $\alpha : PI \to 2^{\mathcal{N}} \times 2^{\mathcal{N}}, (p, c, v_i) \mapsto (\{p_1, \ldots, p_{|p|}\}, \{q\})$, where

(1) $p = p_1 \sqcup \cdots \sqcup p_{|p|}$ is the unique decomposition of p into size 1 partial states,
(2) $q \in PS$ is defined by $U_q := \{v_i\}$ and $q(v_i) := c$.

The prime implicant graph has exactly one arc for every prime implicant, i.e., $\mathcal{A} := \{\alpha(p, c, v_i) \mid (p, c, v_i) \in PI\}$. The *head* of an arc $a = (\{p_1, \ldots, p_k\}, \{q\})$ is denoted by $H(a) := q$, and its *tail* by $T(a) = p_1 \sqcup \cdots \sqcup p_k$. The prime implicant graph of the running example is given in Fig. 4.

2.4 Prime Implicants and Seeds

Now we establish a relationship between subsets $A \subseteq \mathcal{A}$ and the seeds of a network (V, F). To do so we need the notions of *consistency* and *stability*. A subset $A \subseteq \mathcal{A}$ is *consistent* if for all $a_1, a_2 \in A$ the partial states $H(a_1)$ and $H(a_2)$ are consistent. If $A = \{a_1, \ldots, a_m\} \subseteq \mathcal{A}$ is consistent, the union $H(a_1) \sqcup \cdots \sqcup H(a_m)$ is called the *induced partial state* of A and denoted by $H(A)$. For the special case $A = \varnothing$ we define $H(A) := \varnothing$. A subset $A \subseteq \mathcal{A}$ is *stable* if for every $a \in A$ there is a consistent subset $B_a \subseteq A$ such that $T(a) \leq H(B_a)$. Intuitively, in this case the requirement $T(a)$ for each implication $a \in A$ to become effective is met by some assumptions B_a. The stable and consistent subsets of $\mathcal{A}$ for the running example and their induced partial states are given in Fig. 4. The central idea for the computation of *Seeds* is given in the next result:

Theorem 3. *$p \in Seeds$ if and only if there is a stable and consistent $A \subseteq \mathcal{A}$ such that $H(A) = p$.*

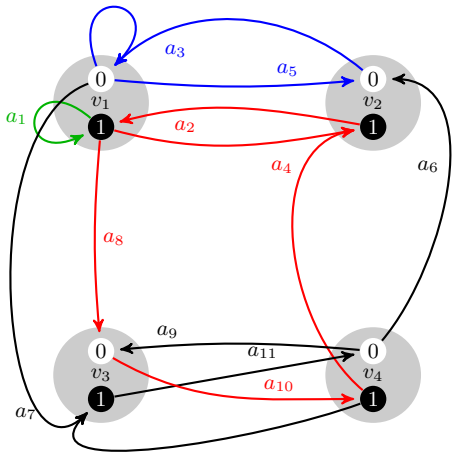
Proof. The statement of the theorem is trivially true for $p = \varnothing$ and $A = \varnothing$.

Hence, assume $\varnothing \neq p$. Let $v_i \in U_p$. Since $F[p] \geq p$ it follows that $f_i[p] = p(v_i)$ and hence that there is a $p(v_i)$-prime implicant q_i of f_i that satisfies $q_i \leq p$. The set $A := \{\alpha(q_i, p(v_i), v_i) \mid v_i \in U_p\}$ is, by definition, consistent and satisfies $H(A) = p$. But it is also stable because $\forall a \in A : T(a) \leq p$. Let $\varnothing \neq A \subseteq \mathcal{A}$ be stable and consistent. Then $\forall v_i \in U_{H(A)} : \exists a \in A : H(a) = H(A)(v_i)$. Hence $F[H(A)] \geq H(A)$ and $H(A) \in Seeds$.

Corollary 1. *Inclusion-wise maximal stable and consistent arc sets induce maximal seeds and therefore maximal symbolic steady states.*

PI	$\mathcal{A}$
$(1_1, 1, v_1)$	a_1
$(1_2, 1, v_1)$	a_2
$(0_1 0_2, 0, v_1)$	a_3
$(1_1 1_4, 1, v_2)$	a_4
$(0_1, 0, v_2)$	a_5
$(0_4, 0, v_2)$	a_6
$(0_1 1_4, 1, v_3)$	a_7
$(1_1, 0, v_3)$	a_8
$(0_4, 0, v_3)$	a_9
$(0_3, 1, v_4)$	a_{10}
$(1_3, 0, v_4)$	a_{11}

(a)



(b)

$$A = \varnothing, \quad B = \{a_3, a_5\}, \quad C = \{a_1\}, \quad \begin{array}{l} D_1 = \{a_1, a_4, a_8, a_{10}\} \\ D_2 = \{a_2, a_4, a_8, a_{10}\} \\ D_3 = \{a_1, a_2, a_4, a_8, a_{10}\} \end{array} \qquad \text{(c)}$$

Fig. 4. (a) The complete set PI of prime implicants of the example network, given in (p, c, v_i) notation. (b) The prime implicant graph $(\mathcal{N}, \mathcal{A})$ of the running example. The 8 nodes are grouped into 4 pairs $0_i, 1_i$ that belong to the same variable v_i. Gray discs represent the groups and are not elements of $\mathcal{N}$. Nodes that correspond to positive literals are drawn in black and negated literals in white. Hyperarcs are represented by several arcs having a common arrowhead. The colors indicate 3 different stable and consistent arc sets. (c) The stable and consistent arc sets induce the following seeds: $H(A) = \varnothing, H(B) = 0_1 0_2, H(C) = 1_1$ and for $i = 1, 2, 3 : H(D_i) = 1101$.

A special case of seeds was first studied in [5, 6], where the authors describe *self-freezing circuits* that rely on *canalizing effects*. They occur if, in our terminology, there is a stable and consistent arc set $\varnothing \neq A$ that contains exclusively size 1 prime implicants, i.e., for all $a \in A$ we have $|T(a)| = 1$.

2.5 Computation of the Stable and Consistent Arc Sets

We propose an optimization-based method for finding all maximal stable and consistent arc sets in $(\mathcal{N}, \mathcal{A})$. Integer linear programming (ILP) has previously been suggested as a method for solving problems arising in the study of Boolean networks, see e.g. [7] and references therein. As a preliminary step, the prime implicants PI have to be enumerated. This can be achieved with any implementation of the Quine-McCluskey algorithm, see e. g. [8]. Although for sufficiently

complex expressions $f_i \in F$ the enumeration of PI itself can be a hard problem, we found that the complexity of F in typical biological models is low enough for this step to be negligible.

We now formulate a 0-1 optimization problem to compute maximal stable and consistent arc sets $A \subseteq \mathcal{A}$. For every arc $a = (p, c, v_i) \in PI$ we introduce a variable $x_a \in \{0, 1\}$ indicating whether or not a is a member of the set $A \subseteq \mathcal{A}$ that we want to compute. We denote these variables by $X := \{x_a \mid a \in PI\}$. In addition, we introduce for every $v_i \in V$ two variables $y_i^0, y_i^1 \in \{0, 1\}$ that indicate whether v_i is in the domain of the induced partial state and, if so, what value it takes. We denote them by $Y := \{y_i^c \mid c \in \mathbb{B}, v_i \in V\}$. For any $c \in \mathbb{B}$, we require $y_i^c = 1$ if and only if $v_i \in U_{H(A)}$ and $H(A)(v_i) = c$. To encode this requirement, we use the logical constraints

$$y_i^c \iff \bigvee_{a \in B_i^c} x_a, \qquad \text{for all } c \in \mathbb{B}, v_i \in V. \tag{C1}$$

Here, $B_i^c := \{a \in \mathcal{A} \mid \{v_i\} = U_{H(a)}, H(a)(v_i) = c\}$ denotes the arcs inducing v_i to take the value c, and $\Rightarrow$ and $\vee$ are the standard logical connectives for implication and disjunction. Next, we want to enforce the set $A := \{a \in \mathcal{A} \mid x_a = 1\}$ to be stable and consistent. To achieve this, we add the following constraints $(C2)$ resp. $(C3)$:

$$\overline{y_i^0} \vee \overline{y_i^1}, \qquad \text{for all } v_i \in V, \tag{C2}$$

$$x_a \Rightarrow y_i^{T(a)(v_i)}, \qquad \text{for all } a \in \mathcal{A}, v_i \in U_{T(a)}. \tag{C3}$$

To find a first maximal stable and consistent set $A \subseteq \mathcal{A}$, we solve the 0-1 optimization problem (here $\sum$ denotes addition)

$$\text{maximize} \sum_{x_a \in X} x_a, \quad \text{such that (C1), (C2), (C3).} \tag{0-1}$$

All maximal seeds can be enumerated by iteratively solving problem (0-1). Whenever a new solution $z : X \cup Y \to \{0, 1\}$ is found, we add a so-called *no-good cut*, which prevents this solution from being computed again. For example, we can use the constraint

$$\bigvee_{x_a \in G(z)} x_a, \quad \text{where } G(z) := \{x_a \in X \mid z(x_a) = 0\}. \tag{C4}$$

To solve problem (0-1) in practice, we reformulated the constraints (C1)-(C4) as linear 0-1 inequalities. A Python implementation using the integer programming solver `Gurobi` [9] is available at http://sourceforge.net/projects/boolnetfixpoints.

3 Application to a MAPK pathway model

We computed $\max(Sym\mathcal{S})$ for a network that models the influence of the MAPK pathway on cancer cell fate decisions, published in [10]. It consists of 53 variables

that represent signaling proteins, genes and phenomenological components like *proliferation* or *apoptosis*. We found that there are 18 maximal symbolic steady states, 12 of which are steady states. Hence, following Application 2 in Sect. 2.2, there are at least 6 cyclic attractors whose properties can be comprehensively investigated using the 6 corresponding reduced models. An illustration of the largest maximal symbolic steady state, and therefore the smallest corresponding sub-model, is given in Fig. 3.

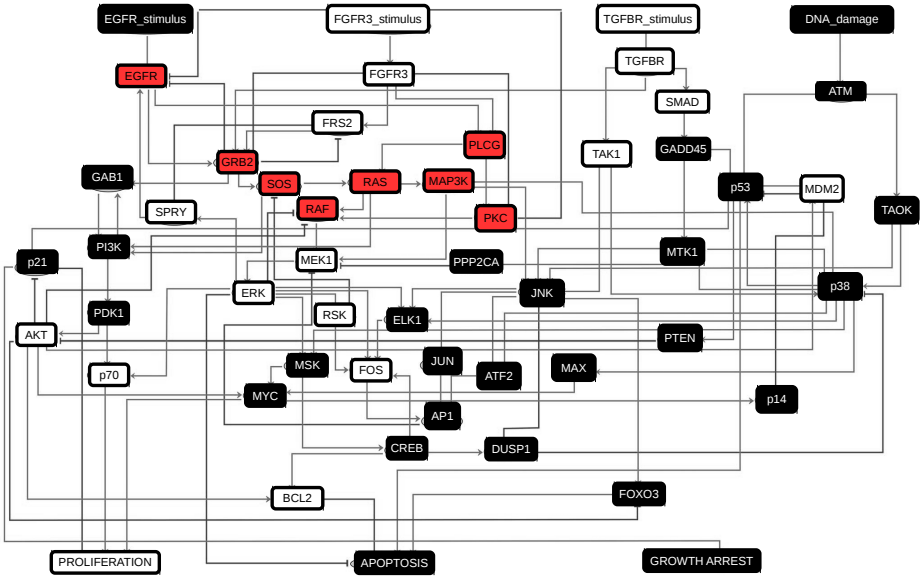


Fig. 5. A representation of the largest of the 6 maximal symbolic steady states p that satisfy $|p| < n$. Here, $|p| = 45$ and the remaining sub-model (V_p, F_p) contains therefore $53 - 45 = 8 = |V_p|$ variables. The underlying *interaction graph* is taken from [10]. Black and white components belong to U_p, with the values 1 and 0 respectively. Red components belong to the remaining sub-model.

4 Discussion

In this paper we propose a way of determining seeds and symbolic steady states of interest, e.g., in the context of model reduction and for estimating the number of cyclic attractors. We provide an optimization-based method for computing seeds exploiting the prime implicant graph. This graph captures properties of fundamental importance for the network behavior, allowing to analyze certain aspects of asymptotic dynamics without having to calculate the state transition graph. Here, we focused on maximal seeds and symbolic steady states, however,

we find that minimal seeds also carry interesting information. We currently work on extending our methods exploiting this potential of the prime implicant graph and seeds in general, particularly for studying questions related to reachability and to decision making.

The optimization-based method for finding maximal symbolic steady states can be extended from Boolean to multi-valued networks by generalizing the notion of prime implicants from Boolean to multi-valued expressions. Theorem 1 holds not only for synchronous or asynchronous but for *any* update rule (see [11] for other update rules) and also stochastic simulations. Just as the steady states of a dynamic Boolean network are independent of the update rule, so are the seeds and symbolic steady states.

Regarding the efficiency of the ILP method, we have performed computational experiments with random Boolean networks that indicate good scalability, yielding results in minutes for networks with restricted maximal in-degree and hundreds of variables. We plan to extend this evaluation and to investigate avenues to increase the efficiency, e.g., by considering not only ILP but also pseudo-Boolean or SAT solvers for handling the problem.

References

1. Wang, R.S., Saadatpour, A., Albert, R.: Boolean modeling in systems biology: an overview of methodology and applications. Physical biology **9**(5) (2012) 055001
2. Dubrova, E., Teslenko, M.: A SAT-based algorithm for finding attractors in synchronous boolean networks. IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB) **8**(5) (2011) 1393–1399
3. Siebert, H.: Analysis of discrete bioregulatory networks using symbolic steady states. Bulletin of Mathematical Biology **73** (2011) 873–898
4. Quine, W.V.: The problem of simplifying truth functions. The American Mathematical Monthly **59**(8) (1952) 521–531
5. Fogelman-Soulie, F.: Parallel and sequential computation on boolean networks. Theoretical computer science **40** (1985) 275–300
6. Kauffman, S.A.: The origins of order: Self organization and selection in evolution. Oxford University Press, USA (1993)
7. Akutsu, T., Yang, Z., Hayashida, M., Tamura, T.: Integer programming-based approach to attractor detection and control of boolean networks. IEICE TRANSACTIONS on Information and Systems **95**(12) (2012) 2960–2970
8. Dick, R.: Quine-McCluskey two-level logic minimization method. http://pypi.python.org/pypi/qm/0.2 (2008) Online, accessed in April-2014.
9. Gurobi Optimization, I.: Gurobi optimizer reference manual (2014)
10. Grieco, L., Calzone, L., Bernard-Pierrot, I., Radvanyi, F., Kahn-Perlès, B., Thieffry, D.: Integrative modelling of the influence of mapk network on cancer cell fate decision. PLoS computational biology **9**(10) (2013) e1003286
11. Gershenson, C.: Updating schemes in random boolean networks: Do they really matter. In: Artificial Life IX Proceedings of the Ninth International Conference on the Simulation and Synthesis of Living Systems, MIT Press (2004) 238–243