Basins of Attraction, Commitment Sets, and Phenotypes of Boolean Networks

Hannes Klarner, Frederike Heinitz, Sarah Nee, and Heike Siebert

Abstract—The attractors of Boolean networks and their basins have been shown to be highly relevant for model validation and predictive modeling, e.g., in systems biology. Yet, there are currently very few tools available that are able to compute and visualize not only attractors but also their basins. In the realm of asynchronous, non-deterministic modeling not only is the repertoire of software even more limited, but also the formal notions for basins of attraction are often lacking. In this setting, the difficulty both for theory and computation arises from the fact that states may be elements of several distinct basins. In this paper, we address this topic by partitioning the state space into sets that are committed to the same attractors. These commitment sets can easily be generalized to sets that are equivalent w.r.t. the long-term behaviors of pre-selected nodes which leads us to the notions of markers and phenotypes which we illustrate in a case study on bladder tumorigenesis. For every concept, we propose equivalent CTL model checking queries and an extension of the state of the art model checking software NuSMV is made available that is capable of computing the respective sets. All notions are fully integrated as three new modules in our Python package PyBoolNet, including functions for visualizing the basins, commitment sets, and phenotypes as quotient graphs and pie charts.

Index Terms—Systems biology, boolean networks, attractors

1 Background

Boolean networks offer an intuitive approach to simulating the dynamics of interaction networks. In cell biology these are usually gene regulatory or signal transduction networks. Each biological component is modeled by a Boolean variable that can only switch between two states, true and false. Depending on the component, these states represent whether a protein is present at high or low concentration levels, a gene is being transcribed at or above its base rate, a signaling molecule is phosphorylated or not, and so on. The conditions under which variables switch on and off are specified by Boolean functions that involve other variables of the network. Together with a transition relation, which determines how many and which components are allowed to change during a state transition, Boolean networks can be used to predict transient and long term activity profiles of the involved components. Although the restriction to discrete states is arguably an over-simplification for many processes, Boolean networks have frequently been shown to be fruitful to predict drug targets, phenotypes, novel interactions, etc. This paper is particularly relevant to models whose state transitions are non-deterministic. A prominent example are the so-called Thomas networks (after René Thomas, see e.g., [1]) where changes in state variables are modeled as processes that are not synchronized. Asynchronous transition systems are especially applicable when details about the precise mechanisms of the processes, e.g., the activation and inhibition delays, are unknown or uncertain. While the results of this paper are applicable to the deterministic case, they are particularly relevant in a non-deterministic setting. We consider here mainly the Thomas formalism whose state transition graph captures all possible behaviors that result from considering one value change per transition without assigning specific probabilities. We do not consider probabilistic Boolean networks in the traditional sense, and none of our analysis is simulation-based. However, the notions are applicable to any given fixed transition system, and analysis of the Thomas state transition graph yields constraints for the behavior of any model employing randomized asynchronous update schemes. For a detailed introduction to Boolean networks and qualitative modeling more generally see [2].

Attractor detection is a frequent starting point for testing the consistency of models with data and also for obtaining model predictions, see e.g., [3, Section 3.2]. An extension of this approach considers not only the location of attractors but also their basins of attraction. These regions and transitions between them determine the likelihood of reaching particular attractors and not others. From a practical, wet-lab point of view, the motivation for studying basins and attractor commitment sets is to find out how the underlying decision processes are implemented within the interaction networks of living cells. Computational models and the formalization of notions like basin, commitment set and phenotype add to this research by making hypothesis testable in silico. When these models are validated they may then be used to measure the robustness of interaction networks with respect to knock-outs or knock-ins and other perturbations. Due to the high computational cost of analyzing the complete state transition graph for larger networks, such studies often use aggregated results...
from simulations [4], [5]. While analytic results are mostly focused on Boolean networks with deterministic updates [6], [7], there are still some algorithms for computing the basins of Boolean networks with asynchronous updates [8], [9], [10]. These all have to deal with the inherent difficulties in analyzing asynchronous state transition graphs, which in turn results in restricted scalability.

In general, the computation of basins of attraction is challenging because the size of the state space grows exponentially with the number of variables of the model. A competitive approach for dealing with large transition systems is to use so-called symbolic model checking algorithms [11]. Model checking is a method from computer science to decide whether formal hardware designs satisfy desired specifications. The designs are formalized as so-called Kripke structures, which are labeled, directed graphs, and the specifications to be verified are translated into statements of a chosen temporal logic. Computation Tree Logic (CTL) is an example of such a logic. It allows statements about the alternative futures originating in states [11]. These are particularly useful when dealing with properties related to attractors. NuSMV is an open source symbolic model checking tool allowing for the efficient analysis of transition systems [12].

It has been observed that the state transition graphs (STGs) of Boolean networks are Kripke structures and that model checking is therefore applicable to Boolean networks [13]. There are custom model checking tools that are specifically developed for biological networks, e.g., [10], [14], but their additional capabilities may come at the cost of efficiency in more basic queries. Still, we believe that for the problem of computing basins of attractors and related objects state of the art software designed to answer reachability queries exactly such as the model checking software NuSMV provides the ideal basis for implementation.

In this paper we, on the one hand, formally develop several notions related to basins of attraction and, on the other hand, provide efficient tools to compute these objects in practice using a model checking approach. Namely, we created NuSMV-a, an extension of NuSMV 2.6, which implements model checking with accepting states. We complement the theoretical part in later sections by giving an impression of the tool efficiency and scalability and by considering a case study on bladder tumorigenesis, focusing on analysis of different phenotypes of the cancer cells.

We aim at both clear and intuitive descriptions, further technical details can be found in the supplementary material, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TCBB.2018.2879097. In addition, scripts and tutorials to reproduce the results are available, integrated in our PyBoolNet software [15].

2 Model Checking with Accepting States

CTL model checking was originally designed to answer queries by either yes or no depending on whether the transition system satisfies the query or not. When model checking is used commercially, i.e., for the verification of software and hardware designs, it turned out to be useful to analyze the reasons for why a design does not satisfy a specification. This led to the development of algorithms that generate counterexamples when queries are false [16]. When model checking is applied to systems biology a similar analysis is desirable. In particular, the theory which is developed in the subsequent sections requires the capability of computing the exact set of states that satisfies a query, an idea that is more basic than computing counterexamples. The accepting states of a CTL formula $\varphi$ and STG $(S, \rightarrow)$ are defined by

$$\text{Accept}(S, \rightarrow, \varphi) := \{ x \in S \mid x \models \varphi \},$$

where $x \models \varphi$ means that $\varphi$ holds in state $x$ of the transition system $(S, \rightarrow)$. For more details on model checking and CTL, see the supplementary material, available online. Note that the accepting states are defined without initial states. By definition, see [11], a transition system $(S, \rightarrow)$ satisfies a model checking query with CTL formula $\varphi$ and initial states $I \subseteq S$ if $I \subseteq \text{Accept}(S, \rightarrow, \varphi)$.

For our purposes, the accepting states of one query are often re-used in the CTL formula of a subsequent query. We therefore need to be able to encode sets of states as Boolean expressions. Since the state space is already defined in terms of states of Boolean variables, we may use a disjunctive normal form (DNF) in which each term references a single state of the set. Clearly, this is not a very efficient encoding and there are usually other, shorter expressions that also represent the set. We denote by $\text{Enc}(X)$ an arbitrary Boolean expression that represents $X \subseteq S$. Therefore, $\text{Enc}(\text{Accept}(S, \rightarrow, \varphi))$ is an expression that represents the accepting states of $\varphi$ in $(S, \rightarrow)$.

3 Basins of Attraction

Basins of attraction describe regions of state space that lead into specific attractors. Large basins suggest a certain robustness of their attractors against perturbations, see e.g., [17], while small basins suggest unlikely attractors. The study of basins is therefore essential for understanding the likely long-term behaviors of Boolean networks.

The basins of deterministic and non-deterministic STGs are very different because states in deterministic STGs can always reach exactly one attractor while states in non-deterministic STGs can usually reach several attractors. It has therefore been suggested, for example in [18], to distinguish between weak and strong basins of attraction where the weak basin contains all states that can reach a given attractor while the strong basin contains all states that can only reach a given attractor.

For an attractor $A \subseteq S$ and an arbitrary state $a \in A$, the weak basin of attraction is defined by

$$\text{WeakBasin}(S, \rightarrow, a) := \{ x \in S \mid \exists \pi \in \text{Paths}(x) : a \in \pi \},$$

where $\text{Paths}(x)$ is the set of all paths starting from $x$ in the $(S, \rightarrow)$ and $a \in \pi$ means that $a$ appears somewhere along $\pi$. Note that, as attractors are strongly connected, it is irrelevant which $a \in A$ is chosen. Biologically, these basins represent the weakest form of commitment to an attractor and the system might never actually reach the given attractor, either because there are other attractors that are more likely or because the system remains in an intermediate equilibrium.

To define the strong basin one may be tempted to require that all paths lead to the attractor. But this definition is stronger than the notion that a state can only reach $A$. It excludes
states that are on cycles, even if they can, in terms of attractors, only reach A. Nevertheless this type of basin is interesting and since it is cycle-free, we call it CycleFreeBasin(S, \rightarrow, A) and define it by

\[ \{ x \in S \mid \forall \pi \in \text{Paths}(x) : \exists a \in A : a \in \pi \}. \]

Note that this definition is in terms of A instead of an arbitrary \( a \in A \). The reason is that in specific cases it makes a difference which \( a \in A \) is chosen, e.g., if \( A \) is a complex attractor that contains a cycle that does not visit all states of \( A \). These basins form the strictest notion of attraction, the inevitable fall into the given attractor without the possibility of intermediate oscillations or the potential of reaching another attractor. Our motivation for introducing cycle-free basins lies in trying to give a broad picture of different types of basins, but is also due to their mathematical properties which, e.g., prohibit cyclic behavior outside of attractors. This might enable us to link them to the structure of the interaction graph, i.e., to the absence of negative feedback in the local interaction graph of a cycle-free basins, see [19]. In application, the cycle-free basins allow a clear-cut distinction of transient and sustained behavior, which is not the case if the system can be caught in intermediate cycles outside of the attractor.

The strong basin in the usual sense of all states that can only reach \( A \) is called StrongBasin(S, \rightarrow, A) and defined by

\[ \{ x \in S \mid \forall \pi \in \text{Paths}(x) : \forall y \in \pi : \exists a' \in \text{Paths}(y) : a \in \pi' \}. \]

As the system transitions from a weak to a strong basin it looses the potential of reaching certain other attractors and hence these transitions hint at hidden decision processes within the changing activity patterns. These three notions are illustrated for a toy transition system in Fig. 1. The Boolean equations for the toy transition system were created using iBoolNet [20].

To apply symbolic model checking to the computation of weak and strong basins we need CTL queries that capture these definitions. We only need three operators, namely the exists finally operator EF, the always finally operator AF and the all globally operator AG. These operators can be linked to the path-based definitions of the basins, since a state \( x \) satisfies EF(M) for a state set \( M \) if there exists a path starting in \( x \) leading to some state in \( M \). Similarly AF(M) encodes a condition for all paths starting in \( x \) and the combination AG(EF(M)) is satisfied if from every state reachable from \( x \) there exists a path to a state in \( M \). This last condition would in particular not be true for a state \( x \) that can reach an attractor \( M \) and in addition a second attractor \( M' \), since then the states of \( M' \) are reachable from \( x \), but \( M \) is not reachable from \( M' \). These considerations lead us to the following CTL queries

\[
\begin{align*}
&\text{Accept}(S, \rightarrow, \alpha), \quad \text{where } \alpha := EF(Enc(a)) \\
&\text{Accept}(S, \rightarrow, \beta), \quad \text{where } \beta := AF(Enc(A)) \\
&\text{Accept}(S, \rightarrow, \gamma), \quad \text{where } \gamma := AG(\text{EF}(\text{Enc}(a))),
\end{align*}
\]

for WeakBasin(a), CycleFreeBasin(A) and StrongBasin(a), respectively. Note that the three types of basins satisfy the inclusions

\[ A \subseteq \text{CycFreeBasin}(A) \subseteq \text{StrongBasin}(a) \subseteq \text{WeakBasin}(a), \quad (1) \]

and that strong basins are disjoint.

With the definitions of the three types of basins translated into CTL queries we can now compute the sizes of the basins of any transition system that is encodable in NuSMV, including STGs that are obtained by adding priority classes, restricting the state space to specific initial states, multi-valued models, mixed synchronous-asynchronous updates and so on. In general, basins will differ depending on the respective choices and comparisons can help to glean information, e.g., regarding the robustness of the system with respect to the update strategy.

To visualize the relative sizes of the basins we suggest to use pie charts and stacked bar plots. The bar plots highlight the inclusion relationship while the pie charts give an impression of the relative sizes of the basins. These three notions are illustrated for the toy transition system in Fig. 2. At this point we would like to remark that the size of a basin and the likelihood of reaching its attractor from an initial state are only loosely related. The same is true for the size of a basin and its robustness in terms of perturbing state activities and asking whether a returning path exists. We think of the size of these sets as state-based measures of likelihood. To relate state- and trajectory-based likelihoods in a precise manner requires information about the transition probabilities, but it in general it should be clear that larger basins tend to affect simulation-based likelihoods positively.

The notion of basin can be generalized from single to multiple attractors simply by adjusting the state formulas accordingly. Suppose we are interested in the basins of \( k \) attractors \( A_1, \ldots, A_k \) with representative states \( a_i \in A_i \). With \( X := \{a_1, \ldots, a_k\} \) and \( Y := A_1 \cup \cdots \cup A_k \) the CTL queries for the generalized basins are obtained by using \( X \) and \( Y \) instead of \( a \) and \( A \) in the definitions above: WeakBasin(X), CycFreeBasin(Y) and StrongBasin(X).
the strong basin of a steady state even if we do not know whether there are additional attractors.

4 COMMITMENT SETS

A different approach to understanding attractors and regions in state space that lead to them is via commitment sets. Commitment sets partition the state space into classes that can reach the exact same attractors. For ease of referencing specific commitment sets, let \( \{A_1, \ldots, A_k\} \) be a sequence of \( k \) attractors with representatives \( \{a_1, \ldots, a_k\} \) and let \( I \subseteq \{1, \ldots, k\} \) be a non-empty subset of indices. The states that are committed to the attractors \( \{A_i \mid i \in I\} \) are defined by

\[
\text{ComSet}(I) := \{x \in S \mid (\exists \pi \in \text{Paths}(x) : a_i \in \pi) \Rightarrow (i \in I)\}.
\]

Commitment sets are the finest resolution of state space with respect to the potential of reaching certain attractors. An analysis of the similarities of states that belong to the same set is biologically as relevant as understanding the differences between states of different commitment sets. In particular, it remains an open question to understand the decision processes that are realized by transitions between commitment sets, see also [21].

To encode this in CTL we observe that a state is committed to the attractors indexed by \( I \) if and only if there is a path from \( x \) to \( a_i \) for every \( i \in I \) and \( x \) is in the strong basin of \( \{A_i \mid i \in I\} \), i.e., cannot reach other attractors. A possible CTL query for \( \text{ComSet}(I) \) is therefore \( \text{Accept} (S, \rightarrow, \delta) \), where

\[
\delta := \bigwedge_{i \in I} \text{WeakBasin}(a_i) \land \text{StrongBasin}(\{a_j \mid j \in I\}).
\]

Note that, as before, commitment sets can be computed without knowing all attractors of the network. Furthermore, the same queries can be used to compute the basins and commitment sets of subspaces and in particular minimal trap spaces, see [22], which are often in a one-to-one correspondence with the attractors of a network, see [23].

For small networks with up to 15 nodes or so we may visualize commitment sets by different node colors. An example is given in Fig. 3. For larger networks we suggest to visualize the commitment sets and transitions between them by a “graph of commitment sets” in which the nodes are commitment sets and there are edges between the nodes if there are transitions between the corresponding commitment sets in the underlying STG. More precisely, we construct the quotient graph of the STG, induced by the commitment sets. Recall from basic graph theory that the quotient graph of a digraph \( G = (N, \rightarrow) \), induced by a partition \( N_1 \cup \cdots \cup N_m = N \) of its nodes \( N \), is the digraph \( Q \) whose nodes are the blocks \( N_i \) and there is an edge between two blocks \( N_i \rightarrow N_j \) if there are nodes \( n_1 \in N_i \) and \( n_2 \in N_j \) with and edge \( n_1 \rightarrow n_2 \) in \( G \).

In the context of commitment sets we call the quotient graph of an STG induced by the commitment sets its commitment diagram. An example is given in Fig. 3.

As for the nodes, we compute the edges of the commitment diagram by answering CTL model checking queries, using the EX operator. Given two commitment sets \( \text{ComSet}(I) \) and \( \text{ComSet}(J) \) with \( I, J \subseteq \{1, \ldots, k\} \), there is an edge \( \text{ComSet}(I) \rightarrow \text{ComSet}(J) \) if there are states \( x_1 \in \text{ComSet}(I) \) and \( x_2 \in \text{ComSet}(J) \) such that \( x_1 \rightarrow x_2 \) in the STG. Using model checking we first compute

Note that while the distributive law \( \text{EF}(\phi_1 \lor \phi_2) = \text{EF}(\phi_1) \lor \text{EF}(\phi_2) \) for existential CTL operators implies that the weak basin of several attractors is the union of the individual weak basins

\[
\text{WeakBasin}(\{a_i, a_j\}) = \text{WeakBasin}(a_i) \lor \text{WeakBasin}(a_j),
\]

the same does not hold for the strong and cycle-free basins. Instead, the following inclusions hold:

\[
\text{CycFreeBasin}(A_i \cup A_j) \supseteq \text{CycFreeBasin}(A_i)
\]

\[
\text{StrongBasin}(\{a_i, a_j\}) \supseteq \text{StrongBasin}(\{a_i\}).
\]

The reason for the inclusion instead of equality is that a strong basin of two attractors may contain a state that can reach both attractors. Such a state can not belong to the individual strong basins. The same example proves that the inclusion may be strict for cycle-free basins.

Note that in non-deterministic models, the states that can not be linked to a single attractor usually make up a significant portion of state space. The non-determinism can arise from uncertainty or lack of data, but it might also capture biological phenomena such as bistability from stochastic effects. In this setting model predictions can only categorize behavior as feasible or infeasible, with the notions of strong and weak basins allowing the addition of different degrees of confidence to realizations of behavior.

All definitions above work even if not all attractors are known. In practice we may therefore compute, for example,
marketers with respect to three possible outcomes: \textit{inhibited}=$0$, \textit{expressed}=$1$ and \textit{oscillating}=$\ast$. A phenotype analysis can help to understand fundamental system properties by simplifying the attractor landscape of the system. Its main advantage in application, however, is that the measurable components naturally form a set of markers that link the predictions of the Boolean network with observed expression data [3], [4].

We formalize phenotypes as three-valued functions $p: U \rightarrow \{0, 1, \ast\}$ from the set of markers $U \subseteq V$ to the set of possible outcomes $\{0, 1, \ast\}$. The phenotype $p_A$ of an attractor $A \subseteq S$ is defined by

$$p_A(u) = \begin{cases} 0 & \forall a \in A: a(u) = 0 \\ 1 & \forall a \in A: a(u) = 1 \\ \ast & \text{else} \end{cases}$$

Given a set of markers $U$ we say that a phenotype $p$ exists if there is an attractor $A \subseteq S$ such that $p_A = p$. Note that there may be several attractors with identical phenotypes, $p_A = p_{A'}$, but every attractor has a unique phenotype (for fixed markers).

Analogous to the commitment diagram we define the phenotype diagram by partitioning the state space into subsets that are committed to identical phenotypes. Let $p_1, \ldots, p_k$ be the phenotypes of a given STG and given markers. For each phenotype $p_i$, let $A_1, \ldots, A_k$ be all corresponding attractors with representative states $a_i \in A_j$ and denote by $P_i := \{a_i, \ldots, a_k\}$ the union of the representative states. We call the commitment set for the phenotypes specified by $I \subseteq \{1, \ldots, k\}$ the phenotype set. It is called $\text{PhenoSet}(I)$ and defined by

$$\{x \in S \mid (\exists i \in \text{Paths}(x): \forall a \in P_i: a \in x) \Rightarrow (i \in I)\}.$$  

It can be computed using model checking with accepting states $\text{Accept}(S, \rightarrow, \eta)$ where

$$\eta := \bigwedge_{i \in I} \text{WeakBasin}(P_i) \land \text{StrongBasin} \left( \bigcup_{i \in I} P_i \right).$$

Note the similarity with the query for commitment sets. The only difference is that we use sets of representatives $P_i$ instead of individual representatives $a_i$. As before, the existence of edges $\text{PhenoSet}(I) \rightarrow \text{PhenoSet}(J)$ between phenotype sets is determined via the quotient graph relation.

By definition the commitment diagram is a specific case of the phenotype diagram. The former carries the full information on the attractor reachability and as a mathematical object naturally extends the notions of basins of attraction. However, its generalization to the phenotype diagram allows to analyze reachability questions in the context of incomplete data.

5 PhenoTYPES

In this section we extend the notion of commitment sets to make it more suited to application. In experimental studies rarely all components of a system can be measured at the same time, even at all. So when using attractors for model validation and prediction an obvious drawback is that the experimental data yields only partial descriptions that might be valid for several attractors. To be more faithful to this scenario, attractors can be grouped together according to the values of so-called marker components. The resulting object is called a phenotype. More specifically, a phenotype is a pattern that specifies the long-term behaviors of a set of given
the accepting states. We decided to use an output format called factored form that is implemented in the function Cudd_DumpFactoredForm (CUDD 3.1.4). Although the resulting Boolean expressions become quickly unreadable for humans they are in NuSMV syntax and can therefore be used as sub-formulas of subsequent CTL queries. To obtain minimized expressions that may be easier for humans to read we extended PyBoolNet with a module called Boolean-Logic and specifically the function minimize_espresso (...) that uses the program espresso for minimizing Boolean expressions [27].

As of version 2.0, PyBoolNet comes with NuSMV-a and the function

```
check_primes_with_acceptingstates(...),
```

of the package ModelChecking returns a Python dictionary that contains all formulas and cardinalities as strings and integers, respectively. The PyBoolNet modules for computing the various basins, the commitment sets and the phenotypes all require this new model checking function. For tutorials and examples consult the PyBoolNet manual and project homepage.

### 7 Efficiency and Scalability

This section presents a table of running times for computing the accepting states of four Boolean networks of increasing size: tournier_apoptosis [18] with $n = 12$, dahlhaus_neuroplastoma [28] with $n = 24$, grieco_mapk [29] with $n = 53$ and jaoude_thdiff [3] with $n = 103$. The networks are available in the PyBoolNet model repository. We generated three CTL queries of increasing complexity for each network: a weak basin query, a strong basin query and a commitment set query. For state formulas we compute two attractor states $a_i \in A_i$ for $i = 1, 2$ for each network and use a conjunctive encoding for state formulas.

More precisely, the attractors were computed by the minimal trap space method [22], [23]. Each network has more than one attractor and we selected two attractors $A_i$ and attractor states $a_i \in A_i$ for $i = 1, 2$ at random to represent the computational effort. Each line in Table 1 therefore records the running time for a single basin or commitment set. In accordance with the definitions above, the weak basin query is $\text{EF}(\text{Enc}(a_1))$, the strong basin query is $\text{AG}(\text{EF}(\text{Enc}(a_2)))$ and the commitment set query is

$$\text{EF}(\text{Enc}(a_1)) \land \text{EF}(\text{Enc}(a_2)) \land \text{AG}(\text{EF}(\text{Enc}(a_1) \lor \text{Enc}(a_2)))$$

where we used the CNF encoding for $\text{Enc}(a_i)$. NuSMV-a then returns all states that satisfy the query and returns an encoding (a Boolean expression) for these states as well as the size of the set. Note that all states of the state transition graph are therefore taken into consideration. To calculate the basins for all attractors the queries are simply repeated with the respective attractor representative states. For the MAPK network for example there are 18 attractors and to compute all weak basins requires 31 minutes and 12 seconds which corresponds to 18 times 1 minute 45 seconds in Table 1. The number of model checks required to compute the full commitment diagram is in general exponential in the number of attractors, see Section 2 of the supplementary materials, available online. In the case of the MAPK network it requires 10 model checks for 20 nodes and 4 edges (there are 14 trivial cases of a single attractor for an input combination that do not require model checking) and takes 28 minutes and 12 seconds.

We compared the running times of NuSMV-a with Antelope (acronym stands for “Analysis of Networks through Temporal-Logic Specifications”) the only other software we found for computing accepting states of CTL queries. Antelope is a model checking software specifically designed for biological interaction networks, see [10]. Its model checking algorithms are based on “hybrid CTL”, an extension of standard CTL that allows, for example, the specification of cycles of a given length. We confirmed that the size of each set of accepting states returned by Antelope agrees with the size returned by NuSMV-a. We used a 64-bit Ubuntu 16.04 LTS desktop PC with 7.7 GiB memory and 4 CPUs with 2.7 GHz and the NuSMV-a command -dx -dynamic -df -coi -a pr int. The results in Table 1 show that NuSMV can handle networks with up to around 50 variables in reasonable times in the order of minutes. Antelope can answer queries in reasonable time for about 20 variables.

### 8 Case Study: Bladder Tumorigenesis

In this case study we investigate the bladder tumorigenesis model of [4] and compare the results of the trajectory-based
studies on phenotype reachability that are also available in [4]
with our state-based results. The tumorigenesis model has
four inputs which represent the cellular context of the net-
work in terms of DNA damage, the growth factors EGFR and
FGFR3 and growth inhibitors. Three output components,
Growth Arrest, Proliferation and Apoptosis, determine the
response of the cell to a given context. The model was
constructed using the software GINsim [30]. There are five
multi-valued components: ATM, Apoptosis, CHEK1_2, E2F1
and E2F3, each with the activity levels 0=low, 1=medium and
2=high. Since we use NuSMV-a as a plug-in of PyBoolNet
which is designed for Boolean networks we worked with the
equivalent Booleanized version in which multi-valued
components x are replaced by two Boolean components
x_medium and x_high. The update functions are obtained by
the so-called van Ham mapping which has been shown to
result in an equivalent transition system [31]. PyBoolNet
detects variables that represent the same multi-valued vari-
able and removes inadmissible combinations of values from
the initial states of model checking queries. Meaningless
states, e.g., x_medium=0 and x_high=1, that are artifacts of
the Booleanization do therefore not contribute to the basins,
commitment sets and phenotype sets. The Booleanization
was done using GINsim and the Boolean model is available
from the PyBoolNet model repository under the name
remy_tumorigenesis.

The analysis of the wild type multi-valued model consists
of a list of the observed attractors and their phenotypes, see
Table S2 of the supplementary materials, available online, of
[4], here referred to as Table S2, available online, and pie
charts that visualize the reachability of each phenotype under
different conditions, e.g., the wild type but also various
knock-out mutants. The reachability of a phenotype is defined
as the percentage of simulations (random walks on the asyn-
chronous state transition graph) that exhibit a certain pheno-
type. The authors ran 10,000 simulations for each of the 16
input configurations where the remaining variables are ran-
domly initialized.

We first confirmed that the attractors, 20 steady states and
5 cyclic attractors, of Table S2, available online, are identical to
the ones we find using the minimal trap space method [22,
[23] for our Boolean model. The 25 attractors exhibit four pheno-
types for the markers Proliferation, Apoptosis_medium, Apoptosis_high and Growth_arrest. The phenotypes are
listed in Table 2.

The basins of attraction are visualized as a bar plot in Fig. 4.
Since the model has four inputs, its state space is divided into
16 disconnected regions each making up $1/16 = 6.25\%$ of all
states. The maximal height of a bar is therefore at 6.25. The
bars are grouped by input combination which is indicated as
a four digit label that represents the activities of the four input
components in alphabetical order. Nine of the 16 input combi-
inations (0000, 0010, 0100, 0101,..) contain a single attractor and
hence the the whole region is its strong basin. Five input combi-
inations (0001, 1001, 1011, 1101, 1111) contain two attractors
and two input combinations (0011, 0111) contain three attrac-
tors with most of the region being just weakly attracted to
one of the attractors. Note that out of the 16 input combina-
tions only the two that contain three attractors are non-
deterministic with respect to the phenotype markers and both
may either develop towards GA or P.

The commitment diagram also reflects the division into 16
disconnected regions. It is divided into 16 disconnected dia-
agrams of three different shapes where the shape depends
only on how many attractors there are in the respective
region. An example of each is given in Fig. 5. If an input com-
bination contains more than one attractor, Figs. 5a and 5b,
then the majority of states, about 75 percent, can reach all
attractors. For input choices allowing for three attractors there
are combinations of attractors that are reachable from individ-
ual states (e.g., A10 and A8 in Fig. 5a) but also combinations
that are not reachable from individual states (e.g., A6 and
A10). It remains an open question to predict and understand
which combinations are possible and which are impossible.

The analysis in this case study shows a high degree of similar-
ity in the STG for different input combinations (only 3 dia-
gram shapes with identical sizes of commitment sets). Also, it
shows that a commitment to a specific attractor seems to occur
as late as possible in the sense that most states have the poten-
tial to reach every attractor that exists in the input region.

We then compared the simulation-based analysis of
phenotype reachability in [4] with our state-based approach.
The simulation-based analysis computes the percentage of
trajectories that finally exhibit a given phenotype. In [4] the
results are visualized as a pie chart with one slice per pheno-
type, see Fig. 6a (numbers taken from [4]). Our state-based
analysis, which uses model checking with accepting states,
finds that there are five different phenotype commitment

### Table 2

| m₁ | m₂ | m₃ | m₄ | steady | cyclic |
|----|----|----|----|--------|--------|
| GA | 0  | 0  | 0  | 1      | 7      |
| P  | 1  | 0  | 0  | 4      | 0      |
| A  | 0  | 1  | 0  | 1      | 9      |
| OscP/GA | * | 0  | 0  | 0      | 1      |

The color coding is consistent with Fig. 6. The last two columns record the
number of steady states and cyclic attractors of each phenotype. Note that a
phenotype may contain cyclic attractors even though all markers are steady.
sets: one for each phenotype but also one additional slice that represents states in the basin of several phenotypes, namely GA and P, see the white slice in Fig. 6b. To make the comparison between the two methods easier we visualize the phenotype commitment sets using a pie chart with the same colors.

It is remarkable that almost all states (except for the white slice) are committed to a single phenotype whereas in terms of attractors almost all states are committed to more than one attractor (see commitment diagram). This also explains why the pie charts of both methods look almost identical: most states are committed to a single phenotype and hence all simulations will lead to that phenotype. The 12.1 percent of state space that belong to the white slice, on the other hand, may develop towards either growth arrest or proliferation, depending on the particular transitions of a particular trajectory. Linking the phenotypes to the biological phenomenon they represent, we can see that the system, once committed to apoptosis, e.g., by certain components adopting apoptotic activity, will not divert from that fate. Growth arrest, in contrast, is not necessarily as strictly regulated. The white slice in the pie chart represents states where although some conditions leading to growth arrest are met, the system can still reverse them and enter proliferation if other factors outweigh the impact of the original setting at some point.

To summarize our observations: the state-based method can locate regions of state space that are “simulation-sensitive” (white slice) and regions that are deterministic in terms of the reachable system fate. Also, we observe that as we move from the attractor level to the phenotype level the degree of non-determinism seems to reduce drastically—at least for this particular model. The slices of Fig. 6b trivially correspond to likelihoods since they are all committed to a single phenotype.

**9 Discussion**

This paper puts the notions of a basin of attraction for non-deterministic transition systems on firm mathematical ground. We propose definitions of varying strictness and focus that allow to partition state space in a way providing
insight in the sequences of both reversible and non-reversible decisions ultimately leading the system into an attractor. For each introduced set and its corresponding reachability properties, we developed means of visualization, either in the form of a diagram or a pie chart.

On the computational side we define what CTL model checking with accepting states is and offer an extension to NuSMV that is capable of returning the accepting states of a formula. This extension, NuSMV-a, as well as the evaluation and visualization tools for all notions introduced here are now part of PyBoolNet which offers a high level interface to model checking Boolean networks with many functions for model and query generation already implemented. The comparison of NuSMV-a with Antelope suggests that computing accepting states with NuSMV-a is competitive and networks with more than 50 nodes can be handled on a scale of minutes.

Observations made in the case study lead us to the hypothesis that a model may be highly non-deterministic in terms of attractors and at the same time be almost deterministic in terms of phenotypes. This observation partially straightforward from a mathematical point of view, since the consideration of phenotypes based on a small number of markers would be expected to lower resolution significantly in most cases. However, the degree of this shift that we can observe in the case study is extremely high and might very well be related to the choice of markers that carry biological meaning and are expected to characterize system behavior well. In terms of higher level functions (commitment to phenotypes) this determinism suggests a robustness against lower level implementation details (the update order). This observation might be specific to the particular tumorigenesis model or even biological models and choices of relevant markers. In any case, it is also an interesting mathematical phenomenon that would be worthwhile to consider for broad classes of models, e.g., in randomly generated networks or across a repository of models.

We believe that in the context of non-deterministic transition systems a range of notions concerning attractor reachability are of interest. The cycle-free basins springing from the natural condition that all paths originating in a state should lead to the same attractor show that further distinctions concerning the asymptotic behavior of trajectories are possible and useful. The question in how far non-terminal cycles in the state transition graph can play a similar role to attractors lead to questions about the stability of such cycles. Progress concerning this issue in turn could potentially be highly meaningful for application, since it may allow to uncover robustly maintainable system behaviors in the models that are overlooked when only focusing on the attractors. A possible way to address this question mathematically and computationally may be in combining our state based analysis of commitment sets with trajectory-based analysis as we have outlined in the case study for the reachability analysis of phenotypes.

In a similar step to higher resolution, the commitment sets can be further refined by considering not only which attractors are reachable but also which other commitment sets, i.e., two states may both be committed to the same attractors but they traverse different commitment sets on their way to the attractors. It is certainly of mathematical interest to find the rule allowing to link the properties of the resulting graphs on the different resolution levels, which in turn links to the most exciting open question that results from this research: characterizing the control points and decision processes that occur at the border between commitment sets. Which changes in activity of which variables are responsible for the loss of a previously attainable attractor? What are minimal interventions that restore the reachability? Questions along these lines are crucial for understanding the dynamics of qualitative models and in practice for controlling cell differentiation and cell reprogramming. In particular, it remains an open question to understand the decision processes that are realized by transitions between commitment sets, see also [21].

ACKNOWLEDGMENTS

The authors thank the participants of the discussion about basins of attraction at Luminy University in Marseille in 2016: Elisabeth Remy, Claudine Chauouiya, Laurence Calzone, Denis Thieffry, Aurélien Naldi, Pedro Monteiro and Laurent Tournier. The work was partially funded by the German Federal Ministry of Education and Research (BMBF), grant no. 0316195.

REFERENCES

[1] R. Thomas, “Regulatory networks seen as asynchronous automata: A logical description,” J. Theoretical Biol., vol. 153, no. 1, pp. 1–23, 1991.
[2] A. Naldi, P. T. Monteiro, C. Müessel, Consortium for Logical Models, Tools, H. A. Kestler, D. Thieffry, I. Xenarios, J. Saez-Rodriguez, T. Helikar, and C. Chauoiuya, “Cooperative development of logical modelling standards and tools with CoLoMoTo,” Bioinf., vol. 31, no. 7, pp. 1154–1159, 2015.
[3] W. Abou-Jaoude, P. T. Monteiro, A. Naldi, M. Grandclaudon, V. Soumelis, C. Chauoiuya, and D. Thieffry, “Model checking to assess T-helper cell plasticity,” Frontiers Biöeng. Biotechnol., vol. 2, 2015, Art. no. 86.
[4] E. Remy, S. Rebouissou, C. Chauoiuya, A. Zinovyev, F. Radvanyi, and L. Calzone, “A modeling approach to explain mutually exclusive and co-occurring genetic alterations in bladder tumorigenesis,” Cancer Res., vol. 75, no. 19, pp. 4042–4052, 2015.
[5] N. D. Mendes, R. Henriques, E. Remy, J. Carneiro, P. T. Monteiro, and C. Chauoiuya, “Estimating attractor reachability in asynchronous logical models,” Frontiers Physiology, vol. 9, 2018, Art. no. 1161.
[6] K. Willadsen and J. Wiles, “Robustness and state-space structure of Boolean gene regulatory models,” J. Theoretical Biol., vol. 249, no. 4, pp. 749–765, 2007.
[7] A. Wuenische, “Complex and chaotic dynamics, basins of attraction, and memory in discrete networks,” Acta Physica Polonica B-Proc. Suppl., vol. 3, pp. 463–478, 2001.
[8] J. M. Babu and S. Comassot-Vivier, “Basins of attraction in fully asynchronous discrete-time discrete-state dynamic networks,” IEEE Trans. Neural Netw., vol. 17, no. 2, pp. 397–408, Mar. 2006.
[9] I. H. Dinwoodie, “Computational methods for asynchronous basins,” Discrete Continuous Dynamical Syst. Series B, vol. 21, no. 10, 2016, Art. no. 3391.
[10] G. Arelano, J. Argil, E. Azpeitia, M. Benitez, M. A. Carrillo, P. C. Gongora, D. A. Rosenthal, E. R. Alvarez-Buylla, et al., “Antelope: A hybrid-logic model checker for branching-time Boolean GRN analysis,” BMC Bioinf., vol. 12, no. 1, 2011, Art. no. 490.
[11] C. Baier and J.-P. Katoen, Principles of Model Checking. Cambridge, MA, USA: MIT Press, 2008.
[12] A. Cimatti, E. M. Clarke, E. Giunchiglia, F. Giunchiglia, M. Pistore, M. Roveri, R. Sebastiani, and A. Tacchella, “NuSMV 2: An eXtensible engine for symbolic model checking,” in Proc. 14th Int. Conf. Comput. Aided Verification, 2002, pp. 359–364.
[13] G. Bernt, J.-P. Comet, A. Richard, and J. Guespin, “Application of formal methods to biological regulatory networks: Extending Thomas’ asynchronous logical approach with temporal logic,” J. Theoretical Biol., vol. 229, no. 3, pp. 339–347, 2004.
[14] A. Streck, “Toolkit for reverse engineering of molecular pathways via parameter identification,” PhD thesis, Dept. Math. Comput. Sci., Freie Univ. Berlin, Berlin, Germany, 2016.
H. Klarner, A. Streck, and H. Siebert, “PyBoolNet: A python package for the creation, manipulation and analysis of Boolean networks,” *Bioinf.*, vol. 33, pp. 770–772, 2017.

E. Clarke, S. Jha, Y. Lu, and H. Veith, “Tree-like counterexamples in model checking,” in *Proc. 17th Annu. IEEE Symp. Logic Comput. Biotecn.*, pp. 19–29.

M. I. Davidich and S. Bornholdt, “Boolean network model predicts cell cycle sequence of fission yeast,” *PloS One*, vol. 3, no. 2, 2008, Art. no. e1672.

L. Tournier and M. Chaves, “Uncovering operational interactions in genetic networks using asynchronous Boolean dynamics,” *J. Theoretical Biol.*, vol. 260, no. 2, pp. 196–209, 2009.

M. I. Davidich and S. Bornholdt, “Boolean network model predicts cell cycle sequence of fission yeast,” *PloS One*, vol. 3, no. 2, 2008, Art. no. e1672.

H. Klarner, A. Bockmayr, and H. Siebert, “Computing maximal and minimal trap spaces of Boolean networks,” *Natural Comput.*, vol. 14, no. 4, pp. 378–392, 2010.

H. Klarner, “Interactive Boolean network designer,” 2018. [Online]. Available: http://page.mi.fu-berlin.de/klarner/iBoolNet/iBoolNet.html, Accessed on: Feb. 2018.

D. Bérenguier, C. Chaouiya, P. T. Monteiro, A. Naldi, E. Remy, D. Thieffry, and L. Tichit, “Dynamical modeling and analysis of large cellular regulatory networks,” *Chaos: An Interdisciplinary J. Nonlinear Sci.*, vol. 23, no. 2, 2013, Art. no. 025114.

H. Klarner, A. Bockmayr, and H. Siebert, “Computing maximal and minimal trap spaces of Boolean networks,” *Natural Comput.*, vol. 14, no. 4, pp. 535–544, 2015.

H. Klarner and H. Siebert, “Approximating attractors of Boolean networks by iterative CTL model checking,” *Frontiers Bioeng. Biotechn.*, vol. 3, 2015, Art. no. 130.

J. G. T. Zanudo and R. Albert, “Cell fate reprogramming by control of intracellular network dynamics,” *PLoS Comput. Biol.*, vol. 11, Apr. 2015, Art. no. e1004193.

G. Yang, J. Gómez Tejeda Zanudo, and R. Albert, “Target control in logical models using the domain of influence of nodes,” *Frontiers Physiology*, vol. 9, 2018, Art. no. 454.

H. Klarner, S. Nee, F. Heinitz, and R. Albert, “NuSMV-A: An extension of NuSMV2.6 for CTL model checking with accepting states,” 2016. [Online]. Available: github.com/hklarner/NuSMV-a, Accessed on: Sep. 2016.

R. K. Brayton, G. D. Hachtel, C. McMullen, and A. Sangiovanni-Vincentelli, *Logic Minimization Algorithms for VLSI Synthesis*, vol. 2. Berlin, Germany: Springer, 1984.

M. Dahlhaus, A. Burkovski, F. Hertwig, C. Mussel, R. Volland, M. Fischer, K.-M. Debatin, H. A. Kestler, and C. Beltinger, “Boolean modeling identifies Greactwal/MASTL as an important regulator in the AURKA network of neuroblastoma,” *Cancer Lett.*, vol. 371, no. 1, pp. 79–89, 2016.

L. Grieco, L. Calzone, I. Bernard-Pierrot, F. Radvanyi, B. Kahn-Perles, and D. Thieffry, “Integrative modelling of the influence of MAPK network on cancer cell fate decision,” *PLoS Comput. Biol.*, vol. 9, no. 10, 2013, Art. no. e1003286.

A. Naldi, D. Bérenguier, A. Fauré, F. Lopez, D. Thieffry, and C. Chaouiya, “Logical modelling of regulatory networks with GInSim 2.3,” *Biosoftware*, vol. 97, no. 2, pp. 134–139, 2009.

G. Didier, E. Remy, and C. Chaouiya, “Mapping multivalued onto Boolean dynamics,” *J. Theoretical Biol.*, vol. 270, no. 1, pp. 177–184, 2011.

For more information on this or any other computing topic, please visit our Digital Library at www.computer.org/csdl.