Switching Gene Regulatory Networks

Yoli Shavit[1,2], Boyan Yordanov[2], Sara-Jane Dunn[2], Christoph M. Wintersteiger[2], Youssef Hamadi[2], and Hillel Kugler[2,3]

[1] University of Cambridge, UK
[2] Microsoft Research
[3] Bar-Ilan University, Israel

Abstract. A fundamental question in biology is how cells change into specific cell types with unique roles throughout development. This process can be viewed as a program prescribing the system dynamics, governed by a network of genetic interactions. Recent experimental evidence suggests that these networks are not fixed but rather change their topology as cells develop. Currently, there are limited tools for the construction and analysis of such self-modifying biological programs. We introduce *Switching Gene Regulatory Networks* to enable the modeling and analysis of network reconfiguration, and define the synthesis problem of constructing switching networks from observations of cell behavior. We solve the synthesis problem using Satisfiability Modulo Theories (SMT) based methods, and evaluate the feasibility of our method by considering a set of synthetic benchmarks exhibiting typical biological behavior of cell development.

Keywords: Gene regulatory networks (GRNs), Boolean networks, Biological Modeling, Satisfiability Modulo Theories (SMT), Synthesis, Self-modifying Code.

1 Introduction

The cell is a fundamental unit of biological systems. Many aspects of cellular function are interpreted as the consequence of a series of genetic interactions that ultimately determine the expression levels of genes within the cell. Such interactions are composed into Gene Regulatory Networks (GRNs), which describe how individual genes regulate one another. Computational modeling allows us to represent a mechanistic understanding of GRNs, to formally compare model simulations to experimental data, explore new hypotheses and perform *in-silico* experiments.

Recent findings suggest that the process through which cells take on a specific role, termed differentiation, might be implemented by changing the accessibility of binding sites required for regulation [23], essentially enabling and disabling interactions in the GRN. When considering network reconfiguration in cells, self-modifying programs come to mind. Self-modifying programs are not a new concept in software, but they have not become mainstream, mainly because in most contexts they do not add expressive power, and they are hard to write

and analyze. Consequently, modern program analysis tools have no, or very limited, means of reasoning about such programs. It does appear, however, that for biological modeling, supporting the concept of switching networks can provide a useful abstraction for capturing the processes at work as cells change type.

To capture these phenomena, we introduce the concept of a *Switching Gene Regulatory Network* (SGRN), a framework for the analysis and synthesis of self-modifying biological programs. An SGRN is constructed to incorporate knowledge of network topology and to reproduce and explain experimental observations of system dynamics, by integrating known biological hypotheses. We formalize our approach and provide an encoding of SGRNs and bounded temporal constraints representing known experimental data, within a framework based on Satisfiability Modulo Theories (SMT) solvers. This builds upon and extends our previous work in the area, which supported only fixed GRNs [8, 26]. Finally, we evaluate the performance of our approach on a set of synthetic benchmarks in terms of running time, accuracy, and precision and we show that our method is scalable and that it reliably recovers the changes taking place in the network topology.

2 Background

We focus on *Boolean networks (BNs)* [13], a class of GRN models that are Boolean abstractions of genetic systems, *i.e.* every gene is represented by a Boolean variable specifying whether the gene is active or not. The concept of an *Abstract Boolean Network (ABN)* was introduced in [8] to allow the representation of models with initially unknown network topologies and dynamics. ABNs were then used to investigate the decision-making in pluripotent stem cells. In the following, we briefly review the relevant definitions from [8], which serve as a basis for the modeling approach described in later sections.

Let G be a finite set of genes and let $E : G \times G \times \mathbb{B} \to \mathbb{B}$ denote the set of directed edges between elements of G, labeled with a regulation activity ($\top$ for positive and $\bot$ for negative). Given genes g and g', we call g an *activator* of g' if $(g, g', \top) \in E$, a *repressor* if $(g, g', \bot) \in E$ and a *regulator* if it is either of those. Due to the Boolean abstraction of genetic states, the state space $Q = \mathbb{B}^{|G|}$ is induced implicitly where, for a given state $q \in Q$ and gene $g \in G$, $q(g) \in \mathbb{B}$ denotes the state of g. An update function $f_g : Q \to \mathbb{B}$ defines the dynamics of gene g. For a Boolean network with synchronous updates, the dynamics of the system are defined in terms of the update functions of all genes applied at each step, where given a current and next state $q, q' \in Q$, $\bigwedge_{g \in G} q'(g) = f_g(q)$. Although the presentation and examples in this paper focus on synchronous semantics, we also support asynchronous updates, where at each step the update function of only one gene is applied, while the value of all the other genes remains unchanged.

A set of 18 biologically plausible update function templates, which are called *regulation conditions*, was proposed in [8]. For a given gene $g \in G$, each regulation condition defines an update function $f_g : E \times Q \to \mathbb{B}$ that respects biologically-

inspired constraints. One such constraint is monotonicity, where the availability of additional activators does not lead to the inactivation of a gene, i.e., if a gene is expressed in q' when only some of its activators are expressed in q, then it must also be expressed in q' if all its activators are expressed in q and there is no change in the presence of repressors. These regulation conditions only consider whether none, some, or all potential activators or repressors of g are expressed in a state q.

To capture the possible uncertainty in the precise network topology, the ABN formalism allows some interactions to be marked as *optional* (denoted by the set $E^?$), each of which could be included in a synthesized *concrete model* (a model where all interactions are definite). Thus, in terms of network topology, an ABN model specifies a set of $2^{|E^?|}$ concrete models, each corresponding to a unique selection of optional interactions. Additionally, a choice of several possible regulation conditions for each gene is allowed, leading to the following definition:

Definition 1 (Abstract Boolean Network [8]). *An abstract Boolean network (ABN) is a tuple $\mathcal{N} = (G, E, E^?, R)$, where G is a finite set of genes, $E : G \times G \times \mathbb{B} \to \mathbb{B}$ is the set of definite (positive and negative) interactions between them, $E^? : G \times G \times \mathbb{B} \to \mathbb{B}$ is the set of optional interactions and $R = \{R_g \mid g \in G\}$, where R_g specifies a (non-empty) set of possible regulation conditions for each gene $g \in G$.*

An ABN is transformed into a concrete model by selecting a subset of the optional interactions to be included and assigning a specific regulation condition for each gene. Formally, let $\hat{E^?} \subseteq E^?$ denote the set of selected optional interactions, $\hat{E} = E \cup \hat{E^?}$ denote the set of all selected interactions and $\hat{R}_g \in R_g$ denote the specific regulation condition chosen for each gene $g \in G$. The semantics of such a concrete model are defined in terms of a transition system $\mathcal{T} = (Q, T)$, where $Q = \mathbb{B}^{|G|}$ is the set of states ($q(g) \in \mathbb{B}$ is the state of gene g in $q \in Q$) and the transition relation $T : Q \times Q \to \mathbb{B}$ is defined as

$$\forall q, q' \in Q \,.\, T(q, q') \leftrightarrow \bigwedge_{g \in G} q'(g) = \hat{R}_g(\hat{E}, q). \tag{1}$$

A finite *trajectory* of a concrete model is defined as a sequence of states $t = q_0, q_1, \ldots, q_K$ where $q_i \in Q$ and $\forall_{0 \le i < K} \,.\, T(q_i, q_{i+1})$. The semantics of an ABN can be understood in terms of the choice of optional interactions $\hat{E^?}$ and the choice of a regulation condition for each gene, $\hat{R}_g$, together with the transition system $\mathcal{T}$ representing the resulting concrete model.

A set of *experimental observations* that each concrete model needs to be able to satisfy are encoded as predicates over system states which limits the possible consistent choices of $\hat{E^?}$ and $\hat{R}_g$. For instance, an *experiment* in which genes g and g' are observed to be initially active and become inactive at step K is formalized as a constraint requiring the existence of a trajectory $t = q_0, \ldots, q_K$ such that $q_0(g) \wedge q_0(g') \wedge \neg q_K(g) \wedge \neg q_K(g')$. The approach developed in [8] allows GRN synthesis for non-switching networks: given an ABN and a set of experiments,

find a choice of interactions $\hat{E^?}$ and regulation conditions $\hat{R}_g$ for each gene, guaranteeing that the resulting concrete model is consistent with all experimental observations.

3 Switching Gene Regulatory Networks

We propose an extension of the ABN formalism, where transitions between unique cell types, characterized by potentially different network topologies, are directly supported. Let C denote a set of cell types sharing a set of genes G and regulation conditions R. Each cell type $c \in C$ is modeled as an ABN $\mathcal{N}_c = (G, E_c, E_c^?, R)$, where the set of definite interactions E_c and optional interactions $E_c^?$ could be different for different cell types. Note that, while the network topology is allowed to change between different cell types, we assume that the possible regulation conditions $R_g \in R$ depend only on a gene $g \in G$ and remain consistent across cell types.

Arbitrary transitions between different cell types are not plausible in most biological systems. For example, two distinct cell types $c, c' \in C$ can represent a progenitor cell c and a differentiated cell c' that is derived from c. While the progenitor can become a differentiated cell, the reverse does not occur under normal conditions. For each cell $c \in C$, we capture this information using the (non-empty) subset $D_c \subseteq C$ of all possible cell types that c can transition into directly. In order to capture mechanistic details within the model, our framework also supports the addition of guards, encoded as state predicates, to further constrain cell type switches. In the absence of restrictive guards, switching between cell types is represented as a nondeterministic choice (when $|D_c| > 1$), without explicitly modeling either the mechanism or preconditions on the system state required for such a switch.

This leads to the following definition of SGRNs:

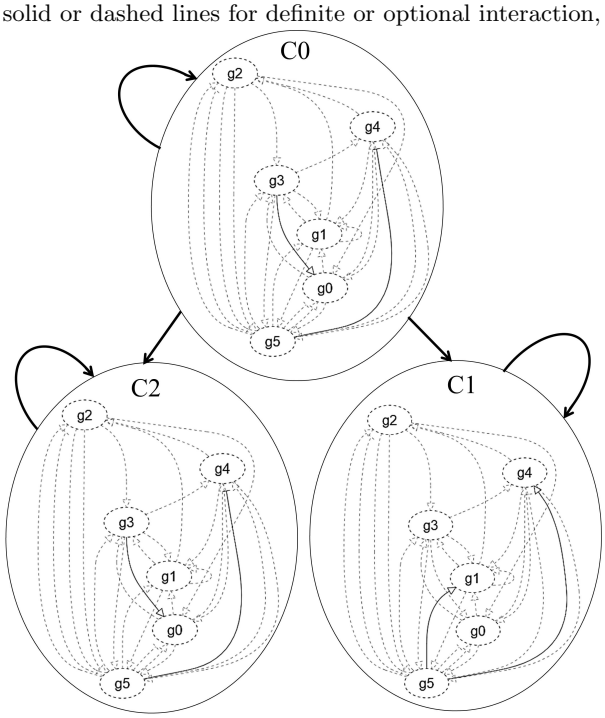
Definition 2 (Switching Gene Regulatory Network). *A Switching Gene Regulatory Network (SGRN) is a tuple $\mathcal{N}_S = (G, C, D_c, E_c, E_c^?, R)$, where*

- *G is the finite set of genes,*
- *C is a finite set of cell types,*
- *for each $c \in C$, $D_c \subseteq C$ is the set of cell types that c can transition into directly,*
- *$E_c : G \times G \times \mathbb{B} \to \mathbb{B}$ is the set of definite interactions between genes for each $c \in C$,*
- *$E_c^? : G \times G \times \mathbb{B} \to \mathbb{B}$ is the set of optional interactions for cell type c, and*
- *$R = \{R_g \mid g \in G\}$, where R_g specifies a (non-empty) set of possible regulation conditions for each gene $g \in G$.*

Fig. 1 shows an SGRN with 3 cell types: $C = \{c_0, c_1, c_2\}$, and 6 genes: $G = \{g_0, g_1, g_2, g_3, g_4, g_5\}$. In this example, a (progenitor) cell type, c_0, may change into cell types c_1 or c_2, by reconfiguring its network, so that $D_{c_0} = \{c_1, c_2\}$, while c_1 and c_2 cannot switch their identity (thus $D_{c_1} = \{c_1\}$ and $D_{c_2} = \{c_2\}$). For

each cell type, edges between genes appear in solid or dashed lines for definite (E_c) or optional ($E_c^?$) interactions respectively. Genes appear in dashed circles to indicate that R consists of multiple possible regulations conditions for each gene.

Fig. 1. An SGRN with 3 cell types ($c0$-$c2$) with 6 genes ($g0$-$g5$), illustrating a typical setting where one cell type ($c0$) can maintain its identity (self-loops) or give rise to other cell types by switching its interactions. Edges between genes represent regulatory interactions, with a bar representing repression and an arrow representing activation, and appear as solid or dashed lines for definite or optional interaction, respectively.



As in Section 2, the semantics of SGRNs are defined in terms of a transition system $\mathcal{T} = (Q, T)$. Here $Q = \mathbb{B}^{|G|} \times C$ is the set of states where, for a given state $q \in Q$, $q = (q_G, q_C)$, $q_G(g) \in \mathbb{B}$ indicates the state of a given gene $g \in G$ and $q_C \in C$ indicates the current cell type. For a concrete switching GRN model, let $\hat{E}_c^? \subseteq E_c^?$ denote the set of selected optional interactions and $\hat{E} = E_c \cup \hat{E}_c^?$ denote the set of all selected interactions for each cell type $c \in C$. Let $\hat{R}_g \in R_g$ denote the specific regulation condition selected for each gene $g \in G$, which is

the same for all cell types. The transition relation $T : Q \times Q \to \mathbb{B}$ is defined as

$$\forall q, q' \in Q.\, T(q, q') \leftrightarrow \bigwedge_{c \in C} \left[q_C = c \to \left(q'_C \in D_c \wedge \bigwedge_{g \in G} q'_G(g) = \hat{R}_g(\hat{E}_c, q_G) \right) \right]. \tag{2}$$

Intuitively, Eqn. 2 captures the fact that all genes are updated according to the selected regulation conditions $\hat{R}_g$ and the network topology $\hat{E}_c$ corresponding to the particular cell type c in the current state q. In the next state q', the cell type can be updated (non-deterministically) to one of the possible cell types $D_c \subseteq C$ that c can transition into directly. As for ABNs, given an assignment of the optional interactions $\hat{E}_c^?$ for each cell type $c \in C$, and a specific regulation condition $\hat{R}_g$ for each gene $g \in G$, Eqn. 2 allows us to define finite trajectories of the resulting concrete SGRN models as a sequence of states $t = q_0, q_1, \ldots, q_K$ from Q where $\forall_{0 \leq i = 0 < K}\, .\, T(q_i, q_{i+1})$.

4 SGRN model synthesis

We are interested in concrete SGRN models that are consistent with given experimental observations. In this section, we formalize this as a synthesis problem and present the details of our solution and implementation.

A SGRN model $\mathcal{N}_S = (G, C, D_c, E_c, E_c^?, R)$ is transformed into a concrete model by selecting a specific regulation condition $\hat{R}_g \in R_g$ for each gene $g \in G$ and a subset of the optional interactions $\hat{E}_c^? \subseteq E_c^?$ to be included for each cell type $c \in C$. Each possible concrete model is represented as a transition system $\mathcal{T} = (Q, T)$, where the system set of states is $Q = \mathbb{B}^{|G|} \times C$.

Let $\pi : Q \to \mathbb{B}$ denote a state predicate capturing some observed gene states or cell type and the tuple (π, n) denote a constraint that, for a given trajectory $t = q_0, \ldots, q_K$, t satisfies π at step n (*i.e.* $\pi(q_n) = \top$). An *experiment* $\mathcal{E} = \{(\pi_i, n_i) \,|\, i = 0 \ldots M\}$, where π_i is a state predicate and $n_i \in [0, K]$ for all $i \in [0, M]$, is expressed as a finite set of such constraints and formalizes the gene expressions or cell types observed during a particular execution of the system. We write $t \vDash \mathcal{E}$ when trajectory t satisfies experiment $\mathcal{E}$ (*i.e.* when $\bigwedge_{(\pi,n) \in \mathcal{E}} \pi(q_n)$). More complicated expressions can also be constructed as part of an experiment by combining terms (π, n) using the logical operators $\{\wedge, \vee, \Rightarrow, \Leftrightarrow, \neg\}$.

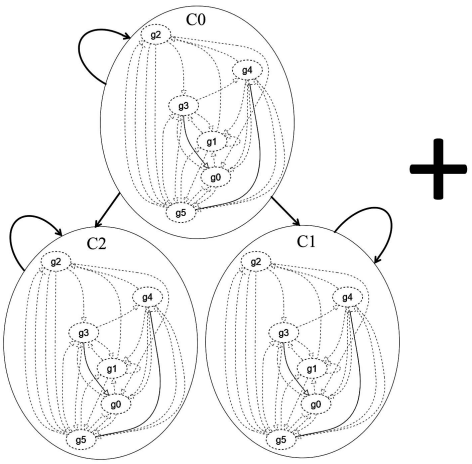
The main problem we consider in this paper is the following:

Problem 1 (Lineage Synthesis) *Given an SGRN $\mathcal{N}_S = (G, C, Dc, E_c, E_c^?, R)$ and a finite set of experiments $\mathcal{E}_0, \ldots, \mathcal{E}_m$, find an assignment $\hat{E}_c^?$ of the optional interactions $E_c^?$ for each cell type $c \in C$ and a single regulation condition $\hat{R}_g \in R_g$ for each gene $g \in G$ such that, for each $i = 0, \ldots, m$ there exists a trajectory t_i of the resulting concrete model that satisfies $\mathcal{E}_i$ (*i.e.* $t_i \vDash \mathcal{E}_i$).*

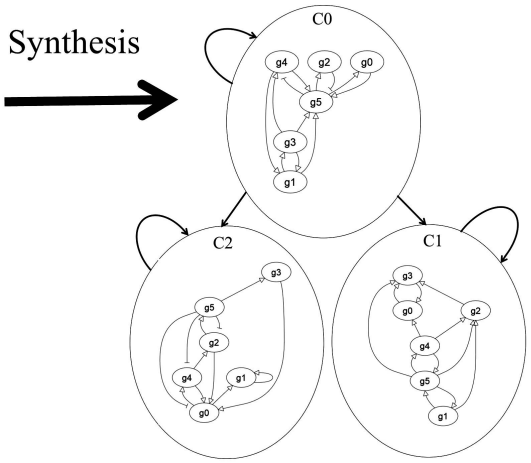
Fig. 2 illustrates a lineage synthesis problem for an example SGRN.

Given an SGRN $\mathcal{N}_S = (G, C, Dc, E_c, E_c^?, R)$ we encode the choice of optional interactions $\hat{E}_c^?$ for each cell type $c \in C$ using a unique Boolean choice variable

Fig. 2. A lineage synthesis problem. The SGRN from Fig. 1 and a finite set of experiments define a lineage synthesis problem. A solution for this problem includes the assignment of definite interactions for each cell type and the choice of a single regulation condition for each gene.



Step	Cell	g0	g1	g2	g3	g4	g5
Experiment 1							
0	C0	1	0	1	1	1	0
1	C0	0	1	0	0	1	1
8	C0	1	0	1	0	0	1
10	C1	1	0	1	0	0	1
Experiment 2							
0	C0	1	0	0	1	0	0
2	?	1	0	1	0	0	1
10	C2	1	0	0	1	0	1
…							
Experiment m							
0	C0	1	1	0	0	1	0
10	C0	1	0	1	0	0	1



for each interaction, or more conveniently, as a single bit-vector using the respective SMT theory. Additionally, a single regulation condition $\hat{R}_g$ from the set of allowed conditions R_g must be selected for each gene $g \in G$. We encode this as the synthesis of a single bit-vector or integer 'coefficient' for each gene, which is shared across all cell types.

The choice variables for optional interactions of each cell type and regulation conditions for each gene allow us to consider the transition system $\mathcal{T} = (Q, T)$ as defined in Section 3, which represents a given concrete model. The set of states $Q = \mathbb{B}^{|G|} \times C$ is finite since both the number of genes G and the number of cell

types C are finite. Furthermore, for a given state $q \in Q$ where $q = (q_G, q_C)$, the component of the state space describing the state of all genes q_G is encoded as a single bit-vector using the SMT theory of bit-vectors. In our implementation, we represent the cell type component of a state q_C using a "one-hot" encoding, where $q_C \in \mathbb{B}^{|C|}$ with the guarantee that the cardinality of q_C for any state $q \in Q$ is 1. This allows us to represent the entire state (q_G, q_C) as individual Boolean variables or as a single bit-vector.

We follow a bounded model checking (BMC) approach [4], and unroll the transition relation T of $\mathcal{T}$ to define a trajectory t_i for each experiment $\mathcal{E}_i$ (see Problem 1), for which the corresponding experimental observations from $\mathcal{E}_i$ are asserted. Note that while a separate trajectory t_i is used for each experiment $\mathcal{E}_i$, we do not require these trajectories to be unique (*i.e.* it is possible that a single trajectory $t = t_i = t_j$ satisfies the constraints of both experiments $\mathcal{E}_i$ and $\mathcal{E}_j$).

Finally, we employ an SMT solver to determine the satisfiability of all generated constraints (our choice of SMT solver is Z3 [16]). Here, we exploit the fact that SMT solvers such as Z3 produce an assignment of all the constants used in the encoding of the problem, which is presented as a certificate of the satisfiability of all constraints. When such an assignment (referred to as a "model" in this context) is found, we extract the optional interactions $\hat{E}_c^?$ selected for each cell type and the regulation condition $\hat{R}_g$ selected for each gene. In addition, since each trajectory t_i was represented explicitly as part of the problem, the exact sequence of states is recovered from the model synthesized by the SMT solver, to serve as an example demonstrating exactly how the SGRN reproduces the behavior observed in each experiment $\mathcal{E}_i$. In addition to the sequence of gene expressions at each time point, this information also reveals the cell types along executions of the system, allowing for further investigation of the captured cellular differentiation processes.

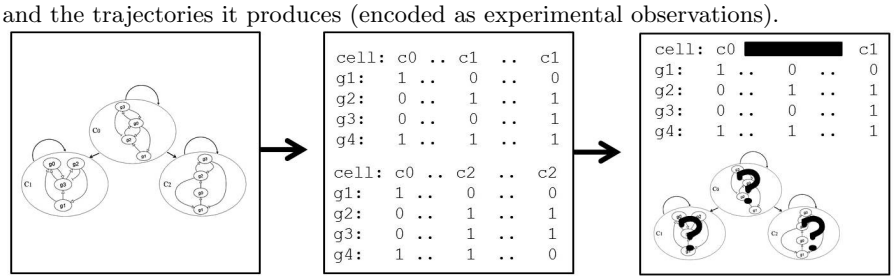
5 Experimental Results

In order to test our approach and systematically evaluate its performance we require benchmarks of lineage synthesis problems for SGRNs with different number of genes and cell types. This is achieved by producing synthetic problems, following the main steps summarized in Fig. 3 and described in Subsection 5.1. Subsection 5.2 gives the results of our evaluation in terms of accuracy, precision and running time.

5.1 Benchmark Design

Cell types are defined by directed networks with a scale-free topology (the degree of the vertices follows a power-law distribution), which is a common feature of GRNs and other biological networks [2], with the exponent of the degree distribution set to 2 (for both in- and out- degree distributions). Interactions are labelled with either a positive or negative sign, such that each gene has at least one activator. This is in keeping with the assumption that, by default, genes are

Fig. 3. The three steps for generating an *in-silico* lineage synthesis problems involve: (a) randomly generating a concrete SGRN, where all interactions are definite and a single regulation condition is allowed for each gene. (b) Generating trajectories of the concrete SGRN model from (a). This essentially amounts to simulation, which is possible since the model does not include any uncertainty. (c) Generating a lineage synthesis problem with partial information about the interactions in the system (encoded as an SGRN) and the trajectories it produces (encoded as experimental observations).



(a) Generate the SGRN. Assign interactions between genes in each cell type.

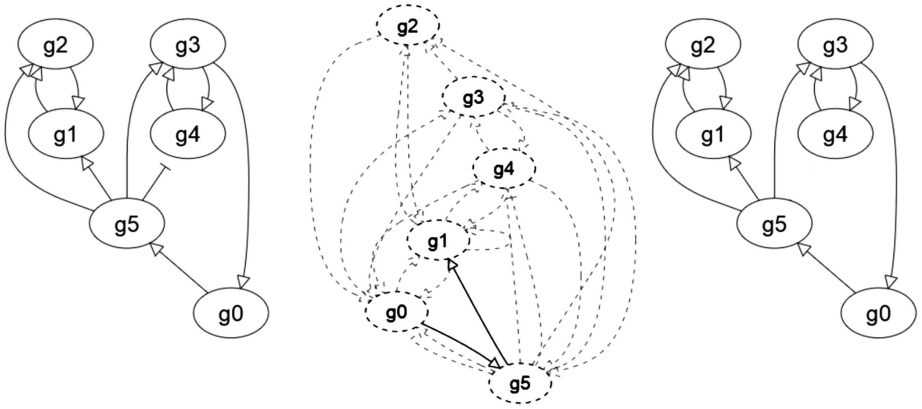
(b) Generate trajectories. Trace the gene states as the cell switches its type.

(c) Set up the problem. Hide information about switches and cells' topology.

repressed in higher organisms, and must be "switched on" to be expressed and behave as regulators of their target genes [19]. A regulation condition is randomly assigned to each gene from a set of 16 out of the 18 regulation conditions defined in [8], excluding the two functions that allow activation of a gene in the absence of any activators. For a given model with m cell types, n genes, and a progenitor cell type c_0, we generate $2 \cdot m \cdot n$ trajectories of length $K = 11$ starting at c_0 with a gene state configuration j, and switching to cell type c_i at a randomly selected time point s, for $i = 0 \,..\, m$, $j = 1 \,..\, 2n$ and $1 \leq s \leq K$. In order to create the set of $2n$ starting gene state configurations, we randomly select $2n - 2$ integer values in the range $(0, 2^n - 1)$ (exclusive) and add the values 0 and $2^n - 1$, representing the extreme configurations of the system. System states are represented by bit-vectors of size $|G|$, where the k^{th} position in the vector represents the state of the k^{th} gene.

To construct an instance of the lineage synthesis problem, each model (generated as described above) is used to produce a SGRN and its trajectories are encoded as experimental observations. We assume no information about the exact regulation conditions available and, therefore, all 16 choices are allowed for each gene. Let $E_c^\star$ denote the interactions of cell type c in the "true" model and $E^\star = \cup_{c \in C} E_c^\star$ denote the interactions appearing in any cell type. We construct the SGRN by assigning a small proportion (20%) of $E_c^\star$ as definite for cell type c (representing known interactions) and marking the rest of $E^\star$ as optional, which defines the sets E_c and $E_c^?$ respectively (Fig. 4). Each trajectory is then used to generate an experiment with the gene states observed at each time step, and the cell type observed at the start and at the end of the experiment (time steps 0 and 10, correspondingly). In total, this amounts to $2 \cdot m \cdot n$ experiments included in a lineage synthesis problem of m cells and n genes.

Fig. 4. A true, a hypothesized, and a synthesized progenitor cell type in an SGRN with 6 genes ($g0$-$g5$) and 3 cell types. The true cell type (a) was generated with a scale-free topology. The union of all cell types in the SGRN was used to create the hypothesized cell type (optional interactions appear as dashed lines) with a small proportion of its true interactions known (solid lines). Genes appear with dashed circles to indicate that their regulation condition is not fixed. The synthesized cell type is part of our solution for a lineage synthesis problem generated for this SGRN and recovers the true cell type with the exception of the negative interaction from $g5$ to $g4$.



(a) True (b) Hypothesis (c) Synthesis

5.2 Results

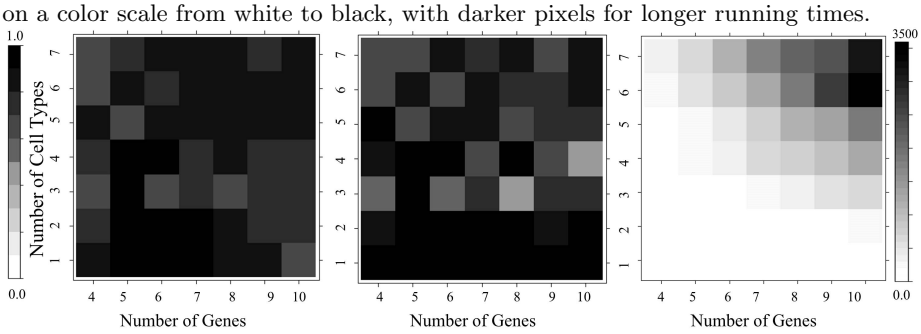
We demonstrate our technique on benchmarks of lineage synthesis problems with 1-7 cell types and 4-10 genes, generated as described above. For each problem we record the running time required to solve and we evaluate solutions by means of accuracy and precision in relation to the 'hidden' true model from which each problem was generated.

Let $E_c^\star$ denote the "true" interactions of cell type c, E_c ($E_c^?$) denote the definite (optional) interactions of the corresponding SGRN cell type, and $\hat{E}_c$ ($\hat{E}_c^?$) denote the synthesized (optional) interactions. A *True Positive* (*Negative*) is an interaction that is (not) in $\hat{E}_c^?$ and (not) in $E_c^\star$ (note that we evaluate the synthesis of only those interactions that were optional in the SGRN since definite interactions will always be part of the synthesized model). A *False Positive* is an interaction in $\hat{E}_c^?$ that is not in $E_c^\star$ and a *False Negative* is an interaction in $E_c^\star$ that is not in $\hat{E}_c^?$. The precision of a solution for a given cell type is then defined as $\frac{TP}{TP+FP}$, and its accuracy as $\frac{TP+TN}{TP+FP+TN+FN}$, with TP, TN, FP and FN, the number of True Positives, True Negatives, False Positives and False Negatives, respectively. The total precision and accuracy of a solution is the mean precision and accuracy across all cell types in the problem.

The results of our evaluation (Fig. 5a,b) show that our approach can successfully recover hidden topologies of SGRNs, achieving 0.81 accuracy and 0.78 precision (on average, across 2-7 cell types and 1-10 genes). As evident from

the heatmaps in Fig. 5a,b, cell types are synthesized with good accuracy across problems (17% with accuracy > 0.9, 86% of cases with accuracy > 0.7 and all problems with accuracy > 0.6) and with good precision in the majority of cases (71% of cases with precision > 0.7). For our benchmarks, the performance seems to be independent of the number of cells or genes. The running time of our synthesis is also feasible for the SGRNs under consideration, with all problems in the benchmark set solved in under an hour on a personal computer (Intel Core i3-4010U 1.7GHz, 4GB RAM, Windows 8.1 64-bit OS) and with an average running time of 730.25 seconds (Fig. 5c).

Fig. 5. Heatmaps of experimental results for a benchmark of lineage synthesis problems with 1-7 cells and 4-10 genes. Darker pixels indicate higher accuracy (a) and precision (b), while lighter pixels indicate poorer performance. Running times (c) are indicated on a color scale from white to black, with darker pixels for longer running times.



6 Related Work

Since the early days of computer science, the concept of self-modifying programs has been a natural one to explore, especially after the introduction of the Von Neumann architecture [17], in which both the program and the data were stored in the same memory, leading to the possibility of allowing program modification during runtime. This model was supported in early computer architectures (cf. e.g., [3]) and applied in some specific domains, however it did not become a mainstream paradigm.

Boolean networks have been suggested for studying cell differentiation [13, 24]. In this context the concept of *switching* was mainly used to describe changes in the state of the nodes (genes) rather than the reconfiguration of the topology of the network itself. The change in the gene's state could be a result of executing the GRN and by including additional effects such as the spatio-temporal dynamics of the neighbouring cellular (tissue) environment (for example: [7, 10]). However, little attention was given to the rewiring of the network as a mechanism to achieve differentiation or changes in the cellular function.

Petrinets and their extensions have been used in modeling of GRNs (see e.g., [5, 11]) and in particular the extension of self-modifying nets [25] enables to describe reconfiguration of Petrinets. This is achieved by allowing an arc to refer to a place, implying that the number of tokens in this place should be added/removed while firing the transition. The number of tokens in a place can change during execution leading to the 'reconfiguration' of the net. Self-modifying nets and further extensions have been used in modeling of metabolic networks [12], where self-modification permits the representation of concentrations and kinetic effects. It is known that self-modifying Petrinets are more expressive than conventional Petrinets, making the reachability problem undecidable [25], whereas in our work we defined a framework in which the basic dynamic properties of the system remain decidable.

Bayesian networks have been extensively applied to the problem of inference of gene regulatory networks from time series data [9]. Unlike our work, these methods handle continuous variables and stochastic events, but they lack some of the general advantages of reasoning based approaches, including proofs that solutions do not exist and effective ways to symbolically reason about sets of solutions. More recently, there has been research on generalizing Bayesian networks inference to the case of time varying networks (e.g., [21, 22, 1, 18, 6, 14]).

Related concepts of *switching* have also been introduced and explored in other fields. For example, mode-automata was proposed as a formalism for modelling reactive systems, in order to capture explicitly a decomposition of the system's global behaviour into multiple independent tasks [15]. In our work, however, such a decomposition is not fully known a priori and our focus is on synthesizing the structure of the system in different cell types, which can be viewed as modes, together with the transitions between them. Thus, our approach is also related to methods for the synthesis of controllers for discrete event systems (e.g. [20]) - a problem that has received considerable attention. However, the problem we address requires the synthesis of a system for each cell type such that the overall behaviour reproduces certain experimental observations, rather than synthesizing a controller that, when coupled with the system, restricts its behaviour to some desirable subset.

7 Conclusion

Computational methods are becoming a powerful tool for experimental biologists to improve the understanding of cellular decision-making. In particular, formal reasoning and different synthesis approaches are attractive as they enable the automatic generation of models that are guaranteed to satisfy a given set of constraints representing known experimental measurements. Motivated by recent biological evidence suggesting that it makes sense to view a molecular program within a cell as a self-modifying program, we introduce a framework that allows us to represent cellular reconfiguration, and effectively synthesize models that are consistent with experimental constraints and hypotheses. This opens the way to combined computational and experimental research to improve our un-

derstanding of how cells differentiate into specific cell types during development, as well as how cells may modify their behavior under artificial culture conditions used for research and medical applications. A long-term research goal is to gain a mechanistic understanding of how self-modifying biological programs operate and investigate whether the underlying principles nature utilizes can inspire new directions for the design of self-modifying software.

Acknowledgments. Yoli Shavit is supported by the Cambridge International Scholarship Scheme (CISS). The research was carried out during her internship at Microsoft Research Cambridge, UK.

References

1. Ahmed, A., Xing, E.: Recovering time-varying networks of dependencies in social and biological studies. Proc. Nat. Acad. of Sciences 106(29) (2009)
2. Albert, R.: Scale-free networks in cell biology. J Cell Sci. 118 (2005)
3. Bashe, C., Johnson, L., Palmer, J., Pugh, E.: IBM's early computers. MIT Press (1986)
4. Biere, A., Cimatti, A., Clarke, E., Zhu, Y.: Symbolic model checking without BDDs. In: TACAS, vol. 1579 of LNCS, Springer (1999)
5. Chaouiya, C.: Petri net modelling of biological networks. Briefings in Bioinformatics 8(4) (2007)
6. Dondelinger, F., Lébre, S., Husmeier, D.: Non-homogeneous dynamic bayesian networks with bayesian regularization for inferring gene regulatory networks with gradually time-varying structure. Machine Learning 90(2) (2013)
7. Doursat, R.: The growing canvas of biological development: Multiscale pattern generation on an expanding lattice of gene regulatory nets. In: Minai, A., Braha, D., Bar-Yam, Y. (eds.) Unifying Themes in Complex Systems, pp. 205–210. Springer Berlin Heidelberg (2008)
8. Dunn, S., Martello, G., Yordanov, B., Emmott, S., Smith, A.: Defining an essential transcription factor program for naïve pluripotency. Science 344(6188) (2014)
9. Friedman, N., Linial, M., Nachman, I., Pe'er, D.: Using Bayesian networks to analyze expression data. J. Comp. Bio. 3(7) (2000)
10. Giavittob, J., Klaudela, H., Pommereau, F.: Integrated regulatory networks (IRNs): Spatially organized biochemical modules. Theoretical Computer Science 431(0), 219–234 (2012)
11. Heiner, M., Gilbert, D., Donaldson, R.: Petri nets for systems and synthetic biology. FMCSB 5016 (2008)
12. Hofestädt, R., Thelen, S.: Quantitative modeling of biochemical networks. In Silico Biology 1(1) (1998)
13. Kauffman, S.: Metabolic stability and epigenesis in randomly constructed genetic nets. Journal of Theoretical Biology 22(3) (1969)
14. Khan, J., Bouaynaya, N., Fathallah-Shaykh, H.: Tracking of time-varying genomic regulatory networks with a lasso-kalman smoother. EURASIP J. Bioinf. and Sys. Bio. 3 (2014)
15. Maraninchi, F., Rémond, Y.: Mode-automata: About modes and states for reactive systems. In: European Symposium On Programming (1998)

16. de Moura, L., Bjørner, N.: Z3: An Efficient SMT Solver. In: TACAS. LNCS, vol. 4963. Springer (2008)
17. von Neumann, J.: First draft of a report on the EDVAC. Tech. Rep. Contract No. W670ORD4926, Moore School of Elec. Eng., Univ. of Pennsylvania (1945)
18. Parikh, A., Wu, W., Curtis, R., Xing, E.: TREEGL: reverse engineering tree-evolving gene networks underlying developing biological lineages. Bioinf. 27(13) (2011)
19. Phillips, T.: Regulation of transcription and gene expression in eukaryotes. Nature Education 1 (2008)
20. Ramadge, P.J., Wonham, W.M.: Supervisory control of a class of discrete event processes. SIAM J. Control Optim. 25(1), 206–230 (1987)
21. Rao, A., Hero, A., States, D., Engel, J.: Inferring time-varying network topologies from gene expression data. EURASIP J. Bioinformatics Syst. Biol. 1 (2007)
22. Song, L., Kolar, M., Xing, E.: Time-varying dynamic Bayesian networks. In: Advances in Neural Information Processing Systems (NIPS) (2009)
23. Stergachis et al.: Developmental fate and cellular maturity encoded in human regulatory DNA landscapes. Cell 154 (2013)
24. Thomas, R., Kaufman, M.: Multistationarity, the basis of cell differentiation and memory. ii. logical analysis of regulatory networks in terms of feedback circuits. Chaos 11(1), 180–95 (2001)
25. Valk, R.: Self-modifying nets, a natural extension of Petri nets. In: Proc. 5th Colloquium on Automata, Languages and Programming. LNCS, vol. 62. Springer (1978)
26. Yordanov, B., Wintersteiger, C., Hamadi, Y., Kugler, H.: Z34Bio: An SMT-based framework for analyzing biological computation. In: SMT (2013)