BioSystems xxx (2016) xxx–xxx

Contents lists available at ScienceDirect

BioSystems

journal homepage: www.elsevier.com/locate/biosystems

Automated Synthesis and Analysis of Switching Gene Regulatory Networks[☆]

Yoli Shavit[a,b], Boyan Yordanov[b], Sara-Jane Dunn[b], Christoph M. Wintersteiger[b], Tomoki Otani[a], Youssef Hamadi[b], Frederick J. Livesey[a], Hillel Kugler[b,c,*]

[a] University of Cambridge, UK
[b] Microsoft Research, UK
[c] Bar-Ilan University, Israel

ARTICLE INFO

Article history:
Received 1 March 2016
Accepted 30 March 2016
Available online xxx

Keywords:
Gene regulatory networks (GRNs)
Boolean networks (BNs)
Biological modelling
Satisfiability Modulo Theories (SMT)
Synthesis
Self-modifying code
Cell fate
Mammalian cortex

ABSTRACT

Studying the gene regulatory networks (GRNs) that govern how cells change into specific cell types with unique roles throughout development is an active area of experimental research. The fate specification process can be viewed as a biological program prescribing the system dynamics, governed by a network of genetic interactions. To investigate the possibility that GRNs are not fixed but rather change their topology, for example as cells progress through commitment, we introduce the concept of *Switching Gene Regulatory Networks* (SGRNs) to enable the modelling and analysis of network reconfiguration. We define the synthesis problem of constructing SGRNs that are guaranteed to satisfy a set of constraints representing experimental observations of cell behaviour. We propose a solution to this problem that employs methods based upon Satisfiability Modulo Theories (SMT) solvers, and evaluate the feasibility and scalability of our approach by considering a set of synthetic benchmarks exhibiting possible biological behaviour of cell development. We outline how our approach is applied to a more realistic biological system, by considering a simplified network involved in the processes of neuron maturation and fate specification in the mammalian cortex.

1. Introduction

Gene regulatory networks (GRNs) are used to describe how individual genetic components regulate each other to determine gene expression patterns and, consequently, cellular decision-making. Computational modelling of GRNs (Davidson et al., 2002; Le Novère, 2015) can be used effectively to complement experimental work, to elucidate and summarise a mechanistic understanding of a system precisely, to check if models reproduce experimental data, to explore new hypotheses, and to make predictions that can then be tested experimentally. Despite the broad spectrum of languages and formalisms now available to model GRNs, which may be gainfully used to study the cellular decision-making that occurs during differentiation – the process through which cells take on a specific role – the majority hold the assumption that the network topology is fixed throughout. Recent findings suggest that differentiation might arise as the accessibility of binding sites required for genetic regulation change (Stergachis et al., 2013), essentially enabling and disabling interactions in the GRN (Yosef et al., 2013).

To capture these phenomena, we introduce the concept of a *Switching Gene Regulatory Network* (SGRN), which is a modelling language and framework for the analysis and synthesis of reconfiguring GRNs. An SGRN is constructed to incorporate knowledge of network topology and to reproduce and explain experimental observations of system dynamics by integrating known gene expression measurements and biological hypotheses. At the core of the approach is a synthesis algorithm, which can decide algorithmically whether there exists an SGRN that is guaranteed to satisfy given experimental observations, and a set of assumptions on the possible cell types, switches, and interactions, which are specified as constraints. Towards this goal, we formalise our modelling framework and provide an encoding of SGRNs together with bounded temporal(-logic) constraints representing known experimental data, within a framework based on Satisfiability Modulo Theories (SMT) solvers. This approach naturally builds upon and extends our previous work in the area, in which we propose

[☆] An abridged version of this article appeared in Proc. 10th International Conference on Information Processing in Cells and Tissues (IPCAT 2015), LNCS vol. 9303, Springer.

[*] Corresponding author at: Faculty of Engineering, Bar-Ilan University, Ramat Gan, Israel.

E-mail address: hillelk@biu.ac.il (H. Kugler).

http://dx.doi.org/10.1016/j.biosystems.2016.03.012

various techniques for analysis and synthesis of fixed (non-switching) GRNs, which was successfully applied to uncover an essential pluripotency program for embryonic stem cells (Dunn et al., 2014) and a range of other biological programs (Yordanov et al., 2016).

We evaluate the performance of our approach on a set of synthetic benchmarks in terms of running time, accuracy, and precision. We show that our method is scalable and that it reliably recovers the changes taking place in the network topology. Finally, we outline a case study of a particular SGRN that is used to describe cell-fate decision making and maturation of neurons within the mammalian cortex.

2. Materials and methods

In this paper we introduce SGRNs as an extension of *Boolean Networks* (BNs) (Kauffman, 1969). The main ideas do, in principle, generalise to less abstract formalisms such as qualitative networks, chemical reaction networks or differential equation models, but here we focus on SGRNs as an extension of Boolean networks.

BNs are 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 inactive (on or off). The concept of *Abstract Boolean Networks* (ABNs) (Dunn et al., 2014) was introduced to allow the representation of models with network topologies and dynamics that are initially unknown or uncertain. Those models were then used to investigate decision-making in pluripotent stem cells. In the following, we briefly review the relevant definitions (Dunn et al., 2014; Yordanov et al., 2016), which serve as a basis for the modelling approach described in the following sections.

We begin to describe the formalisation by letting G be a finite set of genes. Let $\mathbb{B} = \{\top, \bot\}$ be the Boolean domain and let $\mathbb{B}^n$ be all vectors of n Booleans. Let D be a set of directed edges between elements of G, i.e., $D \subseteq G \times G \times \mathbb{B}$. To specify the sign of an interaction, we require an additional label on each regulation activity, which is either $\top$ for positive or $\bot$ for negative activities, and we attach one such label to each edge within a network, thus obtaining a set of labelled interactions $E \subseteq D \times \mathbb{B} = G \times G \times \mathbb{B} \times \mathbb{B}$. Let g and g' be genes from G. We call g an *activator* of g' iff $(g, g', \top) \in E$, a *repressor* iff $(g, g', \bot) \in E$, and a *regulator* iff it is either an activator or a repressor, i.e., when $\{(g, g', \top)\} \cup \{(g, g', \bot)\} \cap E \neq \emptyset$. In line with the adoption of a Boolean abstraction of genetic states, we define the state space of a system as $Q = \mathbb{B}^{|G|}$. For a given state $q \in Q$ and a gene $g \in G$, we denote by $q(g)$ the state of g in q.

Each gene is associated with an update function f_g with a signature $f_g : Q \rightarrow \mathbb{B}$ defining its dynamics. For *synchronous* updates, the dynamics of the system are defined in terms of the update functions of all genes applied at each transition (time step), where, given a current state q and next state q', we always have that $\bigwedge_{g \in G} q'(g) = f_g(q)$. In this paper, we focus on synchronous semantics, but if so desired, asynchronous semantics are incorporated by requiring that at each transition the update function of only one, non-deterministically chosen gene, is applied, while the value of all other genes remains unchanged. Formally, for the case of asynchronous updates, given a current and next state $q, q' \in Q$, we require that $\bigvee_{g \in G} (q'(g) = f_g(q) \wedge \bigwedge_{g' \in G, g' \neq g} q'(g') = q(g'))$. Our prototype implementation of the synthesis framework and algorithms supports both synchronous and asynchronous semantics.

Dunn et al. (2014) propose and define a set of 18 biologically plausible update function templates, dubbed *regulation conditions* and this notion was refined and explained in detail via illustrative biological case studies in Yordanov et al. (2016). Introducing these function 'templates' aims to reduce the number of Boolean functions that need to be considered (thus simplifying analysis) while still maintaining and emphasising biological and experimental plausibility. Note that these functions preserve the concepts of activators and repressors, thus allowing the integration of known experimental evidence that supports such regulation, but they abstract from the exact numbers and types of activators (repressors) that need to be present (absent) for a gene to turn on (off).

One constraint that is imposed on the regulation conditions is monotonicity, where the increased availability of an expressed activator 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. Similarly, if a gene is not expressed in q' when only some of its repressors are expressed in q, then it cannot be expressed in q' if all of its repressors are expressed in q and there is no change in the presence of activators.

To capture possible uncertainty and partial knowledge of the precise network topology, we allow some interactions to be marked as *possible* (denoted by the set $E^?$), each of which could be included in a synthesised *concrete model* (a model where all interactions are known, i.e., there are no possible interactions). Thus, in terms of network topology, this means a set of $2^{|E^?|}$ concrete models, each of which corresponds to a unique selection of possible interactions. Additionally, a choice of several possible regulation conditions for each gene is taken into consideration, leading to the following definition:

Definition 1 *(Abstract Boolean Network (ABN))*. An abstract Boolean network (ABN) is a four-tuple $\langle G, E, E^?, R \rangle$, where G is a finite set of genes, $E \subseteq G \times G \times \mathbb{B} \times \mathbb{B}$ is a set of definite (positive and negative) and directed interactions between them, $E^? : G \times G \times \mathbb{B} \times \mathbb{B}$ is a set of possible interactions and $R = \{R_g \mid \forall g \in G\}$, where R_g specifies a (non-empty) set of admissible regulation conditions for gene g (Dunn et al., 2014; Yordanov et al., 2016).

An ABN is transformed into a concrete Boolean network by selecting a subset of the possible interactions to be included (or excluded) and assigning a specific regulation condition to each gene; thus, for a concrete Boolean network, $E^? = \emptyset$ and $\forall g \in G \cdot |R_g| = 1$. The semantics of such a concrete model is defined in terms of a transition system $\mathcal{T} = (Q, T)$, where $Q = \mathbb{B}^{|G|}$ is the state space and T is a transition relation defined in terms of the predicate $T : Q \times Q \rightarrow \mathbb{B}$. The semantics of the (synchronous) transition system is then given by

$$\forall q, q' \in Q \quad \cdot \quad T(q, q') \leftrightarrow \bigwedge_{g \in G} q'(g) = R_g(q).$$

A finite *trajectory* of length k is defined as a sequence of states $q_0, q_1, \ldots, q_{k-1}$ where $\bigwedge_{0<i<k} q_i \in Q \wedge T(q_{i-1}, q_i)$. The semantics of an ABN can be understood in terms of the choice of possible interactions and the choice of a regulation condition for each gene, together with the transition system $\mathcal{T}$ resulting in a concrete model. Intuitively, an ABN therefore captures the semantics of all trajectories of all of the concrete Boolean networks it describes.

A set of *experimental observations* that a BN must be able to satisfy are encoded as predicates over system states, which limits the feasible choices of possible interactions and regulation conditions yielding consistent models (networks that are guaranteed to satisfy all observations and network constraints). For instance, an *experiment* in which genes g and g' are observed to be initially active and are inactive at step k is formalised by a constraint requiring the existence of a trajectory $q_0, \ldots, q_{k-1}$ such that $q_0(g) \wedge q_0(g') \wedge \neg q_{k-1}(g) \wedge \neg q_{k-1}(g')$. The approach developed and described in (Dunn et al., 2014; Yordanov et al., 2016) allows GRN synthesis for non-switching networks: given an ABN and a set of experiments, find a choice of interactions and regulation conditions that guarantees that the resulting concrete

BN is consistent with all experimental observations. The synthesis algorithm constructs concrete, consistent models if they exist, or formally proves no solution exists (empty solution set). The approach is implemented in the Reasoning Engine for Interaction Networks (RE:IN, see http://rein.cloudapp.net). RE:IN also supports editing and visualisation of ABNs, experimental observations and solutions, an it enables the user to make predictions based on the set of all consistent concrete models.

An alternative interpretation of ABNs in terms of sets of finite-state machines (FSMs) may be helpful to consider. To this end we can define a BN to be a non-deterministic state machine with state space $Q := \langle g_0, \ldots, g_{n-1}\rangle = \mathbb{B}^n$, a set of initial states $Q_0 \subseteq Q$, a set of final states $F \subseteq Q$, and a transition function δ that is composed of Boolean update functions f_g for each gene. Note that, unusually, we use a *set* of initial states instead of a single initial state $q_0 \in Q$.

Definition 2 *(Boolean network (alternative)).* A Boolean Network of n genes is a non-deterministic finite-state machine with

- finite state-space $Q = \mathbb{B}^n$,
- empty input alphabet,
- set of initial states $Q_i \subseteq Q$,
- set of final states $F \subseteq Q$, and
- transition relation $\delta : Q \times Q = (q, \langle R_1(q), \ldots, R_n(q)\rangle)$, for all $q \in Q$ and a fixed (Boolean) regulation function R_i for each gene g_i.

BNs according to this definition are non-deterministic, as they have more than one initial state. Depending on the context (and especially if asynchronous semantics are required), it may also be helpful to consider non-deterministic transition relations δ, which means that

$$\exists q, q', q'' \in Q \quad \cdot \quad (q, q') \in \delta \wedge (q, q'') \in \delta \wedge q' \neq q'' \quad .$$

If such non-deterministic transitions are permitted, then it is easy to convert the set of initial states Q_i into a unique initial state q^* by adding an initialisation transition to δ, e.g., via

$$\delta' := \delta \cup \{(q^*, q) \quad | \quad q \in Q_i\} \quad .$$

Definition 3 *(Abstract Boolean Network (alternative)).* An *Abstract* Boolean Network (ABN) is a Boolean network with set of genes G with $|G| = n$, state-space Q, initial states Q_0, final states F, regulation function sets $R_i \subseteq \mathcal{P}(\mathbb{B}^n \times \mathbb{B})$, and further equipped with restrictions on interactions $I := I_{def} \cup I_{pos}$ with

$$i_k \in I := \langle g_1, g_2, reg, pos\rangle,$$

where $g_1, g_2 \in G$, $reg \in \{\top, \bot\}$, $pos \in \{\top, \bot\}$, which indicates an interaction between genes g_1, g_2 which is either positive or negative (according to *reg*) and possible or definite (as specified by *pos*).

Each instantiation of an ABN to a particular choice of possible interactions and to a concrete regulation function for each of the genes, uniquely defines δ in the straight-forward way. Conversely, the introduction of possible interactions in ABNs means that every ABN represents a set of concrete FSMs, where each previously possible interaction is either present or not present. Thus, the sets G and I define the abstract network topology which is a set of $2^{|I_{pos}|}$ concrete unique FSMs, in which all interactions and regulation functions are definite (and thus part of δ).

3. Results

In this section we describe a formal framework for specifying switching gene regulatory networks and the associated synthesis problem. We explain how synthesis of switching networks is algorithmically solved using SMT-based methods, and we evaluate the performance of a prototype implementation of our algorithms on synthetic benchmarks. Finally, we apply the method to a simplified network involved in the processes of neuron maturation in the mammalian cortex.

3.1. Switching Gene Regulatory Networks

We propose an extension of the ABN formalism, where transitions between unique cell types, characterised 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 modelled as an ABN $\langle G, E_c, E_c^?, R\rangle$, where the set of definite interactions E_c and possible interactions $E_c^?$ are allowed to differ between cell types. Note that, while the network topology may change between different cell types, we assume that the dependencies as specified by regulation conditions remain consistent across cell types. Modification of these dependencies, if desired, or when sufficient experiments suggest it, is possible as a modification to the ABN, but that modification spans across cell types.

Arbitrary transitions between different cell types are not desired or plausible in many biological systems. For example, two distinct cell types $c, c' \in C$ may represent a progenitor cell c and a differentiated cell c' that is derived from c. While the progenitor could differentiate into a cell type different from c', the reverse does not usually 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 differentiates 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 non-deterministic choice (when $|D_c| > 1$), without explicitly modelling either the mechanism or preconditions on the system state required for such a switch.

Using these concepts, we define SGRNs as follows:

Definition 4 *(Switching Gene Regulatory Network (SGRN)).* A Switching Gene Regulatory Network is a tuple $\langle G, C, D_c, E_c, E_c^?, R\rangle$, where

- G is the finite set of genes,
- C is a finite set of cell types,
- $D_c \subseteq C$ is the set of cell types that cell type c can differentiate into,
- $E_c : G \times G \times \mathbb{B} \times \mathbb{B}$ is the set of definite interactions for cell type c,
- $E_c^? : G \times G \times \mathbb{B} \times \mathbb{B}$ is the set of possible interactions for cell type c, and
- $R = \{R_g \,|\, g \in G\}$, defines admissible regulation conditions for each gene g.

Fig. 1 shows an SGRN with 3 cell types: $C = \{C_0, C_1, C_2\}$, and 6 genes: $G = \{g_0, \ldots, g_5\}$. In this example, a (progenitor) cell type, C_0, may change into cell types C_1 or C_2, by reconfiguring its network, such that $D_{C_0} = \{C_0, C_1, C_2\}$, while C_1 and C_2 cannot switch their network (thus $D_{C_1} = \{C_1\}$ and $D_{C_2} = \{C_2\}$). For each cell type, edges between genes appear in solid (dashed) lines for definite (possible) interactions respectively. Genes appear in dashed nodes to indicate that an R_g permits multiple regulation conditions for gene g.

As in Section 2, the semantics of SGRNs is defined in terms of a transition system $\mathcal{T} = (Q, T)$ with the extended state space $Q = \mathbb{B}^{|G|} \times C$. For a given state $q \in Q$, the cell type of the system at that state is denoted as q_c. Let $\hat{R}_g \in R_g$ be the specific regulation condition selected for each gene $g \in G$ and note that this is independent of c, i.e., it is the same for all cell types. The transition relation $T : Q \times Q \to \mathbb{B}$ for synchronous updates is now adjusted to

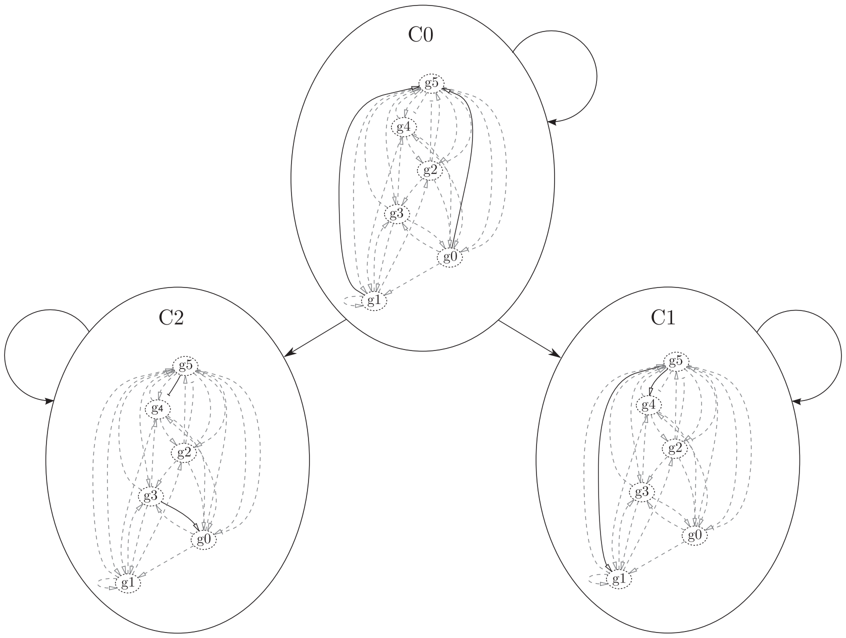


Fig. 1. An SGRN with 3 cell types (C0–C2) and 6 genes (g0–g5), illustrating a typical setting where one cell type (C0) can maintain its identity (self-loops) or differentiate into other cell types by switching its interactions. Edges between genes represent regulatory interactions, with a bar representing repression and an arrow representing activation, and solid (dashed) lines for definite (possible) interactions, respectively.

$$\forall q, q' \in Q \quad \cdot \quad T(q, q') \leftrightarrow \bigwedge_{k \in C} \left[q_c = k \rightarrow \left(q'_c \in D_k \wedge \bigwedge_{g \in G} q'(g) = \hat{R}_g(q) \right) \right]. \qquad (1)$$

Intuitively, Eq. (1) captures the fact that all genes are updated according to the selected regulation conditions $\hat{R}_g$ and the regulators corresponding to the 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 is permitted to transition into. As for ABNs, given an assignment of the possible interactions $E_c^?$ for each cell type c, and a specific regulation condition $\hat{R}_g$ for each gene g, Eq. (1) allows us to define finite trajectories of length k in the resulting concrete SGRN models as a sequence of states $q_0, q_1, \ldots, q_{k-1}$ subject to $\bigwedge_{0<i<k} T(q_{i-1}, q_i)$.

3.2. SGRN model synthesis

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

An abstract SGRN $\langle G, C, D_c, E_c, E_c^?, R\rangle$ is transformed into a concrete SGRN with $E_c^? = \emptyset$ by selecting a specific regulation condition for each gene g, thus fixing $|R_g| = 1$, and by instantiating a concrete set of interactions $\hat{E}_c^? \subseteq E_c^?$ to be included for each cell type c. Let $\pi_q : Q \rightarrow \mathbb{B}$ denote a predicate that recognises a (partial) state q (and no others), i.e., $\pi_q(s)$ means that the gene states and the cell type in s are exactly as in q.

Definition 5 *(Experiment).* An *experiment* $\mathcal{E}$ is a set of constraints that describe experimental observations based on a (partial) trajectory $t = q_0, \ldots, q_{k-1}$ with $\pi_t = \bigwedge_{0 \le i < k} \pi_{q_i}$ of length at most k. Thus, $\mathcal{E} = \{(\pi_t, n)\}$, where $n \le k$.

We write $u \vDash \mathcal{E}$ when trajectory u satisfies experiment $\mathcal{E}$, i.e., when $\pi_t(u)$. More complex expressions may also be constructed as part of an experiment by combining terms (π_t, n) using the Boolean operators $\wedge, \vee, \Rightarrow, \Leftrightarrow, \neg$, *etc*.

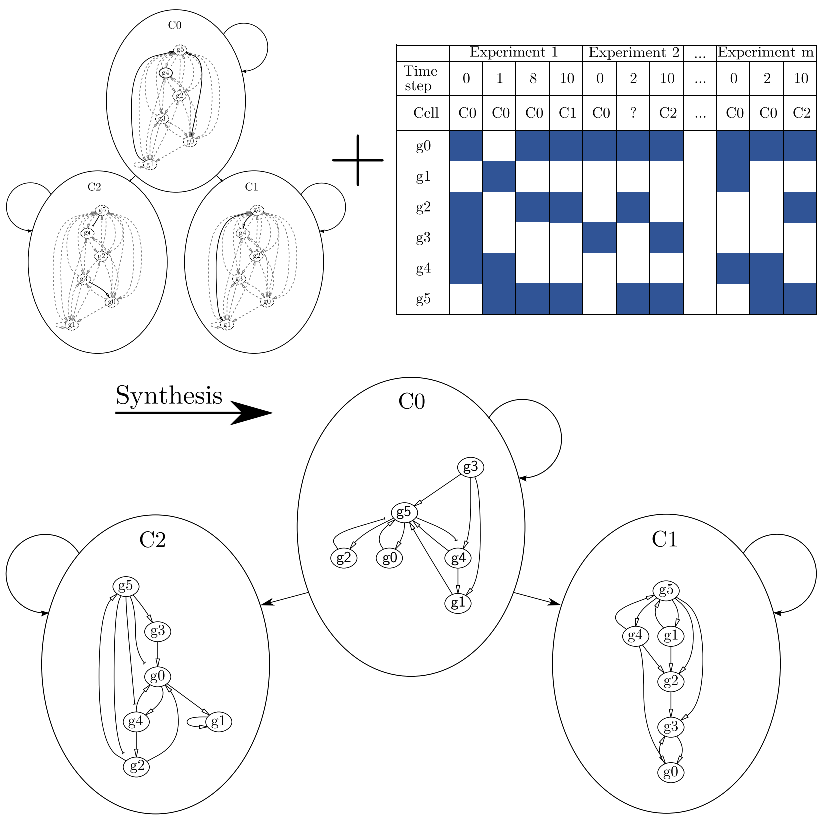
The main problem we consider in this paper is the following (see Fig. 2 for a concrete example):

Problem 1 *(Lineage synthesis).* Given an SGRN $\langle G, C, D_c, E_c, E_c^?, R\rangle$ and a finite set of experiments $\mathcal{E}_0, \ldots, \mathcal{E}_m$, find an assignment $\hat{E}_c^?$ to the possible interactions $E_c^?$ for each cell type c and a single regulation condition $\hat{R}_g$ for each gene g such that there exists a trajectory t_i of the resulting concrete model with $t_i \vDash \mathcal{E}_i$ for $0 \le i \le m$.

Given an SGRN $\langle G, C, D_c, E_c, E_c^?, R\rangle$ we encode the choice of possible interactions $\hat{E}_c^?$ for each cell type c using a unique Boolean choice variable for each interaction, or more conveniently, as a bit-vector using the SMT theory of quantifier-free bit-vectors (QF_BV). Additionally, a single regulation condition $\hat{R}_g$ from the set of admissible conditions R_g must be selected for each gene g. We encode this as the synthesis of a bit-vector (or bounded integer) 'coefficient' on each gene that selects one out of the regulation conditions as proposed by Dunn et al. (2014).

The choice variables for possible 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 3.1, which represents a given concrete BN for each cell type. The state space of T is always finite since both the number of genes G and the number of cell types C are finite. For a given state $q \in Q$, the component of the state space describing the state of all genes is encoded as a single bit-vector variable. In our prototype 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 is exactly 1. This allows us to represent the entire state as individual Boolean variables or as a single bit-vector that combines its components.

For further analysis of the dynamics of SGRNs, we adopt a simple bounded model checking (BMC) approach (Biere et al., 1999) and we 'unroll' the transition relation 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 or non-overlapping, i.e., it



	Experiment 1				Experiment 2			...	Experiment m		
Time step	0	1	8	10	0	2	10	...	0	2	10
Cell	C0	C0	C0	C1	C0	?	C2	...	C0	C0	C2
g0	🟦		🟦	🟦	🟦	🟦	🟦		🟦	🟦	🟦
g1		🟦							🟦		
g2	🟦		🟦	🟦		🟦					🟦
g3	🟦				🟦		🟦				
g4	🟦	🟦							🟦	🟦	
g5		🟦	🟦	🟦		🟦	🟦			🟦	🟦

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.

is possible that one trajectory $t = t_i = t_j$ satisfies the constraints of both experiments $\mathcal{E}_i$ and $\mathcal{E}_j$.

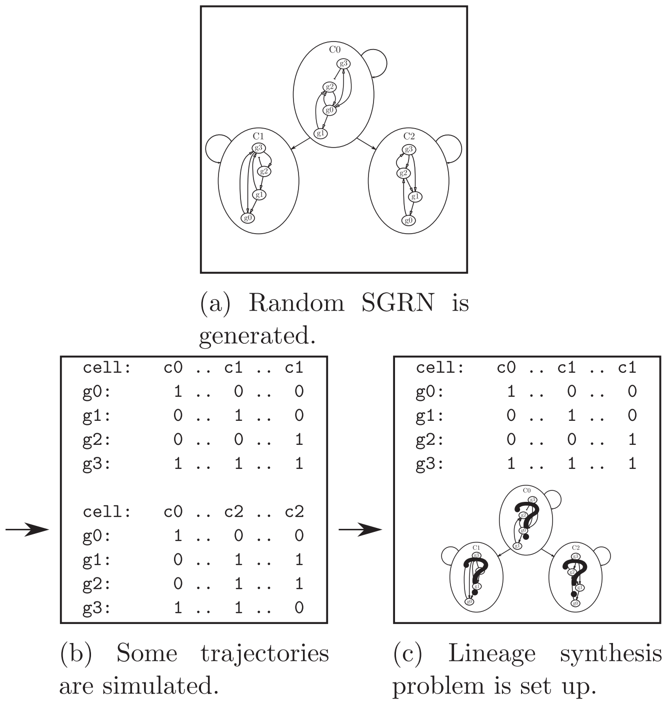
Finally, we employ an SMT solver to determine the satisfiability of all constraints we encode (we use the SMT solver Z3 (de Moura and Bjorner, 2008; Yordanov et al., 2013)). Further, we exploit the fact that SMT solvers usually produce an assignment of all the variables used in the encoding of the problem, which is presented as a certificate of the satisfiability of all constraints. When such an assignment (also referred to as a 'model' in this context) is found, we extract the possible interactions $\hat{E_c^?}$ that were selected for each cell type and the regulation conditions $\hat{R_g}$ 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 solution, to serve as an example demonstrating exactly how the SGRN reproduces the behaviour observed in each experiment. In addition to the sequence of gene expression values at each time point, this information also reveals the cell types along executions of the system, allowing for further investigation of the cellular differentiation processes.

3.3. Synthetic benchmarks

In order to test our approach and systematically evaluate its performance we require benchmarks of lineage synthesis problems for SGRNs with different numbers of genes and cell types. This is achieved by producing synthetic problems, following the main steps summarised in Fig. 3 and described below.

3.3.1. 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



(a) Random SGRN is generated.

```
cell:   c0 .. c1 .. c1
g0:      1 ..  0 ..  0
g1:      0 ..  1 ..  0
g2:      0 ..  0 ..  1
g3:      1 ..  1 ..  1

cell:   c0 .. c2 .. c2
g0:      1 ..  0 ..  0
g1:      0 ..  1 ..  1
g2:      0 ..  1 ..  1
g3:      1 ..  1 ..  0
```

(b) Some trajectories are simulated.

```
cell:   c0 .. c1 .. c1
g0:      1 ..  0 ..  0
g1:      0 ..  1 ..  0
g2:      0 ..  0 ..  1
g3:      1 ..  1 ..  1
```

(c) Lineage synthesis problem is set up.

Fig. 3. The three steps for generating *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).

networks (Albert, 2005), with the exponent of the degree distribution set to 2 (for both in-degree 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 repressed in higher organisms, and must be 'switched on' to be expressed and behave as regulators of their target genes (Phillips, 2008). A regulation condition is randomly assigned to each gene from a set of 16 out of the 18 regulation conditions defined by Dunn et al. (2014).

For a given model with m differentiated 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 $i = 1, \ldots, m$, $j = 1, \ldots, 2n$ and $1 \leq s \leq K$. In order to create the set of $2n$ initial gene state configurations, we randomly select $2n - 2$ integer values in the range $(0, 2^n - 1)$ (exclusive) and add the values 0 and $2^n - 1$, which represent the extreme configurations of the system. System states are encoded as bit-vectors of size $|G| + |C|$, where the kth position in the leading $|G|$ bits represents the state of the kth gene, while the remaining $|C|$ bits encode the cell type.

To construct an instance of the lineage synthesis problem, each model (generated as described above) is used to produce an 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 possible, 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 only at the start and at the end of the experiment (time steps 0 and 10, respectively), thus the exact timing when the progenitor cell switches to a differentiated cell type is not known. In total, this amounts to $2 \cdot m \cdot n$ experiments included in a lineage synthesis problem of m cells and n genes.

3.3.2. *Synthesis evaluation*

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 the synthesis problem 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 denote the definite interactions and $E_c^{?}$ denote the possible interactions of the corresponding SGRN cell type. Let $\hat{E}_c$ denote the synthesised interactions obtained as a result of selecting the possible interactions $\widehat{E_c^{?}}$, where $\hat{E}_c = E_c \cup \widehat{E_c^{?}}$. A *True Positive* is an interaction that is in $\widehat{E_c^{?}}$ and in $E_c^{\star}$. A *True Negative* is an interaction that is not in $\widehat{E_c^{?}}$ and not in $E_c^{\star}$. Note that we evaluate the synthesis of only those interactions that were possible in the SGRN since definite interactions will always be part of the synthesised model. A *False Positive* is an interaction in $\widehat{E_c^{?}}$ that is not in $E_c^{\star}$ and a *False Negative* is an interaction that is not in $\widehat{E_c^{?}}$ and is in $E_c^{\star}$. 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.

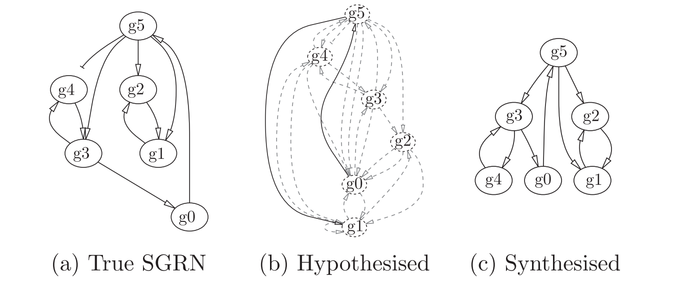


Fig. 4. A true, a hypothesised, and a synthesised progenitor cell type in an SGRN with 6 genes (g0–g5) and 3 cell types. The true cell type (a) was generated with a scale-free topology. The union of all cell types in the SGRN was used to create the hypothesised cell type (possible interactions appear 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 synthesised 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.

The results of our evaluation (Fig. 5a and b) show that our approach successfully recovers 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 heat-maps in Fig. 5a and b, cell types are synthesised 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 1.7 GHz, 4 GB RAM, Windows 8.1 64-bit OS) and with an average running time of 730.25 s (Fig. 5c).

In order to test further whether our approach is useful for alternative biological scenarios, where environmental conditions

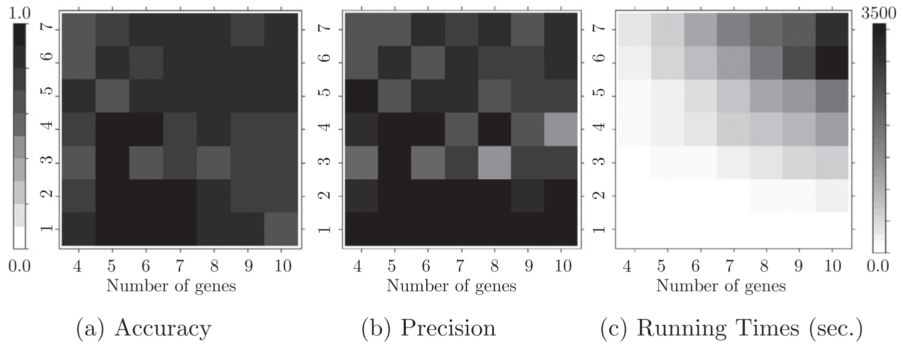


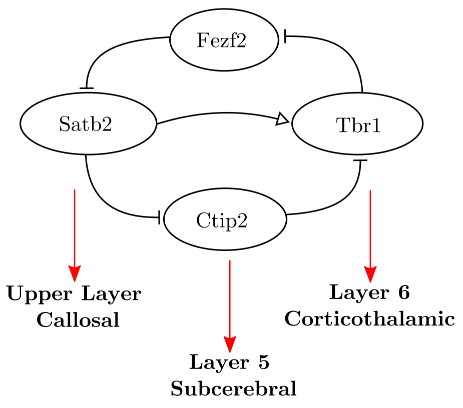
Fig. 5. Heat-maps of experimental results for a benchmark of lineage synthesis problems with 1–7 cells and 4–10 genes. Darker pixels indicate higher accuracy (a) and precision (b), while lighter pixels indicate poorer performance. Running times (c) are indicated on a colour scale from white to black,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

induce changes in the topology of the network, we repeat our evaluation for benchmark problems designed to represent this behaviour. These problems are generated as described above, with the exception that all cell types cannot change their identity (i.e., no progenitor cell is defined). This aims to represent a system in which cells are cultured in different conditions (for example, in the presence or absence of a nutrient or signal). In this setting our approach recovers cell topologies with 0.92 accuracy and 0.94 precision (on average, across 2–7 cell types and 1–10 genes) in an average running time of 158.57 s.

3.4. Illustrative case study

Next, we illustrate how our approach is applied to real biological problems, by considering the process of neuron maturation and cell fate specification in the mammalian cortex. Neurons within each layer of the mammalian cortex acquire specific projection identities, which are controlled by key regulatory genes. Here we focus on four key genes involved in this process: *Fezf2*, *Satb2*, *Ctip2* and *Tbr1*. Srinivasan et al. (2012) study a static network specifying projection fates by generating double mutants of *Fezf2*, *Satb2*, *Ctip2* – phenotypes in which these genes are knocked out – and by integrating previous experimental observations. They further suggest a network model summarising the mechanistic understanding, which is illustrated in Fig. 6.

Using our SGRN approach, we re-frame the problem of deriving unique networks for the different cell types under consideration. As illustrated in Fig. 7(a), we define four possible cell states, and the allowable cell switches: a common progenitor cell, P, can switch to either an upper layer (UL) cell, a Layer 5 (L5) or Layer 6 (L6) cell, which correspond to the different projection identities. Next,



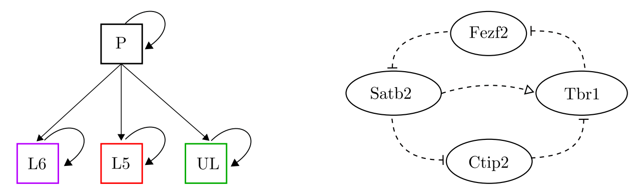
	Callosal		
	Initial	UL	UL
Time step	0	10	11*
Fezf2	▨	⬜	⬜
Satb2	▨	🟦	🟦
Ctip2	▨	▨	▨
Tbr1	▨	▨	▨

	Subcerebral		
	Initial	L5	L5
Time step	0	10	11*
Fezf2	▨	🟦	🟦
Satb2	▨	⬜	⬜
Ctip2	▨	🟦	🟦
Tbr1	▨	⬜	⬜

	Corticothalamic		
	Initial	L6	L6
Time step	0	10	11*
Fezf2	▨	▨	▨
Satb2	▨	⬜	⬜
Ctip2	▨	⬜	⬜
Tbr1	▨	🟦	🟦

🟦 Activated ⬜ Repressed ▨ Not constrained

Fig. 6. A network of genetic interactions specifying neuron projection fates in the wild type cortex, from (Srinivasan et al., 2012). Arrows denote positive interactions, while bars denote inhibitory interactions. The expression of specific genes marks cell fates, as indicated by the coloured arrows and tables of gene expression constraints. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)



(a) A hypothetical progenitor cell P can switch into an upper layer neuron (UL), a layer 5 neuron (L5) or a layer 6 neuron (L6).

(b) Dashed lines in the switching network indicate that the interactions are possible ones and can thus be selected to be definite in each cell type.

Fig. 7. A switching network of genetic interactions specifying neuron projection fates in the wild type cortex.

using our ABN synthesis approach, we attach precise semantics to networks such as the one described in Fig. 6 by selecting which of the interactions are instantiated in cells of the different layers, and by choosing a regulation condition that determines the expression of a gene based on activity of its regulators. By encoding the interactions identified by Srinivasan et al. (2012) as possible in each layer, as illustrated in Fig. 7(b), we successfully synthesised models that generate the corresponding callosal, sub-cerebral and corticothalamic fates using the synthesis algorithm described earlier.

Crucially, if we instead require all interactions to be present in all cell types, which amounts to disallowing switching, no models can be identified that are consistent with (some) experimental observations. An alternative mechanism might allow differences between the layers to emerge as a result of non-determinism and such behaviour can be modelled by allowing asynchronous updates in the Boolean network. While this allows more flexibility in model behaviour, interestingly, for this system no consistent models are found even when considering asynchronous updates. For this specific model and assumptions, this illustrates how switching provides cells with more flexibility to regulate gene expression patterns and acquire specific fates that may be harder to a achieve with either a 'rigid' static network topology, or only through non-determinism, which would make it harder to discover the actual laws underlying this process.

4. Discussion

The potential for network reconfiguration in cells is suggestive of self-modifying biological programs. 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 are hard to write and analyse. Consequently, modern program analysis tools have no, or very limited, means of reasoning about such programs. It does appear, however, that supporting the concept of switching networks in a biological context may provide a useful abstraction for capturing the bio-molecular processes at work as cells change type.

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 (von Neumann, 1945), 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., Bashe et al., 1986) and applied in some specific domains (e.g., in computer graphics Keppel et al., 1991), but did not become a mainstream paradigm. One of the reasons for the limited use of self-modifying program may be that they are more complex to understand and maintain, and the advantages that they offer in terms of program size and performance are less significant in modern computer architectures.

Boolean networks have been suggested as a useful abstraction for the study of cell differentiation (Kauffman, 1969; Thomas and Kaufman, 2001). In this context the concept of *switching* was mainly used to describe changes in the state of the nodes (genes) rather than a reconfiguration of the network topology itself. The change in a gene's state could be the result of executing the GRN and by including additional effects such as the spatio-temporal dynamics of the neighbouring cellular (tissue) environment (e.g. see Doursat, 2008; Giavitto et al., 2012). Recently there has been growing interest in formal reasoning and synthesis approaches for logical models in biology, e.g. Guziolowski et al., 2013; Paoletti et al., 2014; Fisher et al., 2015. However, little attention has been given to the rewiring of the network as a mechanism to achieve differentiation, or changes in the cellular function.

Petri-nets and their extensions have also been used in modelling of GRNs (see for instance (Chaouiya, 2007; Heiner et al., 2008)). In particular, the extension of self-modifying nets (Valk, 1978) enables one to describe the reconfiguration of Petri-nets. 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. Therefore a self modifying net can be viewed as a Petri-net that is able to modify its own firing rules, in contrast to an ordinary Petri-net that employs fixed firing rules. Self-modifying nets and further extensions have been used in modelling of metabolic networks (Hofestädt and Thelen, 1998), where self-modification permits the representation of concentrations and kinetic effects. It is known that self-modifying Petri-nets are more expressive than conventional Petri-nets, making the reachability problem undecidable (Valk, 1978). In contrast, we define a framework in which the basic dynamic properties of the system remain decidable when considering bounded trajectories.

Bayesian networks have been extensively applied to the problem of inference of gene regulatory networks from time series data (Friedman et al., 2000). 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 reason about sets of solutions symbolically. However, in relation to our focus here, more recently there has been research on generalising Bayesian network inference to the case of time-varying networks (see e.g., (Rao et al., 2007; Song et al., 2009; Ahmed and Xing, 2009; Parikh et al., 2011; Dondelinger et al., 2013; Khan et al., 2014)).

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 (Maraninchi and Rémond, 1998). In our work, however, such a decomposition is not fully known *a priori*, and our focus is on synthesising 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. Ramadge and Wonham, 1987) – 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. This is in contrast to synthesising a controller that, when coupled with the system, restricts its behaviour to some desirable subset.

The neuronal maturation example we discuss illustrates a biological system and scenario in which it is necessary to investigate the possibility that switching is used by cells to regulate fate specification. A detailed and realistic study of the question is beyond the scope of this paper, and is a topic we are now actively pursuing. It will require an investigation of whether additional genes, beyond those studied in (Srinivasan et al., 2012), directly regulate fate specification, and remove the need for switching. Our preliminary results from a more detailed SGRN model that includes *Sox5* in addition to the current network components (*Fezf2*, *Satb2*, *Ctip2* and *Tbr1*) support the idea that switching may play a role in the neuron fate specification process.

An additional route to fate specification is for a cell to respond to external signals from the cell environment. In this scenario, cells arising at different developmental times, and in different regions, may be exposed to varying input signals that determine fate specification accordingly. As shown in (Dunn et al., 2014; Yordanov et al., 2016), signals can naturally be incorporated within the synthesis framework. Ultimately, we are interested in synthesising predictive models, and thus for a switching network model to be useful there is a need to demonstrate that model predictions can be validated experimentally, and are more accurate than those derived from static network models. Such a goal will require us to represent a rich set of observations encompassing the state-of-the art in the experimental knowledge of projection fate specification in the mammalian cortex.

To summarise, 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. Our method provides a computational framework to study how cells differentiate into specific cell types during development, in particular making explicit the role of switching and reconfiguration of gene networks governing cellular decision making. A long-term research goal is to gain a mechanistic understanding of how biological programs operate and experimentally to investigate the existence and design principles of (switching) biological programs, such as used to orchestrate neuron development in the mammalian cortex.

Acknowledgements

Yoli Shavit is supported by the Cambridge International Scholarship Scheme (CISS). Most of the research underlying the results in this paper was carried out during her internship at Microsoft Research.

References

Ahmed, A., Xing, E., 2009. Recovering time-varying networks of dependencies in social and biological studies. Proc. Natl. Acad. Sci. 106 (29), 11878–11883.
Albert, R., 2005. Scale-free networks in cell biology. J. Cell Sci. 118, 4947–4957.
Bashe, C., Johnson, L., Palmer, J., Pugh, E., 1986. IBM's Early Computers. MIT Press.
Biere, A., Cimatti, A., Clarke, E.M., Zhu, Y.,1999. Symbolic model checking without BDDs. In: Proc. of the 5th Intl. Conf. Tools and Algorithms for Construction and Analysis of Systems (TACAS'99), vol. 1579 of Lecture Notes in Computer Science. Springer, pp. 193–207.
Chaouiya, C., 2007. Petri net modelling of biological networks. Brief. Bioinform. 8 (4), 210–219.
Davidson, E.H., Rast, J.P., Oliveri, P., Ransick, A., Calestani, C., Yuh, C.-H., Minokawa, T., Amore, G., Hinman, V., Arenas-Mena, C., Otim, O., Brown, C.T., Livi, C.B., Lee, P.Y., Revilla, R., Rust, A.G., Pan, Z.j., Schilstra, M.J., Clarke, P.J.C., Arnone, M.I., Rowen, L., Cameron, R.A., McClay, D.R., Hood, L., Bolouri, H., 2002. A genomic regulatory network for development. Science 295 (5560), 1669–1678.
de Moura, L., Bjørner, N.,2008. Z3: an efficient SMT solver. In: Proc. of the 14th Intl. Conf. on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'08), vol. 4963 of Lecture Notes in Computer Science. Springer, pp. 337–340.
Dondelinger, F., Lèbre, S., Husmeier, D., 2013. Non-homogeneous dynamic Bayesian networks with Bayesian regularization for inferring gene regulatory networks with gradually time-varying structure. Mach. Learn. 90 (2), 191–230.
Doursat, R.,2008. The growing canvas of biological development: multiscale pattern generation on an expanding lattice of gene regulatory nets. In: Unifying Themes in Complex Systems: Proc. of the 6th Intl. Conf. on Complex Systems. Springer, pp. 205–210.
Dunn, S.-J., Martello, G., Yordanov, B., Emmott, S., Smith, A., 2014. Defining an essential transcription factor program for naïve pluripotency. Science 344 (6188), 1156–1160.
Fisher, J., Köksal, A.S., Piterman, N., Woodhouse, S.,2015. Synthesising executable gene regulatory networks from single-cell gene expression data. In: Proc. of

the 27th Intl. Conf. on Computer Aided Verification (CAV'15), vol. 9206 of Lecture Notes in Computer Science. Springer, pp. 544–560.

Friedman, N., Linial, M., Nachman, I., Pe'er, D., 2000. Using Bayesian networks to analyze expression data. J. Comput. Biol. 3 (7), 601–620.

Giavitto, J., Klaudel, H., Pommereau, F., 2012. Integrated regulatory networks (IRNs): spatially organized biochemical modules. Theor. Comput. Sci. 431, 219–234.

Guziolowski, C., Videla, S., Eduati, F., Thiele, S., Cokelaer, T., Siegel, A., Saez-Rodriguez, J., 2013. Exhaustively characterizing feasible logic models of a signaling network using answer set programming. Bioinformatics 29 (18), 2320–2326.

Heiner, M., Gilbert, D., Donaldson, R.,2008. Petri nets for systems and synthetic biology. In: Formal Methods for Computational Systems Biology: Proc. of the 8th Intl. School on Formal Methods for the Design of Computer, Communication, and Software Systems (SFM'08). Springer, pp. 215–264.

Hofestädt, R., Thelen, S., 1998. Quantitative modeling of biochemical networks. In Silico Biol. – J. Biol. Syst. Model Simul. 1 (1), 39–53.

Kauffman, S., 1969. Metabolic stability and epigenesis in randomly constructed genetic nets. J. Theor. Biol. 22 (3), 437–467.

Keppel, D., Eggers, S., Henry, R., 1991. A Case for Runtime Code Generation. Technical Report 91-11-04, Department of Computer Science and Engineering, University of Washington.

Khan, J., Bouaynaya, N., Fathallah-Shaykh, H., 2014. Tracking of time-varying genomic regulatory networks with a LASSO-Kalman smoother. EURASIP – J. Bioinform. Syst. Biol. 2014 (3).

Le Novère, N., 2015. Quantitative and logic modelling of molecular and gene networks. Nat. Rev. Genet. 16 (3), 146–158.

Maraninchi, F., Rémond, Y.,1998. Mode-automata: About modes and states for reactive systems. In: Proc. of the 7th European Symp. on Programming (ESOP'98), vol. 1381 of Lecture Notes in Computer Science. Springer, pp. 185–199.

Paoletti, N., Yordanov, B., Hamadi, Y., Wintersteiger, C.M., Kugler, H.,2014. Analyzing and synthesizing genomic logic functions. In: Proc. of the 26th Intl. Conf. on Computer Aided Verification (CAV'14), vol. 8559 of Lecture Notes in Computer Science. Springer, pp. 343–357.

Parikh, A., Wu, W., Curtis, R., Xing, E., 2011. TREEGL: reverse engineering tree-evolving gene networks underlying developing biological lineages. Bioinformatics 27 (13), i196–i204.

Phillips, T., 2008. Regulation of transcription and gene expression in eukaryotes. Nat. Educ. 1 (1), 199.

Ramadge, P.J., Wonham, W.M., 1987. Supervisory control of a class of discrete event processes. SIAM – J. Control Optim. 25 (1), 206–230.

Rao, A., Hero, A., States, D., Engel, J., 2007. Inferring time-varying network topologies from gene expression data. EURASIP J. Bioinform. Syst. Biol. 2007 (1).

Song, L., Kolar, M., Xing, E.P., 2009. Time-varying dynamic Bayesian networks. In: Advances in Neural Information Processing Systems: Proc. of the Conf. on Neural Information Processing Systems (NIPS'09), vol. 22, pp. 1732–1740.

Srinivasan, K., Leone, D., Bateson, R., Dobreva, G., Kohwi, Y., Kohwi-Shigematsu, T., Grosschedl, R., McConnell, S., 2012. A network of genetic repression and derepression specifies projection fates in the developing neocortex. Proc. Natl. Acad. Sci. U.S.A. 109 (47), 19071–19078.

Stergachis, A.B., Neph, S., Reynolds, A., Humbert, R., Miller, B., Paige, S.L., Vernot, B., Cheng, J.B., Thurman, R.E., Sandstrom, R., Haugen, E., Heimfeld, S., Murry, C.E., Akey, J.M., Stamatoyannopoulos, J.A., 2013. Developmental fate and cellular maturity encoded in human regulatory DNA landscapes. Cell 154 (4), 888–903.

Thomas, R., Kaufman, M., 2001. Multistationarity, the basis of cell differentiation and memory. II. Logical analysis of regulatory networks in terms of feedback circuits. Chaos 11 (1), 180–195.

Valk, R.,1978. Self-modifying nets, a natural extension of Petri nets. In: Proc. of the 5th Colloquium on Automata, Languages and Programming, vol. 62 of Lecture Notes in Computer Science. Springer, pp. 464–476.

von Neumann, J., 1945. First Draft of a Report on the EDVAC. Tech. Rep. Contract No. W670ORD4926, Moore School of Elec. Eng., Univ. of Pennsylvania.

Yordanov, B., Dunn, S.-J., Kugler, H., Smith, A., Martello, G., Emmott, S., 2016. A method to identify and analyze biological programs through automated reasoning. npj Syst. Biol. Appl. (in press).

Yordanov, B., Wintersteiger, C., Hamadi, Y., Kugler, H., 2013. Z34Bio: an SMT-based framework for analyzing biological computation. In: Proc. of the 11th Intl. Workshop on Satisfiability Modulo Theories (SMT'13).

Yosef, N., Shalek, A.K., Gaublomme, J.T., Jin, H., Lee, Y., Awasthi, A., Wu, C., Karwacz, K., Xiao, S., Jorgolli, M., Gennert, D., Satija, R., Shakya, A., Lu, D.Y., Trombetta, J.J., Pillai, M.R., Ratcliffe, P.J., Coleman, M.L., Bix, M., Tantin, D., Park, H., Kuchroo, V.K., Regev, A., 2013. Dynamic regulatory network controlling th17 cell differentiation. Nature 496 (7446), 461–468.