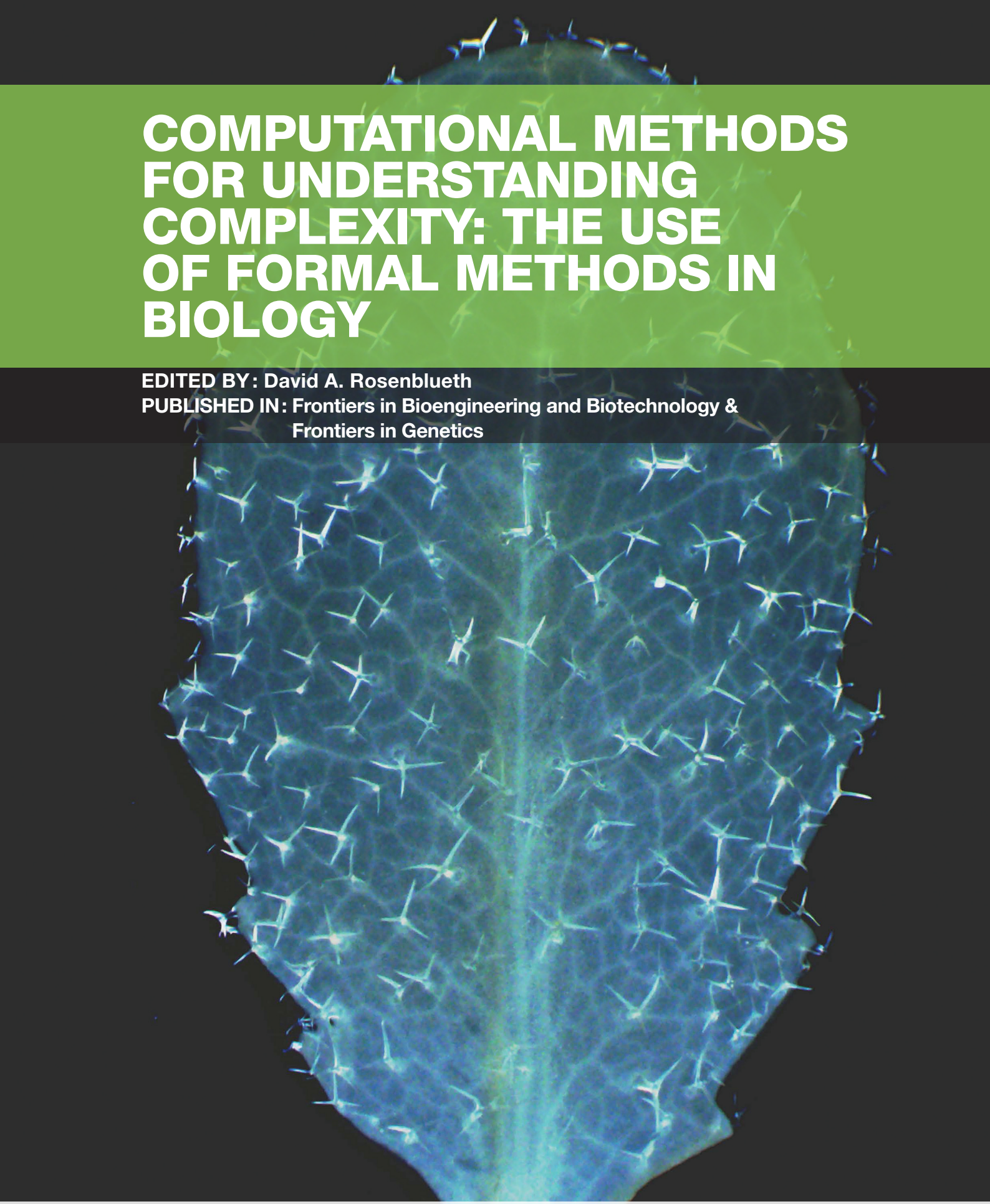
COMPUTATIONAL METHODS FOR UNDERSTANDING COMPLEXITY: THE USE OF FORMAL METHODS IN BIOLOGY

EDITED BY: David A. Rosenblueth
PUBLISHED IN: Frontiers in Bioengineering and Biotechnology &
Frontiers in Genetics



frontiers Research Topics

frontiers

ISSN 1664-8714
ISBN 978-2-88945-042-8
DOI 10.3389/978-2-88945-042-8

About Frontiers

Frontiers is more than just an open-access publisher of scholarly articles: it is a pioneering approach to the world of academia, radically improving the way scholarly research is managed. The grand vision of Frontiers is a world where all people have an equal opportunity to seek, share and generate knowledge. Frontiers provides immediate and permanent online open access to all its publications, but this alone is not enough to realize our grand goals.

Frontiers Journal Series

The Frontiers Journal Series is a multi-tier and interdisciplinary set of open-access, online journals, promising a paradigm shift from the current review, selection and dissemination processes in academic publishing. All Frontiers journals are driven by researchers for researchers; therefore, they constitute a service to the scholarly community. At the same time, the Frontiers Journal Series operates on a revolutionary invention, the tiered publishing system, initially addressing specific communities of scholars, and gradually climbing up to broader public understanding, thus serving the interests of the lay society, too.

Dedication to quality

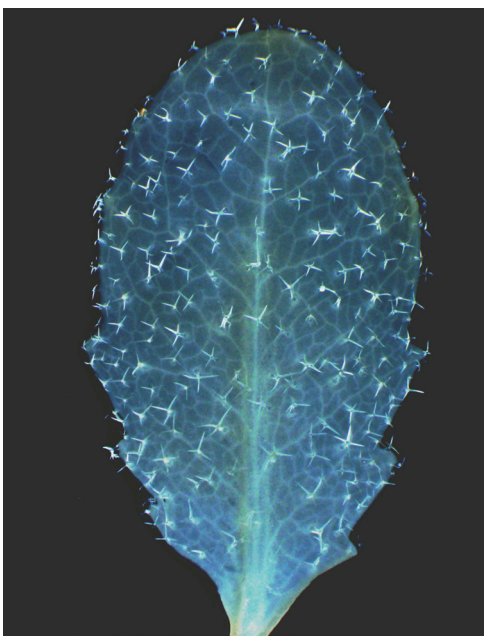
Each Frontiers article is a landmark of the highest quality, thanks to genuinely collaborative interactions between authors and review editors, who include some of the world's best academicians. Research must be certified by peers before entering a stream of knowledge that may eventually reach the public - and shape society; therefore, Frontiers only applies the most rigorous and unbiased reviews.
Frontiers revolutionizes research publishing by freely delivering the most outstanding research, evaluated with no bias from both the academic and social point of view.
By applying the most advanced information technologies, Frontiers is catapulting scholarly publishing into a new generation.

What are Frontiers Research Topics?

Frontiers Research Topics are very popular trademarks of the Frontiers Journals Series: they are collections of at least ten articles, all centered on a particular subject. With their unique mix of varied contributions from Original Research to Review Articles, Frontiers Research Topics unify the most influential researchers, the latest key findings and historical advances in a hot research area! Find out more on how to host your own Frontiers Research Topic or contribute to one as an author by contacting the Frontiers Editorial Office: **researchtopics@frontiersin.org**

COMPUTATIONAL METHODS FOR UNDERSTANDING COMPLEXITY: THE USE OF FORMAL METHODS IN BIOLOGY

Topic Editor:
David A. Rosenblueth, Universidad Nacional Autónoma de México, Mexico



Leaf of Arabidopsis thaliana with enhanced trichomes.

Photograph: Andrea Domínguez Román. Edition: Stalin Muñoz

The complexity of living organisms surpasses our unaided habilities of analysis. Hence, computational and mathematical methods are necessary for increasing our understanding of biological systems.

At the same time, there has been a phenomenal recent progress allowing the application of novel formal methods to new domains. This progress has spurred a conspicuous optimism in computational biology, promoting, in turn, a rapid increase in collaboration between specialists of biology with specialists of computer science.

Through sheer complexity, however, many important biological problems are at present intractable, and it is not clear whether we will ever be able to solve such problems. We are in the process of learning what kind of model and what kind of analysis and synthesis techniques to use for a particular problem. Some existing formalisms have been readily used in biological problems, others have been adapted to biological needs, and still others have been especially developed for biological systems.

This Research Topic has examples of cases (1) employing existing methods, (2) adapting methods to biology, and (3) developing new methods. We can also see discrete and Boolean

models, and the use of both simulators and model checkers. Synthesis is exemplified by manual and by machine-learning methods. We hope that the articles collected in this Research Topic will stimulate new research.

Citation: Rosenblueth, D. A., ed. (2016). Computational Methods for Understanding Complexity: The Use of Formal Methods in Biology. Lausanne: Frontiers Media. doi: 10.3389/978-2-88945-042-8

Table of Contents

frontiers in Bioengineering and Biotechnology

EDITORIAL
published: 22 August 2016
doi: 10.3389/fbioe.2016.00068

Editorial: Computational Methods for Understanding Complexity: The Use of Formal Methods in Biology

*David A. Rosenblueth**

Instituto de Investigaciones en Matemáticas Aplicadas y en Sistemas, Universidad Nacional Autónoma de México, México D.F., Mexico

Keywords: Boolean networks, gene regulatory networks, biochemical networks, attractors of Boolean networks, logic programing, model checking, answer set programing, synthesis of biochemical models

The Editorial on the Research Topic

Computational Methods for Understanding Complexity: The Use of Formal Methods in Biology

The functional properties of living organisms have a complexity exceeding the human capacity for analysis. A basic conviction in computational biology is that it should be possible to develop computational tools allowing us to considerably increase our understanding of such functional properties.

Understanding a fragment of reality is closely related to having a model of such a fragment. Hence, model construction is a high priority on the agenda of computational biology. Once available, a model can then be analyzed with different techniques. These two processes, however, are often intertwined, as analysis can guide the construction of a model.

Among the *models* for biochemical and gene networks (de Jong, 2002; Fages and Soliman, 2008), ordinary differential equations are of prime importance. Stochastic models based on Gillespie's method (identified with continuous-time Markov chains) represent perhaps a most concrete model. Discrete models (e.g., Petri nets) are prominent, as abstractions from stochastic techniques, where both the concentrations and time have been discretized. Finally, Boolean formalisms are abstractions of discrete models. Boolean models were initially studied with propositional logic (i.e., Boolean logic). Later, however, close connections with more expressive logics have been established, such as those underlying Logic Programing (Kowalski, 2014) and Model Checking (Clarke et al., 1999).

Analysis techniques vary in the direction of treatment of time. Simulators normally deal with time in a forward manner by reproducing in the model a single behavior among all possible behaviors from an initial state. Model checkers, by contrast, often proceed backwards by analyzing, in reverse, all possible behaviors ending in a given set of final states.

Model-construction techniques, in turn, range from those completely performed by a human being to those entirely mechanized.

Understanding a living system through a model could be a goal per se. The model of a system, nevertheless, can also be used for predicting or even controlling its behavior. Moreover, a model can be instrumental in the synthesis of a system itself.

The aim of the present research topic is to explore the application of formal methods for understanding biological systems. This research topic comprises nine articles. Five of them belong to the category Original Research, two are Reviews, one a Technology Report, and the last one an Opinion Article.

"Model Checking to Assess T-Helper Cell Plasticity," by Abou-Jaoudé et al., is based on the discrete, asynchronous formalism developed by Thomas and D'Ari (1990). This work uses GINsim along with Model Checking for Action-Restricted Computation-Tree Logic (ARCTL). ARCTL is a generalization of ordinary Computation-Tree Logic (CTL) incorporating actions. This article extends a previously published work so as to cover several novel Th subtypes, and highlights the plasticity of Th cells depending on their microenvironment. The model has 101 variables (most of which, but not all, are Boolean) and 221 regulatory interactions.

OPEN ACCESS

Edited and Reviewed by:
Richard D. Emes, University of Nottingham, UK

***Correspondence:**
David A. Rosenblueth
drosenbl@unam.mx

Specialty section:
This article was submitted to Bioinformatics and Computational Biology, a section of the journal Frontiers in Bioengineering and Biotechnology

Received: *20 July 2016*
Accepted: *04 August 2016*
Published: *22 August 2016*

Citation:
Rosenblueth DA (2016) Editorial: Computational Methods for Understanding Complexity: The Use of Formal Methods in Biology. Front. Bioeng. Biotechnol. 4:68. doi: 10.3389/fbioe.2016.00068

"Approximating Attractors of Boolean Networks by Iterative CTL Model Checking," by Klarner and Heike, is a contribution to the study of asynchronous Boolean networks. This article advocates a method for approximating asynchronous attractors by "minimal trap spaces" using Answer Set Programing (Eiter et al., 2009), a declarative problem-solving paradigm stemming from Logic Programing. Minimal trap spaces can be computed efficiently even for networks with hundreds of variables. To decide whether each minimal trap space contains exactly one attractor, and whether there are attractors outside them, the authors use CTL Model Checking.

"Systems Perturbation Analysis of a Large Scale Signal Transduction Model Reveals Potentially Influential Candidates for Cancer Therapeutics," by Puniya et al., studies perturbations on a signal-transduction Boolean model having 132 variables and 557 interactions. Through simulations using the platform Cell Collective, this work suggests potential therapeutic targets.

"Learning Delayed Influences of Biological Systems," by Ribeiro et al., is based on an extension of ordinary Boolean models with *delays* and employs Inductive Logic Programing to infer such models. Experimental data are a set of traces of observations, used in a bottom-up method that generates hypotheses. This process is illustrated in the yeast cell cycle system.

"Designing experiments to discriminate families of logic models," by Videla et al., studies a method of synthesis of Boolean models employing Answer Set Programing. Through both prior knowledge and multiple-perturbation experiments thousands of logic models are retrieved. This is due to the incomplete and redundant nature of biological data. This work designs optimal experiments finding more specific logic models. The space of possible experiments is iteratively explored imposing constraints to minimize the number of input–output model behaviors at each step. The proposed method is applied to signaling pathways in human liver cells and phosphoproteomic data.

"Towards Synthesizing Executable Models in Biology," by Fisher et al., discusses how Executable Biology can be aided by automatic synthesis of models. They exemplify this approach with several discrete models including a model of the *C. elegans* vulval precursor cells (VPC) system. The technique relies on the translation of the requirements from the model to logical constraints, which are supplied to a solver.

"A Survey about Methods Dedicated to Epistasis Detection," by Niel et al., classifies epistasis-detection methods into those performing exhaustive search and those effecting non-exhaustive search. On the one hand, the exhaustive-search methods may or may not use filtering to reduce the size of the search space. On the other hand, the non-exhaustive-search methods use combinatorial optimization or machine-learning techniques.

"Systems Biology of Cancer: A Challenging Expedition for Clinical and Quantitative Biologists," by Korsunsky et al., relates models with computer tools for computational biology. The models covered include Bayesian networks, Boolean networks, ordinary differential equations, and cellular automata. The computer tools encompass Model Checking and Sensitivity Analysis. Pancreatic cancer is used as an illustration.

"Normal vs. Malignant Hematopoiesis: The Complexity of Acute Leukemia through Systems Biology," by Enciso et al., first observes that the relapse of acute leukemia could be explained as a selection eliminating highly proliferative cells due to chemotherapy, thus favoring slow-cycling cells. Hence, these authors advocate modeling both several hematopoietic populations and the interactions with non-hematopoietic neighboring cells.

We are in the course of learning what kind of model and what kind of analysis and model-building techniques to use for each particular problem. This research topic is a contribution to such exploration. There are articles employing well-established methods, adapting techniques to biology, and developing new approaches. We can also find discrete and Boolean models, and the use of both simulators and model checkers. At the same time, synthesis is exemplified both by manual and machine-learning methods. We believe that the articles in this research topic will stimulate new research.

AUTHOR CONTRIBUTIONS

DAR conceived the idea for this research topic, served as editor for the manuscripts, and wrote the editorial.

ACKNOWLEDGMENTS

The author would like to thank the reviewers, who generously shared their time and expertise. We are also grateful to Wassim Abou-Jaoudé, Carito Guziolowski, Tomáš Helikar, Hannes Klarner, Luis Mendoza, Bud Mishra, Clément Niel, Nir Piterman, Tony Ribeiro, and Denis Thieffry, for their insightful comments. Finally, we happily acknowledge support from PASPA-DGAPA-UNAM and Conacyt grants 221341 and 261225.

Conflict of Interest Statement: The author declares that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

REFERENCES

Clarke, E. M., Grumberg, O., and Peled, D. A. (1999). *Model Checking*. Cambridge, MA: MIT Press.

de Jong, H. (2002). Modeling and simulation of genetic regulatory systems: a literature review. *J. Comput. Biol.* 9, 67–103. doi:10.1089/10665270252833208

Eiter, T., Ianni, G., and Krennwallner, T. (2009). *Answer Set Programming: A Primer*. Springer.

Fages, F., and Soliman, S. (2008). Abstract interpretation and types for systems biology. *Theor. Comp. Sci.* 403, 52–70. doi:10.1016/j.tcs.2008.04.024

Kowalski, R. (2014). "Logic programming," in *Handbook of the History of Logic, Volume 9, Computational Logic (Jörg Siekman, Editor)*, eds D. M. Gabbay and J. Woods (Amsterdam: Elsevier), 523–569.

Thomas, R., and D'Ari, R. (1990). *Biological Feedback*. Boca Raton: CRC Press.

frontiers in
BIOENGINEERING AND BIOTECHNOLOGY

ORIGINAL RESEARCH ARTICLE
published: 28 January 2015
doi: 10.3389/fbioe.2014.00086

Model checking to assess T-helper cell plasticity

Wassim Abou-Jaoudé[1,2,3,4†], Pedro T. Monteiro[5,6†], Aurélien Naldi[7], Maximilien Grandclaudon[8,9], Vassili Soumelis[8,9], Claudine Chaouiya[6] and Denis Thieffry[1,2,3]*

[1] Institut de Biologie de l'Ecole Normale Supérieure, Paris, France
[2] UMR CNRS 8197, Paris, France
[3] INSERM U1024, Paris, France
[4] Laboratoire d'Informatique de l'Ecole Normale Supérieure, Paris, France
[5] INESC-ID, Lisboa, Portugal
[6] Instituto Gulbenkian de Ciência, Oeiras, Portugal
[7] Centre Intégratif de Génomique, Université de Lausanne, Lausanne, Switzerland
[8] Laboratoire d'Immunologie Clinique, Institut Curie, Paris, France
[9] INSERM U932, Paris, France

Edited by:
David A. Rosenblueth, Universidad Nacional Autonoma de Mexico, Mexico

Reviewed by:
Monika Heiner, Brandenburg University of Technology Cottbus-Senftenberg, Germany
Ion Petre, Åbo Akademi University, Finland

***Correspondence:**
Denis Thieffry, Institut de Biologie de l'Ecole Normale Supérieure (IBENS), 46 Rue d'Ulm, Paris 75005, France
e-mail: thieffry@ens.fr

†*Wassim Abou-Jaoudé and Pedro T. Monteiro have contributed equally to this work.*

Computational modeling constitutes a crucial step toward the functional understanding of complex cellular networks. In particular, logical modeling has proven suitable for the dynamical analysis of large signaling and transcriptional regulatory networks. In this context, signaling input components are generally meant to convey external stimuli, or environmental cues. In response to such external signals, cells acquire specific gene expression patterns modeled in terms of attractors (e.g., stable states). The capacity for cells to alter or reprogram their differentiated states upon changes in environmental conditions is referred to as cell plasticity. In this article, we present a multivalued logical framework along with computational methods recently developed to efficiently analyze large models. We mainly focus on a symbolic model checking approach to investigate switches between attractors subsequent to changes of input conditions. As a case study, we consider the cellular network regulating the differentiation of T-helper (Th) cells, which orchestrate many physiological and pathological immune responses. To account for novel cellular subtypes, we present an extended version of a published model of Th cell differentiation. We then use symbolic model checking to analyze reachability properties between Th subtypes upon changes of environmental cues. This allows for the construction of a synthetic view of Th cell plasticity in terms of a graph connecting subtypes with arcs labeled by input conditions. Finally, we explore novel strategies enabling specific Th cell polarizing or reprograming events.

Keywords: logical modeling, signaling networks, T-helper lymphocyte, cell differentiation, cell plasticity, model checking

1. INTRODUCTION

Cellular signaling pathways and regulatory circuits are progressively deciphered, with a recent acceleration allowed by the development of powerful high-throughput experimental approaches. Computational modeling constitutes a crucial step toward the functional understanding of the resulting intertwined networks. Different formalisms have been commonly used to model complex biological networks, with different levels of abstraction (de Jong, 2002; Karlebach and Shamir, 2008; Albert et al., 2013; Samaga and Klamt, 2013). Among these formalisms, the discrete, logical approach is particularly useful to model biological systems for which detailed kinetic data are lacking, which is often the case (Bornholdt, 2008; Wang et al., 2012; Naldi et al., 2014). Moreover, logical modeling allows the consideration and the dynamical analysis of comprehensive signaling/regulatory networks. Here, we rely on the multivalued formalism initially introduced by Thomas and D'Ari (1990).

Following Thomas, we model networks in terms of a *logical regulatory graph* (LRG), where nodes represent regulatory components, while edges denote regulatory interactions (activations or inhibitions). Each component is associated with a discrete variable denoting its (current) functional level of activity. In addition, a *logical rule* (or *logical function*) describes the evolution of this level, depending on the values of the regulators of the component. The regulatory graph together with the logical rules enable the computation of the dynamical behavior of the model, which is usually represented in terms of a State Transition Graph (STG), where each node represents a *state* of the system (i.e., a vector listing the values of all the variables), while arcs represent enabled *state transitions*. The terminal strongly connected components (SCC) of an STG denote the attractors of the underlying network, i.e., capture its asymptotic behavior in terms of stable states or (potentially complex) dynamical cycles. Consequently, the identification of these attractors and the evaluation of their reachability from given initial condition(s) are paramount to understand network behaviors. However, as the number of states may increase exponentially with the number of components, advanced computational methods are needed to analyze the dynamics of discrete

models. In this respect, several strategies have been developed to efficiently assess dynamical properties of comprehensive logical models.

Here, we focus on the analysis of networks encompassing input components that embody external signals, instructing intertwined signaling pathways with feedback regulations. Each (fixed) combination of input values (i.e., environmental cues) defines a specific region of the state space where the dynamics and its associated attractors are confined. In the case of models of networks controlling cell differentiation, attractors correspond to differentiated patterns of gene expression (or protein activity). We call these attractors *differentiated states*, which are generally stable states [see e.g., Naldi et al. (2010)], but can also be complex attractors denoting homeostasis or oscillatory behavior [see e.g., Bonzanni et al. (2013)]. It is of particular interest to assess how input value changes affect differentiated states, sometimes resulting in functional reprograming. The capacity of cells to change their asymptotic behaviors depending on environmental cues is referred to as *cell plasticity* [see e.g., O'Shea and Paul (2010)]. In this manuscript, we present a methodology to assess cell plasticity, relying on the logical formalism assets and recent computational methods, including model checking techniques.

Model checking is a computer science technique for the verification of large discrete dynamical systems (Clarke et al., 1999). It has been recently applied to the analysis of biological networks (Chabrier and Fages, 2003; Batt et al., 2005; Schwarick and Heiner, 2009; Arellano et al., 2011; Brim et al., 2013). Properties are formalized in terms of temporal logic statements, and the verification process explores (restricted) regions of the state space, in order to check the truthfulness of the properties. Here, we consider a further improvement that consists in defining input values as labels of the transitions in STGs, thereby reducing the number of states. This allows to efficiently assess input conditions when verifying, for example, reachability properties between differentiated states. For this, we use a specific symbolic model checker called *NuSMV-ARCTL*, along with a temporal logical semantics enabling the specification of properties with restrictions on the input valuations (Lomuscio et al., 2007).

We consider the case of T-helper (Th) cell differentiation to demonstrate the assets of the logical framework and the power of model checking to elucidate how cells respond to environmental stimuli. More precisely, we model the cellular network controlling the differentiation of Th cells, which regulate many physiological and pathological immune responses. Upon activation by antigen presenting cells (APCs), naive Th cells polarize into distinct Th subtypes expressing different sets of cytokines, tailoring appropriate immune responses to the invading pathogen. Recent experimental data highlight the ability of Th subtypes to alter and even reprogram their phenotypes, according to environmental cues (Nakayamada et al., 2012). These observations challenge the classical linear view of Th differentiation into distinct lineages, raising fundamental questions regarding the mechanisms underlying Th differentiation and plasticity.

In order to get insights into the dynamical behavior of Th cell differentiation, several models describing the regulatory network controlling Th commitment have been proposed, relying on quantitative modeling approaches (van den Ham and de Boer, 2008, 2012; Mendoza and Pardo, 2010) or using discrete qualitative frameworks (Mendoza, 2006; Naldi et al., 2010; Martinez-Sosa and Mendoza, 2013). Here, the logical model of Th cell differentiation of Naldi et al. (2010) is extended to cover several novel Th subtypes. Focusing on Th polarization and reprograming events, we show how biologically relevant properties can be formalized and tested using model checking. More precisely, we compute all reprograming events between Th subtypes under specific documented polarizing cytokine environments, providing a global and synthetic representation of Th plasticity in response to these environmental cues. This analysis leads to the prediction of Th-subtypes conversions, which will need to be assessed experimentally. Finally, we delineate several strategies for Th subtype reprograming, as well as for naive Th cell polarization toward a novel hybrid Th subtype (predicted by our model).

This manuscript is organized as follows. Section 2 briefly reviews the basics of the logical modeling framework, including model definition and an overview of computational methods to analyze dynamical properties. We also introduce the use of model checking to enhance the analysis of logical models, in particular when these include input components. This methodology is then applied to a logical model for Th differentiation in Section 3, which includes a presentation of the resulting biological insights. Section 4 concludes the manuscript with a discussion and some prospects.

2. MATERIALS AND METHODS

In this section, we introduce the logical framework, presenting the rationale underlying the model definition. We further describe model modifications accounting for genetic perturbations (e.g., gene knock-out or knock-in) along with a model reduction method. Next, we briefly present computational strategies to efficiently analyze properties of logical models. Finally, we focus on the assets of model checking to enhance the dynamical analysis of large signaling/regulatory logical networks. **Figure 1** illustrates the workflow for logical model definition and analysis, on which we rely to address the question of Th cell plasticity. Most methods presented in this section are implemented in GINsim (Chaouiya et al., 2012)[1].

2.1. LOGICAL MODEL CONSTRUCTION

This subsection shortly introduces the definition of multivalued logical models [for more details and formal definitions, see e.g., (Thomas and D'Ari, 1990; Chaouiya et al., 2003)].

2.1.1. Logical formalism

A logical model of a regulatory and/or signaling network is defined as an LRG, where:

- $\{s_1, \ldots, s_n\}$ is the set of nodes, which embody the components of the network; these may correspond to proteins, genes, or phenomenological signals (e.g., the node APC in **Figure 2** denotes an Antigen Presenting Cell, present or not).
- Each component s_i is associated with a discrete (positive integer) variable, which takes its values in $S_i = \{0, \ldots, \max_i\}$; for simplicity, we denote both the component and its associated variable by

[1] http://ginsim.org

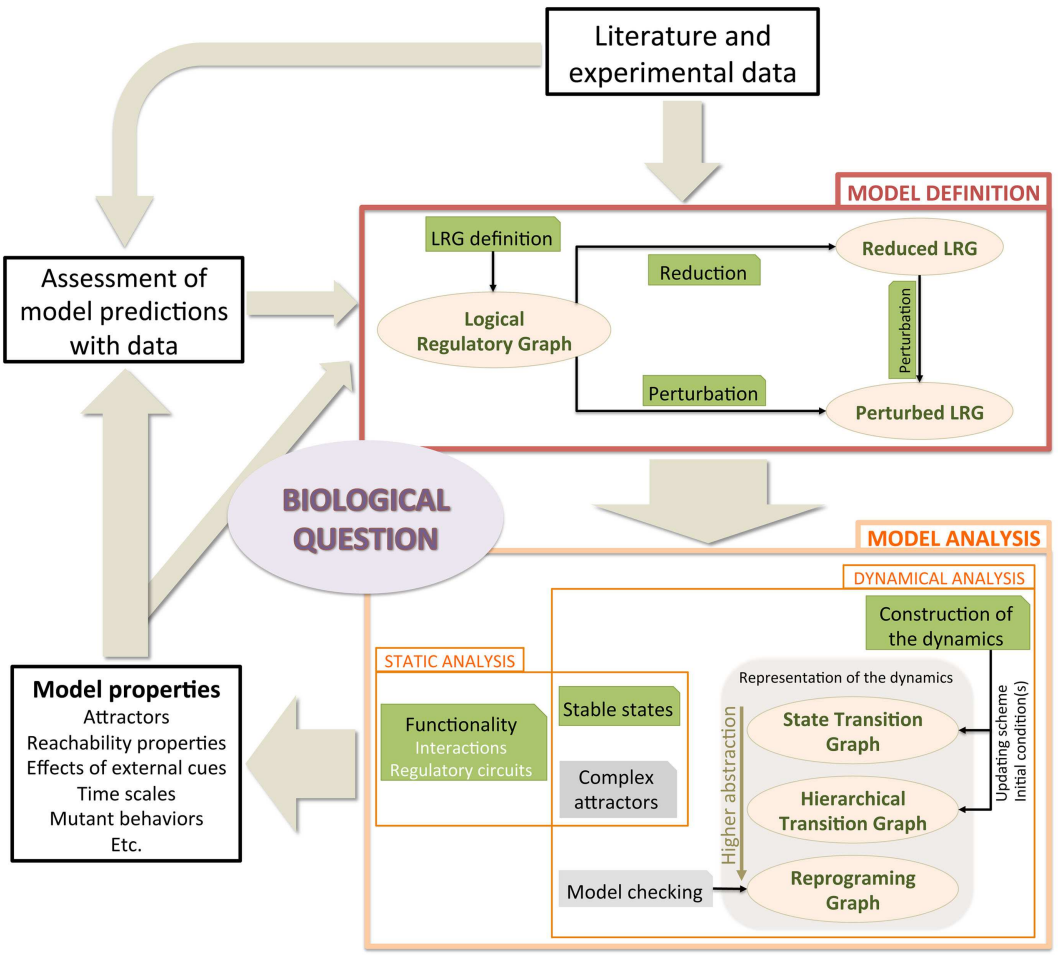


FIGURE 1 | Typical workflow to tackle a central biological question using logical model construction and analysis. A model is defined, relying on literature and experimental data (box *Model Definition*). The model is then analyzed (boxes *Static analysis* and *Dynamical analysis*). The identification of the attractors is performed either by static methods (see Sections 2.2.1 and 2.2.2) or by inspecting the dynamics (see Sections 2.2.3 and 2.3). Dynamics are represented at different levels of abstraction, from the comprehensive state transition graphs to the reprograming graphs. Resulting properties are confronted with biological observations, leading to predictions and/or to model revision. Ellipsoid boxes relate to the different model versions and behavior representations. Green boxes denote methods that are available in GINsim, whereas gray boxes denote analyses performed with other software tools.

s_i, embodying the component level of activity or concentration. In general, the maximum level of s_i, denoted $\max_i$, is set to 1 (i.e., Boolean variable), but it can take higher values to convey qualitatively distinct functional levels.

- Each interaction (s_i, s_j, θ) is defined by its source s_i, its target s_j and a threshold θ; the interaction is said to be effective when $s_i \geq \theta$; note that $\theta \leq \max_i$ (the threshold cannot exceed the maximal level of the source).
- The state space of the LRG is given by $\mathbb{S} = \Pi_{i=1,\ldots,n} S_i$; hence a state of the model is a vector $\mathbf{s} = (s_i)_{i=1,\ldots,n}$.
- The model behavior is specified in terms of *logical rules* (or *logical functions*): the evolution of s_i is defined by $\mathcal{K}_i: \mathbb{S} \rightarrow S_i$ with $\mathcal{K}_i(s)$ specifying the target value of s_i when the system is in state s.

The software GINsim provides a graphical interface for the LRG definition, including the components (nodes) and their ranges (maximum values), the interactions (signed arcs) and their thresholds, along with the logical rules [using Boolean expressions or *logical parameters* (Thomas and D'Ari, 1990)].

The behavior of an LRG is classically represented in terms of a STG, which encompasses the initial model state(s) together with their direct and indirect successors. A transition between two states corresponds to the update of specific components. These updates are dictated by the logical rules. When several components are called to change their values at a given state, these updates are performed according to an updating scheme. The most used updating schemes are the fully synchronous updating (all changes are performed simultaneously, leading to a unique successor), and the fully asynchronous updating (all changes are performed independently, leading to as many successors as the number of updated components). Further details on STG and updating schemes are provided in Section 2.2.3.

In such dynamical models, the asymptotic behavior of the system is captured by the *attractors*. These correspond to the terminal

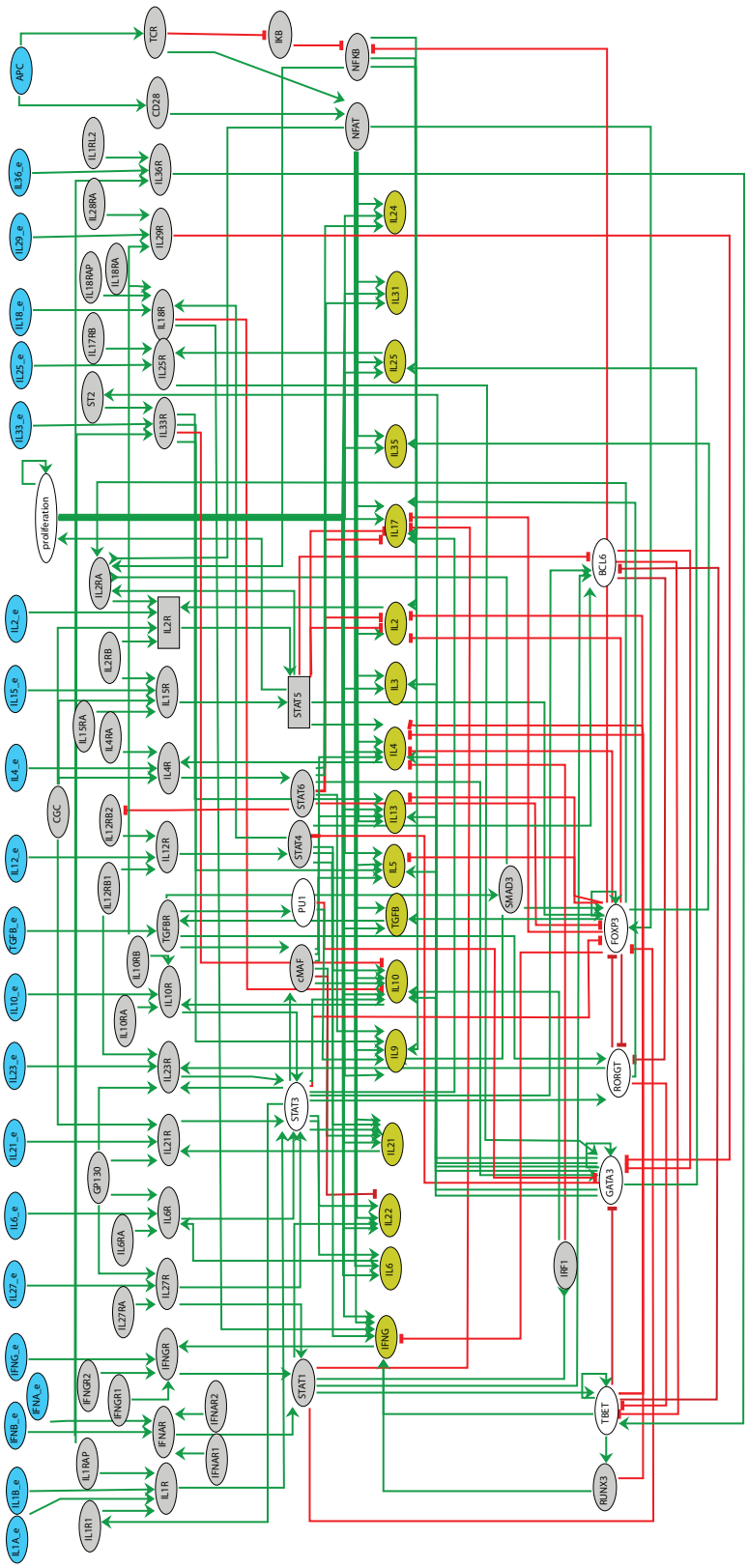


FIGURE 2 | Regulatory graph of Th differentiation logical model. The model encompasses 101 components (among which 21 input nodes) and 221 interactions. The components denoting the inputs are in blue, those denoting the secreted cytokines in olive. Green edges correspond to activations, whereas red blunt ones denote inhibitions. Ellipses denote Boolean components, whereas rectangles denote ternary ones. Gray-out components are those selected for reduction.

SCC of the STG. An SCC is defined as a maximal set of mutually reachable states. An SCC is denoted *terminal* when no transition leaves this state set (i.e., once the system enters this set, it is trapped there forever). An attractor is defined by either a single state, which corresponds to a stable state denoting a stable pattern of expression often interpreted as a cell differentiation state, or by a larger set of states involved in a dynamical terminal *cycle*, denoting an oscillatory (or homeostatic) behavior. It is therefore important to identify these attractors along with reachability properties (e.g., to determine the attractors reachable from a specific initial state).

2.1.2. Logical modeling of network perturbations

In the logical framework, it is straightforward to define perturbations such as gene knock-out, gene knock-in, or more subtle perturbations (e.g., rendering a component insensitive to the presence of one of its regulators). Modeling such perturbations amounts to specific modifications of the corresponding logical rules. Modifications affecting several components can be easily combined. Given a logical model, one can thus define various perturbations to account for experimental observations or to generate predictions regarding the dynamical role of regulatory components or interactions.

2.1.3. Reduction of logical models

It is often useful to simplify large models by abstracting components, hence diminishing the size of the model state space. In this respect, GINsim implements a reduction method automating the reduction of any component, except those that are self-regulated (Naldi et al., 2011). The computation of a reduced model is performed iteratively: to remove a component, the logical rules of its targets are modified to account for the (indirect) effects of the regulators of this component. This is efficiently done in time polynomial in the number of targets (components regulated by the removed one) and regulators of the removed components. In the case of a Boolean model, removing n components leads to a reduction of the state space by a factor 2^n.

Obviously, such a reduction may change the dynamics. In fact, it conserves the nature (and number) of the stable states and of the terminal elementary cycles [also called simple cycles, with neither repeated states nor repeated transitions (Berge, 2001)]. However, oscillatory components may be split or isolated, and reachability properties only partly conserved. Depending on the type of components that are removed upon reduction, specific dynamical properties are preserved. In Saadatpour et al. (2013), the authors showed that all the attractors of an asynchronous Boolean model are conserved upon reduction of input and pseudo-input components (i.e., components with no regulators or regulated by only input and pseudo-input components). Additionally, Naldi et al. (2012) proved that the reduction of output and pseudo-output components not only preserves the attractors, but also their reachability properties [output components regulate no other components, and pseudo-output components are those regulating only (pseudo-) output components]. In all cases, a trajectory in a reduced model has its counterpart in the original model [see Naldi et al. (2011) for details]. Hereafter, we take advantage of this reduction method to ease the analysis of our Th cell differentiation model (see Section 3).

2.2. MODEL ANALYSIS

Means to investigate the dynamical properties of a model can be subdivided into: (1) static analyses, which infer properties without requiring the construction of the STG; and (2) dynamical analyses, which explore proper representations of the dynamics (see **Figure 1**).

2.2.1. Static analysis – interactions and circuit functionality

The delineation of logical rules for components targeted by several regulators can be relatively tricky. These rules are encoded in GINsim as Multivalued Decision Diagrams, which represent multivalued functions as directed acyclical graphs allowing efficient manipulations (Kam et al., 1998; Naldi et al., 2007).

To help the modeler, GINsim provides a method to check the coherence of the interactions (including their signs) encoded in a regulatory graph with the logical rules associated with its components. Basically, for each interaction (s_i, s_j, θ), GINsim compares the target level of s_j given by its logical function, when (s_i, s_j, θ) is effective ($s_i \geq \theta$) and when it is not ($s_i < \theta$), for all combinations of the remaining regulators of s_j (if any). If both target levels are always equal, we say that this interaction is not *functional*. Relying on this comparison, it is also possible to derive the sign of the interaction (activation or inhibition).

Regulatory circuits (i.e., elementary cycles in the LRG, also called *feedback loops*) drive non-trivial behaviors such as multistability (in the case of positive circuits, involving an even number of negative regulations) or sustained oscillations (negative circuits, involving an odd number of negative regulations) (Thieffry, 2007). Based on the aforementioned method to assess interaction functionality, GINsim enables the delineation of the *functionality context* (if any) of each regulatory circuit (Naldi et al., 2007; Remy and Ruet, 2008). This functionality context is defined as the levels of external regulators that allow each circuit interaction to be functional and thereby affect its target in the circuit. It can be interpreted as the region of the state space where the circuit generates the corresponding dynamical property. This definition enables the identification of the regulatory circuits playing the most important regulatory roles within a complex LRG [see Comet et al. (2013) for further discussion on circuit functionality].

2.2.2. Static analysis – identification of dynamical attractors

Attractors (stable states or terminal cycles) constitute crucial dynamical properties of the model and have thus been the focus of many computational studies. In particular, a SAT-based algorithm was proposed in Dubrova and Teslenko (2011) to compute all the attractors of synchronous Boolean models. However, the problem is harder for the asynchronous updating scheme (see Section 2.2.3). Recently, Zañudo and Albert (2013) introduced a novel method to compute most asynchronous attractors.

Several methods have been proposed to specifically compute the stable states, for example, using constraint programing (Devloo et al., 2003) or polynomial algebra (Veliz-Cuba et al., 2010). To identify all the stable states, GINsim implements an efficient algorithm based on the manipulation of multivalued decision diagrams [see Naldi et al. (2007) for details]. We will rely on this algorithm to compute the stable states of the Th cell differentiation model (see Section 3).

2.2.3. Dynamical analysis – state transition graphs, representation, and analysis

As mentioned above, the discrete dynamics of an LRG can be represented in terms of an STG, where the nodes denote *states* and the arcs represent *transitions* between states. A first approach to investigate dynamical properties consists in analyzing the STG in terms of attractors (terminal SCC), or regarding the existence of paths from an initial state toward specific attractors. The graph of SCC of the STG often provides a convenient, compressed view of the dynamics, in which attractors and reachability properties are easier to visualize. However, this representation may still encompass numerous single state components, hindering the interpretation of the dynamics. To further compress an STG and emphasize its topology, we recently proposed a novel representation, named *hierarchical transition graph* [see Bérenguier et al. (2013) for details].

Still, these representations do not address the identification of the attractors in large STGs. In this respect, Garg et al. (2008) proposed an efficient algorithm to identify all the attractors (synchronous and asynchronous schemes) of Boolean models. Their method relies on a binary decision diagram representation of the STG and can cope with very large models (Xie and Beerel, 1998). An implementation of this algorithm is available along with the software *genYsis*[2].

To further account for kinetic aspects, several strategies have been proposed. One strategy defines priority classes according to biologically founded time scale separations, e.g., fast versus slow processes (Fauré et al., 2006). Alternatively, time delays and constraints on them can be defined and handled with existing methods to analyze timed automata (Siebert and Bockmayr, 2006). Another approach consists in applying continuous time Markov processes on logical state spaces. Based on the delineation of a logical model along with a limited number of kinetic parameters, the software *MaBoSS* uses Monte-Carlo simulations to compute an estimate of the temporal evolution of probability distributions and of the stationary distributions of the logical states (Stoll et al., 2012). Finally, several authors proposed to consider differential models derived from logical models (Mendoza and Xenarios, 2006; Abou-Jaoudé et al., 2009; Wittmann et al., 2009).

2.3. MODEL CHECKING FOR REACHABILITY ANALYSIS

2.3.1. Model checking

The combinatorial explosion of the state spaces of discrete dynamical systems has been addressed during the last 30 years through the development of *model checking*, a computer science technique to verify properties in very large state spaces. The dynamics of discrete systems are directly mapped into a (graph-based) *Kripke structure* (Clarke et al., 1999). Model checkers receive a Kripke structure, either explicitly (representation equivalent to the STG), or implicitly in terms of a transition function specifying the successors of any given state. The latter case corresponds to *symbolic model checking*, which is handled by most model checkers nowadays. To perform a verification, a model checker takes as an input a set of properties denoting real-world observations, specified as temporal logic formulas, and verifies whether each of these properties is satisfied by the Kripke structure induced by the model under study.

Temporal logic formulas specify an order of sequences of transitions between states, without explicit time quantification. Several temporal logics have been defined with different expressive powers, using different types of operators. In the case of asynchronous updating, one might be interested in the study of each alternative path separately. This suggests the use of a temporal logic that provides path quantifiers where, at each step, a choice can be made between multiple paths, i.e., a branching-time temporal logic. Within the family of branching-time temporal logics, *Computation Tree Logic* (CTL) is the most used one. Basic CTL operators are obtained by combining path quantifiers, **E**xists and **A**ll, with temporal operators, ne**X**t, **F**uture, **G**lobally, and **U**ntil (Clarke et al., 1999).

Different model checkers are available, differing in characteristics such as the underlying structure to represent the model dynamics or the supported temporal logics. A few examples are: CADP (Garavel et al., 2007), which uses labeled transition systems, supporting temporal logics with high expressive power like *Computation Tree Regular Logic* (CTRL) (Mateescu et al., 2011) or μ-calculus (Kozen, 1983); Antelope (Arellano et al., 2011), which uses STGs, supporting Hybrid CTL, an extension of CTL with a special operator capable of selecting partly characterized states; and NuSMV (Cimatti et al., 2002), a symbolic model checker, which uses multilevel decision diagrams, supporting the verification of properties through CTL or *Linear Temporal Logic* (LTL) (Clarke et al., 1999). As an open source project providing a generic description language for the specification of discrete dynamical systems, NuSMV is particularly prone to be extended by other research groups with additional features (see next subsection).

2.3.2. Model checking applied to the analysis of logical models of signaling networks

Systems biology is a recent, successful application field for model checking techniques, covering a variety of modeling formalisms and/or type of properties to be verified [for details see Brim et al. (2013)]. Here, we use GINsim, our modeling tool, which automatically exports logical models under the asynchronous scheme into NuSMV specifications. Biological observations are then expressed as sets of temporal logic formulas.

Computational models of signaling/regulatory networks aim at unraveling how external stimuli are processed to determine cell responses. In these networks, input nodes convey environmental cues, which are often assumed to be constant. Each combination of constant values of the inputs defines an STG, which is disconnected from the STGs defined by different combinations of input values. In other words, each fixed environmental condition defines a specific region of the state space in which the system is trapped. Rather than having input variables being part of the state definitions, we label each transition with the input values enabling this transition. This yields a state space defined solely by non-input variables and therefore a unique STG (Monteiro and Chaouiya, 2012). The extent of this reduction depends on the number of input components and on their value ranges.

In order to take advantage of this reduction, we need to be able to verify properties referring to both states and transition labels.

[2]http://www.vital-it.ch/software/genYsis/

NuSMV can only verify properties on state characterizations. We thus use *NuSMV-ARCTL*, which verifies properties combining both state and transition characterizations (Lomuscio et al., 2007). For the verification of such properties, *NuSMV-ARCTL* considers a CTL extension called *Action-Restricted CTL* (ARCTL). **Table 1** describes the syntax and semantics of the main ARCTL operators. With ARCTL, reachability properties are specified not only by characterizing the set of initial and target states, but also by constraining the values of some input components (transition labels), while the remaining input components are allowed to freely vary.

Here, we take advantage of the expressiveness of ARCTL to study the influence of specific environmental conditions on the reprograming of chosen cell types (see Section 3). As presented hereafter, the Th cell differentiation model specified in GINsim is exported into a NuSMV specification, while properties of biological interest are specified as ARCTL temporal formulas. This allows us to define a novel, abstracted view of the dynamical behaviors called *reprograming graph*, which reveals switches between attractors upon changes in the input component values: the nodes of this graph represent the model attractors; and the arcs, labeled by specific combinations of input values, denote paths between those attractors.

Table 1 | Syntax and semantics of the main ARCTL temporal operators [for a complete description see Lomuscio et al. (2007)].

Syntax	Semantics
EAF (α) (ϕ)	There is at least one path leading to a state that satisfies ϕ and the input restriction α must be satisfied along that path
AAF (α) (ϕ)	All the paths lead to a state that satisfies ϕ and the input restriction α must be satisfied along all the paths
EAG (α) (ϕ)	There is at least one path along which all the states satisfy ϕ and the input restriction α is satisfied along that path
AAG (α) (ϕ)	All the states of all the paths satisfy ϕ and the input restriction α is satisfied along all the paths
EA (α)[$\phi \cup \psi$]	There is at least one path along which all the states satisfy ϕ, leading to a state that satisfies ψ and the input restriction α is satisfied along the path
AA (α)[$\phi \cup \psi$]	All the states of all the paths satisfy ϕ, leading to a state that satisfies ψ and the input restriction α is satisfied along all the paths

α denotes a restriction, defined only by the input variables, which must be satisfied (true) along the path; ϕ and ψ denote the restrictions, defined only by non-input variables, which must be satisfied at the target state or along the path.

3. APPLICATION: T-HELPER CELL DIFFERENTIATION

T-helper (CD4+) lymphocytes play a key role in the regulation of the immune response. Upon activation by APC, naive CD4 T cells (Th0) differentiate into specific Th subtypes producing different cytokines, which affect the activity of immune effector cell types (e.g., B lymphocytes, effector CD8 T cells, macrophages, etc.).

Three main types of signals are involved in this Th cell differentiation process (Figure S1 in Supplementary Material): (i) the presentation of antigenic peptide in conjunction with the major histocompatibility complex class II molecules (MHC-II) stimulate specific T cell receptors (TCR); (ii) co-stimulatory molecules further contribute to T cell activation and clonal proliferation; (iii) cytokines secreted by APCs and other cells bind their specific receptor(s) on the surface of Th0 cells, thereby affecting Th differentiation.

The cytokine environment instructs Th0 to enter a specific differentiation program in order to match the type of pathogen primarily stimulating the APCs. Over the last decade, a variety of Th subtypes have been discovered (Nakayamada et al., 2012), well beyond the initial identification of Th1 and Th2 dichotomy (Mosmann et al., 1986; Mosmann and Coffman, 1989).

Currently, several Th subtypes (Th1, Th2, Th17, Treg, Tfh, Th9, and Th22) have been well established. These *canonical* subtypes are characterized by their ability to express specific sets of cytokines under the control of a *master regulator* transcription factor (Figure S1 in Supplementary Material). However, various hybrid Th subtypes expressing several master regulators have been recently identified (Ghoreschi et al., 2010; Duhen et al., 2012; Peine et al., 2013). Evidences for substantial plasticity in Th differentiation have also been reported, including reprograming events between Th subtypes under specific cytokine environments (Yang et al., 2008; Lee et al., 2009; Hegazy et al., 2010). These findings challenge the classical linear view of Th differentiation and raise the question of which mechanisms underlie the observed diversity and plasticity of Th phenotypes.

Unraveling the complexity of Th differentiation and plasticity requires the development of an integrative and systematic approach articulating experimental analysis with computational modeling. We are currently setting a multi-parametric *in vitro* experimental approach to decipher how the microenvironment globally controls Th cell differentiation. In parallel, we are developing a comprehensive logical model of Th differentiation covering all parameters assessed in our experimental setup. Extending the modeling study reported in Naldi et al. (2010), the model presented here includes additional transcription factors and cytokine pathways and hence accounts for the differentiation of several novel Th subtypes. On the basis of this model, we illustrate how the computational methods described in Section 2, in particular model checking, can be used to assess biologically relevant dynamical properties. The model file as well as the steps to reproduce all the results described below are available from the model repository of the GINsim web site.

3.1. MODEL DESCRIPTION

Our Th differentiation model encompasses different layers (see **Figure 2**), namely:

- the cytokine inputs along with the APCs;
- the cytokine receptors and their subchains, along with the TCR and the co-stimulatory receptor CD28;
- the intracellular signaling factors, including "Stat" family proteins (Stat1, Stat3, Stat4, Stat5, and Stat6), the TCR and co-stimulatory signaling components (NFAT, IκB, and NFκB), the master regulators (Tbet, Gata3, Rorγt, Foxp3, and Bcl6),

along with additional transcription factors involved in Th differentiation (cMaf, PU.1, Smad3, IRF1, and Runx3);
- the main cytokines secreted by Th cells;
- a component modeling the proliferation of the cell.

By and large, the model encompasses 21 signaling pathways (comprising external cytokines, receptor chains, etc.), 17 transcription factors, 17 cytokines expressed by Th cells, and 1 node accounting for cell proliferation, amounting to 101 components in total. In comparison with the model reported in Naldi et al. (2010), this model integrates factors characterizing novel Th subtypes (Tfh, Th9, and Th22) as well as additional signaling pathways and secreted cytokines involved in the differentiation and the definition of Th cellular types. A complete list of the components of the model along with supporting evidence is provided in Table S1 in Supplementary Material. The logical rules associated with the components are listed in the Table S2 in Supplementary Material.

As in Naldi et al. (2010), a gene expression pattern is associated with each canonical Th subtype, based on experimental evidence (**Table 2**). Each pattern represents a restriction of Th cell states to a subset defined by the activation or the inactivation of critical markers characterizing the corresponding canonical Th subtype. In the following sections, we present the results obtained by the application of the aforementioned computational methods to our Th differentiation model.

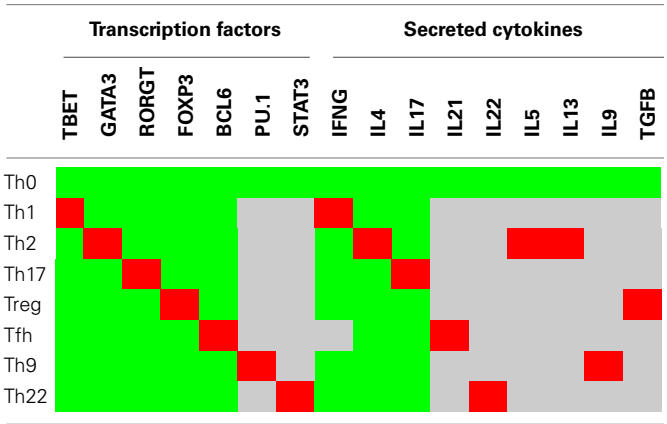
3.2. STATIC ANALYSIS

We first checked the consistency of the rules inferred from experimental data (Table S2 in Supplementary Material) with the interactions composing the regulatory graph of **Figure 2**. An analysis of interaction functionality led to the identification of a single non-functional interaction (IL10R → Stat3). Although the role of this interaction is not yet clear, we kept it in the regulatory graph as it is documented (see Table S1 in Supplementary Material).

Next, to ease the model analysis, we derived a reduced version of this model using the reduction method described in Naldi et al. (2011), keeping internal components characterizing the canonical Th patterns (cf. **Figure 2**, where the gray nodes denote the components selected for reduction).

Using the method described in Naldi et al. (2007), we computed all the stable states for all the input combinations and grouped them according to phenotypic markers (see also Subsection 2.2.2 above). Since the reduction preserves the stable states, each stable state of the reduced model strictly corresponds to one stable state of the original model (and vice versa). This analysis led to the identification of 82 context-dependent stable states, including sets of stable states matching the activity patterns associated with each canonical Th subtype (see Table S3 in Supplementary Material). This analysis further predicts the existence of stable states representing hybrid cellular types, i.e., expressing several master regulators, including four hybrids expressing two master regulators, which have been recently reported in the literature, and another one ($Tbet^+Gata3^+Foxp3^+$) expressing three master regulators, which has not yet been experimentally observed. Each of the stable states found is associated with a subset of input combinations. One can actually recover the input configurations associated with each stable state, getting a first insight into the role of environmental cues in controlling the asymptotic behaviors of the system (see Section 3.3.2 for an illustration of this analysis).

Table 2 | Logical expression patterns for the canonical Th subtypes.



Red and green cells denote the activation and inactivation of the components (column entries), with respect to the canonical Th subtype (row entries). Gray cells represent components that can be either activated or inactivated for the corresponding canonical Th subtype. The components not mentioned are considered to be either activated or inactivated, except in the case of Th0, where they are all inactivated.

3.3. REACHABILITY ANALYSIS

As mentioned above, static analysis of the logical model allows for the identification of stable Th cellular types along with their associated input configurations. Our next aim is to determine how environmental cues control the differentiation and plasticity of these Th cell types. This question amounts to check whether a cellular type is reachable from a given initial state for specific input conditions, under the asynchronous update. This kind of questions can be efficiently addressed using model checking, by verifying temporal properties under constant or varying input conditions.

We first carried out a systematic analysis of reachability properties between the canonical Th subtypes as defined in **Table 2**, under specific constant polarizing cytokine environments. We consider nine prototypic environmental conditions (listed in **Table 3**) for this reachability analysis, including seven documented polarizing cytokine environments known to commit Th cells into the canonical subtypes.

We used the *NuSMV-ARCTL* model checker and instantiated the following generic property with values from **Tables 2** and **3**:

$$\text{INIT } c_1; \text{EAF (e) } (c_2 \wedge \text{AAG (e) } (c_2)) \quad (1)$$

This property asserts the existence of a path from a canonical Th pattern c_1, instantiated with values from **Table 2**, toward a (stable) canonical Th pattern c_2, also instantiated with values from **Table 2**, under an input condition e, instantiated with values from **Table 3**.

Checking this property for all the combinations of canonical Th patterns and input conditions, one can represent the verified properties through a reprograming graph, which here abstracts paths between Th patterns and recapitulates the polarizing and reprograming events predicted by our model (**Figure 3**). This graph

Table 3 | Prototypic environmental conditions.

	Environmental conditions								
	APC	IL12_e	IL4_e	IL6_e	TGFB_e	IL1B_e	IL23_e	IL21_e	IL2_e
No stimulation									
APC only									
proTh1									
proTh2									
proTh17									
proTreg									
proTfh									
proTh9									
proTh22									

*Each row corresponds to a prototypic environment defined as a combination of APC and cytokine inputs (columns). These environments encompass seven documented polarizing environments (denoted "proThX") known to polarize naive Th cells into the canonical subtypes (defined in **Table 2**). Red/green cells represent present/absent inputs. Non-mentioned inputs are considered as absent.*

provides a global and synthetic representation of Th plasticity depending on environmental cues. Focusing on polarizing events from naive Th0 cells to the other Th subtypes, our model is consistent with experimental data, showing that each canonical subtype can be reached from the naive state Th0 (blue arcs starting from Th0 in **Figure 3**) in the presence of specific polarizing cytokine combinations (denoted by the labels associated with the blue arcs in **Figure 3**). The remaining Th subtype conversions present in the reprograming graph would need to be assessed experimentally.

An extensive discussion of all these Th type conversions is beyond the scope of this article. However, one interesting outcome is the inherent dissymmetry of this graph, with some Th subtypes apparently very stable under the environments considered (e.g., Th1 node, with seven incoming arcs but only one outgoing one), while others need very specific conditions for their maintenance (e.g., Th9 node, with six outgoing arcs and only one incoming one).

Hereafter, we focus on specific biological questions regarding Th differentiation and plasticity and show how model checkers can be applied to address these questions. Two biological questions will be considered: (i) the delineation of reprograming strategies to convert Th1 into Th2, and vice versa; (ii) the identification of relevant environmental conditions enabling the polarization to the $Tbet^{+}Gata3^{+}Foxp3^{+}$ hybrid Th subtype identified in the course of the stable state analysis.

3.3.1. Reprograming between Th1 and Th2

Since the discovery of Th1 and Th2 subtypes, Th1 and Th2 commitments have been for a long time considered as mutually exclusive (Murphy and Reiner, 2002). However, recent experimental observations challenged this Th1/Th2 dichotomy (Hegazy et al., 2010; Antebi et al., 2013; Peine et al., 2013), raising the question of which environmental conditions can instruct Th1 or Th2 interconversions.

We first address this question by investigating Th1–Th2 reprograming strategies for the prototypic input conditions (listed in **Table 3**). From the reprograming graph (**Figure 3**), two strategies emerge: (1) although there is no direct path from Th1 cells toward Th2 cells, one could consider a two-step approach to reprogram Th1 cells into Th2 cells by applying a proTh17 condition, followed by a proTh2 condition; (2) as there is a direct path from Th2 to Th1 labeled with proTfh conditions, the application of a proTfh environment would potentially reprogram Th2 cells into Th1 cells.

We then ask whether other (constant or varying) input condition strategies could be identified for the reprograming between Th1 and Th2, beyond the prototypic environmental conditions. This question can be addressed using the following ARCTL formulas:

$$\begin{aligned} &\text{INIT Th1; EAF } (\neg e)\ (\text{Th2}) \\ &\text{INIT Th2; EAF } (\neg e)\ (\text{Th1}) \end{aligned} \tag{2}$$

where e denotes the set of all the prototypic inputs (and consequently $\neg e$ denotes the set of all the input combinations except the prototypic ones). *NuSMV-ARCTL* evaluates both formulas as true, implying that it must exist at least one non-prototypic (constant or varying) input condition allowing for the reprograming of Th1 into Th2, and vice versa.

To further illustrate the power of model checking to analyze cell plasticity, we focus on Th2 reprograming into Th1. Our initial analysis predicts that the prototypic proTh1 cytokine environment does not enable this reprograming (see **Figure 3**). However, looking more closely at the regulatory graph, we see that the TGFβ signaling pathway inhibits Gata3, the master regulator of Th2 cells (**Figure 2**). This suggests an alternative two-step strategy to reprogram Th2 into Th1, by applying first TGFβ in the cell environment to inhibit Gata3, and thereby block its inhibitory effects on Th1 differentiation, followed by the application of a proTh1 environment to induce Th1 polarization. We can assess this strategy using the following ARCTL formula:

$$\begin{aligned} &\text{INIT Th2;} \\ &\text{EAF } (e)\ \big(\text{true} \wedge \text{EAF}\ \big(\text{proTh1}\big)\ \big(\text{Th1} \wedge \text{AAG}\ \big(\text{proTh1}\big)\ (\text{Th1})\big)\big) \end{aligned} \tag{3}$$

where e is an input condition restricting only TGFβ to ON (all other inputs can freely vary). This property is evaluated as true. We can thus conclude that this alternative strategy could also be used to reprogram Th2 into Th1 cells.

Beyond this analysis, one can further investigate network perturbations (e.g., gene knock-in or knock-out) enabling Th1–Th2 reprograming. This type of questions can be assessed using model checking of perturbed models. Here, we focus again on reprograming Th2 cells into Th1 cells under the prototypic proTh1 input condition. Over-expression of a Gata3 (Th2 signature) inhibitor (e.g., PU.1 or Bcl6) would be a relevant option. However, Bcl6 should be discarded because it also inhibits Tbet (Th1 signature) (cf. the logical rule of Tbet in Table S2 in Supplementary Material). Using the generic property (1), the analysis of a perturbed model with ectopically expressed PU.1 suggests that this perturbation can

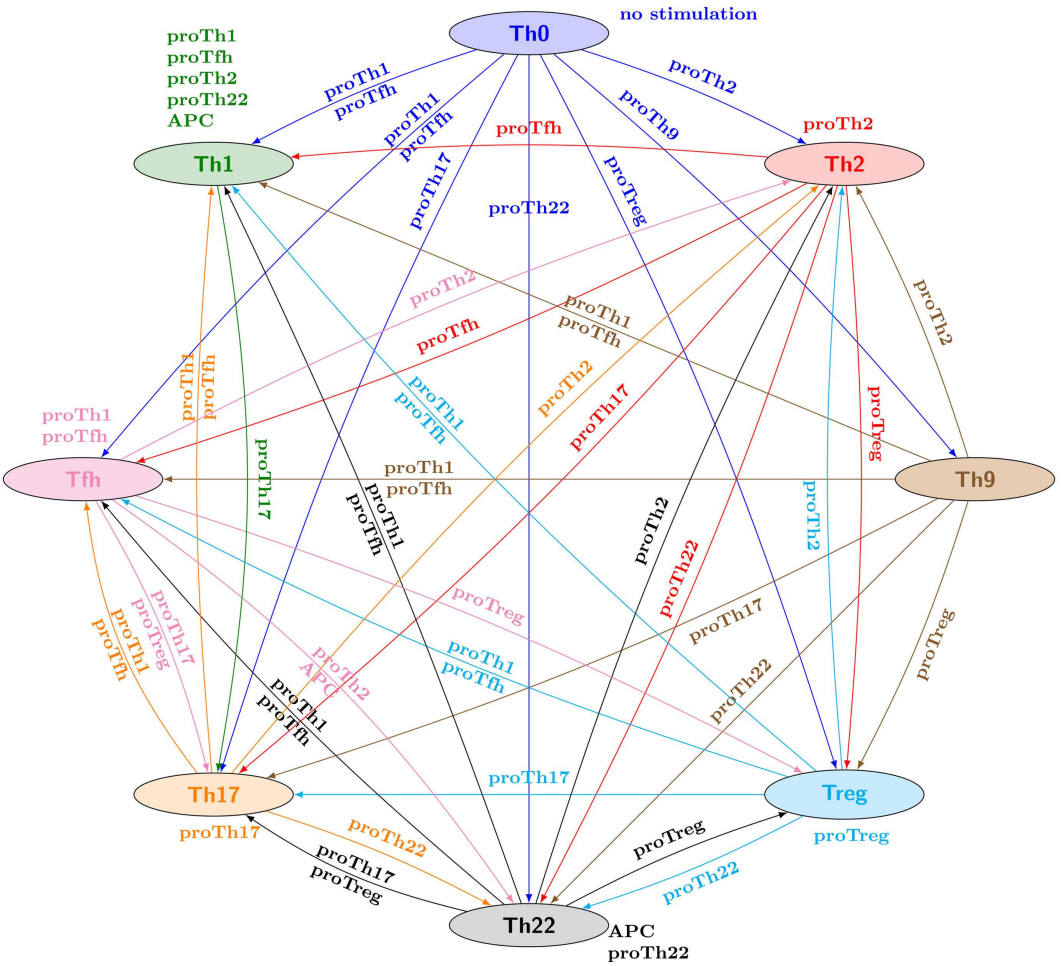


FIGURE 3 | Reprograming graph, considering all canonical Th subtypes, generated with the model checker *NuSMV-ARCTL*. Nodes represent sets of states characterizing the canonical Th subtypes defined in **Table 2**. There is an arc labeled with *e*, going from node c_1 to node c_2, whenever the following ARCTL temporal logic formula is verified: INIT c_1; EAF (*e*) ($c_2 \wedge$ AAG (*e*)(c_2)). It should be noted that the existence of a single reprograming path from a Th subtype to another one does not necessarily imply the stability of the target Th subtype, since *NuSMV-ARCTL* considers that a property is true if and only if it is verified by the whole set of states in the initial conditions. Hence, if at least one state associated with a given subtype points to a state not associated with this subtype (for given input conditions), then the stability of the Th subtype is not represented (see for example, Th9 subtype, which is not considered stable under proTh9 input condition).

indeed induce the reprograming of Th2 into Th1 in the presence of the prototypic proTh1 input condition.

Finally, we can study the role of critical regulatory interactions underlying such reprograming events through model checking analyses of perturbed models. Turning back to the reprograming strategies 1 and 2 presented above, we now focus on the inhibitory interactions acting upon Tbet and Gata3, the master regulators of Th1 and Th2 cell types, respectively. For example, in **Figure 2**, we see that Rorγt inhibits Tbet, which could be relevant for reprograming strategy 1, while Bcl6 inhibits Gata3, which might be relevant for reprograming strategy 2. Analyses of perturbed models, using the ARCTL generic property (1), where either one or the other interaction is suppressed, suggest that the inhibition of Tbet by Rorγt is indeed necessary for reprograming strategy 1, whereas the inhibition of Gata3 by Bcl6 is indeed necessary for reprograming strategy 2.

3.3.2. *Reachability of the triple hybrid subtype Tbet$^+$ Gata3$^+$ Foxp3$^+$*

The steady state analysis of our model in Section 3.2 predicts the existence of a stable hybrid Th subtype co-expressing Tbet (characteristic of the Th1 signature), Gata3 (Th2 signature), and Foxp3 (Treg signature), which has not been yet experimentally reported.

Using model checking, we can evaluate environmental conditions that might enable the polarization of naive Th0 cells into this hybrid subtype. First, the input combinations for which this hybrid subtype is stable can be extracted directly from the steady state analysis (not shown). In these combinations, some cytokines appear to be either always ON, namely IL15, or always OFF, TGFβ. Moreover, TGFβ signaling, via Smad3, is clearly needed to activate Foxp3 (see logical rule of Foxp3 in Table S2 in Supplementary Material), suggesting that a transient TGFβ environment is necessary to polarize naive Th0 cells into the hybrid subtype

$Tbet^+Gata3^+Foxp3^+$. This last hypothesis can be verified using the ARCTL formula:

$$\text{INIT Th0; AAG } (e) \left(\neg \left(Tbet^+Gata3^+Foxp3^+\right)\right) \quad (4)$$

where e denotes an input condition restricting only TGFβ to OFF (all other inputs can freely vary). This formula states that the hybrid pattern cannot be reached from whatever path leaving the canonical Th0 pattern under the input restriction e. As the property is evaluated as true, we conclude that a strategy without (transient) TGFβ in the environment cannot polarize Th0 into the hybrid subtype, confirming our hypothesis.

Therefore, a two-step approach to polarize naive Th0 cells into the hybrid subtype $Tbet^+Gata3^+Foxp3^+$ could be considered, applying TGFβ transiently, before applying an environment containing IL15. This strategy can be evaluated using the ARCTL formula:

$$\text{INIT Th0; EAF } (e_1) \left(\text{true} \wedge \text{EAF } (e_2) \left(Tbet^+Gata3^+Foxp3^+ \wedge \text{AAG } (e_2) \left(Tbet^+Gata3^+Foxp3^+\right)\right)\right) \quad (5)$$

where e_1 denotes the first input combination (in which TGFβ and APC are ON), and e_2 denotes the second input combination (in which IL15 and APC are ON and TGFβ is OFF). Two additional input cytokines were also considered in these combinations: IFNγ for Tbet activation and IL25 for Gata3 activation. We consider 18 strategies (input configurations), six of them are able to polarize Th0 into the hybrid subtype (see Table S4 in Supplementary Material). Interestingly, these six strategies have all IFNγ switched OFF in the first input combination and turned ON in the second input combination.

4. CONCLUSION AND PROSPECTS

Considering logical models of large cellular regulatory networks, we have focused on model checking to explore induced dynamical properties. Over the last decades, computer scientists have made spectacular advances in the development of powerful model checkers, regarding both performances and expressivity power. Several model checkers are freely available and can be used to check specific properties of dynamical models of biological systems. As illustrated above, asynchronous dynamics of logical models integrating signaling pathways with transcriptional networks can be readily translated into explicit or implicit Kripke structures, and thereby become amenable to standard or action-restricted model checking.

We have applied this approach to the analysis of a logical model for a comprehensive signaling/regulatory network controlling Th cell differentiation, which encompasses 101 components (most but not all Boolean) and 221 regulatory interactions. As the state space induced by this network is gigantic (encompassing over 2^{100} states), scalable formal methods enabling the exploration of interesting dynamical properties are paramount. In this respect, we have combined three complementary approaches: (i) a formal reduction method conserving the main dynamical properties, including the stable states (described in Section 2.1.3); (ii) an algorithm enabling the identification of all the stable states in large logical models (described in Section 2.2.2); (iii) the use of model checking to verify the reachability of specific stable patterns (reprograming of specific Th cell subtypes) from given initial conditions, in the presence or absence of network perturbations.

We have illustrated the power of the model checking approach by addressing key biological questions related to Th differentiation and plasticity in response to environmental cues. To this end, we have formulated two main types of queries: (i) is it possible to reprogram a specific Th subtype into another one, using specific fixed (or any free) cytokine combinations, in a single (or a multiple) step(s)? (ii) does such reprograming depend on specific regulatory components or interactions (using perturbed models)? We have shown that such biological questions can be efficiently assessed using action-restricted model checking. Using the model checker *NuSMV-ARCTL*, we could confirm that our model is consistent with the polarization of naive Th cells into the canonical Th subtypes under specific cytokine input environments, and delineated several strategies allowing the reprograming between specific Th subtypes (Th1 and Th2) as well as the polarization of naive Th cells toward a novel Th hybrid subtype predicted by our analysis ($Tbet^+Gata3^+Foxp3^+$).

Although our logical model for Th cell differentiation should be further refined using a comprehensive experimental data set (work in progress), it could be already used as a framework to design informative experiments regarding the identification of Th hybrid subtypes, or yet to characterize Th cell plasticity. Some of the resulting predictions (e.g., the existence of $Tbet^+Gata3^+Foxp3^+$ Th hybrid) currently serve as a basis to design experiments *in vitro*.

More generally, we wish to stress that formal modeling can be used at various stages of the deciphering of complex regulatory networks, provided that the formal framework and methods used, as well as the modeling scope, are adapted to the data available. In this respect, qualitative (Boolean or multivalued) logical modeling is well suited to model large biological regulatory networks, for which reliable quantitative data are often lacking (Saez-Rodriguez et al., 2007; Grieco et al., 2013).

Beyond the proof of concept, the development of user-friendly tools is required for a wider use of model checking in systems biology. In this respect, we are currently working on improving the interaction between GINsim and *NuSMV-ARCTL* in two distinct ways, which will be made available in a forthcoming release of GINsim: (1) implementing recurrent temporal logic patterns into our software GINsim to ease the definition of temporal logic formulas; (2) automating the interaction with the model checker and the parsing of the results, as well as the generation of the reprograming graph.

AUTHOR CONTRIBUTIONS

The model was defined by Wassim Abou-Jaoudé with the help of Maximilien Grandclaudon under the supervision of Vassili Soumelis and Denis Thieffry. All model analyses were performed by Wassim Abou-Jaoudé and Pedro T. Monteiro under the supervision of Claudine Chaouiya and Denis Thieffry. Aurélien Naldi wrote some of the scripts and implemented several algorithms used in this study. All authors contributed to the writing of the manuscript and agreed with its final content.

ACKNOWLEDGMENTS

Wassim Abou-Jaoudé has been supported by postdoctoral grants from the LabEx MemoLife and from the Ecole Normale Supérieure. Pedro T. Monteiro was supported by the FCT grants PEst-OE/EEI/LA0021/2013 and IF/01333/2013. Maximilien Grandclaudon was funded by the Ph.D. program of the Agence National de Recherche sur Le Sida (ANRS). This work was supported by a European Research Council consolidator grant to Vassili Soumelis.

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at http://www.frontiersin.org/Journal/10.3389/fbioe.2014.00086/abstract

REFERENCES

Abou-Jaoudé, W., Ouattara, D. A., and Kaufman, M. (2009). From structure to dynamics: frequency tuning in the p53-Mdm2 network I. Logical approach. *J. Theor. Biol.* 258, 561–577. doi:10.1016/j.jtbi.2009.02.005

Albert, R., Collins, J. J., and Glass, L. (2013). Introduction to focus issue: quantitative approaches to genetic networks. *Chaos* 23, 025001. doi:10.1063/1.4810923

Antebi, Y. E., Reich-Zeliger, S., Hart, Y., Mayo, A., Eizenberg, I., Rimer, J., et al. (2013). Mapping differentiation under mixed culture conditions reveals a tunable continuum of T cell fates. *PLoS Biol.* 11:e1001616. doi:10.1371/journal.pbio.1001616

Arellano, G., Argil, J., Azpeitia, E., Benítez, M., Carrillo, M., Góngora, P., et al. (2011). "Antelope": a hybrid-logic model checker for branching-time Boolean GRN analysis. *BMC Bioinformatics* 12:490. doi:10.1186/1471-2105-12-490

Batt, G., Ropers, D., de Jong, H., Geiselmann, J., Mateescu, R., Page, M., et al. (2005). Validation of qualitative models of genetic regulatory networks by model checking: analysis of the nutritional stress response in Escherichia coli. *Bioinformatics* 21(Suppl. 1), i19–i28. doi:10.1093/bioinformatics/bti1048

Bérenguier, D., Chaouiya, C., Monteiro, P. T., Naldi, A., Remy, E., Thieffry, D., et al. (2013). Dynamical modeling and analysis of large cellular regulatory networks. *Chaos* 23, 025114. doi:10.1063/1.4809783

Berge, C. (2001). *The Theory of Graphs*. New York: Courier Dover Publications.

Bonzanni, N., Garg, A., Feenstra, A., Schutte, J., Kinston, S., Miranda-Saavedra, D., et al. (2013). Hard-wired heterogeneity in blood stem cells revealed using a dynamic regulatory network model. *Bioinformatics* 13, i80–i88. doi:10.1093/bioinformatics/btt243

Bornholdt, S. (2008). Boolean network models of cellular regulation: prospects and limitations. *J. R. Soc. Interface* 5(Suppl. 1), S85–S94. doi:10.1098/rsif.2008.0132.focus

Brim, L., Češka, M., and Šafránek, D. (2013). "Model checking of biological systems," in *Formal Methods for Dynamical Systems, Volume 7938 of Lecture Notes in Computer Science*, eds M. Bernardo, E. de Vink, A. Di Pierro, and H. Wiklicky (Bertinoro: Springer), 63–112.

Chabrier, N., and Fages, F. (2003). "Symbolic model checking of biochemical networks," in *Computational Methods in Systems Biology, Volume 2602 of Lecture Notes in Computer Science*, ed. C. Priami (Rovereto: Springer), 149–162.

Chaouiya, C., Naldi, A., and Thieffry, D. (2012). "Logical modelling of gene regulatory networks with GINsim," in *Bacterial Molecular Networks, Volume 804 of Methods in Molecular Biology*, eds J. van Helden, A. Toussaint, and D. Thieffry (Rome: Springer), 463–479.

Chaouiya, C., Remy, E., Mossé, B., and Thieffry, D. (2003). "Qualitative analysis of regulatory graphs: a computational tool based on a discrete formal framework," in *Positive Systems, Volume 294 of Lecture Notes in Control and Information Science*, eds L. Benvenuti, A. De Santis, and L. Farina (Rome: Springer), 119–126.

Cimatti, A., Clarke, E., Giunchiglia, E., Giunchiglia, F., Pistore, M., Roveri, M., et al. (2002). "NuSMV2: an opensource tool for symbolic model checking," in *Computer Aided Verification, Volume 2404 of Lecture Notes in Computer Science*, eds E. Brinksma and K. Larsen (Berlin: Springer), 359–364.

Clarke, E., Grumberg, O., and Peled, D. (1999). *Model Checking*. Cambridge: MIT Press.

Comet, J.-P., Noual, M., Richard, A., Aracena, J., Calzone, L., Demongeot, J., et al. (2013). On circuit functionality in Boolean networks. *Bull. Math. Biol.* 75, 906–919. doi:10.1007/s11538-013-9829-2

de Jong, H. (2002). Modeling and simulation of genetic regulatory systems: a literature review. *J. Comput. Biol.* 9, 67–103. doi:10.1089/10665270252833208

Devloo, V., Hansen, P., and Labbé, M. (2003). Identification of all steady states in large networks by logical analysis. *Bull. Math. Biol.* 65, 1025–1051. doi:10.1016/S0092-8240(03)00061-2

Dubrova, E., and Teslenko, M. (2011). A SAT-based algorithm for finding attractors in synchronous Boolean networks. *IEEE/ACM Trans. Comput. Biol. Bioinform.* 8, 1393–1399. doi:10.1109/TCBB.2010.20

Duhen, T., Duhen, R., Lanzavecchia, A., Sallusto, F., and Campbell, D. J. (2012). Functionally distinct subsets of human FOXP3+ Treg cells that phenotypically mirror effector Th cells. *Blood* 119, 4430–4440. doi:10.1182/blood-2011-11-392324

Fauré, A., Naldi, A., Chaouiya, C., and Thieffry, D. (2006). Dynamical analysis of a generic Boolean model for the control of the mammalian cell cycle. *Bioinformatics* 22, 124–131. doi:10.1093/bioinformatics/btl210

Garavel, H., Mateescu, R., Lang, F., and Serwe, W. (2007). "CADP 2006: a toolbox for the construction and analysis of distributed processes," in *Computer Aided Verification, Volume 4590 of Lecture Notes in Computer Science*, eds W. Damm and H. Hermanns (Berlin: Springer), 158–163.

Garg, A., Di Cara, A., Xenarios, I., Mendoza, L., and De Micheli, G. (2008). Synchronous versus asynchronous modeling of gene regulatory networks. *Bioinformatics* 24, 1917–1925. doi:10.1093/bioinformatics/btn336

Ghoreschi, K., Laurence, A., Yang, X. P., Tato, C. M., McGeachy, M. J., Konkel, J. E., et al. (2010). Generation of pathogenic TH17 cells in the absence of TGF-β signalling. *Nature* 467, 967–971. doi:10.1038/nature09447

Grieco, L., Calzone, L., Bernard-Pierrot, I., Radvanyi, F., Kahn-Perlès, B., and Thieffry, D. (2013). Integrative modelling of the influence of MAPK network on cancer cell fate decision. *PLoS Comput. Biol.* 9:21003286. doi:10.1371/journal.pcbi.1003286

Hegazy, A., Peine, M., Helmstetter, C., Panse, I., Fröhlich, A., Bergthaler, A., et al. (2010). Interferons direct Th2 cell reprogramming to generate a stable GATA-3(+)T-bet(+) cell subset with combined Th2 and Th1 cell functions. *Immunity* 32, 116–128. doi:10.1016/j.immuni.2009.12.004

Kam, T., Villa, T., Brayton, R. K., and Sangiovanni-Vincentelli, A. L. (1998). Multi-valued decision diagrams: theory and applications. *Multiple-Valued Logic* 4, 9–62.

Karlebach, G., and Shamir, R. (2008). Modelling and analysis of gene regulatory networks. *Nat. Rev. Mol. Cell Biol.* 9, 770–780. doi:10.1038/nrm2503

Kozen, D. (1983). Results on the propositional μ-calculus. *Theor. Comp. Sci.* 27, 333–354. doi:10.1016/0304-3975(82)90125-6

Lee, Y. K., Turner, H., Maynard, C. L., Oliver, J. R., Chen, D., Elson, C. O., et al. (2009). Late developmental plasticity in the T helper 17 lineage. *Immunity* 30, 92–107. doi:10.1016/j.immuni.2008.11.005

Lomuscio, A., Pecheur, C., and Raimondi, F. (2007). "Automatic verification of knowledge and time with NuSMV," in *International Joint Conference on Artificial Intelligence*, ed. M. M. Veloso (Hyderabad: Morgan Kaufmann Publishers Inc), 1384–1389.

Martinez-Sosa, P., and Mendoza, L. (2013). The regulatory network that controls the differentiation of t lymphocytes. *Biosystems* 2, 96–103. doi:10.1016/j.biosystems.2013.05.007

Mateescu, R., Monteiro, P. T., Dumas, E., and de Jong, H. (2011). CTRL: extension of CTL with regular expressions and fairness operators to verify genetic regulatory networks. *Theor. Comp. Sci.* 412, 2854–2883. doi:10.1016/j.tcs.2010.05.009

Mendoza, L. (2006). A network model for the control of the differentiation process in Th cells. *Biosystems* 84, 101–114. doi:10.1016/j.biosystems.2005.10.004

Mendoza, L., and Pardo, F. (2010). A robust model to describe the differentiation of T-helper cells. *Theory Biosci.* 129, 283–293. doi:10.1007/s12064-010-0112-x

Mendoza, L., and Xenarios, I. (2006). A method for the generation of standardized qualitative dynamical systems of regulatory networks. *Theor. Biol. Med. Model.* 3, 13. doi:10.1186/1742-4682-3-13

Monteiro, P. T., and Chaouiya, C. (2012). "Efficient verification for logical models of regulatory networks," in *Practical Applications on Computational Biology & Bioinformatics, Volume 154 of Advances in Intelligent and Soft Computing*, eds M. P. Rocha, N. Luscombe, F. Fdez-Riverola, and J. M. C. Rodríguez (Salamanca: Springer), 259–267.

Mosmann, T. R., Cherwinski, H., Bond, M. W., Giedlin, M. A., and Coffman, R. L. (1986). Two types of murine helper T cell clone. I. Definition according to profiles of lymphokine activities and secreted proteins. *J. Immunol.* 136, 2348–2357.

Mosmann, T. R., and Coffman, R. L. (1989). TH1 and TH2 cells: different patterns of lymphokine secretion lead to different functional properties. *Annu. Rev. Immunol.* 7, 145–173. doi:10.1146/annurev.iy.07.040189.001045

Murphy, K. M., and Reiner, S. L. (2002). The lineage decisions of helper T cells. *Nat. Rev. Immunol.* 2, 933–944. doi:10.1038/nri954

Nakayamada, S., Takahashi, H., Kanno, Y., and O'Shea, J. J. (2012). Helper T cell diversity and plasticity. *Curr. Opin. Immunol.* 24, 297–302. doi:10.1016/j.coi.2012.01.014

Naldi, A., Carneiro, J., Chaouiya, C., and Thieffry, D. (2010). Diversity and plasticity of Th cell types predicted from regulatory network modelling. *PLoS Comput. Biol.* 6:e1000912. doi:10.1371/journal.pcbi.1000912

Naldi, A., Monteiro, P. T., and Chaouiya, C. (2012). "Efficient handling of large signalling-regulatory networks by focusing on their core control," in *Computational Methods in Systems Biology, Volume 7605 of Lecture Notes in Computer Science*, eds D. Gilbert and M. Heiner (London: Springer), 288–306.

Naldi, A., Monteiro, P. T., Mussel, C., Kestler, H. A., Thieffry, D., Xenarios, I., et al. (2014). Cooperative development of logical modelling standards and tools with CoLoMoTo. *bioRxiv*. doi:10.1101/010504 Available at: http://biorxiv.org/content/early/2014/10/19/010504

Naldi, A., Remy, E., Thieffry, D., and Chaouiya, C. (2011). Dynamically consistent reduction of logical regulatory graphs. *Theor. Comp. Sci.* 412, 2207–2218. doi:10.1016/j.tcs.2010.10.021

Naldi, A., Thieffry, D., and Chaouiya, C. (2007). "Decision diagrams for the representation and analysis of logical models of genetic networks," in *Computational Methods in Systems Biology, Volume 4695 of Lecture Notes in Computer Science*, eds M. Calder and S. Gilmore (Edinburgh: Springer), 233–247.

O'Shea, J., and Paul, W. (2010). Mechanisms underlying lineage commitment and plasticity of helper CD4+ T cells. *Science* 327, 1098–1102. doi:10.1126/science.1178334

Peine, M., Rausch, S., Helmstetter, C., Fröhlich, A., Hegazy, A., Kühl, A., et al. (2013). Stable T-bet+GATA-3+ Th1/Th2 hybrid cells arise *in vivo*, can develop directly from naive precursors, and limit immunopathologic inflammation. *PLoS Biol.* 11:e1001633. doi:10.1371/journal.pbio.1001633

Remy, E., and Ruet, P. (2008). From minimal signed circuits to the dynamics of Boolean regulatory networks. *Bioinformatics* 24, i220–i226. doi:10.1093/bioinformatics/btn287

Saadatpour, A., Albert, R., and Reluga, T. (2013). A reduction method for Boolean network models proven to conserve attractors. *SIAM J. Appl. Dyn. Syst.* 12, 1997–2011. doi:10.1137/13090537X

Saez-Rodriguez, J., Simeoni, L., Lindquist, J. A., Hemenway, R., Bommhardt, U., Arndt, B., et al. (2007). A logical model provides insights into T cell receptor signaling. *PLoS Comput. Biol.* 3:e163. doi:10.1371/journal.pcbi.0030163

Samaga, R., and Klamt, S. (2013). Modeling approaches for qualitative and semi-quantitative analysis of cellular signaling networks. *Cell Commun. Signal.* 11, 43. doi:10.1186/1478-811X-11-43

Schwarick, M., and Heiner, M. (2009). "CSL model checking of biochemical networks with interval decision diagrams," in *Computational Methods in Systems Biology, Volume 5688 of Lecture Notes in Computer Science*, eds P. Degano and R. Gorrieri (Bologna: Springer), 296–312.

Siebert, H., and Bockmayr, A. (2006). "Incorporating time delays into the logical analysis of gene regulatory networks," in *Computational Methods in Systems Biology, Volume 4210 of Lecture Notes in Computer Science*, ed. C. Priami (Trento: Springer), 169–183.

Stoll, G., Viara, E., Barillot, E., and Calzone, L. (2012). Continuous time Boolean modeling for biological signaling: application of Gillespie algorithm. *BMC Syst. Biol.* 6:116. doi:10.1186/1752-0509-6-116

Thieffry, D. (2007). Dynamical roles of biological regulatory circuits. *Brief. Bioinformatics* 8, 220–225. doi:10.1093/bib/bbm028

Thomas, R., and D'Ari, R. (1990). *Biological Feedback*. Boca Raton: CRC Press.

van den Ham, H. J., and de Boer, R. J. (2008). From the two-dimensional Th1 and Th2 phenotypes to high-dimensional models for gene regulation. *Int. Immunol.* 20, 1269–1277. doi:10.1093/intimm/dxn093

van den Ham, H. J., and de Boer, R. J. (2012). Cell division curtails helper phenotype plasticity and expedites helper T-cell differentiation. *Immunol. Cell Biol.* 90, 860–868. doi:10.1038/icb.2012.23

Veliz-Cuba, A., Jarrah, A. S., and Laubenbacher, R. (2010). Polynomial algebra of discrete models in systems biology. *Bioinformatics* 26, 1637–1643. doi:10.1093/bioinformatics/btq240

Wang, R.-S., Saadatpour, A., and Albert, R. (2012). Boolean modeling in systems biology: an overview of methodology and applications. *Phys. Biol.* 9, 055001. doi:10.1088/1478-3975/9/5/055001

Wittmann, D. M., Krumsiek, J., Saez-Rodriguez, J., Lauffenburger, D. A., Klamt, S., and Theis, F. J. (2009). Transforming Boolean models to continuous models: methodology and application to T-cell receptor signaling. *BMC Syst. Biol.* 3:98. doi:10.1186/1752-0509-3-98

Xie, A., and Beerel, P. A. (1998). Efficient state classification of finite-state Markov chains. *IEEE Trans. Comput. Aided Des. Integrated Circ. Syst.* 17, 1334–1339. doi:10.1109/43.736573

Yang, X. O., Nurieva, R., Martinez, G. J., Kang, H. S., Chung, Y., Pappu, B. P., et al. (2008). Molecular antagonism and plasticity of regulatory and inflammatory T cell programs. *Immunity* 29, 44–56. doi:10.1016/j.immuni.2008.05.007

Zañudo, J. G. T., and Albert, R. (2013). An effective network reduction approach to find the dynamical repertoire of discrete dynamic networks. *Chaos* 23, 025111. doi:10.1063/1.4809777

Conflict of Interest Statement: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Received: 31 July 2014; accepted: 20 December 2014; published online: 28 January 2015.
Citation: Abou-Jaoudé W, Monteiro PT, Naldi A, Grandclaudon M, Soumelis V, Chaouiya C and Thieffry D (2015) Model checking to assess T-helper cell plasticity. Front. Bioeng. Biotechnol. ***2****:86. doi: 10.3389/fbioe.2014.00086*
This article was submitted to Bioinformatics and Computational Biology, a section of the journal Frontiers in Bioengineering and Biotechnology.

frontiers in Bioengineering and Biotechnology

ORIGINAL RESEARCH
published: 08 September 2015
doi: 10.3389/fbioe.2015.00130

Approximating attractors of Boolean networks by iterative CTL model checking

Hannes Klarner and Heike Siebert*

Fachbereich Mathematik und Informatik, Freie Universität Berlin, Berlin, Germany

This paper introduces the notion of approximating asynchronous attractors of Boolean networks by minimal trap spaces. We define three criteria for determining the quality of an approximation: "faithfulness" which requires that the oscillating variables of all attractors in a trap space correspond to their dimensions, "univocality" which requires that there is a unique attractor in each trap space, and "completeness" which requires that there are no attractors outside of a given set of trap spaces. Each is a reachability property for which we give equivalent model checking queries. Whereas faithfulness and univocality can be decided by model checking the corresponding subnetworks, the naive query for completeness must be evaluated on the full state space. Our main result is an alternative approach which is based on the iterative refinement of an initially poor approximation. The algorithm detects so-called autonomous sets in the interaction graph, variables that contain all their regulators, and considers their intersection and extension in order to perform model checking on the smallest possible state spaces. A benchmark, in which we apply the algorithm to 18 published Boolean networks, is given. In each case, the minimal trap spaces are faithful, univocal, and complete, which suggests that they are in general good approximations for the asymptotics of Boolean networks.

Keywords: Boolean networks, asynchronous dynamics, attractors, CTL model checking, ASP, signaling, gene regulation

OPEN ACCESS

Edited by:
David A. Rosenblueth, Universidad Nacional Autónoma de México, Mexico

Reviewed by:
Stalin Muñoz, Universidad Nacional Autónoma de México, Mexico
Sepinoud Azimi, Åbo Akademi University, Finland

****Correspondence:***
Hannes Klarner, Fachbereich Mathematik und Informatik, Freie Universität Berlin, Arnimallee 6, Berlin 14195, Germany
hannes.klarner@fu-berlin.de

Specialty section:
This article was submitted to Bioinformatics and Computational Biology, a section of the journal Frontiers in Bioengineering and Biotechnology

Received: *30 June 2015*
Accepted: *14 August 2015*
Published: *08 September 2015*

Citation:
Klarner H and Siebert H (2015) Approximating attractors of Boolean networks by iterative CTL model checking. Front. Bioeng. Biotechnol. 3:130. doi: 10.3389/fbioe.2015.00130

1. Introduction

Boolean and multi-valued networks are frequently used to model the dynamics of biological processes that involve gene regulation and signal transduction. The dynamics of such models is captured by the state transition graph, a directed graph that relates states to potential successor states. Different transition relations have been suggested, among them the synchronous update of Kauffman (1993) and the asynchronous update of Thomas (1991). An important type of prediction that can be obtained from such models concerns the long-term behavior of the represented processes. Formally, the long-term behaviors correspond to the minimal trap sets of the state transition graph which are also called its attractors.

Recently, we have suggested to compute the minimal trap spaces of a network to obtain an approximation for its cyclic attractors (Klarner et al., 2014) and proposed an efficient, *Answer Set Programing* (ASP)-based method for their computation. This paper presents an iterative algorithm that combines *Computation Tree Logic* (CTL) model checking with the computation of minimal trap spaces to determine the quality of the approximation.

The paper is organized as follows. Section 2 recapitulates the background including directed graphs, the dynamics of Boolean networks, trap spaces, and model checking. It is only meant to introduce the notation required for the subsequent sections. Section 3 briefly discusses the attractor detection problem. In Section 4, we describe three conditions under which there is a one-to-one correspondence between the minimal trap spaces and the attractors of a network, and how CTL queries may be used to decide whether they hold. The computationally most challenging one is treated in Section 5. In Section 6, we present a full analysis of a MAPK signaling network as well as the results for 18 Boolean models that are currently in the GINSIM repository. Section 7 is an outlook and conclusion. There is a Supplementary Material that contains proofs for the formal statements in the main text.

2. Background

2.1. Directed Graphs

Since several aspects of Boolean networks involve directed graphs (digraphs) we introduce the general terminology. Let (V, A) be a digraph with vertices V and arcs $A \subseteq V \times V$.

An **infinite path** in (V, A) is an infinite sequence of vertices $\pi = (v_0, v_1, \ldots)$ such that $(v_i, v_{i+1}) \in A$ for all $i \in \mathbb{N}_0$. **Finite paths** are defined analogously for finite sequences. In particular, $\pi = (v_0)$ is an admissible finite path. We denote the set of all infinite paths that start in $v \in V$ by $InfPaths(v)$ and finite paths by $FinPaths(v)$. The ith vertex of π is denoted by $\pi[i] := v_i$. For finite paths we denote by $FinPaths(u, v)$ all finite paths that start with u and end with v. The number of vertices in a finite path $\pi = (v_0, v_1, \ldots, v_k)$ is denoted by $len(\pi) := k + 1$.

A vertex $v \in V$ is **reachable** from $u \in V$ iff $FinPaths(u, v) \neq \varnothing$. We denote by $Above(v)$ the vertices that can reach v. A subset $U \subseteq V$ is **strongly connected** if every $u \in U$ is reachable from every other $v \in U$. A **strongly connected component** (SCC) is an inclusion-wise maximal subset $U \subseteq V$ that is strongly connected. We denote the set of SCCs of a digraph by $SCCs(V, A)$. Note that since $\pi = (v_0)$ is an admissible finite path, every vertex is trivially reachable from itself. Hence, each node belongs to some SCC and $SCCs(V, A)$ is a partition of V.

2.2. Boolean Networks

We consider variables from the Boolean domain $\mathbb{B} = \{0,1\}$, where 1 and 0 represent the truth values *true* and *false*. A Boolean **expression** f over the variables $V = \{v_1, \ldots, v_n\}$ is defined by a formula over the grammar

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

where $v \in V$ signifies a variable, $\bar{f}$ the negation, $f_1 \cdot f_2$ the conjunction and $f_1 + f_2$ the (inclusive) disjunction of the expressions f, f_1, and f_2. Given an **assignment** $x : V \to \mathbb{B}$, an expression f can be evaluated to a value $f(x) \in \mathbb{B}$ by substituting the values $x(v)$ for the variables $v \in V$. If $f(x) = f(y)$ for all assignments $x, y : V \to \mathbb{B}$, we say f is **constant** and write $f = c$, with $c \in \mathbb{B}$ being the constant value. A **Boolean network** (V, F) consists of n variables $V = \{v_1, \ldots, v_n\}$ and n corresponding Boolean expressions $F = \{f_1, \ldots, f_n\}$ over V. In this context, an assignment $x : V \to \mathbb{B}$ is also called a **state** of the network and the **state space** $S = S_V$ consists of all possible 2^n states. We specify states by a sequence of n values that correspond to the variables in the order given in V, i.e., $x = 110$ should be read as $x(v_1) = 1$, $x(v_2) = 1$, and $x(v_3) = 0$. The expressions F can be thought of as a function $F : S \to S$ governing the network behavior. The **image** $F(x)$ of a state x under F is defined to be the state y that satisfies $y(v_i) = f_i(x)$.

The **interaction graph** of a network (V, F) captures the dependencies between the variables and their expressions. It is a digraph $(V, \to)$ where $\to \subseteq V \times V$ and $(u, v) \in \to$ iff there are $x, y \in S$ such that $x(w) = y(w)$ for all $w \in V \setminus \{u\}$ and $f_v(x) \neq f_v(y)$. As for state transitions we write $u \to v$ iff $(u,v) \in \to$.

The **state transition graph** (STG) of a Boolean network (V, F) is the digraph $(S, \to)$ where the transitions $\to \subseteq S \times S$ are obtained from F via a given update rule. We usually write $x \to y$ iff $(x, y) \in \to$. We mention two update rules here, the **synchronous rule** and its transition relation $\twoheadrightarrow \subseteq S \times S$, and the **asynchronous rule** and its transition relation $\hookrightarrow \subseteq S \times S$. The former is defined by $x \twoheadrightarrow y$ iff $F(x) = y$. To define $\hookrightarrow$ we need the Hamming distance $\Delta : S \times S \to \{1, \ldots, n\}$ between states which is given by $\Delta(x, y) := |\{v \in V \mid x(v) \neq y(v)\}|$. We define $x \hookrightarrow y$ iff either $x = y$ and $F(x) = x$ or $\Delta(x,y) = 1$ and $\Delta(y, F(x)) < \Delta(x, F(x))$. In the context of the STG, the expressions $f \in F$ are also called **update functions**.

A non-empty set $T \subseteq S$ is a **trap set** of $(S, \to)$ iff for every $x \in T$ and $y \in S$ with $x \to y$ it holds that $y \in T$. An inclusion-wise minimal trap set is also called an **attractor** of $(S, \to)$. Every trap set contains at least one minimal trap set and therefore at least one attractor. A variable $v \in V$ is **steady** in an attractor $A \subseteq S$ iff $x(v) = y(v)$ for all $x, y \in A$ and **oscillating** otherwise. We distinguish two types of attractors depending on their size. If $A \subseteq S$ is an attractor and $|A| = 1$, then A is called a **steady state** and if $|A| > 1$, we call it a **cyclic attractor**. The cyclic attractors of $(S, \twoheadrightarrow)$ are, in general, different from the cyclic attractors of $(S, \hookrightarrow)$. The steady states, however, are identical in both transition graphs because $x \in S$ is steady iff $x \to x$ which is characterized, for both update rules, by the equation $F(x) = x$. Hence, we may omit the update rule and denote the set of steady states by S_F.

A **subspace** of S is characterized by its fixed and free variables. It may be specified by an assignment $p : D \to \mathbb{B}$ where $D \subseteq V$ is the subset of **fixed** variables, $p(u)$ the value of $u \in D$ and the remaining variables, $V \setminus D$, are said to be **free**. Subspaces are sometimes referred to as "symbolic states" (Siebert, 2011) or "partial states" (Irons, 2006). We specify subspaces like states but allow in addition the symbol $\star$ to indicate that a variable is free, i.e., $p = {\star\star}10$ means $D = \{v_3, v_4\}$ and $p(v_3) = 1$, $p(v_4) = 0$. The set $S^\star = S_V^\star$ denotes all possible 3^n subspaces. States are therefore a special kind of subspace and $S \subset S^\star$ holds. We denote the fixed variables D of a specific $p \in S^\star$ by D_p. A subspace p references the states $S[p] := \{x \in S \mid \forall v \in D_p : x(v) = p(v)\}$. We denote the unique subspace that does not fix any variables by $\epsilon \in S^\star$, i.e., $D_\epsilon = \varnothing$. Two subspaces $p, q \in S^\star$ are said to be **consistent** iff $p(v) = q(v)$ for all $v \in D_p \cap D_q$. We define the **intersection** $z := q \sqcap p$ of two consistent $p, q \in S^\star$ to be the unique $z \in S^\star$ that satisfies $S[z] = S[p] \cap S[q]$.

A **trap space** is a subspace that is also a trap set. Trap spaces are therefore trap sets with a particularly simple geometry. They

generalize the notion of steadiness from states to subspaces. In Klarner et al. (2014), we proved that trap spaces are independent of the update strategy. It is therefore meaningful to denote the trap spaces of $(S, \hookrightarrow)$ by $S_F^\star$ independent of $\to$. If a network (V, F) satisfies $S_F^\star = \{\epsilon\}$, then we say it is **trap-space-free**. We also showed that the dynamics inside a trap space p is fully specified by the **reduced network** (V_p, F_p) with

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

where $f[p]$ denotes the Boolean expression that is obtained by substituting the values $p(v)$ for $v \in D_p$ into $f \in F$, as introduced in Section 2.1 of Klarner et al. (2014).

Since every trap set contains at least one attractor, inclusion-wise minimal trap spaces can be used to predict the location of a particular attractor. Hence, we define a partial order on $S^\star$ based on whether the referenced subspaces are nested: $p \leq q$ iff $S[p] \subseteq S[q]$. The **minimal** trap spaces are defined by $\min(S_F^\star) := \{p \in S_F^\star \,|\, \nexists\, q \in S_F^\star : \; q < p\}$.

2.3. CTL Model Checking

Model checking is a formal method from computer science to determine whether a transition system satisfies a temporal specification. See Carrillo et al. (2012) for a review of its application to computational biology.

A **transition system** is a 5-tuple $TS = (S, \to, AP, L, I)$ where $(S, \to)$ is a state transition graph, AP a set of atomic propositions, $L : S \to 2^{AP}$ a labeling function and $I \subseteq S$ a set of initial states. We use the atomic propositions $AP := \{v = c, \delta_v = d \mid v \in V, c \in \mathbb{B}, d \in \{-1, 0, 1\}\}$ and define the labeling function L by

$$v = c \in L(x) :\Leftrightarrow x(v) = c$$
$$\delta_v = d \in L(x) :\Leftrightarrow f_v(x) - x(v) = d$$

for all $c \in \mathbb{B}$, $d \in \{-1, 0, 1\}$ and $x \in S$. The label $\delta_v = d$ therefore indicates whether a variable v is decreasing, steady or increasing in a state. In addition to "$=$" we need the inequality operator "$\neq$", e.g., $v \neq c \in L(x)$ iff $x(v) \neq c$, and the special atom *true* which satisfies $true \in L(x)$ for all $x \in$ S.

Next, we define a fragment of the temporal specification language CTL that is sufficient for the subsequent sections. A formula φ of this fragment is defined by

$$\varphi ::= a \;\Big|\; \varphi_1 \wedge \varphi_2 \;\Big|\; \varphi_1 \vee \varphi_2 \;\Big|\; \mathbf{EF}(\varphi) \;\Big|\; \mathbf{AG}(\varphi)$$

where $\alpha \in AP$, **EF** is the "exists finally" operator and **AG** the "always globally" operator. The semantics of the operators and the satisfaction relation $\models$ for transitions systems and CTL formulas is defined in **Table 1**. Since the atomic propositions and labeling function are fixed for the remainder of this article, we will specify transition systems by 3-tuples $TS = (S, \to, I)$. In practice, we use the model checking tool NUSMV (Cimatti et al., 2000) to decide whether a given transition system satisfies a CTL query.

TABLE 1 | The satisfaction relation $\models$ for CTL formulas φ, states $x \in S$, and transition systems $TS = (S, \to, AP, L, I)$.

$x \models a$	$:\Leftrightarrow$	$a \in L(x)$
$x \models \varphi_1 \wedge \varphi_2$	$:\Leftrightarrow$	$x \models \varphi_1$ and $x \models \varphi_2$
$x \models \varphi_1 \vee \varphi_2$	$:\Leftrightarrow$	$x \models \varphi_1$ or $x \models \varphi_2$
$x \models \mathbf{EF}(\varphi)$	$:\Leftrightarrow$	$\exists \pi \in InfPaths(x) : \exists i \in \mathbb{N}_0 : \pi[i] \models \varphi$
$x \models \mathbf{AG}(\varphi)$	$:\Leftrightarrow$	$\forall \pi \in InfPaths(x) : \forall i \in \mathbb{N}_0 : \pi[i] \models \varphi$
$TS \models \varphi$	$:\Leftrightarrow$	$\forall x \in I: x \models \varphi$

3. The Attractor Detection Problem

The naive approach to find all attractors of a given network, i.e., a full exploration of its STG, is limited by the state explosion problem. Several groups have developed tools and algorithms that address this problem. They may be grouped into those for deterministic updates (Irons, 2006; Dubrova and Teslenko, 2011; Akutsu et al., 2012; Veliz-Cuba et al., 2014) and non-deterministic updates (Garg et al., 2008; Skodawessely and Klemm, 2011; Berntenis and Ebeling, 2013). The average running times are usually given in terms of randomly generated networks and a connectivity parameter k that describes the distribution of in-degrees in the interaction graph. It seems that finding deterministic attractors is easier than non-deterministic attractors. Intuitively, computing the terminal SCCs of digraphs with all out-degrees equal to one is easier than for digraphs with higher out-degrees. The average running times for synchronous STGs with hundreds of variables is, for example, on the order of seconds with the tool BNS (Dubrova and Teslenko, 2011), which is based on a variant of *bounded linear time logic* (LTL) model checking and uses a *satisfiability* (SAT) solver to detect attractors.

Algorithms for non-deterministic STGs, on the other hand, are likely to run for hours or days for networks with less than even 100 variables (see Section 2). Garg et al. (2008) and the tool GENYSIS is based on the symbolic manipulation of reachable states using *binary decision diagrams* (BDDs), while Skodawessely and Klemm (2011) and Berntenis and Ebeling (2013) rely on a guided exploration and enumeration of the state space.

3.1. Attractor Detection Pre-Process

If $v \in V$ is a constant with $f_v = c$ and A an attractor, then $x(v) = c$ for every $x \in A$. Hence, before we start an attractor detection algorithm, we may safely remove all constants. The result is a reduced network whose attractors are in a one-to-one relationship with the attractors of the original network. During the removal of constants, update functions that depend on them may in turn become constant. The pre-process is therefore improved by an iterative substitution until there are no more constants.

The **percolation** operator $\vec{\bullet} : S_F^\star \to S_F^\star$ is defined on the set of trap spaces by the following recursion. Let p be the initial trap space, for example, defined by the constants $C \subseteq V$ of a network ($D_p := C$ and $p(v) := f_v$). The initial percolation is $\vec{p}_0 := p$ and for each $k \in \mathbb{N}_0$ we define $\vec{p}_{k+1}$ by

$$D_{\vec{p}_{k+1}} := \{v \in V \,|\, f_v[\vec{p}_k] \text{ is constant}\}$$
$$\vec{p}_{k+1}(v) := f_v[\vec{p}_k], \quad \text{for all } v \in D_{\vec{p}_{k+1}}.$$

Note that $f[p]$ denotes the Boolean expression obtained by substituting the values $p(v)$ into f, as introduced in Section 2.1 of Klarner et al. (2014). Because $\vec{p}_0 = p$ it follows that $\vec{p}_{k+1} \leq \vec{p}_k$ and $\vec{p}_k \in S_F^\star$, for all $k \in \mathbb{N}_0$. Since V is finite, there is some $K \in \mathbb{N}_0$ such that $\vec{p}_K = \vec{p}_{K+1}$ and $\vec{p} := \vec{p}_K$ is well-defined. Percolations

are cheap to compute and have the following implication for the location of attractors (see Siebert (2011)):

Proposition 1. If p is a trap space and $A \subseteq S[p]$ an attractor of $(S, \hookrightarrow)$, then $A \subseteq S[\vec{p}\,]$.

In the following sections, we will assume that the initial network is constant-free.

3.2. Attractor Detection by Random Walks

Given a trap space p, for example, the whole space $p = \epsilon$, we can find an attractor $A \subseteq S[p]$ by a sufficiently long random walk $(x_0, x_1, \ldots, x_k)$ where $x_0 \in S[p]$. In practice, we use $k = 10|V|$ and found that so far, without exception, random paths of this length have reached an attractor. To decide whether x_k does really belong to an attractor we use the CTL query of 2. It uses the CTL formula φ_p defined by $\varphi_p := \bigwedge_{v \in D_p} (v = p(v))$ if $p \neq \epsilon$, and $\varphi_p = ture$ otherwise.

Proposition 2 (Attractor State). Let p be a trap space and $x \in S[p]$. The state x belongs to an attractor $A \subseteq S[p]$ of $(S, \hookrightarrow)$ iff

$$TS = (S_{V_p}, \hookrightarrow, \{y\}) \models \mathbf{AG}(\mathbf{EF}(\varphi_y))$$

where $y \in S_{V_p}$ is the projection of $x \in S_V$ onto V_p, i.e., $y(v) := x(v)$ for all $v \in V_p$.

Starting from $x \in A$, we can then enumerate A by listing all states reachable from x. Note that model checking is performed on the reduced system $(S_{V_p}, \hookrightarrow)$ rather than the full system $(S, \hookrightarrow)$ and that there is no equivalent LTL query to decide whether x belongs to an attractor ($\mathbf{G}(\mathbf{F}(\varphi_y))$ does not work). Also, the observation that finding a single attractor is easy using a random walk does not contradict the fact that finding all attractors is hard.

4. Approximating Attractors by Subspaces

The result of attractor detection algorithms are usually sets of states that make up each attractor. The notion of an approximation of an attractor is instead based on information regarding steady and oscillating variables. An **approximation** of the attractors of a STG is a set $P \subseteq S^*$ such that each $S[p]$ contains an attractor. The trivial approximation for any network is $P := \{\epsilon\}$. Approximations differ in what can be learned from them about the number of attractors and their locations. The best approximation for a single attractor is the smallest subspace it is contained in. The **smallest subspace** that contains $A \subseteq S$ is $p \in S^*$ defined by $D_p := \{v \in V \mid \forall x, y \in A : x(v) = y(v)\}$ and $p(v) := x(v)$ for $x \in A$ arbitrary. We denote it by $Sub(A)$. Note that in general, $A \neq Sub(A)$ and that there may be two attractors $A, B \in S$ with $A \neq B$ such that $Sub(A) = Sub(B)$. The quality of an approximation is defined in terms of the following criteria.

Definition 1. A subspace p is **faithful** in $(S, \hookrightarrow)$ iff $Sub(A) = p$ for every attractor $A \subseteq S[p]$ of $(S, \hookrightarrow)$. An approximation P is faithful iff each $p \in P$ is faithful.

Definition 2. A subspace p is **univocal** in $(S, \hookrightarrow)$ iff there is a unique attractor A of $(S, \hookrightarrow)$ such that $A \subseteq S[p]$. An approximation P is univocal iff each $p \in P$ is univocal.

Definition 3. An approximation P is **complete** in $(S, \hookrightarrow)$ iff for every attractor $A \subseteq S$ of $(S, \hookrightarrow)$ there is $p \in P$ such that $A \subseteq S[p]$.

Note that the three properties are independent of each other. If P is faithful, univocal, and complete, then we call it a **perfect approximation**. If P is perfect, then all attractors can be found by the random walk method above.

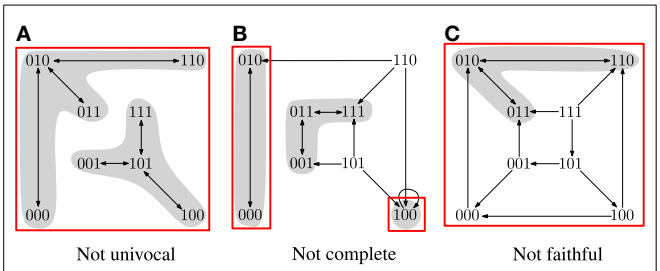


FIGURE 1 | The asynchronous STGs of three different Boolean networks. The minimal trap spaces are indicated by boxes. **(A)** Two attractors in the same box. **(B)** An attractor outside of the boxes. **(C)** An attractor that does not oscillate in all dimensions of the box. Equations for the networks are given in the Supplementary Material.

In Klarner et al. (2014), we observed that $\min(S_F^\star)$ is a good candidate for a perfect approximation. We showed that steady states are minimal trap spaces ($S_F \subseteq \min(S_F^\star)$) and that every $p \in \min(S_F^\star) \setminus S_F$ contains only cyclic attractors. Given that $\min(S_F^\star)$ can be computed efficiently using ASP, we would like to have an efficient method for determining its quality as an approximation. **Figure 1** demonstrates that $\min(S_F^\star)$ is, in general, neither univocal, complete nor faithful.

4.1. Univocality

Proposition 3 (Univocality). Let p be a trap space and $x \in A$ such that $A \subseteq S[p]$ is an attractor of $(S, \hookrightarrow)$. p is univocal in $(S, \hookrightarrow)$ iff

$$TS = (S_{V_p}, \hookrightarrow, S_{V_p}) \models \mathbf{EF}(\varphi_y)$$

where $y \in S_{V_p}$ is the projection of $x \in S_v$ onto V_p.

The intuition behind this proposition is that if A is the only attractor inside the trap space p then x must be reachable from all states S_{V_p}.

4.2. Faithfulness

Proposition 4 (Faithfulness). A trap space p is faithful in $(S, \hookrightarrow)$ iff

$$TS = (S_{V_p}, \hookrightarrow, S_{V_p}) \models \bigwedge_{v \in V_p} \mathbf{EF}(\delta_v \neq 0).$$

This proposition is true because a variable v oscillates in an attractor A iff there is a state $x \in A$ such that $x \models (\delta_v \neq 0)$.

4.3. Completeness

Proposition 5 (Completeness). A set of trap spaces p is complete in $(S, \hookrightarrow)$ iff

$$TS = (S, \hookrightarrow, S) \models \bigvee_{p \in P} \mathbf{EF}(\varphi_p).$$

Although we may restrict the initial states to $S \setminus \cup_{p \in P} S[p]$, the completeness query is still essentially dealing with the whole transition system and is therefore much less efficient than the queries of Proposition 2–4 (which are decided on reduced systems). In Klarner (2015b), we benchmarked NUSMV and found that Boolean networks with $n \approx 39$–55 variables may be considered infeasible for queries of this type. The next section develops a refinement-based approach to decide completeness that can deal with much larger networks.

5. Deciding Completeness by Iterative Refinement

The central idea for the refinement-based approach is to exploit hierarchies in the interaction graph and to use model checking on subnetworks that are in the upper layers of the hierarchy rather than the whole network. Given a complete set of trap spaces p, we keep replacing each $p \in P$ by smaller trap spaces until either $P = \min(S_F^\star)$ and we declare victory, or we find some $p \in P$ that satisfies the failure criterion below which implies that $\min(S_F^\star)$ can not be complete.

Proposition 6 (Refinement). Let $P \subseteq S_F^\star$ be complete in $(S, \hookrightarrow)$ and $p \in P$ some trap space. If $Q \subseteq S_{F_p}^\star$ is complete in $(S_{V_p}, \hookrightarrow)$ then $P' := (P \setminus \{p\}) \cup \{p \sqcap q \mid q \in Q\}$ is complete in $(S, \hookrightarrow)$.

Note that the intersection $p \sqcap q$ is necessary to position the trap space q of $(S_{V_p}, \hookrightarrow)$ correctly in the full transition system $(S_V, \hookrightarrow)$ and that $(p \sqcap q) \leq p$. An example of a refinement is the percolation operator. By Proposition 1, if P is complete, then $\vec{P} := \{\vec{p} \mid p \in P\}$ is also complete. The **failure criterion** is based on the observation that if $\min(S_F^\star)$ is complete in $(S_{V_p}, \hookrightarrow)$, then $\min(S_{F_p}^\star)$ must be complete in $(S_{V_p}, \hookrightarrow)$ for every $p \in S_F^\star$.

Proposition 7 (Failure Criterion). If there is a trap space p such that $\min(S_{F_p}^\star)$ is not complete in $(S_{V_p}, \hookrightarrow)$, then $\min(S_F^\star)$ is not complete in $(S, \hookrightarrow)$.

Example 1. Consider the network defined by $V = \{v_1, v_2, v_3\}$ and F with $f_1 = \overline{v_1}\,\overline{v_2} + v_1 v_2 + v_2 \overline{v_3}$, $f_2 = \overline{v_1}\,\overline{v_2} v_3 + v_1 v_2 v_3$ and $f_3 = v_2 + v_3$. The minimal trap spaces are {111,*00}. The trap space $p := {}^{**}1$ satisfies the failure criterion because $\min(S_{F_p}^\star) = \{11\}$ is not complete in $(S_{V_p}, \hookrightarrow)$ as there is, for example, no path from 01 to 11 in $(S_{V_p}, \hookrightarrow)$. It follows that $\min(S_F^\star)$ is not complete.

5.1. Autonomous Sets

To find the initial $P \subseteq S_F^\star$ and then $Q \subseteq \min(S_{F_p}^\star)$ for a given $p \in P$ we use Proposition 8 below. It is based on so-called autonomous sets, a generalization of inputs. The variables $U \subseteq V$ are **autonomous** iff $Above(U) = U$ in the interaction graph. An autonomous U induces a **restricted network** $(U, F_{|U})$ where $F_{|U} := \{f_u \in F \mid u \in U\}$. Note that if U is autonomous, then $(U, F_{|U})$ is a well-defined network.

Proposition 8. Let U be autonomous and $Q := \min(S_{F_{|U}}^\star)$ the minimal trap spaces of the restriction $(U, F_{|U})$.

(a) If Q is complete in $(S_U, \hookrightarrow)$, then Q is also complete in $(S, \hookrightarrow)$.
(b) If Q is not complete in $(S_U, \hookrightarrow)$, then $\min(S_F^\star)$ is not complete in $(S, \hookrightarrow)$.

Note that the inputs $I \subseteq V$ of a network are autonomous and that P defined by $P := \{p \in S_F^\star \mid D_p = I\}$ (the $|P| = 2^{|I|}$ input combinations) is complete in $(I, F_{|I})$. Proposition 8(a) implies that P is also complete in (V, F). $\vec{P}$ is a refinement of P and if any $\vec{p} \in \vec{P}$ satisfies the failure criterion then $\min(S_F^\star)$ is not complete.

Example 2. Consider the network with $V = \{v_1, \ldots, v_4\}$ and F with $f_1 = v_1$, $f_2 = v_2$, $f_3 = v_1 \overline{v_4}$, $f_4 = v_2 v_3$. The minimal trap spaces are {0000, 0100, 1000, 11**}. To decide whether they are complete we observe that the network has two inputs $\{v_1, v_2\}$ and four input combinations whose minimal trap spaces are $P = \{00^{**}, 01^{**}, 10^{**}; 11^{**}\}$. Since $\vec{P} = \min(S_F^\star) = \{0000, 0100, 1000, 11^{**}\}$, we deduce that $\min(S_F^\star)$ is complete.

5.2. Minimal Autonomous Sets

A refinement-based algorithm requires choosing an autonomous set U and deciding whether Q is complete in $(S_U, \hookrightarrow)$ using the query of Proposition 5. The best performance in terms of model checking is expected if the minimal sets are as small as possible. **Minimal autonomous sets** (set-inclusion-wise) are located in the top layer of the interaction graph $(V, \rightarrow)$ and can be found using any SCC algorithm.

Proposition 9. Let $U \subseteq V$. The following statements are equivalent:

(a) U is a minimal autonomous set of $(V, \rightarrow)$.
(b) U is autonomous and $U \in SCCs(V, \rightarrow)$.

Once it is confirmed that the minimal trap spaces of each restriction are complete, we may consider their **intersection**.

Proposition 10. If $P, Q \subseteq S_F^\star$ are complete in $(S, \hookrightarrow)$ then $P \sqcap Q := \{p \sqcap q \mid p \in P, q \in Q : p \text{ and } q \text{ are consistent}\}$ is also complete in $(S, \hookrightarrow)$.

Note that if P and Q are complete, then for each $p \in P$, there is necessarily a $q \in Q$ such that p and q are consistent. Similarly, for each attractor $A \subseteq S[p]$, there is some consistent $q \in Q$ such that $A \subseteq p \sqcap q$. Hence $P \sqcap Q$ is non-empty and complete. Also, unless there is $p \in P$ with $p \in Q$ we get $|P \sqcap Q| = |P| \cdot |Q|$. Finally, inputs are minimal autonomous sets and if a network has no other minimal autonomous sets, then the intersection is equal to the input combinations. Taking the intersection therefore generalizes the approach of inputs and input combinations.

Example 3. Consider the network with $V = \{v_1, \ldots, v_6\}$ and F with $f_1 = v_2$, $f_2 = v_1$, $f_3 = v_4$, $f_4 = v_3$, $f_5 = v_2 \overline{v_6}$ and $f_6 = v_3 v_5$. The minimal trap spaces are $\min(S_F^\star) = \{000000, 001100, 110010, 1111^{**}\}$. The network has two minimal autonomous sets $U_1 = \{v_1, v_2\}$ and $U_2 = \{v_3, v_4\}$. The corresponding restrictions are $(U_1, F_{|U_1})$ and $(U_2, F_{|U_2})$ with the minimal trap spaces $Q_1 := \min(S_{F_{|U_1}}^\star) = \{11, 00\}$ and $Q_2 := \min(S_{F_{|U_2}}^\star) = \{11, 00\}$. Model checking (or inspection of the STGs) confirms that they are complete in their respective restricted systems. The intersection $P := Q_1 \sqcap Q_2$ and the percolation $\vec{P}$ are $P = \{0000^{**}, 0011^{**}, 1100^{**}, 1111^{**}\}$ and $\vec{P} = \{000000, 001100, 110000, 1111^{**}\}$. As before in Example 2, $\vec{P} = \min(S_F^\star)$ and we deduce that $\min(S_F^\star)$ must be complete in $(S, \hookrightarrow)$.

5.3. Extending Minimal Autonomous Sets

Although minimal autonomous sets are favorable for efficient model checking, there is no guarantee that the respective restricted systems do actually contain non-trivial trap spaces. A refinement based on the trivial trap space ϵ, i.e., $Q = \{\epsilon\}$, is useless because it means replacing p with $p \sqcap \epsilon = p$, that is, with itself. A possible solution is to increase the size of checked autonomous sets until we find non-trivial trap spaces. The question is: by how many variables should we extend an autonomous set U? On the one hand, we want to be generous because new variables increase the chances for finding new trap spaces. On the other hand, we want to add as few variables as possible because the failure criterion requires CTL model checking.

What is the best extension for a given U whose restricted system is trap-space-free? Adding only outputs or cascades to U is not

enough as the emergence of trap spaces requires "self-freezing", positive feedback circuits, see Section 4.7 in Klarner (2015b). Intuitively, we want to extend down to the next SCC.

For a clean definition, we introduce the following notions. The set of **cascade components** consists of all single element SCCs in the interaction graph, whose nodes do not have self-loops. The remaining components are the **non-cascade components**.

$$Casc(V, \to) := \{U \in SCCs(V, \to) \mid \exists v \in V : U = \{v\}, v \not\to v\}$$
$$NonCasc(V, \to) := SCCs(V, \to) \setminus Casc(V, \to)$$

The **condensation graph** $(Z, \rhd)$ of the interaction graph is then the digraph with vertices $Z := NonCasc(V, \to)$ such that an arc $U \rhd W$ indicates whether there is a cascade from U to W. More precisely, $U \rhd W$ iff $U \neq W$ and there is $u \in U$, $w \in W$ such that

$$\exists \pi \in FinPaths(u, w) : \forall 1 \leq i \leq len(\pi) - 2 : \{\pi[i]\} \in Casc(V, \to).$$

Note that $(Z, \rhd)$ is acyclic and so we can partition its vertex set into classes, which we call **layers**, depending on the longest path that reaches them.

$$Lay(W) := \max\{len(\pi) \mid \pi \in FinPaths(U, W), U \in Z\}$$

Note that $Lay(W) \geq 1$ because $\pi = (W)$ is an admissible path from W to W and $len(W) = 1$ and that all minimal autonomous sets can then be found in the first layer of the condensation graph, i.e., $U \subseteq V$ is minimal and autonomous iff $U \in Z$ and $Lay(U) = 1$.

To illustrate how the condensation graph is used for extending autonomous sets, consider the network given in **Figure 2**. First, we compute its minimal autonomous sets, i.e., the top layer of $(Z, \rhd)$. In this example, there is a unique $W \in Z$ with $Lay(W) = 1$. The restriction $(W, F_{|W})$ consists of an isolated negative feedback circuit and is trap-space-free. To determine the smallest extension that contains new feedback circuits, we first compute the graph $(Z', \rhd)$, which is obtained from the condensation graph $(Z, \rhd)$ by removing all $U \in Z$ that satisfy $U \cap W \neq \varnothing$. For each $Y \in Z'$ that satisfies $Lay(Y) = 1$, we get an extended autonomous set W' by considering the variables above Y in the interaction graph $(V, \to)$. In the example, there is again a unique Y and the restriction to $W' := Above(Y)$ contains a non-trivial trap space p. The failure criterion is not satisfied by p and so we have found an initial complete set, namely $P := \{p\}$. Note that in general, there will be several minimal autonomous sets and several possible extensions. We are now ready to design an efficient algorithm for deciding completeness.

5.4. The Algorithm

The first step of the algorithm in **Figure 3** is to compute the minimal trap spaces of a given network using the ASP-based method proposed in Klarner et al. (2014). If the network is trap-space-free, then $\min(S_F^\star) = \{\epsilon\}$ is, by definition, complete and we stop and return *true*. Otherwise the variable *CurrentSet* is initialized. It consists of tuples (p, W), where p is a trap space and $W \subseteq V$ are the variables of the network (V_p, F_p) that have previously been subjected to model checking. The tuples correspond to those trap spaces of a complete set that need further refinement (i.e., are not minimal). Initially $CurrentSet := \{(\epsilon, \varnothing)\}$ because $\{\epsilon\}$ is trivially complete and we have not started model checking $W = \varnothing$. The lines 5–24 execute the iterative refinement of *CurrentSet* until we either find a p that satisfies the failure criterion in line 17 or $CurrentSet = \varnothing$ in which case every p is equal to some minimal trap space (only non-minimal trap spaces are put back onto *CurrentSet*, see lines 23, 24).

The next steps are to select an arbitrary (p, W) for refinement (line 6), compute the reduced network (V_p, F_p), its condensation graph $(Z, \rhd)$ and the graph $(Z', \rhd)$ described in the previous section. The top layer elements U of $(Z', \rhd)$ are minimal autonomous sets if $Z = Z'$ or extended autonomous sets if $Z \neq Z'$. In the latter case, the restricted networks that correspond to minimal autonomous sets of (V_p, F_p) must have previously been found to be trap-space-free. For each U, the variables above U are autonomous (in (V_p, F_p)). If the minimal trap spaces of the restricted networks are complete in $(S_{U'}, \hookrightarrow)$ then, by Proposition 8(a), they are also complete in $(S_{V_p}, \hookrightarrow)$. Otherwise it follows, by Proposition 8(b), that $\min(S_{F_p}^\star)$ is not complete in $(S_{V_p}, \hookrightarrow)$ and hence that p satisfies the failure criterion and we stop and return *false* in line 18.

The variable *Refinement* stores all complete sets that were found in the upper layers of (V_p, F_p), while W' keeps track of the variables

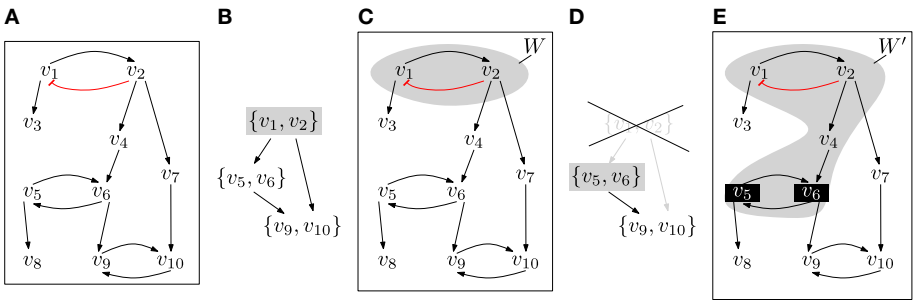


FIGURE 2 | (A) The interaction graph of an example network where each f_i is the disjunction of its inputs (e.g., $f_6 = v_5 + v_4$) except for v_1 which is inhibited by v_2 ($f_1 := \overline{v_2}$). **(B)** The condensation graph $(Z, \rhd)$ with the unique top layer node $\{v_1\ v_2\}$. **(C)** The corresponding minimal autonomous set W. The restriction $(W, F_{|W})$ is trap-space-free. **(D)** To extend W, we compute the graph $(Z', \rhd)$, which is obtained from $(Z, \rhd)$ by removing the node that intersects W. The new top layer node is $Y := \{v_5, v_6\}$. **(E)** The extended autonomous set W' is obtained by considering $Above(Y)$ in the interaction graph. It has a minimal trap space p that is defined by fixing v_5 and v_6 to 1.

Algorithm 1 (V, F)

Input: (V, F) is a constant-free Boolean network
Output: **true** if $\min(S^\star_F)$ is complete, **false** otherwise

1: $MinTrapSpaces \leftarrow$ MINTRAPSPACES(V, F)
2: **if** $MinTrapSpaces = \{\epsilon\}$ **then**
3: **return true**
4: $CurrentSet \leftarrow \{(\epsilon, \varnothing)\}$
5: **while not** $CurrentSet = \varnothing$ **do**
6: $(p, W) \leftarrow CurrentSet$.POP()
7: $(V_p, F_p) \leftarrow$ REDUCEDNETWORK(V, F, p)
8: $(Z, \rhd) \leftarrow$ CONDENSATIONGRAPH(V_p, F_p)
9: $(Z', \rhd) \leftarrow$ REMOVECOMPONENTS$(Z, \rhd, W)$
10: $W' \leftarrow$ COPY(W)
11: $Refinement \leftarrow \varnothing$
12: **for** $U \in$ TOPLAYER$(Z', \rhd)$ **do**
13: $U' \leftarrow$ ABOVE(V_p, F_p, U)
14: $(U', F_{|U'}) \leftarrow$ RESTRICTION(V_p, F_p, U')
15: $Q \leftarrow$ MINTRAPSPACES$(U', F_{|U'})$
16: $\varphi \leftarrow$ COMPLETENESSQUERY(Q)
17: **if not** CTLMODELCHECKING$(S'_U, \hookrightarrow, \varphi)$ **then**
18: **return false**
19: $Refinement$.APPEND(INTERSECTION(p, Q))
20: $W' \leftarrow$ SETUNION(W', U')
21: **for** $q \in$ INTERSECTION$(Refinement)$ **do**
22: $\vec{q} \leftarrow$ PERCOLATION(V, F, q)
23: **if not** $\vec{q} \in MinTrapSpaces$ **then**
24: $CurrentSet$.APPEND$(\vec{q}, W')$
25: **return true**

FIGURE 3 | The iterative, refinement-based algorithm for deciding the question of completeness. See main text for a detailed description.

that were subjected to model checking. Line 21 is an application of Proposition 10, i.e., the intersection of all complete sets is taken (generalization of input combinations). For each trap space q in the intersection, we check whether the percolation $\vec{q}$ needs further refinement (not a minimal trap space of (V, F)) and if so add it back onto *CurrentSet*.

Note that (V_p, F_p) must have non-trivial trap spaces for each $(p, W) \in$ *CompleteSets* (see lines 23, 24). Hence, although it may happen that (p, W) is replaced by (p, W') (if $Q = \{\epsilon\}$ in line 15) eventually it will be replaced by smaller trap spaces. The algorithm is implemented and available as part of our PYTHON toolbox for Boolean networks (Klarner, 2015a).

5.5. Counterexamples for Attractor Detection

If $\min(S^\star_F)$ is not a perfect approximation, we would like to know why. Model checking tools like NUSMV are capable of producing a counterexample in case a formula does not hold. Intuitively, a counterexample is a finite path from an initial state that proves that the query is false. If $\min(S^\star_F)$ is not complete, then the algorithm of the previous section can be used to return some $p \in S^\star_F$ that satisfies the failure criterion together with a counterexample to the respective completeness query for (V_p, F_p) and $\min(S^*_{F_p})$. Every attractor that is reachable from its last state, say x, must then be outside of $\min(S^*_{F_p})$. We then use the random walk approach to find state a z that belongs to an attractor $A \subseteq S_{V_p}$ outside of $\min(S^*_{F_p})$. If the modified completeness query

$$TS = (S_{V_p}, \hookrightarrow, S_{V_p}) \models \varphi_z \vee \bigvee_{q \in \min(S^\star_{F_p})} \mathbf{EF}(\varphi_q)$$

holds then A is the only outside attractor, otherwise we use the next counterexample to find the next outside attractor until they are all found. Note that p is an extension of a minimal autonomous set. A similar approach is possible for trap spaces that are not faithful or not univocal. We end up with a set of states that captures the attractors outside of P, the number of attractors inside $S[p]$ for each $p \in P$ and whether they are faithful or not.

6. Results

All computations in this section were done on a 32-bit Linux laptop with 4×2.60 GHz and 8 GB memory.

6.1. MAPK Case Study

In this case study, we consider the network published in Grieco et al. (2013), which models the influence of the MAPK pathway on cancer cell fate decisions and consists of 53 variables. Using Klarner (2015a), we compute $\min(S^\star_F)$ in under one second. It consists of 12 steady states and six trap spaces that contain only cyclic attractors. The single query approach to deciding completeness runs 35 min, while the refinement-based algorithm confirms completeness in only 28 s. For the six trap spaces in $\min(S^\star_F) \setminus S_F$ we confirmed univocality in 261 s (44 s on average per trap space) and faithfulness in 74 s (12 s on average per trap space) using the CTL queries of Section 4. Hence, $\min(S^\star_F)$ is a perfect approximation of the attractors of $(S, \hookrightarrow)$ and for each attractor we can find an internal state by the random walk approach of Section 4. We stopped GENYSIS after seven hours without a result.

Figure 4 is an illustration of the steps performed during the iterative refinement for the MAPK network. The information is represented as a **decision tree**. The root represents the initial and trivially complete set $P := \{\epsilon\}$. Boxes are split into a left side, representing the size $|U|$ of a minimal autonomous set (or an extension), and a right hand side that is split vertically into cells that contain the numbers $|D_q|$ of fixed variables for each minimal trap space q of $(U, F_{|U})$. Boxes are colored according to whether $(U, F_{|U})$ is trap-space-free (white) or not in which case model checking is required to find out whether the minimal trap spaces of $(U, F_{|U})$ are complete (failure criterion). Boxes with more than one minimal trap space are outlined in red to emphasize that a decision process between competing trap spaces exists. The intersection of several autonomous sets is indicated by $\otimes$ but occurs for this network only for the inputs. Arcs are labeled by the number of variables that are fixed during percolations, i.e., $\left|D_q \setminus D_{\vec{q}}\right|$ (see line 22 in **Figure 3**). If a restricted network is trap-space-free, the extension is indicated by a dashed arc. Along each branch of the decision tree, the number of fixed and oscillating variables must add up to 53. The bottom branch, for example, starts with four fixed variables, percolates seven more, extends an autonomous set whose restriction consists of four variables and is trap-space-free, finds a single trap space with three fixed variables and finishes as the remaining trap space is minimal (and $4 + 7 + 3 + 0 + 39 = 53$).

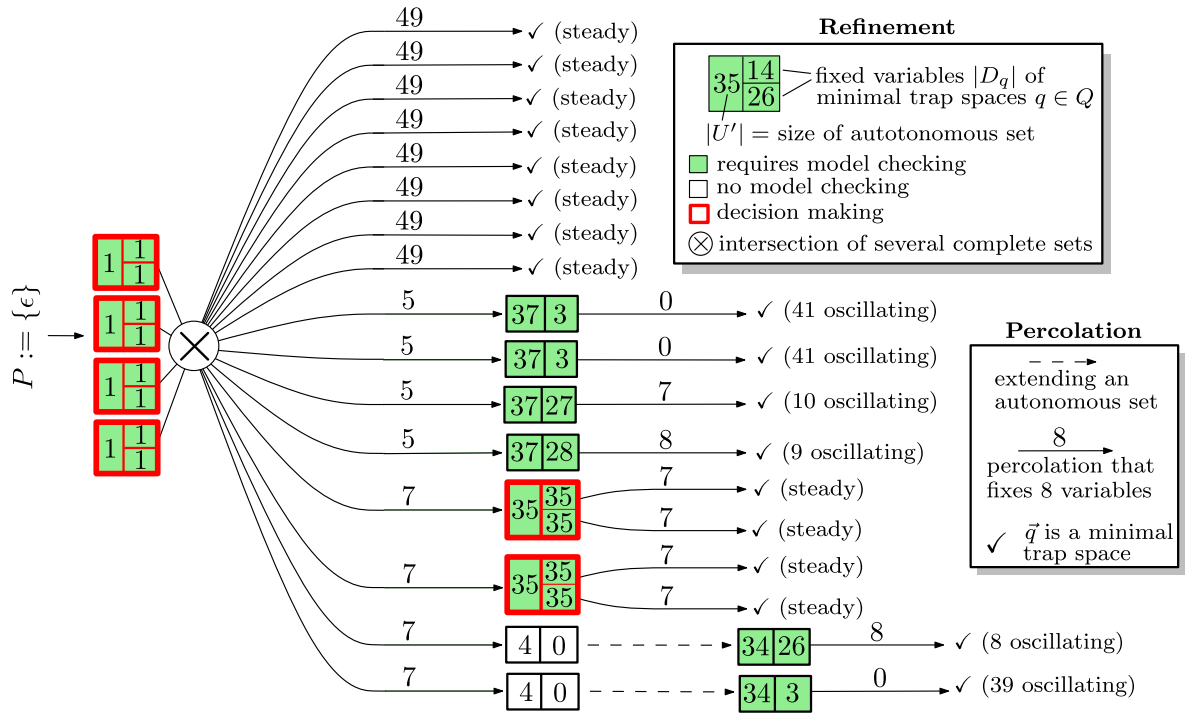


FIGURE 4 | An illustration of the how the iterative refinement algorithm confirms that $\min(S_F^\star)$ of the MAPK network is complete. Altogether 12 restricted systems are model checked instead of the single full system. A more detailed description is given in the main text.

TABLE 2 | The minimal trap spaces of all Boolean models in the GINSIM repository are perfect approximations of the attractors of $(S, \hookrightarrow)$.

Network file (.zginml)	$\|V\|$	Steady	Cyclic	Faithful	Univocal	Complete
buddingYeastOrlando2008	9	1	–	0.08*s*	0.03*s*	0.23*s*
fissionYeastDavidich2008	10	12	–	0.01*s*	0.02*s*	0.08*s*
boolean_cell_cycle	10	1	1	0.03*s*	0.19*s*	0.12*s*
Toll_Pathway_12Jun2013	11	4	–	0.01*s*	0.01*s*	0.09*s*
drosophilaCellCycleVariants	14	1	–	0.01*s*	0.05*s*	0.11*s*
MAPK_red3_19062013	16	12	6	0.15*s*	1.11*s*	0.93*s*
MAPK_red1_19062013	17	12	6	0.18*s*	1.25*s*	0.87*s*
VEGF_Pathway_12Jun2013_0	18	256	–	0.04*s*	0.05*s*	0.28*s*
MAPK_red2_19062013	18	12	6	0.14*s*	1.26*s*	0.67*s*
buddingYeastIrons2009	18	–	1	0.16*s*	0.48*s*	0.02*s*
ErbB2_model	20	1	–	0.08*s*	0.00*s*	0.02*s*
FGF_Pathway_12Jun2013	23	512	–	0.09*s*	0.09*s*	0.51*s*
Hh—Pathway_11Jun2013_0	24	8192	–	1.29*s*	1.43*s*	6.34*s*
Spz—Processing_12Jun2013	24	64	–	0.04*s*	0.03*s*	0.22*s*
Wg_Pathway_11Jun2013	26	16384	–	2.38*s*	2.38*s*	17.16*s*
TCRsig40	40	7	1	1.07*s*	3.34*s*	0.12*s*
MAPK_large_19june2013	53	12	6	40.15*s*	565.84*s*	20.72*s*
T_LGL	60	86	70	1.07*s*	6.57*s*	5669.57*s*

The number of variables, steady states, and cyclic attractors are recorded in the first three columns. The remaining three columns record the time needed to confirm faithfulness, univocality, and completeness.

Note that the algorithm encounters roughly four types of refinements. The first type (branches 1–8) leads directly to a steady state. The second type (branches 9–12) discovers a single minimal autonomous set consisting of 37 variables, whose restriction has a single minimal trap spaces in which between nine and 41 variables oscillate. The third type (branches 13–14) discovers a single minimal autonomous set that has two minimal trap spaces that commit the network to different steady states. The fourth type (branches 15–16) discovers a single minimal autonomous set consisting of four variables that is trap-space-free. An extension leads an autonomous set of 34 variables with a single minimal trap space.

6.2. GINsim Repository Benchmark

To test whether the MAPK network is unusual in that its minimal trap spaces are perfect approximations, we ran the same analysis for every Boolean model currently in the GINSIM model repository (see Naldi et al. (2009)). In every case, the minimal trap spaces are perfect approximations of the attractors of $(S, \hookrightarrow)$. The time

needed to confirm faithfulness, univocality and completeness is given in **Table 2**. We confirmed the number of steady states and cyclic attractors with GINSIM and GENYSIS. The execution of GINSIM is, like the computation of minimal trap spaces, instantaneous. The running times of GENYSIS are comparable to that of our algorithm for networks with $|\mathrm{V}| < 40$ and on the order of 24–72 h for the three networks with $|V| \geq 40$. The networks and attractors are available for benchmarking at Klarner (2015a).

7. Conclusion and Outlook

In this paper, we developed the notion of an approximation of attractors of a Boolean network. Minimal trap spaces are approximations that can be computed for networks with hundreds of variables using ASP solvers. Since available attractor detection tools for asynchronous systems are only feasible for about 50 variables, approximations via minimal trap spaces might yield attractor information otherwise inaccessible. We defined three criteria to assess the quality of an approximation and showed that they can be decided using model checking. The main contribution in this paper is an algorithm that improves the efficiency of deciding completeness by dividing the problem into smaller subproblems according to autonomous sets in the interaction graph.

We ran the algorithm on the 18 Boolean networks that are currently in the GINSIM repository and found that each time, the minimal trap spaces are a perfect approximation of the asynchronous attractors, i.e., that we can find all asynchronous attractors using random walks and $\min(S_F^\star)$.

Section 5.3 explains that autonomous sets must be extended if the corresponding restricted systems are trap-space-free. Strategies by which extensions are constructed must compromise between adding variables to increase the likelihood of discovering non-trivial trap spaces and the efficiency of model checking the respective transition systems. The strategy in Section 5.3 can be considered optimal in the sense that it adds as few variables at a time as necessary for the emergence of new trap spaces.

There are several directions in which the algorithm may be improved further, for example, by removing so-called "mediator variables" (see, e.g., Saadatpour et al. (2013)) from the interaction graph of the subnetworks. The relationship to other reduction methods, e.g., Naldi et al. (2011) or Veliz-Cuba (2011), may also yield improvements by reducing the size of the transition systems passed to the model checking software further.

The decision tree in **Figure 4** might be an interesting tool for questions regarding network control, an idea that was recently developed in Zañudo and Albert (2015). It also suggests that the dynamics of Boolean networks is governed by two very different regimes: the *percolation regime* in which the long-term activities are pre-determined, and the *decision-making regime* in which the long-term activities are determined by which of the competing trap spaces is reached first.

Supplementary Material

The Supplementary Material for this article can be found online at http://journal.frontiersin.org/article/10.3389/fbioe.2015.00130

References

Akutsu, T., Yang, Z., Hayashida, M., and Tamura, T. (2012). Integer programming-based approach to attractor detection and control of boolean networks. *IEICE Trans. Inf. Syst.* 95, 2960–2970. doi:10.1587/transinf.E95.D.2960

Berntenis, N., and Ebeling, M. (2013). Detection of attractors of large boolean networks via exhaustive enumeration of appropriate subspaces of the state space. *BMC Bioinformatics* 14:361. doi:10.1186/1471-2105-14-361

Carrillo, M., Góngora, P. A., and Rosenblueth, D. A. (2012). An overview of existing modeling tools making use of model checking in the analysis of biochemical networks. *Front. Plant Sci.* 3:155. doi:10.3389/fpls.2012.00155

Cimatti, A., Clarke, E., Giunchiglia, F., and Roveri, M. (2000). NuSMV: a new symbolic model checker. *Int. J. Software Tool. Technol. Tran.* 2, 410–425. doi:10.1007/s100090050046

Dubrova, E., and Teslenko, M. (2011). A SAT-based algorithm for finding attractors in synchronous boolean networks. *IEEE/ACM Trans. Comput. Biol. Bioinform.* 8, 1393–1399. doi:10.1109/TCBB.2010.20

Garg, A., Di Cara, A., Xenarios, I., Mendoza, L., and De Micheli, G. (2008). Synchronous versus asynchronous modeling of gene regulatory networks. *Bioinformatics* 24, 1917–1925. doi:10.1093/bioinformatics/btn336

Grieco, L., Calzone, L., Bernard-Pierrot, I., Radvanyi, F., Kahn-Perlès, B., and Thieffry, D. (2013). Integrative modelling of the influence of mapk network on cancer cell fate decision. *PLoS Comput. Biol.* 9:e1003286. doi:10.1371/journal.pcbi.1003286

Irons, D. J. (2006). Improving the efficiency of attractor cycle identification in boolean networks. *Physica D* 217, 7–21. doi:10.1016/j.physd.2006.03.006

Kauffman, S. A. (1993). *The Origins of Order: Self Organization and Selection in Evolution*. Oxford University Press.

Klarner, H. (2015a). Available at: http://sourceforge.net/Projects/BoolNetFixpoints

Klarner, H. (2015b). *Contributions to the Analysis of Qualitative Models of Regulatory Networks*. PhD thesis, Freie Universität Berlin, Germany.

Klarner, H., Bockmayr, A., and Siebert, H. (2014). "Computing symbolic steady states of boolean networks," in *Cellular Automata*, Vol. 8751. eds J. Was, G. C. Sirakoulis, and S., Bandini (Switzerland: Springer International Publishing), 561–570.

Naldi, A., Berenguier, D., Fauré, A., Lopez, F., Thieffry, D., and Chaouiya, C. (2009). Logical modelling of regulatory networks with ginsim 2.3. *BioSystems* 97, 134–139. doi:10.1016/j.biosystems.2009.04.008

Naldi, A., Remy, E., Thieffry, D., and Chaouiya, C. (2011). Dynamically consistent reduction of logical regulatory graphs. *Theor. Comp. Sci.* 412, 2207–2218. doi:10.1016/j.tcs.2010.10.021

Saadatpour, A., Albert, R., and Reluga, T. C. (2013). A reduction method for boolean network models proven to conserve attractors. *SIAM J. Appl. Dyn. Syst.* 12, 1997–2011. doi:10.1137/13090537X

Siebert, H. (2011). Analysis of discrete bioregulatory networks using symbolic steady states. *Bull. Math. Biol.* 73, 873–898. doi:10.1007/s11538-010-9609-1

Skodawessely, T., and Klemm, K. (2011). Finding attractors in asynchronous boolean dynamics. *Adv. Complex Syst.* 14, 439–449. doi:10.1142/S0219525911003098

Thomas, R. (1991). Regulatory networks seen as asynchronous automata: a logical description. *J. Theor. Biol.* 153, 1–23. doi:10.1016/S0022-5193(05)80350-9

Veliz-Cuba, A. (2011). Reduction of boolean network models. *J. Theor. Biol.* 289, 167–172. doi:10.1016/j.jtbi.2011.08.042

Veliz-Cuba, A., Aguilar, B., Hinkelmann, F., and Laubenbacher, R. (2014). Steady state analysis of boolean molecular network models via model reduction and computational algebra. *BMC Bioinformatics* 15:221. doi:10.1186/1471-2105-15-221

Zañudo, J. G. T., and Albert, R. (2015). Cell fate reprogramming by control of intracellular network dynamics. *PLoS Comput. Biol.* 11:e1004193. doi:10.1371/journal.pcbi.1004193

Conflict of Interest Statement: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

frontiers in Bioengineering and Biotechnology

ORIGINAL RESEARCH
published: 11 February 2016
doi: 10.3389/fbioe.2016.00010

Systems Perturbation Analysis of a Large-Scale Signal Transduction Model Reveals Potentially Influential Candidates for Cancer Therapeutics

*Bhanwar Lal Puniya[1], Laura Allen[2], Colleen Hochfelder[3], Mahbubul Majumder[2] and Tomáš Helikar[1]**

[1] Department of Biochemistry, University of Nebraska-Lincoln, Lincoln, NE, USA, [2] Department of Mathematics, University of Nebraska at Omaha, Omaha, NE, USA, [3] Albert Einstein College of Medicine, New York, NY, USA

Dysregulation in signal transduction pathways can lead to a variety of complex disorders, including cancer. Computational approaches such as network analysis are important tools to understand system dynamics as well as to identify critical components that could be further explored as therapeutic targets. Here, we performed perturbation analysis of a large-scale signal transduction model in extracellular environments that stimulate cell death, growth, motility, and quiescence. Each of the model's components was perturbed under both loss-of-function and gain-of-function mutations. Using 1,300 simulations under both types of perturbations across various extracellular conditions, we identified the most and least influential components based on the magnitude of their influence on the rest of the system. Based on the premise that the most influential components might serve as better drug targets, we characterized them for biological functions, housekeeping genes, essential genes, and druggable proteins. The most influential components under all environmental conditions were enriched with several biological processes. The inositol pathway was found as most influential under inactivating perturbations, whereas the kinase and small lung cancer pathways were identified as the most influential under activating perturbations. The most influential components were enriched with essential genes and druggable proteins. Moreover, known cancer drug targets were also classified in influential components based on the affected components in the network. Additionally, the systemic perturbation analysis of the model revealed a network motif of most influential components which affect each other. Furthermore, our analysis predicted novel combinations of cancer drug targets with various effects on other most influential components. We found that the combinatorial perturbation consisting of PI3K inactivation and overactivation of IP3R1 can lead to increased activity levels of apoptosis-related components and tumor-suppressor genes, suggesting that this combinatorial perturbation may lead to a better target for decreasing cell proliferation and inducing apoptosis. Finally, our approach shows a potential to identify and prioritize therapeutic targets through systemic perturbation analysis of large-scale computational models of signal transduction. Although some components of the presented computational results have been validated against independent gene expression data sets, more laboratory experiments are warranted to more comprehensively validate the presented results.

Keywords: computational modeling, *in silico* perturbation analysis, signal transduction, cancer, therapeutic targets

OPEN ACCESS

Edited by:
David A. Rosenblueth, Universidad Nacional Autónoma de México, México

Reviewed by:
Reka Albert, Pennsylvania State University, USA
Ovidiu Radulescu, Université de Montpellier 2, France

****Correspondence:***
Tomáš Helikar
thelikar2@unl.edu

Specialty section:
This article was submitted to Bioinformatics and Computational Biology, a section of the journal Frontiers in Bioengineering and Biotechnology

Received: *01 November 2015*
Accepted: *25 January 2016*
Published: *11 February 2016*

Citation:
Puniya BL, Allen L, Hochfelder C, Majumder M and Helikar T (2016) Systems Perturbation Analysis of a Large-Scale Signal Transduction Model Reveals Potentially Influential Candidates for Cancer Therapeutics. Front. Bioeng. Biotechnol. 4:10. doi: 10.3389/fbioe.2016.00010

INTRODUCTION

Recent advances in systems biology and computational biology have introduced methods for the visualization, comprehension, and interpretation of big data in biomedical research. These fields provide an array of methodologies including computer simulations that can be used to generate new hypotheses and identify which hypotheses might be more productive to undertake experimentally, and eliminate hypotheses with little chance of success (Kitano, 2002a,b; Ghosh et al., 2011). These methods can be effective in navigating complex network problems associated with diseases. Many diseases and pathologies can be characterized by the dysregulation or dysfunction of multiple molecular components that are connected within these highly intertwined biological and biochemical networks (Loscalzo and Barabasi, 2011). Biological networks, including biochemical signal transduction networks, consist of a large number of highly interconnected pathways that give rise to complex, non-linear dynamics governing various cellular functions (Helikar et al., 2008; Helikar and Rogers, 2009). Disruptions of these networks, such as mutations or disease states can have drastic effects upon the whole system. These effects are difficult to predict from static network diagrams.

However, understanding the hierarchy of these changes remains a paramount problem. Often the specific causal interactions of the disease state are hidden within the massive cell-wide alterations, making attempts to reverse a disease state more challenging. In addition, the specific causal interactions are difficult to predict making the development of a potential therapeutic target results in unforeseen side effects (Singh and Singh, 2012). The unwanted effects of these drugs are often drastic as seen with many cancer medications (Kayl and Meyers, 2006; Lotfi-Jam et al., 2008; Singh and Singh, 2012). These challenges are further exacerbated by drug resistance that can render therapies ineffective. Therefore, it is necessary to gain a systems level understanding of the components associated with the disease states.

In recent years, targeted therapy has been used for multiple diseases, e.g., cancer (Vanneman and Dranoff, 2012), and often involve the activation or inactivation of a specific component in a biological network by a small molecule or drug, for instance. Perturbation analyses allow one to interrogate the structure and dynamic footprint of the underlying biological system. Perturbation biology has been proposed as an approach to reduce the collateral damage caused by non-specific drugs. Computational network perturbations and new methods to evaluate the robustness of a given network can help identify more effective network components to target in order to obtain desired outcomes with minimal disruption to the rest of the network (Molinelli et al., 2013).

In order to fully leverage the potential of computational network perturbation analyses large dynamical models are necessary. A wide spectrum of modeling approaches exists ranging from detailed (but less scalable) differential equation-based systems to large (but not dynamic) static networks. In the middle are approaches such as logical modeling that are relatively scalable while capable of capturing the dynamic nature of biological systems (Le Novère, 2015). Logical networks, namely Boolean networks, have been used to describe and simulate a wide spectrum of biological systems ranging in size as well as contextual application (Naldi et al., 2010; Helikar et al., 2012; Madrahimov et al., 2013; Rocha et al., 2013; Conroy et al., 2014). Thus, applying perturbation analysis to large-scale logical models may provide new insights into the system, which could be used to identify novel therapeutic targets.

Herein, we present results from a system-wide perturbation analysis of a large-scale Boolean model of a signal transduction network widely present in many types of cells. Specifically, the model previously described in Helikar et al. (2008) represents signaling events within the integrated epidermal growth factor (EGF), G protein-coupled receptor, and integrin signaling network. The model consists of 137 components (mostly proteins) and 557 biochemical interactions. The simulation-based, system-wide perturbation analyses enabled us to identify the most and least influential components (ones with the most and least impact on the rest of the network). To explore the role and effects of these perturbations in the context of the complex extracellular environment, the simulations and analyses were conducted under four biologically relevant environmental conditions known to stimulate cell growth, cell death, motility, and quiescence (in addition to a set of randomly generated environmental stimuli). In order to investigate potential therapeutic targets, we performed functional annotation and analysis of the most influential signal transduction components under both inactivating (e.g., knockout) and activating (e.g., overexpression) perturbations. The most influential components were found to be enriched with many biological processes and druggable targets. Also, the most influential components under activating perturbations were enriched with more essential genes than the least influential components. We used the most influential components and their upstream regulators to identify novel interactions. We also identified a network of the most influential components consisting of drug targets considered in multiple cancer types. The highest ranked among the most influential components were already explored as drug targets against cancer, including EGFR, PI3K, Raf, Ras, and Erk. Because some of these targets have been reported to be associated with drug resistance (Holohan et al., 2013; Rodon et al., 2013; Wagle et al., 2014), we analyzed additional components of the signal transduction network that could potentially complement drug-resistant targets. As a result of the systemic analysis, we identified one novel combinatorial target, PI3K–IP3R1, with consistent occurrence in all simulated environmental conditions. This combination could be used to suppress cell proliferation while increasing the rate of apoptosis. We simulated the effect of combinatorial perturbation and the results were correlated with the literature, further supporting our predictions.

MATERIALS AND METHODS

Computational Model

The computational model analyzed in this work is a Boolean model of signal transduction in a generic cell type. In Boolean models, each component can assume an active (1) or inactive

(0) state at any time *t*. The activity state of the model's internal components is determined by the regulatory mechanisms of other directly interacting components. These regulatory mechanisms are described with Boolean functions (in the form of truth tables or Boolean expressions). To represent the milieu of stimuli in the extracellular environment, the model contains external components that represent various ligands. The activity level of these components is specified as a probability to simulate different levels of concentrations. This methodology was previously detailed and exemplified in Helikar et al. (2008, 2012), Helikar and Rogers (2009), and Todd and Helikar (2012).

The signal transduction model, previously detailed in Helikar et al. (2008), was constructed manually from around 500 published papers. The model consists of several main signaling pathways, including the receptor tyrosine kinase (EGF receptor), G protein-coupled receptors (G-alpha *i*, G-alpha *q*, G-alpha *s*, and G-alpha 12/13), and the integrin signaling pathways. Each of the 130 components in the model corresponds to a signaling molecule (mainly protein). The model also contains nine external components that represent the extracellular environment (mostly composed of receptor ligands). These external components include the EGF, extracellular matrix (ECM), calcium pump, interleukin 1, and tumor necrosis factor (TNF), ligands for four types of G protein-coupled receptors (αi, αq, αs, and 12/13), and a general stress signal. The final model consists of 137 components (130 internal and 7 external) connected with 557 interactions. The model is fully annotated and freely available *via* the Cell Collective software (Helikar et al., 2012, 2013) at www.thecellcollective.org (under Published Models). Cell Collective, an interactive modeling environment, can be used to download the model (and other logical models published by the community) in several file formats (SBML qual, text file of logical functions, truth tables, etc.), as well as simulate directly on the platform. For convenience, the model SBML file is provided as File S1 (Data Sheet 1) in Supplementary Material.

Model Simulations

The Cell Collective platform was used to perform all computational simulations of the model. Although the model is built by using discrete mathematics the output activity levels (AL) can be continuous (ranging from 0 to 100) as previously described in Helikar et al. (2008) and Helikar and Rogers (2009). Each simulation is synchronous and consists of 800 steps, where the activity level of the measured output component is calculated as the fraction of ones (active states) over the last 300 iterations that describe the network's steady behavior (Helikar et al., 2008; Helikar and Rogers, 2009).

Let $x_j(t_i)$ denotes a node's activity on the ith iteration and jth simulation where $i = 1, 2, \ldots, T$ and $j = 1, 2, \ldots, N$, total is the simulation out of N total simulations. We obtain AL as below.

$$\mathrm{AL} = \frac{1}{300N} \sum_{j=1}^{N} \sum_{i=T-300}^{T} x_j(t_i)$$

The model was simulated and analyzed under four biologically relevant environmental conditions that stimulate cell growth, cell death, quiescence, motility (and randomly generated behaviors), as established and detailed in Helikar et al. (2008). The environmental conditions that stimulate each of these cellular responses were obtained based on Helikar et al. (2008) where the model's responses were characterized based on 10,000 combinations of randomly generated environmental signals. For example, cell growth behavior is characterized by higher AL of Erk (marker for proliferation) and Akt (marker for anti-apoptosis). Cell motility behavior was characterized by higher AL of Cdc42 and Rac. Quiescence response is considered when the activity level of Akt is medium to low, and proliferation (Erk) and motility (Cdc42 and Rac) are low or inactive (Helikar et al., 2008). Each environmental condition is defined by different combinations of AL of external components (ligands). The activity level ranges of the environmental conditions were further determined by an optimization method whereby 2,000 simulations were run with all external stimuli ranging from 0 to 100 (except for IL1_TNF and Stress that were limited to low AL). Subsequently, environmental activity level combinations that stimulated cell growth, cell death, motility, and quiescence most effectively were selected as the corresponding environmental conditions (**Table 1**). This is directly analogous to optimization experiments in laboratory studies (e.g., determining the optimal medium and plating conditions of a cell before performing a growth factor titration).

A wild type (WT) experiment (used as a reference) was conducted under each environmental condition without any perturbations. Subsequently, systematic perturbation experiments were conducted under each condition, whereby each component of the model was constitutively activated (activity stuck at 1; gain-of-function/overexpression) or inactivated (activity stuck at 0; loss-of-function/knockout). Each experiment consisted of randomly selecting 100 combinations of AL of the external stimuli from each condition activity range. (The only exception was the random environmental condition, which was simulated 2,000 times.) Each of the 100 combinations were simulated 30 times (i.e., 30 replicates) to ensure consistency of the dynamics in response to a specific combination of stimuli. These replicates were subjected to a Fligner Killeen test of homogeneity of variances, which confirmed that the measured AL of the network components, were homologous for identical combinations of AL of the environmental stimuli.

Model Analysis

The Kolmogorov–Smirnov (KS) test (Wang et al., 2003) was used to compare the WT dynamics (under each environmental condition) with the dynamics of each perturbation experiment. If the KS test resulted in a p-value <0.05, then it has a difference value (DV) equal to the test statistic; otherwise, the DV for a component is 0. Because we are looking at how a node's perturbation affects the rest of the network, its DV when it is the perturbed node is set to 0.

Most and Least Influential Components

The most influential components are defined as components that induce the largest changes in the network under a given perturbation. The ranking of the perturbations is derived by calculating

TABLE 1 | Activity level ranges of environmental stimuli for cell death, growth, motility, quiescence, and random environments.

External	Death	Growth	Motility	Quiescence	Random
Extracellular matrix (ECM)	10–72	26–82	81–99	7–30	0–100
Epidermal growth factor (EGF)	3–15	72–97	29–83	43–56	0–100
Calcium pump (ExtPump)	35–87	24–83	41–92	17–82	0–100
GPCR q ligand (alpha_qL)	13–58	18–78	17–74	4–84	0–100
GPCR i ligand (alpha_iL)	1–4	15–77	30–82	31–83	0–100
GPCR s ligand (alpha_sL)	30–87	24–80	20–77	19–46	0–100
GPCR 12/13 ligand (alpha_1213L)	14–65	18–78	12–77	18–67	0–100
IL1_TNF	4–13	8–15	4–13	2	2
Stress	2–5	2–5	2–5	2–3	2

an influence score (IS) for the *i*th node, which is found by summing the DV for all *M* nodes in the network. The top 10% are considered most influential, and the bottom components with IS value 0 were considered the least influential. The cutoffs were set to 10% because only a few components had a high influence on the network.

$$\mathrm{IS}_i = \sum_{j=1}^{M} \mathrm{DV}_{ij}$$
$$i = 1, \ldots, 130$$

Most Affected Components to a Specific Perturbation

For each perturbation induced, the components that are most sensitive to that perturbation are ranked in decreasing order to be able to characterize downstream effects of the perturbation on the network.

Annotation and Biological Relevance of Signal Transduction Components

All model components were first annotated using the appropriate NCBI gene IDs (Pruitt et al., 2007) for associated genes and UniProt IDs (Consortium, 2011) for protein products of the genes. All components were then further characterized using online resources such as DrugBank (Wishart et al., 2006).

The biological process enrichment analysis of the most influential components was done using DAVID (Huang et al., 2008), with high stringency. Gene Ontology (Ashburner et al., 2000), SP_PIR keywords, and KEGG pathways (Kanehisa, 2002) were obtained using FDR < 5%.

Essentiality data were obtained from the Online GEne Essentiality (OGEE) database and mapped on the most and least influential components (Chen et al., 2012). DrugBank data were used to obtain druggability information for each component in the network. Data on cancer-associated genes were obtained from The Cancer Genome Atlas (TCGA) (Weinstein et al., 2013) and mapped on the most influential components to identify cancer-associated most influential components. The enrichment of essential genes and druggable proteins was computed based on the number of genes mapped on most or least influential components out of the total number of most and least influential components.

Network Motif Analysis

Network motif analysis in the directed signal transduction network was performed using FANMOD tool (Wernicke and Rasche, 2006). The default parameters were used that include 100,000 samples to determine the sub-graphs. The significance of network was computed by comparison with 1,000 random networks. Network motifs that have occurrence more than five times and p-value <0.05 were considered as significant. Network motif analysis was previously integrated with logical modeling of signal transduction of epithelial–mesenchymal transition (Steinway et al., 2014).

Gene Expression Analysis

To investigate the functional activity of the components of signal transduction model, we queried publicly available gene expression data in four different databases (Consortium, 2011); Bgee (gives activity level of genes across different species as well as different developmental stages) (Bastian et al., 2008), CleanEx database (Providing heterogenous data from different) (Praz et al., 2004), Expression Atlas database (gene expression data under different biological conditions) (Petryszak et al., 2014), and GeneVisible database (Gene expression in different tissues) (Zimmermann et al., 2004). Out of 109 signal transduction components i.e., proteins, 107 (~98%) showed expression across different species, developmental stages, organs, and tissues – suggesting the biological activity of signal transduction network. Gene expression status of signal transduction components is shown in the Table S1 in Supplementary Material.

The gene expression dataset GSE53309 was obtained from the GEO database (Barrett et al., 2005; Rosich et al., 2014). We selected samples that were treated with pan-PI3K inhibitor and of normal control. The log 2 RMA signal intensities of samples were transformed into *Z*-scores (Cheadle et al., 2003). To compare the *Z*-scores of treated samples with normal control, we used *Z*-ratio approach. Genes with *Z*-ratio ≥1.50 were considered upregulated and with ≤−1.50 were considered as downregulated. The *Z*-ratio cut-off (1.5) is previously found as robust (Cheadle et al., 2003). The genes of signal transduction components whose AL were affected as a result of PI3K inactivation were examined for *Z*-ratios in both the biological replicates. We used DAVID to perform biological process enrichment analysis of upregulated and downregulated genes. The high stringency and FDR < 5% were used to select significantly enriched biological processes.

RESULTS

System-Wide Perturbation Analysis Reveals Core Components of the Signal Transduction Network

A critical objective of biomedical research is the identification and prioritization of novel therapeutic targets. In this context, we performed systematic perturbation analysis in a generic signal transduction model. The workflow used in this work is illustrated in **Figure 1**.

The activating/inactivating perturbation experiments for each component in the model were carried out across four environmental conditions (as described in the Section "Materials and Methods"). Additional randomly generated extracellular conditions were used to check the robustness of the model and results. Perturbation analysis enabled us to identify and rank components of the signaling network that are most and least influential (Table S2 in Supplementary Material). The heatmaps for all the environmental conditions [Figures S1–S10 (Image 1) in Supplementary Material] indicate that a few components had high influence on rest of the system. Therefore, we considered the top 10% of the components from each condition as the most influential. By contrast, the components that had no influence on the system were considered as the least influential (KS = 0).

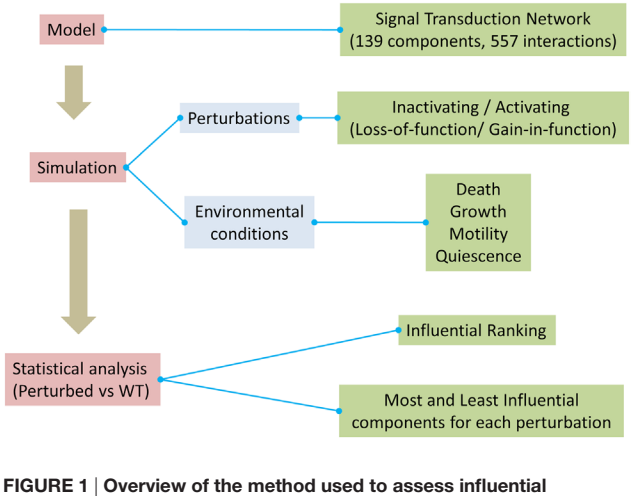


FIGURE 1 | Overview of the method used to assess influential components in the model.

Also, the most influential components correspond to network components that, when perturbed, affect the largest part of the network in terms of the number of affected components and the magnitude of the effect. The most influential components were found for both inactivating (**Figure 2A**) and activating (**Figure 2B**) perturbations under the different environmental conditions. It is interesting to note that many of the most influential components overlap across all environmental conditions. However, the most influential components do not overlap between two types of perturbations (inactivating or activating). We investigated whether the most influential components that spanned different environmental conditions could function as housekeeping genes. Also, the most influential components that are specifically found under one environmental condition should have association with that condition.

Housekeeping Genes Are Enriched in the Most Influential Components Common in Different Environments

Housekeeping genes are defined as genes expressed at constant level in many cells and under many conditions (Eisenberg and Levanon, 2013). Therefore, components that were identified as most influential under all of the simulated environmental conditions can be hypothesized to have housekeeping function. To investigate this, we compared these most influential components with known housekeeping genes as provided in Eisenberg and Levanon (2013). Under inactivating perturbations, out of the seven components common among the different environmental conditions, PI4K, PI5K, ARF, and PI3K were associated with housekeeping genes (Eisenberg and Levanon, 2013). Under activating perturbations, Trafs, Erk, Mek, and SHP2 (out of nine common components), were associated with housekeeping genes. Housekeeping genes associated with the common components are displayed in **Table 2**. This observation suggests that the most influential components that are common among different environmental conditions are likely to function as housekeeping genes.

Unique Components Associated with Each Environmental Condition Are Found to Be Condition Specific

Under both types of perturbations, certain environmental conditions had several uniquely associated components (**Figure 2**; **Table 3**). Under inactivating perturbations, components uniquely

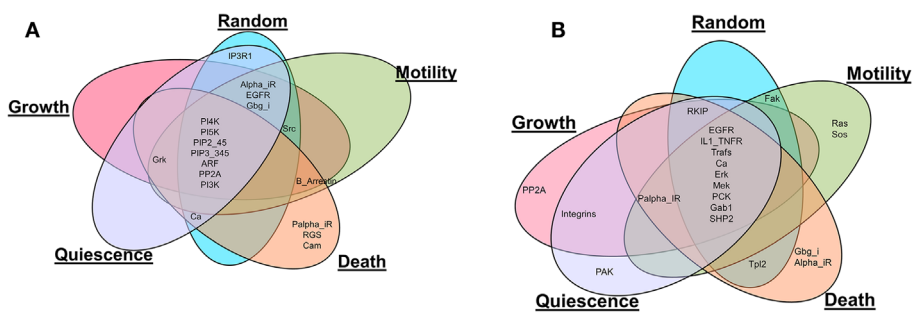


FIGURE 2 | Comparison of the most influential components across simulated environmental conditions. (A) Inactivating perturbations, **(B)** activating perturbations.

TABLE 2 | Housekeeping genes in the most influential components overlapped among different environmental conditions.

Perturbation	Components	Genes	Housekeeping genes[a]
Inactivating	PI4K	PI4KA, PI4KB, PIK4CB	PI4KA, PI4KB
	PI5K	PIP5K1A, PIP5K1B, PIP5K1C	PIP5K1A
	ARF	ARFGAP1, ARFGAP2, ARFGAP3	ARFGAP2, ARFGAP3
	PP2A	PPP2CA	PPP2CA
	PI3K	PIK3CA, PIK3CB, PIK3CD, PIK3CG	PIK3C3, PIK3CB
Activating	EGFR	EGFR	No
	IL1_TNFR	IL1B, TNFRSF1A	No
	TRAFS	TRAF1, TRAF2, TRAF3, TRAF4, TRAF5, TRAF6, TRAF7	TRAF6, TRAF7
	ERK	MAPK1 to MAPK15	MAPK1, MAPK6, MAPK8, MAPK9
	MEK	MAP2K1 to MAP2K7	MAP2K1, MAP2K2, MAP2K5
	PKC	PRKCA, PRKCB, PRKCD, PRKCE, PRKCG, PRKCH, PRKCI, PRKCQ, PRKCZ	No
	GAB1	GAB1	No
	SHP2	PTPN11	PTPN11

[a]*List of housekeeping genes were obtained from Eisenberg and Levanon (2013).*

TABLE 3 | Condition-specific components and literature support.

Perturbations	Environmental condition	Associated components	Literature
Inactivating	Death	*CaM*, *RGS*, Palpha_iR	CaM- and CaM-dependent signaling systems control vertebrate cell proliferation, programed cell death, and autophagy (Berchtold and Villalobo, 2014). RGS is involved in cell death (Fisher, 2009)
Activating	Death	*Gbg_i* (GNB), Alpha_iR	Gbg_i has been hypothesized to be involved in mTOR-mediated anti-apoptotic pathways. Furthermore, it has been functionally annotated with apoptosis, cell death (Wazir et al., 2013)
	Growth	PP2A	Highly regulated family of Ser/Thr phosphatase implicated in cell growth and signaling (Janssens and Goris, 2001)
	Motility	KRAS, Sos	Knockdown of *KRAS* in pancreatic cancer cell lines leads to decreased motility and proliferation. The Grb2–Sos1 complex may promote cell motility, and tumerogenesis (Qu et al., 2014)

associated with the cell death stimulating condition are calmodulin (CaM), RGS, and Palpha_iR. Out of these, CaM and RGS have been previously associated with cell death and apoptosis (Fisher, 2009; Berchtold and Villalobo, 2014). In fact, CaM plays a central role in the regulation of several cellular functions, including programed cell death (Berchtold and Villalobo, 2014). It is also known that RGS protein can regulate cell death, cell cycle, and cell division (Fisher, 2009). Under activating perturbations, the most influential components associated with the cell death-inducing condition include Gbg_i and Alpha_iR. On the other hand, PP2A was found to be most influential under the growth stimulating condition, Ras and Sos under motility stimulating condition, and PAK under quiescence stimulating condition. These results are also further supported by published studies that reported Gbg_i (GNB) to be involved in mTOR-mediated anti-apoptotic pathways; Gbg_i was also functionally annotated with apoptosis and cell death (Wazir et al., 2013). PP2A was reported as a highly regulated Ser/Thr phosphatase involved in cell growth and signaling (Janssens and Goris, 2001). In pancreatic cancer cell lines, the knockdown of KRAS has been found to lead to the decrease in cell motility and proliferation (Rachagani et al., 2011; Birkeland et al., 2012). Furthermore, the Grb2–Sos1 complex has been found to most likely promote cell motility, and tumerogenesis (Qu et al., 2014). These observations suggest that the proteins, which were uniquely associated with simulated environmental conditions, are most likely to have the association with that condition. Finally, the literature evidence obtained for housekeeping, or condition associated genes, further supports our simulation results.

Key Biological Processes Are Enriched in the Most Influential Components

Next, we assessed the enrichment of biological processes or pathways in the most influential components. The most influential components across all four conditions under both types of perturbation showed significant enrichment with key biological processes. The counts and fold differences of enriched biological terms in all the conditions are shown in **Figures 3** and **4**. In the case of inactivating perturbations, inositol phosphate metabolism was enriched under all environmental conditions (**Figure 3**). In the case of activating perturbations, the significantly enriched biological processes include phosphate metabolic processes, kinase activity, apoptosis, and, interestingly, the non-small lung cancer pathway (**Figure 4**). These results illustrate that the group of proteins with similar biological functions appear as the influential components under each type of perturbation.

The Most Influential Components under Activating Perturbations Are Enriched with Essential Genes

Mutations in an essential gene can be lethal. Based on the hypothesis that the influential components might serve as essential for

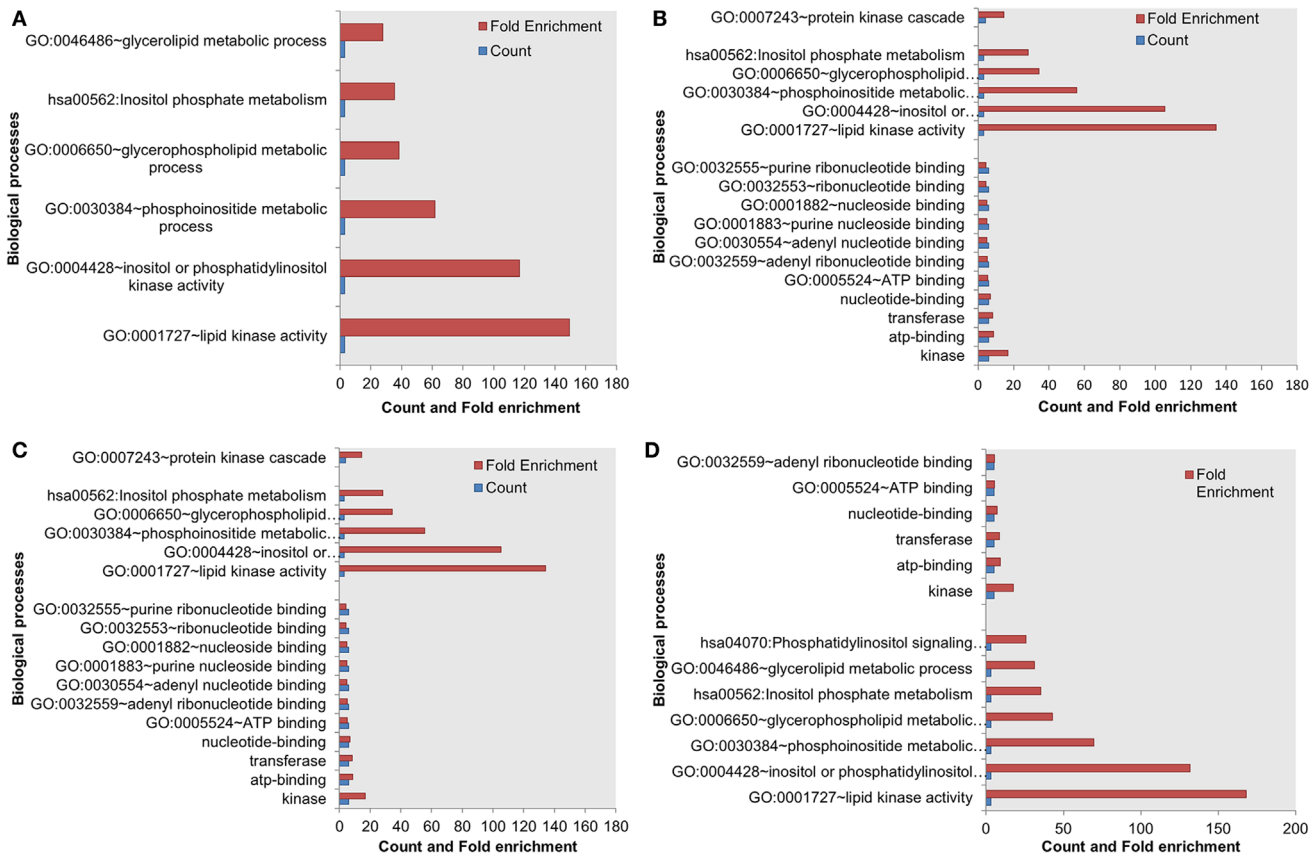


FIGURE 3 | **Enriched biological processes in the most influential components under environmental conditions, and inactivating perturbations**. **(A)** Death **(B)** growth **(C)** motility and **(D)** quiescence.

the survival of the cell, we performed essentiality analysis. We mapped essential genes on the most influential components and on the least influential components. The essential genes mapped on the most influential components were compared with essential genes mapped on the least influential components. Under activating perturbations, more essential genes were found within the most influential components than the least influential components (**Figure 5A**). Under the cell death stimulating condition, a total of 69% of the most influential components were essential; this is in contrast to the least influential components that contained 31% essential genes. Under other environmental conditions stimulating growth, motility, and quiescence, the difference of essential genes between the most influential and the least influential components are 23, 15, and 32%, respectively.

On the other hand, under inactivating perturbations, we found either an equal or larger number of essential genes in the least influential components (**Figure 5B**). The most significant differences were observed under the cell death stimulating conditions: the least influential components have 66% of essential genes in contrast to the 46% essential genes in the most influential. Also, under the growth stimulating conditions, 68 and 53% of essential genes were contained within the least and the most influential components, respectively. Under the motility and quiescence stimulating conditions, there were 3 and 9% more essential genes within the least influential components than the most influential components, respectively. We found that under inactivating perturbations, the number of essential genes among the least influential components was slightly larger than the activating perturbation (**Figures 5C,D**). On the other hand, under activating perturbations, the more essential genes mapped within the most influential components than the least influential components.

Thus, the most influential components are essential under activating perturbations, suggesting an environmental condition-specific essentiality.

The Most Influential Components Are Enriched with Druggable Proteins

To further investigate the importance of the most influential components, we evaluated the distribution of known druggable targets. We obtained druggability data from the DrugBank database (Wishart et al., 2006) and mapped them on the most and least influential components. A total of 51 components in the whole network were enriched with druggable proteins. We compared druggable proteins within the most influential components with druggable proteins within the least influential components. We found that under both types of perturbations and across all environmental conditions more druggable proteins were found among the most influential than the least influential components

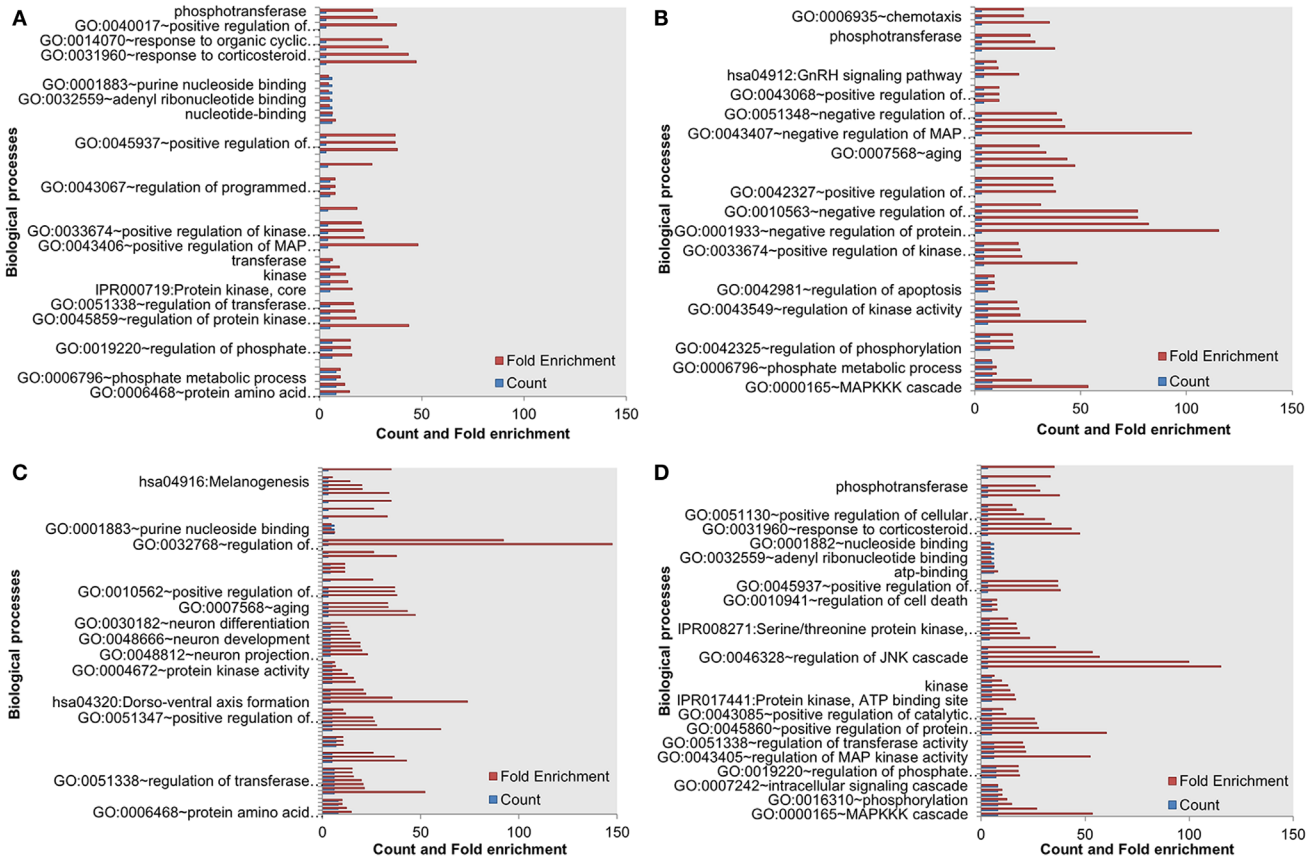


FIGURE 4 | **Enriched biological processes in the most influential components under environmental conditions, and activating perturbations**. **(A)** Death **(B)** growth **(C)** motility and **(D)** quiescence.

(**Figure 6**). Druggable proteins are experimentally characterized or predicted to bind to antagonist or agonist drugs with high affinity. Therefore, enrichment of druggable proteins within the most influential components has the potential to suggest important candidates for therapeutic target discovery.

The Most Influential Components as Drug Targets

Ranked Most Influential Components Based on Downstream Components

We identified the most affected components of the most influential components under both types of perturbations. We combined all environmental conditions to construct networks of the most influential components with their downstream targets. We subsequently mapped druggable proteins and cancer-associated genes on these networks. Under inactivating perturbations, we obtained a network consisting of the most influential components: PI3K, EGFR, PP2A, GRK, and CaM (**Figure 7A**). Under activating perturbations, we obtained a network composed of influential components: EGFR, IL1_TNFR, ERK, SHP2, RKIP, Ras, Gbg_i, Fak, Integrins, and PP2A (**Figure 7B**).

The total number of downstream targets for each of the most influential druggable component under both inactivating and activating perturbations is listed in **Table 4**. We analyzed if these downstream components also affects their upstream component. In the case of PI3K-out of 42 downstream components, two (PIP3_345 and RGS) are part of a feedback system. Other feedback components in downstream targets include alpha_iR for GRK in inactivating perturbations, Gab1 for SHP2, and Palpha_iR for RKIP under activating perturbations. We observed that EGFR, a validated cancer drug target (Mendelsohn, 2001), affects the largest number of components under activating and inactivating perturbations.

The Most Influential Components Mainly Affect Other Most Influential Components

Here, we identified all components that directly affect the activity of each most influential component (KS = 1). Interestingly, most of these direct upstream components were also ranked as the most influential in at least one environmental condition (**Figure 8**). Under inactivating perturbations, 22 components were directly upstream of the most influential components. Of these, 19 were the most influential under at least one environmental condition. On the other hand, under activating perturbations, out of 45 upstream components, 19 were also ranked as most influential. Additionally, under inactivating perturbations, 9 (CaM, EGFR, Gbg_i, GRK, IP3R1, PP2A, PI3K, Ras, and Src) out of total 22 upstream components are druggable. Out of these 22 components, 6 components (CaM, EGFR, Gbg_i, GRK, IP3R1, and

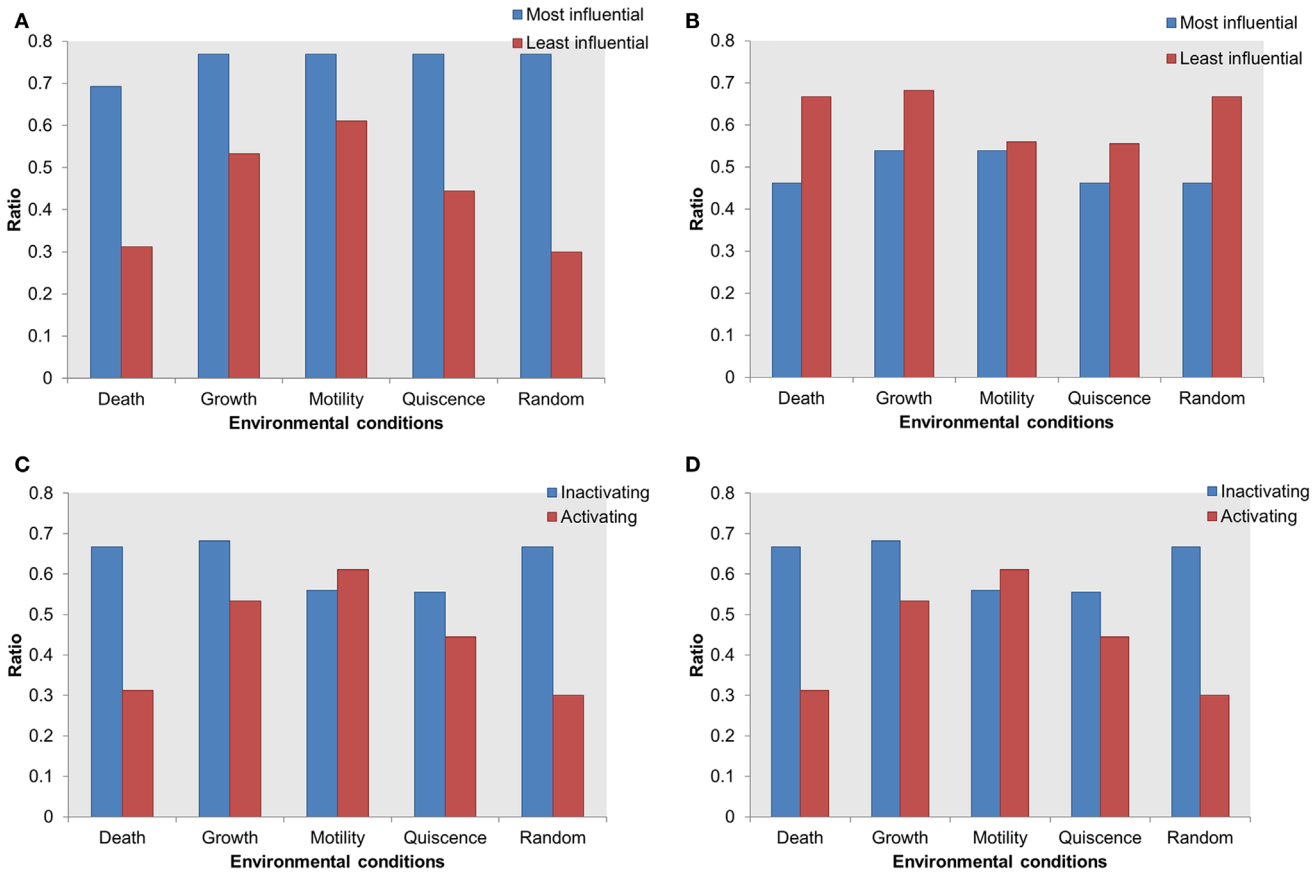


FIGURE 5 | Distribution of essential genes in the most influential components. *X*-axis = environmental conditions, *Y*-axis = ratio of essential genes in total selected most or least influential components in **(A)** most influential vs. least influential components under activating perturbations, **(B)** most influential vs. least influential components under inactivating perturbations, **(C)** essential genes in most influential under inactivating vs. activating perturbations, **(D)** essential genes in least influential components under inactivating vs. activating perturbations.

PP2A) were upstream to the most influential druggable components. Under activating perturbations, 21 (CaM, Cdc42, EGFR, Erk, Fak, Gbg_i, Grb2, GRK, IL1_TNFR, Integrins, IP3R1, PDK1, PI3K, PKA, PP2A, Rac, Raf, Ras, RKIP, SHP2, and Src) out of 45 upstream to the most influential components are associated with druggable proteins. Out of these 21, 10 components were also the most influential. Under both types of perturbations, a total of 18 (alpha_iR, ARF, B_Arrestin, Ca, CaM, EGFR, Gbg_i, GRK, IP3R1, Palpha_iR, PI5K, PIP2_45, PIP3_345, PP2A, RGS, PI3K, Ras, and Src) upstream components were common. Nine of these components (CaM, EGFR, Gbg_i, GRK, IP3R1, PP2A, PI3K, Ras, and Src) were druggable or these were used as the drug targets. The important drug targets, such as EGFR, PI3K, Ras, and Raf, are also appeared as influential upstream components. Together, these results suggest that under inactivating perturbations the activity of the most influential components are likely to be modulated by the other most influential components.

The Most Influential Components as Drug Targets and Drug Resistance

The top most influential components, such as EGFR, PI3K, ERK, and Ras, have been previously explored as drug targets in multiple cancer types. However, it is also evident from literature that several most influential components have been associated with drug resistance. For example, in non-small cell lung cancer, mutation within the kinase domain of EGFR and epithelial–mesenchymal transition are responsible for the development of resistance to gefitinib (Holohan et al., 2013). In colorectal, and head and neck cancers, KRAS mutation, EGFR-S492R mutation, and increased ErBb signaling are responsible for resistance against Cetuximab (Dienstmann et al., 2012; Holohan et al., 2013). Furthermore, PI3K showed drug resistance in breast cancer against rapamycin through the expression of RSK3 and RSK4 (Rodon et al., 2013). Mutations in ERK1 or ERK2 have shown resistance against ERK inhibitors or RAF/MEK inhibitors (Wagle et al., 2014). Tumors with mutation in BRAF V600E can adapt to the RAF inhibitors (Lito et al., 2013; Perna et al., 2015). As such, the identification and prediction of drug targets alone are not sufficient to identify completely useful drug targets. Investigation of the interactions and feedback of these most influential components could be useful to modulate the activity of the most influential component. Thus, we explored the regulatory interactions to investigate the effect of combinatorial perturbations on cell's behavior.

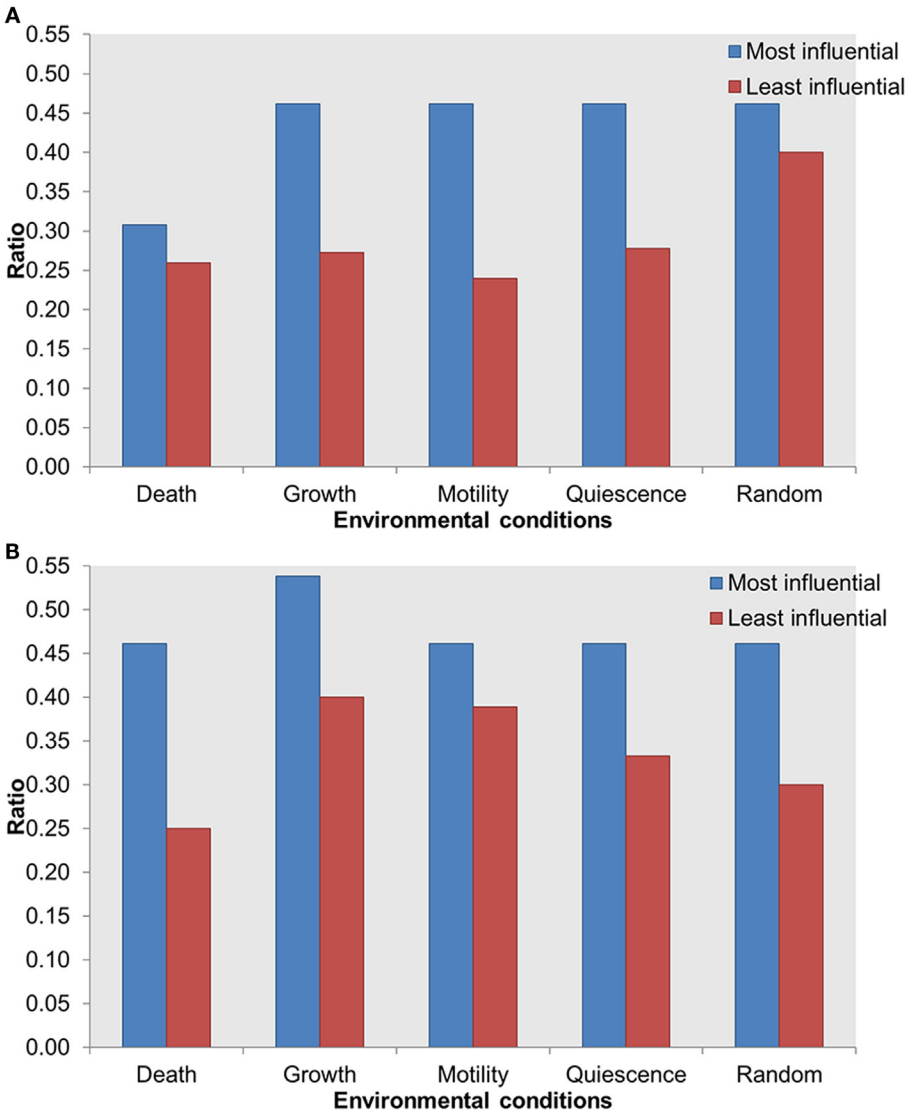


FIGURE 6 | Distribution of druggable proteins within the most influential vs. least influential components. **(A)** Inactivating perturbations, **(B)** activating perturbations. *X*-axis = environmental conditions, *Y*-axis = ratio of druggable proteins in total most or least influential components.

Regulatory Interactions between the Most Influential Components and Their Upstream Components

To develop a better strategy that can account for drug resistance of the most important drug targets, we sought to investigate novel regulatory interactions. We analyzed the previously described interactions between the most influential components and their direct upstream components. We found that some interactions consistently occur in more than one environmental condition. For example, the inactivation of IP3R1 increases the activity of PI3K under all four environmental conditions. However, the maximal effect was observed under the death environmental condition. Additionally, the inactivation of IP3R1 leads to inactive RGS under three environmental conditions stimulating cell growth, motility, and quiescence. These finding also correlate with published studies that found that RGS positively regulates apoptosis (Fisher, 2009). Other examples of consistently occurred interactions include: the activation of Grb2 leads to increased AL of Ras under all four environmental conditions, and increased Sos activity under two environmental conditions stimulating death and quiescence. The activation of Rac increases the activation of PAK under environmental conditions stimulating cell death and growth. Overall, we found three types of interactions: inactivation of one component leads to the increase of activity of another component (PI3K–IP3R1, IP3R1–PI3K, and RGS–IP3R1), inactivation of a component leads to decreased activity of another component (IP3R1–RGS), and activation of a component leads to increased activity of another component (Grb2–Ras, Grb2–Sos, and Rac–PAK).

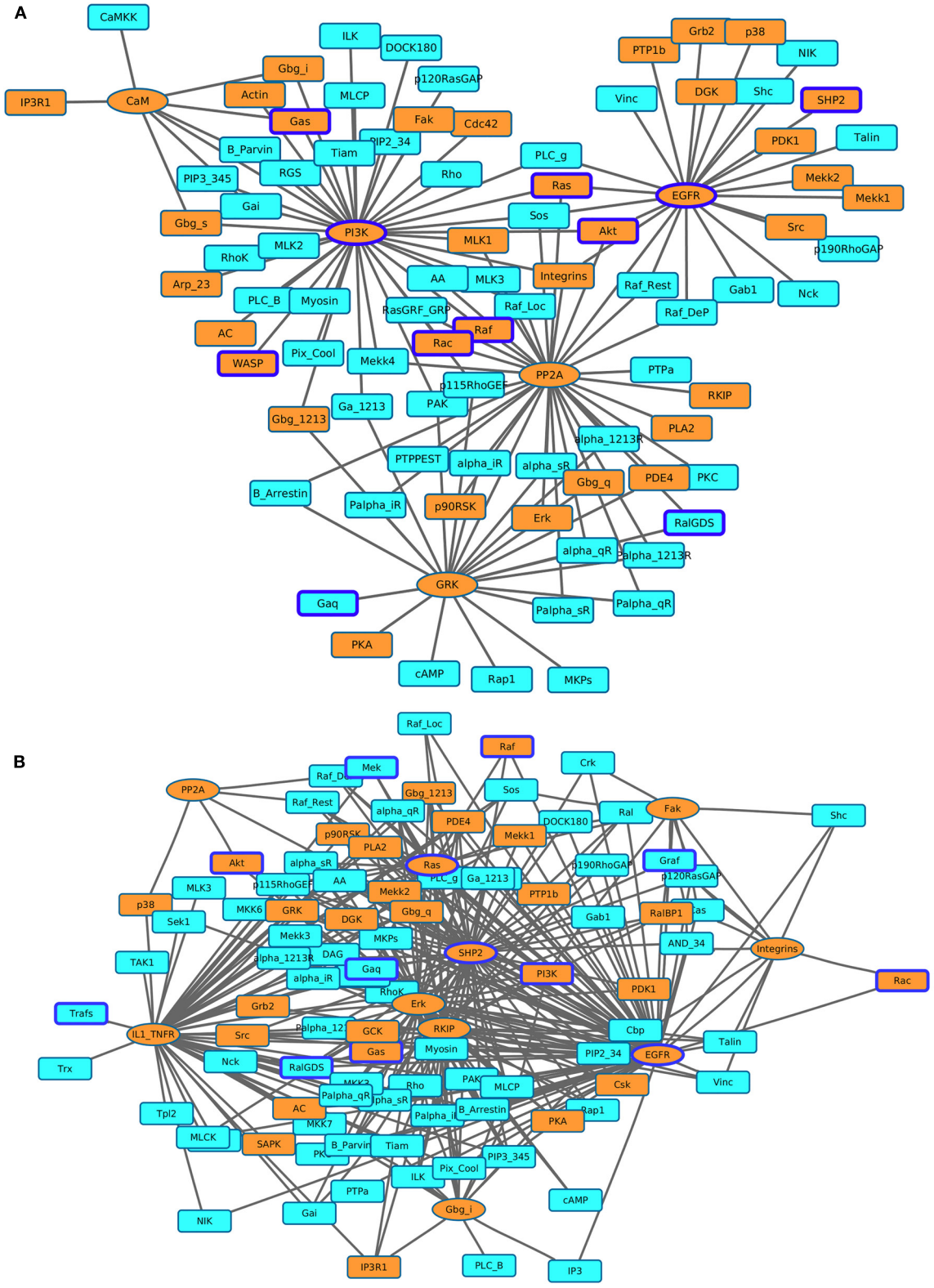


FIGURE 7 | Visualization of the most affected components (KST value = 1) as a result of perturbing the most influential druggable components. (A) Inactivating perturbations, **(B)** activating perturbations. Orange colored eclipeses = most influential druggable components; squares = affected components; orange colored squares = affected druggable components; components with blue borders = experimentally found to be associated with cancer.

TABLE 4 | Number of downstream targets of the most influential druggable components.

	Number of affected components	Number of affected druggable components	Number of cancer-associated components	Feedback components	Perturbation
EGFR	70	25	8		Inactivating
EGFR	24	13	3		Activating
IL1_TNFR	54	14	5		Activating
Erk	54	21	8		Activating
SHP2	53	17		1 (Gab1)	Activating
RKIP	43	12	4	1 (Palpha_iR)	Activating
PI3K	42	17	7	2 (PIP3_345, RGS)	Inactivating
PP2A	36	14	6		Inactivating
PP2A	5	3	2		Activating
Ras	30	13	5		Activating
GRK	22	5	2	1 (alpha_iR)	Inactivating
Gbg_i	15	5	1		Activating
Fak	14	6	4		Activating
Integrins	11	3	3		Activating
CaM	8	5	2		Inactivating

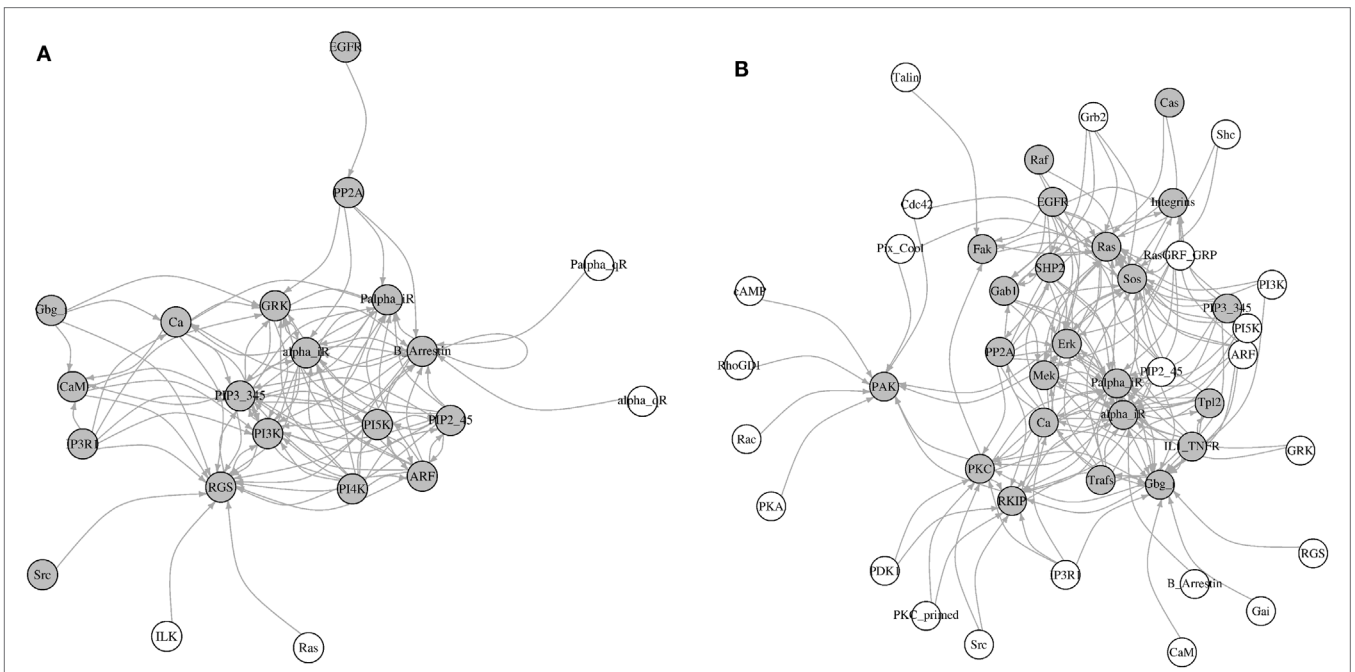


FIGURE 8 | Visualization of the upstream components affecting the most influential components. **(A)** Inactivating perturbations, **(B)** activating perturbations. Gray colored nodes = the most influential components, and white colored nodes = not most influential components. The directions of arrows are from the source (upstream component) to the target (most influential components).

The fold differences of all these interactions are displayed in the **Table 5**. Under the cell death condition, the inactivation of IP3R1 results in PI3K activity increase by 2.38-fold. Similarly, PI3K inactivation leads to a 5.42-fold increase in IP3R1 activity. In the case of other interactions, the inactivation of IP3R1 leads to inactive RGS under the cell growth, motility, and quiescence stimulating conditions. Under the motility and quiescence stimulating conditions, the inactivation of Gbg_i leads to inactive CaM. The activation of Grb2 increases the activity of Ras 7.40-fold under the cell death stimulating conditions, and 2.13-fold under the quiescence stimulating conditions. Grb2 activation also affects Sos 7.8-fold under the cell death stimulating conditions and 2.18-fold under the quiescence stimulating conditions. An activating perturbation of Rac increases the activity of PAK more than 18-fold under the cell death stimulating conditions, and 5.59-fold under the growth stimulating conditions.

To investigate if these interactions are part of any network motifs in the signal transduction network, we performed a

TABLE 5 | Fold differences of the affected most influential component when the upstream component was perturbed.

Perturbed component	Affected component	Fold differences (perturbed/WT)			
		Death	Growth	Motility	Quiescence
IP3R1 (inactivation)	PI3K	2.38-Fold[a]	1.03-Fold	1.04-Fold	1.14-Fold
PI3K (inactivation)	IP3R1	5.42-Fold[a]	1.18-Fold	1.15-Fold	1.24-Fold
IP3R1 (inactivation)	RGS	NSA	Complete inactivation	Complete inactivation	Complete inactivation
RGS (inactivation)	IP3R1	NSA	1.21-Fold	1.18-Fold	1.24-Fold
Gbg_i (inactivation)	CaM	NSA	NSA	Complete inactivation	Complete inactivation
CaM (inactivation)	Gbg_i	NSA	NSA	1.30-Fold	1.43-Fold
Grb2 (activation)	Ras	7.40-Fold[a]	1.32-Fold	1.39-Fold	2.13-Fold
Ras (activation)	Grb2	0.99-Fold	0.97-Fold	0.99-Fold	1.01-Fold
Grb2 (activation)	Sos	7.87-Fold[a]	1.39-Fold	1.53-Fold	2.18-Fold
Sos (activation)	Grb2	1-Fold	0.97-Fold	0.99-Fold	1.01-Fold
Rac (activation)	PAK	18.41-Fold[a]	5.69-Fold	NSA	NSA
PAK (activation)	Rac	1.18-Fold	1.24-Fold	NSA	NSA

NSA, not significantly affected (KST value <1).
[a]*Twofold or above change.*

network motif analysis. We found that all interactions discussed above were part of network motifs (p-value <0.05). IP3R1–PI3K is found in 3 significantly occurred 4-node network motifs and in 15 significantly occurred 5-node network motifs. The other interactions are also found in significantly occurred 4 and 5-node network motifs (Table S3 in Supplementary Material).

These results suggest different types of regulatory effects of activating and inactivating perturbations of direct upstream components of the most influential components.

Cotargeting IP3R1 with PI3K

As discussed earlier, although PI3K was identified as one of the most influential components, it has been also associated with drug resistance. Based on the interactions of upstream regulators of the most influential components discussed above, we further investigated the interactions involving PI3K and IP3R1 with the objective of identifying a secondary drug target that could be potentially used to address the issue of PI3K-associated drug resistance. In contrast to PI3K/Akt signaling, IP3R1 positively regulates apoptosis. We hypothesized that the rate of apoptosis will increase when IP3R1 is overactivated (activating perturbation) and PI3K is inactivated (inactivation perturbation). Despite the strong dynamical relationship between IP3R1 and PI3K, these two components are only connected indirectly through a sub-network. In this sub-network, Gbg_i is upstream of and directly activates both components. IP3R1 regulates PI3K through a Ca → EGFR route, whereas PI3K regulates IP3R1 *via* a PTEN → PIP2_45 → IP3 route (**Figure 9**).

The inactivating perturbation of PI3K resulted in the inactivation of 29 components across all four environmental conditions. To correlate PI3K inhibition results with laboratory experiments, we analyzed a gene expression dataset obtained from cells treated with PI3K inhibitors (Rosich et al., 2014). In two biological replicates, we found that the genes of components with affected AL had shown differential gene expression (at least in one experiment). As a result of the simulated constitutive inhibition of PI3K in the model, the activity level of a total of 15 components (20 genes) increased more than twofold. Nine (60%) of these components were also significantly upregulated in the gene expression dataset (Table S4 in Supplementary Material). Out of these 20 genes, 9 genes (45%) were upregulated in biological replicate 1, whereas 12 (60%) genes were upregulated in biological replicate 2. Cumulatively, 18 genes (90%) were upregulated in both biological replicates. Two of these signal transduction components, Rap1 and PTPPEST, showed significant upregulation in both the biological replicates in gene expression data. Furthermore, the activity of a total of 26 components (41 genes) decreased more than twofold in our model. Genes of eight components (30%) were significantly downregulated in the obtained gene expression data (Table S4 in Supplementary Material). Out of these 41 genes, three genes (7%) were significantly downregulated in biological replicate 1, whereas eight genes (19.5%) were significantly downregulated in biological replicate 2. Cumulatively, 12 genes (29%) were upregulated in both biological replicates. Furthermore, we compared enriched biological processes within the components affected in the model with enriched biological processes in differentially expressed genes. We found that the "regulation of phosphorylation" biological process was enriched for the upregulated genes in both the model and the gene expression data. For downregulated components, "positive regulation of programed cell death" was consistent for both the model and the gene expression data (biological replicate 1). Together, these results suggest that our simulation results are moderately correlated with the results of available gene expression data. In previous integrative studies of gene expression and biochemical models, at best moderate correlations were observed between gene expression and metabolic fluxes (Blazier and Papin, 2012). Post-transcriptional modifications and enzyme kinetics are possible reasons behind poor correlation between gene expression and protein abundance (Washburn et al., 2003; Blazier and Papin, 2012). As such, more laboratory experiments will be needed to further validate our results.

Under PI3K inactivation, the average activity of IP3R1 increased from 71.9% in WT to 85.18%. This perturbation also

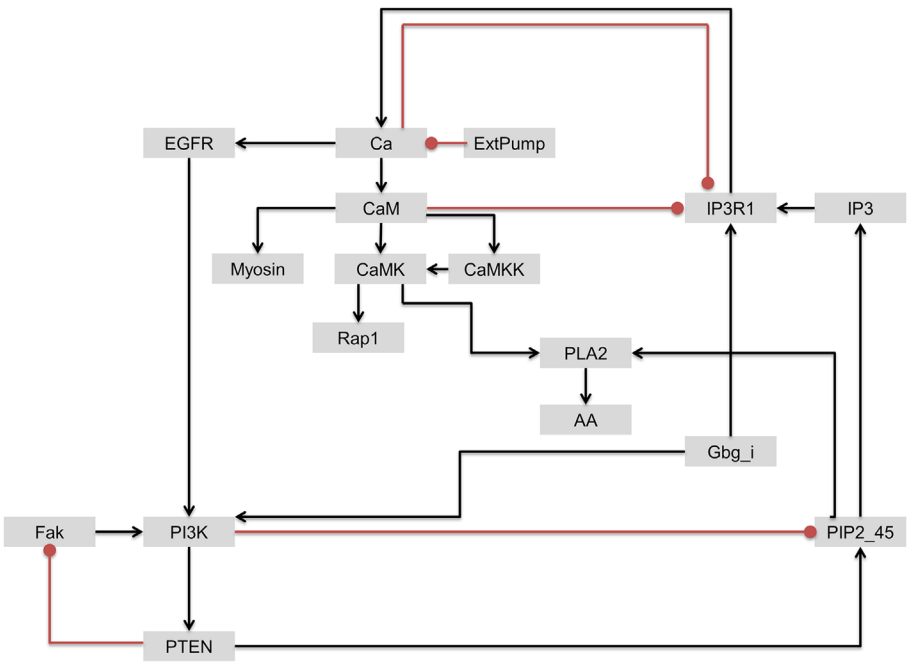


FIGURE 9 | The regulatory circuit connecting IP3R1 and PI3K and downstream components. Edges with arrow = activation. Edges with oval end = inhibition.

TABLE 6 | Activity of affected components under single (PI3K or IP3R1) and double perturbations (PI3K and IP3R1) under the cell growth environmental condition.

Affected components	PI3K inactivation (single perturbation)[a] (fold)	IP3R1 activation (single perturbation)[a] (fold)	Double perturbation[a] (fold)	Functional annotation[b]
Rap1	3.25	1.07	3.90	Tumor-suppressor gene
Ca	1.17	1.41	1.43	Calcium ion, apoptosis
CaM	1.17	1.41	1.43	Cell death
CaMKK	1.17	1.41	1.43	Calcium ion binding, apoptosis
Myosin	0.30	1.004	0.36	Regulatory light chain of myosin
CaMK	1.33	2.09	2.19	May function in dendritic spine and synapse formation and neuronal plasticity
PLA2	0.32	1.24	0.63	Tumor-suppressor gene, apoptosis
AA	0.32	1.24	0.63	Apoptosis

[a]*Compared to the activity of components in wild type.*
[b]*Functional annotations for proteins were obtained from UniProt database and literature.*

led to downregulation of positive regulators of apoptosis phospholipase A2 (PLA2) and arachidonic acid (AA). AA released by PLA2 triggers Ca^{2+}-dependent apoptosis through mitochondrial pathways (Penzo et al., 2004). The elevation in Ca^{2+} is thought to be involved in apoptosis (Pinton et al., 2008). It was shown that blocking calcium channels can directly lead to tumor promotion (Mason, 1999). Thus, inactivation of PI3K can block cell proliferation; simultaneously, it can lower the rate of apoptosis. Interestingly, the positive regulation of the programed cell death biological process was enriched in downregulated genes within the analyzed gene expression data.

Under the cell growth stimulating condition, the activating perturbation of IP3R1 increased the activity of apoptosis-associated components: Ca, CaM, CaMK, CaMKK, and RGS in the range of +1.41- to +2.09-fold when compared to WT.

To simulate the cell death effect under the growth stimulating condition, we carried out a double perturbation of IP3R1 and PI3K, whereby IP3R1 was constitutively activated and PI3K was completely inactivated. Under this combinatorial perturbation, we found 27 proteins including proto-oncogenes such as Akt (which suppresses apoptosis) and Raf to be downregulated. Here, we found eight proteins with more than 19% increased activity than in the case of a single inactivating perturbation of PI3K. These proteins include Rap1 (+1.19-fold), Ca (+1.21-fold), CaM (+1.21-fold), CaMKK (+1.21-fold), Myosin (+1.22-fold), CaMK (+1.65-fold), PLA2 (+1.98-fold), and AA (+1.98-fold)

(**Table 6**; full list of all affected components is given in Table S5 in Supplementary Material). These components were downregulated when only PI3K was inactivated. Under the combinatorial perturbation (PI3K inactivated and IP3R1 activated), the increased activity of these components was achieved by constitutive expression of IP3R1 *via* the following routes: IP3R1 → Ca → CaM → CaMK → Rap1 and IP3R1 → Ca → Ca M → CaMK → PLA2 → AA (**Figure 9**). It is noteworthy that these components have been found to positively regulate apoptosis or cell death. Therefore, under the aforementioned combinatorial perturbation, components involved in cell proliferation were downregulated through the inactivation of PI3K, and the activity of tumor-suppressor genes (PLA2) with arachidonic acid (AA) and other components, including Ca, CaM, and CaMK, was increased as a result of the IP3R1 overactivation.

Together, these results suggest a regulatory interaction between PI3K and IP3R1, and that cotargeting both of these components may serve as therapeutic strategy rather than targeting PI3K alone. Using this combination of targets, we simulated cell death behavior in cell proliferation inducing environmental condition. Thus, we predict that this novel target combination might increase the rate of apoptosis while blocking cell proliferation in tumor cells. However, additional experimental validation is needed to validate this computational result.

DISCUSSION

We have presented a systemic perturbation analysis of a signal transduction network model to identify and characterize functionally important components. We used these components to explore novel therapeutic strategies against cancer. Specifically, we used a logical modeling approach to analyze the dynamics of a large-scale signal transduction model. Logical modeling approaches have been used, for example, to understand the dynamics of signal transduction and gene regulation networks to identify drug synergies in gastric cancers, and to identify potential drug combinations (Flobak et al., 2015). In biochemical networks, combined effect of topology and dynamical features has been shown to have the most significant impact on the dynamics of the network (Kochi et al., 2014). Computational approaches have become indispensable tools to understand biological pathways and disease phenotypes. Examples include computational methods such as molecular modeling, text mining, and network modeling to identify drug targets in a vast array of diseases from pathogens to complex disorders (Flórez et al., 2010; Yao et al., 2010; Folger et al., 2011; Madrahimov et al., 2013; Puniya et al., 2013).

In the present work, the identified most influential components were characterized for biological functions. The relevance of identified influential components was established with pathway analysis, mapping of housekeeping genes, essential proteins, and association with druggable proteins. Interestingly, we found enrichment of housekeeping genes in the most influential components that were independent of the extracellular environments. A notable agreement is obtained from literature surveys for the most influential components, which were unique to specific environmental conditions. Because essential components are important from a disease perspective, the identified most influential components may serve as potential candidates and essential proteins under specific conditions. Under activating perturbations, we found that essential genes were enriched more within the most influential components than within the least influential components. The high association of dysregulated signal transduction proteins with different subtypes of cancers suggests that these components may be important candidates for drug targets. Notably, the most influential components are enriched with several already known drug targets. However, many of these drug targets (EGFR, ERK, Ras, PI3K, etc.) have been associated with drug resistance (West et al., 2002; Kobayashi et al., 2005; Linardou et al., 2008; Wheeler et al., 2010; Dienstmann et al., 2012). The mechanism of drug resistance includes mutation in the targeted protein or expression of other genes (altered expression) to bypass the effect caused by perturbation, deregulation in apoptosis, etc. (Gottesman, 2002; Holohan et al., 2013). Thus, to identify novel regulatory interactions, we explored components that are upstream to the most influential components associated with drug resistance. Interestingly, several upstream components (more than 90% in the case of inactivating perturbations) to the most influential components were also identified as most influential. Thus, the most influential components form a tightly connected sub-network of proteins interacting with each other. In yeast, it has previously shown that the essential proteins are hubs in the network and have more interconnections than non-essential proteins, and form a module or sub-network (Song and Singh, 2013).

The interaction between IP3R1 and PI3K was observed under all environmental conditions. This interaction was also observed as part of network motifs in the modeled signal transduction network. IP3R1 activation, when combined with PI3K inactivation, increases the activities of PLA2 and AA, which are decreased with a single PI3K knockdown. It was already shown that AA released by PLA2 helps to initiate apoptosis (Penzo et al., 2004). In a *Dictyostelium discoideum* chemotaxis experiment, it was also shown that cells with PI3K deficiency were more sensitive to PLA2 inhibition (Chen et al., 2007), which supports our predicted interaction between PI3K and PLA2. To this end, we hypothesized that the PI3K inactivation could be combined with the overactivation of IP3R1 to increase the activity of proteins involved in apoptosis. IP3R1 inactivation can lead to the downregulation of RGS, and reversibly, the overexpression of IP3R1 can lead to increased activity of RGS. Similar to IP3R1, RGS subtype RGS3T has been found to be involved in inducing cell death (Fisher, 2009), and it has also been found that RGS can suppress the PI3K activity downstream of the receptor (Liang et al., 2009). Therefore, the constitutive activation of IP3R1 might also negatively regulate the activity of PI3K. Systemic analysis of the most influential components and their upstream components has led us to identify novel combinations of drug targets. In various studies, combinatorial therapies have shown a decrease in drug resistance in pathogens. In combinatorial therapy, a protein associated with drug resistance can be targeted in combination with different protein of either the same or different pathway (Fischbach, 2011). Clinical trials have also suggested that the efficiency of cytotoxic drugs

increases when given in combinations (Al-Lazikani et al., 2012). If co-occurrence of two genetic events results in cell death, it can be termed as synthetic lethality (Nijman, 2011). The combinatorial perturbation of PI3K and IP3R1 could be considered as synthetically lethal. However, in this perturbation, the activation of IP3R1 is synergistic with the inactivation of PI3K. Upregulation of IP3R1 could be achieved using a targeted drug therapy, such as stress hormone dexamethasone, a synthetic glucocorticoid show to significantly upregulate the expression of IP3R1 in differentiating myoblasts (Chai et al., 2010).

As a validation of model's result, we used previously published gene expression data. Our model's results moderately correlate with this data. This agreement was based on only one dataset of PI3K inhibition with two biological replicates. Further addition of experimental data for other perturbations, including the combinatorial perturbation is required to validate the trends of perturbation analysis in model.

In conclusion, by combining IP3R1 (activation) and PI3K (inactivation), we were able to stimulate cell death under the cell growth stimulating condition. Based on this, one can hypothesize that it might be possible that the decrease in cell proliferation with increased apoptosis as a result of this combinatorial intervention could subsequently increase the rate of clearance of tumor cells, and serve as a novel strategy for important targets associated with drug resistance. However, more laboratory validations will be required to test this hypothesis.

AUTHOR CONTRIBUTIONS

Conceived and designed the experiments: BP, LA, CH, and TH. Performed experiments: LA and MM. Data analysis and interpretation: BP and CH. Wrote the manuscript: BP, LA, CH, and TH.

ACKNOWLEDGMENTS

We would like to thank Resa Helikar for providing feedback on the manuscript.

FUNDING

Office of Sponsored Programs at the University of Nebraska at Omaha *via* Fund for Undergraduate Scholarly Experiences to LA and CH.

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at http://journal.frontiersin.org//article/10.3389/fbioe.2016.00010

TABLE S1 | Functional activity of signal transduction components in model. Gene expression of signal transduction components were queried in four databases: Bgee, CleanEx, Expression Atlas, and Genevisible.

TABLE S2 | Influential ranking of all the components of signal transduction model.

TABLE S3 | Network motifs that containing component pairs selected in Table 5.

TABLE S4 | Significant Z-ratios (cells treated with PI3K-inhibitor/ untreated) of genes of signal transduction components showed activity level changes in PI3K inactivation in model.

TABLE S5 | Activity levels of all the components; on PI3K inactivation (single perturbation), on IP3R1 activation (single perturbation), and on PI3K-IP3R1 (double perturbations).

DATA SHEET 1 | Model file in SBML format.

FIGURES S1–S10 | Heatmaps of raw differences of activity levels between perturbed and WT conditions (for all the simulated environmental conditions under both types of perturbations).

REFERENCES

Al-Lazikani, B., Banerji, U., and Workman, P. (2012). Combinatorial drug therapy for cancer in the post-genomic era. *Nat. Biotechnol.* 30, 679–692. doi:10.1038/nbt.2284

Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., et al. (2000). Gene ontology: tool for the unification of biology. *Nat. Genet.* 25, 25–29. doi:10.1038/75556

Barrett, T., Suzek, T. O., Troup, D. B., Wilhite, S. E., Ngau, W.-C., Ledoux, P., et al. (2005). NCBI GEO: mining millions of expression profiles – database and tools. *Nucleic Acids Res.* 33, D562–D566. doi:10.1093/nar/gki022

Bastian, F., Parmentier, G., Roux, J., Moretti, S., Laudet, V., and Robinson-Rechavi, M. (2008). "Bgee: integrating and comparing heterogeneous transcriptome data among species," in *Data Integration in the Life Sciences*, eds A. Bairoch, S. Cohen-Boulakia, and C. Froidevaux (Berlin; Heidelberg: Springer), 124–131.

Berchtold, M. W., and Villalobo, A. (2014). The many faces of calmodulin in cell proliferation, programmed cell death, autophagy, and cancer. *Biochim. Biophys. Acta* 1843, 398–435. doi:10.1016/j.bbamcr.2013.10.021

Birkeland, E., Wik, E., Mjøs, S., Hoivik, E., Trovik, J., Werner, H. M. J., et al. (2012). KRAS gene amplification and overexpression but not mutation associates with aggressive and metastatic endometrial cancer. *Br. J. Cancer* 107, 1997–2004. doi:10.1038/bjc.2012.477

Blazier, A. S., and Papin, J. A. (2012). Integration of expression data in genome-scale metabolic network reconstructions. *Front. Physiol.* 3:299. doi:10.3389/fphys.2012.00299

Chai, J., Xiong, Q., Zhang, P., Zheng, R., Peng, J., and Jiang, S. (2010). Induction of Ca 2+ signal mediated apoptosis and alteration of IP3R1 and SERCA1 expression levels by stress hormone in differentiating C2C12 myoblasts. *Gen. Comp. Endocrinol.* 166, 241–249. doi:10.1016/j.ygcen.2009.08.011

Cheadle, C., Vawter, M. P., Freed, W. J., and Becker, K. G. (2003). Analysis of microarray data using Z score transformation. *J. Mol. Diagn.* 5, 73–81. doi:10.1016/S1525-1578(10)60455-2

Chen, L., Iijima, M., Tang, M., Landree, M. A., Huang, Y. E., Xiong, Y., et al. (2007). PLA 2 and PI3K/PTEN pathways act in parallel to mediate chemotaxis. *Dev. Cell* 12, 603–614. doi:10.1016/j.devcel.2007.03.005

Chen, W.-H., Minguez, P., Lercher, M. J., and Bork, P. (2012). OGEE: an Online GEne Essentiality database. *Nucleic Acids Res.* 40, D901–D906. doi:10.1093/nar/gkr986

Conroy, B. D., Herek, T. A., Shew, T. D., Latner, M., Larson, J. J., Allen, L., et al. (2014). Design, assessment, and in vivo evaluation of a computational model illustrating the role of CAV1 in CD4+ T-lymphocytes. *Front. Immunol.* 5:599. doi:10.3389/fimmu.2014.00599

Consortium, U. (2011). Reorganizing the protein space at the universal protein resource (UniProt). *Nucleic Acids Res.* 40, D71–D75. doi:10.1093/nar/gkr981

Dienstmann, R., De Dosso, S., Felip, E., and Tabernero, J. (2012). Drug development to overcome resistance to EGFR inhibitors in lung and colorectal cancer. *Mol. Oncol.* 6, 15–26. doi:10.1016/j.molonc.2011.11.009

Eisenberg, E., and Levanon, E. Y. (2013). Human housekeeping genes, revisited. *Trends Genet.* 29, 569–574. doi:10.1016/j.tig.2013.05.010

Fischbach, M. A. (2011). Combination therapies for combating antimicrobial resistance. *Curr. Opin. Microbiol.* 14, 519–523. doi:10.1016/j.mib.2011.08.003

Fisher, R. A. (2009). *Molecular Biology of RGS Proteins*. London: Academic Press.

Flobak, Å., Baudot, A., Remy, E., Thommesen, L., Thieffry, D., Kuiper, M., et al. (2015). Discovery of drug synergies in gastric cancer cells predicted by logical modeling. *PLoS Comput. Biol.* 11:e1004426. doi:10.1371/journal.pcbi.1004426

Flórez, A. F., Park, D., Bhak, J., Kim, B.-C., Kuchinsky, A., Morris, J. H., et al. (2010). Protein network prediction and topological analysis in *Leishmania major* as a tool for drug target selection. *BMC Bioinformatics* 11:484. doi:10.1186/1471-2105-11-484

Folger, O., Jerby, L., Frezza, C., Gottlieb, E., Ruppin, E., and Shlomi, T. (2011). Predicting selective drug targets in cancer through metabolic networks. *Mol. Syst. Biol.* 7, 501. doi:10.1038/msb.2011.35

Ghosh, S., Matsuoka, Y., Asai, Y., Hsin, K.-Y., and Kitano, H. (2011). Software for systems biology: from tools to integrated platforms. *Nat. Rev. Genet.* 12, 821–832. doi:10.1038/nrg3096

Gottesman, M. M. (2002). Mechanisms of cancer drug resistance. *Annu. Rev. Med.* 53, 615–627. doi:10.1146/annurev.med.53.082901.103929

Helikar, T., Konvalina, J., Heidel, J., and Rogers, J. A. (2008). Emergent decision-making in biological signal transduction networks. *Proc. Natl. Acad. Sci. U.S.A* 105, 1913–1918. doi:10.1073/pnas.0705088105

Helikar, T., Kowal, B., McClenathan, S., Bruckner, M., Rowley, T., Madrahimov, A., et al. (2012). The cell collective: toward an open and collaborative approach to systems biology. *BMC Syst. Biol.* 6:96. doi:10.1186/1752-0509-6-96

Helikar, T., Kowal, B., and Rogers, J. A. (2013). A cell simulator platform: the cell collective. *Clin. Pharmacol. Ther.* 93, 393–395. doi:10.1038/clpt.2013.41

Helikar, T., and Rogers, J. A. (2009). ChemChains: a platform for simulation and analysis of biochemical networks aimed to laboratory scientists. *BMC Syst. Biol.* 3:58. doi:10.1186/1752-0509-3-58

Holohan, C., Van Schaeybroeck, S., Longley, D. B., and Johnston, P. G. (2013). Cancer drug resistance: an evolving paradigm. *Nat. Rev. Cancer* 13, 714–726. doi:10.1038/nrc3599

Huang, D. W., Sherman, B. T., and Lempicki, R. A. (2008). Systematic and integrative analysis of large gene lists using DAVID bioinformatics resources. *Nat. Protoc.* 4, 44–57. doi:10.1038/nprot.2008.211

Janssens, V., and Goris, J. (2001). Protein phosphatase 2A: a highly regulated family of serine/threonine phosphatases implicated in cell growth and signalling. *Biochem. J.* 353, 417–439. doi:10.1042/bj3530417

Kanehisa, M. (2002). "The KEGG database," in *In Silico Simulation of Biological Processes: Novartis Foundation Symposium*, Vol. 247. eds G. Bock and J. A. Goode (Chichester: John Wiley & Sons Ltd.), 91–103.

Kayl, A. E., and Meyers, C. A. (2006). Side-effects of chemotherapy and quality of life in ovarian and breast cancer patients. *Curr. Opin. Obstet. Gynecol.* 18, 24–28. doi:10.1097/01.gco.0000192996.20040.24

Kitano, H. (2002a). Computational systems biology. *Nature* 420, 206–210. doi:10.1038/nature01254

Kitano, H. (2002b). Systems biology: a brief overview. *Science* 295, 1662–1664. doi:10.1126/science.1069492

Kobayashi, S., Boggon, T. J., Dayaram, T., Jänne, P. A., Kocher, O., Meyerson, M., et al. (2005). EGFR mutation and resistance of non-small-cell lung cancer to gefitinib. *N. Engl. J. Med.* 352, 786–792. doi:10.1056/NEJMoa044238

Kochi, N., Helikar, T., Allen, L., Rogers, J. A., Wang, Z., and Matache, M. T. (2014). Sensitivity analysis of biological Boolean networks using information fusion based on nonadditive set functions. *BMC Syst. Biol.* 8:92. doi:10.1186/s12918-014-0092-4

Le Novère, N. (2015). Quantitative and logic modelling of molecular and gene networks. *Nat. Rev. Genet.* 16, 146–158. doi:10.1038/nrg3885

Liang, G., Bansal, G., Xie, Z., and Druey, K. M. (2009). RGS16 inhibits breast cancer cell growth by mitigating phosphatidylinositol 3-kinase signaling. *J. Biol. Chem.* 284, 21719–21727. doi:10.1074/jbc.M109.028407

Linardou, H., Dahabreh, I. J., Kanaloupiti, D., Siannis, F., Bafaloukos, D., Kosmidis, P., et al. (2008). Assessment of somatic k-RAS mutations as a mechanism associated with resistance to EGFR-targeted agents: a systematic review and meta-analysis of studies in advanced non-small-cell lung cancer and metastatic colorectal cancer. *Lancet Oncol.* 9, 962–972. doi:10.1016/S1470-2045(08)70206-7

Lito, P., Rosen, N., and Solit, D. B. (2013). Tumor adaptation and resistance to RAF inhibitors. *Nat. Med.* 19, 1401–1409. doi:10.1038/nm.3392

Loscalzo, J., and Barabasi, A. L. (2011). Systems biology and the future of medicine. *Wiley Interdiscip. Rev. Syst. Biol. Med.* 3, 619–627. doi:10.1002/wsbm.144

Lotfi-Jam, K., Carey, M., Jefford, M., Schofield, P., Charleson, C., and Aranda, S. (2008). Nonpharmacologic strategies for managing common chemotherapy adverse effects: a systematic review. *J. Clin. Oncol.* 26, 5618–5629. doi:10.1200/JCO.2007.15.9053

Madrahimov, A., Helikar, T., Kowal, B., Lu, G., and Rogers, J. (2013). Dynamics of influenza virus and human host interactions during infection and replication cycle. *Bull. Math. Biol.* 75, 988–1011. doi:10.1007/s11538-012-9777-2

Mason, R. P. (1999). Effects of calcium channel blockers on cellular apoptosis. *Cancer* 85, 2093–2102. doi:10.1002/(SICI)1097-0142(19990515)85:10<2093::AID-CNCR1>3.0.CO;2-E

Mendelsohn, J. (2001). The epidermal growth factor receptor as a target for cancer therapy. *Endocr. Relat. Cancer* 8, 3–9. doi:10.1677/erc.0.0080003

Molinelli, E. J., Korkut, A., Wang, W., Miller, M. L., Gauthier, N. P., Jing, X., et al. (2013). Perturbation biology: inferring signaling networks in cellular systems. *PLoS Comput. Biol.* 9:e1003290. doi:10.1371/journal.pcbi.1003290

Naldi, A., Carneiro, J., Chaouiya, C., and Thieffry, D. (2010). Diversity and plasticity of Th cell types predicted from regulatory network modelling. *PLoS Comput. Biol.* 6:e1000912. doi:10.1371/journal.pcbi.1000912

Nijman, S. M. (2011). Synthetic lethality: general principles, utility and detection using genetic screens in human cells. *FEBS Lett.* 585, 1–6. doi:10.1016/j.febslet.2010.11.024

Penzo, D., Petronilli, V., Angelin, A., Cusan, C., Colonna, R., Scorrano, L., et al. (2004). Arachidonic acid released by phospholipase A2 activation triggers Ca2+-dependent apoptosis through the mitochondrial pathway. *J. Biol. Chem.* 279, 25219–25225. doi:10.1074/jbc.M310381200

Perna, D., Karreth, F. A., Rust, A. G., Perez-Mancera, P. A., Rashid, M., Iorio, F., et al. (2015). BRAF inhibitor resistance mediated by the AKT pathway in an oncogenic BRAF mouse melanoma model. *Proc. Natl. Acad. Sci. U.S.A* 112, E536–E545. doi:10.1073/pnas.1418163112

Petryszak, R., Burdett, T., Fiorelli, B., Fonseca, N. A., Gonzalez-Porta, M., Hastings, E., et al. (2014). Expression Atlas update – a database of gene and transcript expression from microarray-and sequencing-based functional genomics experiments. *Nucleic Acids Res.* 42, D926–D932. doi:10.1093/nar/gkt1270

Pinton, P., Giorgi, C., Siviero, R., Zecchini, E., and Rizzuto, R. (2008). Calcium and apoptosis: ER-mitochondria Ca2+ transfer in the control of apoptosis. *Oncogene* 27, 6407–6418. doi:10.1038/onc.2008.308

Praz, V., Jagannathan, V., and Bucher, P. (2004). CleanEx: a database of heterogeneous gene expression data based on a consistent gene nomenclature. *Nucleic Acids Res.* 32, D542–D547. doi:10.1093/nar/gkh107

Pruitt, K. D., Tatusova, T., and Maglott, D. R. (2007). NCBI reference sequences (RefSeq): a curated non-redundant sequence database of genomes, transcripts and proteins. *Nucleic Acids Res.* 35, D61–D65. doi:10.1093/nar/gkl842

Puniya, B. L., Kulshreshtha, D., Verma, S. P., Kumar, S., and Ramachandran, S. (2013). Integrated gene co-expression network analysis in the growth phase of *Mycobacterium tuberculosis* reveals new potential drug targets. *Mol. Biosyst.* 9, 2798–2815. doi:10.1039/c3mb70278b

Qu, Y., Chen, Q., Lai, X., Zhu, C., Chen, C., Zhao, X., et al. (2014). SUMOylation of Grb2 enhances the ERK activity by increasing its binding with Sos1. *Mol. Cancer* 13, 95. doi:10.1186/1476-4598-13-95

Rachagani, S., Senapati, S., Chakraborty, S., Ponnusamy, M. P., Kumar, S., Smith, L., et al. (2011). Activated KrasG12D is associated with invasion and metastasis of pancreatic cancer cells through inhibition of E-cadherin. *Br. J. Cancer* 104, 1038–1048. doi:10.1038/bjc.2011.31

Rocha, C., Mendonça, T., and Eduarda Silva, M. (2013). Modelling neuromuscular blockade: a stochastic approach based on clinical data. *Math. Comput. Model. Dyn. Syst.* 19, 540–556. doi:10.1080/13873954.2013.801865

Rodon, J., Dienstmann, R., Serra, V., and Tabernero, J. (2013). Development of PI3K inhibitors: lessons learned from early clinical trials. *Nat. Rev. Clin. Oncol.* 10, 143–153. doi:10.1038/nrclinonc.2013.10

Rosich, L., Montraveta, A., Xargay-Torrent, S., López-Guerra, M., Roldán, J., Aymerich, M., et al. (2014). Dual PI3K/mTOR inhibition is required to effectively impair microenvironment survival signals in mantle cell lymphoma. *Oncotarget* 5, 6788. doi:10.18632/oncotarget.2253

Singh, P., and Singh, A. (2012). Ocular adverse effects of anti-cancer chemotherapy. *J. Cancer Ther. Res.* 1, 5. doi:10.7243/2049-7962-1-5

Song, J., and Singh, M. (2013). From hub proteins to hub modules: the relationship between essentiality and centrality in the yeast interactome at different scales of organization. *PLoS Comput. Biol.* 9:e1002910. doi:10.1371/journal.pcbi.1002910

Steinway, S. N., Zañudo, J. G., Ding, W., Rountree, C. B., Feith, D. J., Loughran, T. P., et al. (2014). Network modeling of TGFβ signaling in hepatocellular carcinoma epithelial-to-mesenchymal transition reveals joint Sonic Hedgehog and Wnt pathway activation. *Cancer Res.* 74, 5963–5977. doi:10.1158/0008-5472.CAN-14-0225

Todd, R. G., and Helikar, T. (2012). Ergodic sets as cell phenotype of budding yeast cell cycle. *PLoS ONE* 7:e45780. doi:10.1371/journal.pone.0045780

Vanneman, M., and Dranoff, G. (2012). Combining immunotherapy and targeted therapies in cancer treatment. *Nat. Rev. Cancer* 12, 237–251. doi:10.1038/nrc3237

Wagle, N., Van Allen, E. M., Treacy, D. J., Frederick, D. T., Cooper, Z. A., Taylor-Weiner, A., et al. (2014). MAP kinase pathway alterations in BRAF-mutant melanoma patients with acquired resistance to combined RAF/MEK inhibition. *Cancer Discov.* 4, 61–68. doi:10.1158/2159-8290.CD-13-0631

Wang, J., Tsang, W. W., and Marsaglia, G. (2003). Evaluating Kolmogorov's distribution. *J. Stat. Softw.* 8, doi:10.18637/jss.v008.i18

Washburn, M. P., Koller, A., Oshiro, G., Ulaszek, R. R., Plouffe, D., Deciu, C., et al. (2003). Protein pathway and complex clustering of correlated mRNA and protein expression analyses in *Saccharomyces cerevisiae*. *Proc. Natl. Acad. Sci. U.S.A* 100, 3107–3112. doi:10.1073/pnas.0634629100

Wazir, U., Jiang, W. G., Sharma, A. K., and Mokbel, K. (2013). Guanine nucleotide binding protein β 1: a novel transduction protein with a possible role in human breast cancer. *Cancer Genomics Proteomics* 10, 69–73. doi:10.1016/j.ejso.2012.07.172

Weinstein, J. N., Collisson, E. A., Mills, G. B., Shaw, K. R. M., Ozenberger, B. A., Ellrott, K., et al. (2013). The Cancer Genome Atlas pan-cancer analysis project. *Nat. Genet.* 45, 1113–1120. doi:10.1038/ng.2764

Wernicke, S., and Rasche, F. (2006). FANMOD: a tool for fast network motif detection. *Bioinformatics* 22, 1152–1153. doi:10.1093/bioinformatics/btl038

West, K. A., Castillo, S. S., and Dennis, P. A. (2002). Activation of the PI3K/Akt pathway and chemotherapeutic resistance. *Drug. Resist. Updat.* 5, 234–248. doi:10.1016/S1368-7646(02)00120-6

Wheeler, D. L., Dunn, E. F., and Harari, P. M. (2010). Understanding resistance to EGFR inhibitors – impact on future treatment strategies. *Nat. Rev. Clin. Oncol.* 7, 493–507. doi:10.1038/nrclinonc.2010.97

Wishart, D. S., Knox, C., Guo, A. C., Shrivastava, S., Hassanali, M., Stothard, P., et al. (2006). DrugBank: a comprehensive resource for in silico drug discovery and exploration. *Nucleic Acids Res.* 34, D668–D672. doi:10.1093/nar/gkj067

Yao, L., Evans, J. A., and Rzhetsky, A. (2010). Novel opportunities for computational biology and sociology in drug discovery: corrected paper. *Trends Biotechnol.* 28, 161–170. doi:10.1016/j.tibtech.2010.01.004

Zimmermann, P., Hirsch-Hoffmann, M., Hennig, L., and Gruissem, W. (2004). GENEVESTIGATOR. Arabidopsis microarray database and analysis toolbox. *Plant Physiol.* 136, 2621–2632. doi:10.1104/pp.104.046367

Conflict of Interest Statement: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Tomáš Helikar has served as a scientific advisor and/or consultant to Discovery Collective.

frontiers in
BIOENGINEERING AND BIOTECHNOLOGY

ORIGINAL RESEARCH ARTICLE
published: 16 January 2015
doi: 10.3389/fbioe.2014.00081

Learning delayed influences of biological systems

Tony Ribeiro[1]*, Morgan Magnin[2,3], Katsumi Inoue[1,2] and Chiaki Sakama[4]

[1] *The Graduate University for Advanced Studies (Sokendai), Tokyo, Japan*
[2] *National Institute of Informatics, Tokyo, Japan*
[3] *Institut de Recherche en Communications et Cybernétique de Nantes (IRCCyN), Nantes, France*
[4] *Department of Computer and Communication Sciences, Wakayama University, Wakayama, Japan*

Edited by:
David A. Rosenblueth, Universidad Nacional Autonoma de Mexico, Mexico

Reviewed by:
Jérôme Feret, Institut National de Recherche en Informatique et Automatique (INRIA), France
Ricardo Gonçalves, Universidade Nova de Lisboa, Portugal

***Correspondence:**
Tony Ribeiro, National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo 101-8430, Japan
e-mail: tony_ribeiro@nii.ac.jp

Boolean networks are widely used model to represent gene interactions and global dynamical behavior of gene regulatory networks. To understand the memory effect involved in some interactions between biological components, it is necessary to include delayed influences in the model. In this paper, we present a logical method to learn such models from sequences of gene expression data. This method analyzes each sequence one by one to iteratively construct a Boolean network that captures the dynamics of these observations. To illustrate the merits of this approach, we apply it to learning real data from bioinformatic literature. Using data from the yeast cell cycle, we give experimental results and show the scalability of the method. We show empirically that using this method we can handle millions of observations and successfully capture delayed influences of Boolean networks.

Keywords: Boolean network, gene regulatory networks, delayed influences, time delay, logic programming, machine learning, state transitions

1. INTRODUCTION

1.1. IMMEDIATE VERSUS DELAYED INFLUENCES

Thanks to the development of recent high-throughput measurement technologies such as DNA microarrays, biologists succeed in obtaining a large amount of gene expression profiles. It then becomes crucial to be able to connect the data and build a predictive model of the gene network. The analysis of biological networks often requires agreeing on an appropriate mathematical or computational model to represent the biological system. Because of the complexity of the system, models usually assume that the modification of one node results in an immediate activation (or inhibition) of its targeted nodes. But this hypothesis is generally unfair: some influences may take some time to operate, thus modifying the behavior of the model. These delayed influences can play a major role in various biological systems of crucial importance, like the mammalian circadian clock [as illustrated by Comet et al. (2012)] or the DNA damage repair [as shown by Abou-Jaoudé et al. (2009)]. We especially need to capture the memory of the system, i.e., keep track of the previous steps.

1.2. MODELING DELAYED INFLUENCES INTO BOOLEAN NETWORKS

Delayed influences have been integrated in each of the most well-known formalisms to model gene regulatory networks. Thanks to their structure, which allows to model both sequentiality and parallelism, Petri nets were able to model complex regulation mechanisms (Chaouiya, 2007). Recent works even considered not only discrete but continuous delays (Siebert and Bockmayr, 2006; Comet et al., 2010) in some hybrid automata paradigms. However, such approaches, because of their complexity, fail to deal with large systems and biological data about quantitative time delays are generally scarce. That is why we chose to focus, in this paper, on Boolean networks, which have proven to be a simple, yet powerful, framework to model and analyze the dynamics of gene regulatory networks. The classical dynamics of Boolean networks is based on the central assumption that a homogeneous transmission delay is involved among all components of the network. This means the modification of one node results in an immediate activation (or inhibition) of its targeted nodes [as studied, e.g., by Akutsu et al. (2003)] for the sake of simplicity. This is quite unrealistic in the sense that, in a real biochemical system, the evolution happens at various time scales.

The urge to incorporate delays into the model is perfectly illustrated by the feedforward loop scheme. The feedforward loop is a network pattern that appears in many cycling processes, e.g., *Escherichia coli* and *Yeast Saccharomyces cerevisiae* [as considered by Koh et al. (2009)]. Biologists like Mangan and Alon (2003) assume these loops play a major role in the acceleration of the response time of transcriptional networks. It consists of the following elements (see **Figure 1**): 3 genes, let us say *a*, *b*, and *c*, with *a* regulating *b*, *b* regulating *c*, and a direct regulation from *a* to *c*. Depending on the nature of the regulations (activations or inhibitions; **Figure 1** arbitrarily considers an inhibition from *a* to *c*) and their delays, the concurrence between the direct regulation from *a* onto *c* and the indirect one through *b* can lead to a drastic different behavior. To analyze feedforward loops, the information about the respective delays of the regulations at stake are crucial. A small change in the delays understanding may lead to a complete different behavior. The thin analysis of the dynamical behavior of such pattern requires to enrich classical discrete models with delays.

To address this issue, different approaches have been designed. The most well-known one is due to Silvescu and Honavar (2001). To understand precisely the dynamics of some biological processes (like cell development) while considering the memory of the system, the authors take their inspiration from the works about Markov models. They introduced an extension of Boolean Networks from a Markov(1) to Markov(k) model, where

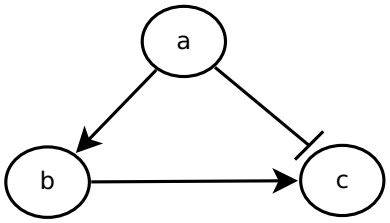


FIGURE 1 | Example of a feedforward loop.

k is the number of time steps during which a gene can influence another gene. This extension is called temporal Boolean networks, abridged as $TBN(n, m, k)$, with n the number of genes and the expression of each gene at time $t+1$ being controlled by a Boolean function of the expression levels of at most m genes at times in $\{t, t-1, \ldots, t-(k+1)\}$. They even extend the formalism to multi-valued discrete networks, calling it temporal discrete networks, $TDN(n, m, k, D)$, where each gene can be expressed at levels 0, 1, ..., $D-1$. Their main results, however, focus on $TBN(n, m, k)$: they design a decision tree learning algorithm that infers a temporal Boolean network from time series data. They also give some bounds on the size of the necessary data to infer temporal Boolean networks. They illustrate their results through artificially generated networks, claiming that the main limit of their method is the lack of real data on large datasets.

Other authors addressed the idea of modeling delayed and indirect influences. We can cite the work of Chueh and Lu (2012), who extended the Boolean network formalism with delays. To model that the induction of a Boolean function may not activate immediately the targeted gene, they replace the classical deterministic relation between the Boolean function and the targeted gene by two relationships: (i) the prerequisite function: it represents the fact that the on-status of the target gene at time $t+1$ requires that the Boolean function at time t is on. (ii) The similarity function: the Boolean function and the target gene are said to be similar if the status of the Boolean function and the status of the target gene are in the same expression. In other words, this means that the classical Boolean operators are encoded in the prerequisite function, while the similarity function allows to model precedence (under the form of delays) between concurrent updates.

Another approach to model indirect influences is given in the 6th chapter of the dissertation by Ghanbarnejad (2011). The author recalls that the time passing between production of a regulating molecule and its binding to a target site depends both on the molecule and its target site. That is why he decides to study the dynamics in such a way that:

$$x_i(t) = f_i(x_1(t-\tau_{i1}), x_2(t-\tau_{i2}), \ldots, x_N(t-\tau_{iN}))$$

with x_i the value of gene i, t the current time step, τ_{ij} the delay for the interaction between a source node j and a target node i. For the purpose of his research, the author draws the delay τ_{ij} as a random integer from a flat distribution on $\{1, 2\bar{\tau}-1\}$ for each pair of nodes i and j, the average delay $\bar{\tau}$ being a tunable parameter. That is introduced as Boolean networks with distributed delays. The semantics here is synchronous, thus very similar to what we aim at.

As these works derive of the seminal formalism proposed by Silvescu and Honavar (2001), we will consider $TBN(n, m, k)$ in this paper and discuss a new learning algorithm.

1.3. LEARNING BOOLEAN NETWORKS WITH DELAYED INFLUENCES

Various approaches have been recently designed to tackle the reverse engineering of gene regulatory networks from expression data. This has led to the emergence of the so-called executable biology, whose goal is to provide formal methods to automatically synthesize models from experiments (Koksal et al., 2013). Most of them are only static. But there has been a growing interest for inference algorithms that incorporate temporal aspects. Koh et al. (2009) recently studied the relevance of these various algorithms. Liu et al. (2004) proposed to infer time-delayed gene regulatory networks through Bayesian networks. Lopes and Bontempi (2013) showed that the inference algorithms that include temporal features perform better than static ones. The main issue is then to be able to infer the appropriate temporal delays between the influences at stake. As this is a hard problem, Zhang (2008) claimed that the key issue when analyzing time series data consists in segmenting time series data in different successive phases. Their contribution then focuses on solving this segmentation problem and shows the merits of their approaches on various case studies.

All these approaches consider gene expression data as input and infer the associated regulations. One common problem of discrete approaches taking expression data as input lies in the determination of a relevant threshold to define the inactive and active states of gene expression. To position this hypothesis in the context of existing approaches to process raw biological data, let us cite the works of some authors, like Soinov et al. (2003), who proposed an alternative methodology that considers not a concentration level, but the way the concentration is changed in the presence/absence of one regulator. The other major modeling problem depends on the quality of the expression data. In other words, noisy data may lead to errors in the inference process. For example, when a gene is expressed at a low level, a low signal-to-noise ratio would result in an inaccurate measurement of the behavior of the gene.

The pre-processing of the data is really critical to the relevance of the inferred relations between components. In this paper, we assume our input data has already been pre-processed and resulted in a reliable set of state-transitions information.

Aside from these intrinsic modeling issues, the existing learning approaches share some computational limitations:

- Because of the complexity of the problem, the size of the inferred model is limited: the inferred gene regulatory network has to be composed of less than 15 components and the memory effect cannot take into account more than $k=15$ steps.
- Many approaches fail when the network involves cyclic interactions.

1.4. OUR CONTRIBUTION

In this paper, we focus on the logical approach to learn gene regulatory networks with delays from an existing knowledge that is

expressed through a set of state transitions. As mentioned in the previous subsection, we assume there has been a pre-processing of the time series data.

In previous works, we exhibited the links between logic programs and Boolean networks on the one hand (Inoue, 2011; Inoue and Sakama, 2012), designed an algorithm that is able to learn Markov(1) state transition systems on the other hand (Inoue et al., 2014). While existing works did not allow to capture the delayed influences between components, this paper designs an algorithm that takes multiple sequences of state transitions as input and builds a logic program that capture the delayed dynamics of a Markov(k) system.

This can be seen as an extension of previous results in the following sense: in Inoue (2011) and Inoue and Sakama (2012), Markov(1) state transition systems are represented with logic programs, in which the state of the world is represented by a Herbrand interpretation and the dynamics that rule the environment changes are represented by a logic program P. The rules in P specify the next state of the world as a Herbrand interpretation through the *immediate consequence operator* (also called the T_P *operator*) [as introduced by Van Emden and Kowalski (1976) and Apt et al. (1988)]. With such a background, Inoue et al. (2014) have recently proposed a framework to learn logic programs from traces of interpretation transitions (LFIT). We extended this body of research: while the previous algorithm dealt only with 1-step transitions (i.e., we assume the state of the system at time t depends only of its state at time $t-1$), we propose here an approach that is able to consider k-step transitions (sequence of at most k state transitions). This means that we are now able to capture delayed influences in the *inductive logic programing* methodology.

1.5. OUTLINE OF THE PAPER

The paper is organized as follows: Section 2 reviews the logical background of this work, and summarizes the main ideas behind the existing LF1T algorithm in order to make its extension to Markov(k) models (i.e., with delayed influences) in Section 3 be more understandable. In Section 4, we apply our methodology to some case studies and highlight its scalability. Finally, we discuss these results and further works in Section 5.

2. BACKGROUND

2.1. BOOLEAN NETWORK

A Boolean network is a simple discrete representation widely used in bioinformatics (Kauffman, 1969; Lähdesmäki et al., 2003; Klamt et al., 2006). A *Boolean network* (Kauffman, 1969) is a pair (N, F) with $N = \{n_1, \ldots, n_k\}$, a finite set of nodes (or variables), and $F = \{f_1, \ldots, f_k\}$, a corresponding set of Boolean functions $f_i : \mathbb{B}^n \to \mathbb{B}$, with $\mathbb{B} = \{0, 1\}$. $n_t(t)$ represents the value of n_i at time step t, and equals either 1 (expressed) or 0 (not expressed). A vector (or *state*) $s(t) = (n_1(t), ..., n_k(t)$ is the expression of the nodes of N at time step t. There are 2^k possible distinct states for each time step. The state of a node n_i at the next time step $t+1$ is determined by $n_i(t+1) = f_i(n_{i_1}(t), \ldots, n_{i_p}(t))$ with $n_{i_1}, \ldots, n_{i_p}$ the nodes directly influencing n_i, called *regulation nodes* of n_i. A Boolean network can be represented by its *interaction graph* (see **Figure 2** left), but its precise regulation relations can only be represented by the Boolean function (see Example 1). For each Boolean network, there is the *state transition diagram* (see **Figure 2** right), which represents the transitions between $n_i(t)$ and $n_i(t+1)$. In the case of a gene regulatory network, nodes represent genes and Boolean functions represent their relations.

Example 1: **Figure 2** shows the interaction graph and the state transitions diagram of a Boolean network B_1 composed of the three following variables: {a, b, c}. The Boolean functions of B_1 are f_a, f_b, and f_c, which are, respectively, the following Boolean functions of a, b, and c:

$$f_a = \neg a \wedge (b \vee c)\,, f_b = a \wedge c, f_c = \neg a$$

Let us consider that the Boolean network B_1, whose graph is depicted in **Figure 2**, is a gene regulatory network so that a, b, and c are genes. According to the interaction graph of B_1: a is not only an *activator* of b and an *inhibitor* of c but also its own inhibitor. The gene b is an activator of a, and the gene c is activator of both a and b. According to the Boolean functions of B_1 in Example 1, to activate a, either b or c has to be present but if a is present, it will prevent its own expression at the next step (f_a). The activation of b requires both a and c to be expressed at the same time step; if one of them is not expressed at time step t then b will not be expressed at $t+1$ (f_b). The presence of a is enough to prevent the expression of c, so that if a is expressed at time step t then c will not be expressed at $t+1$ (f_c).

It is straightforward to generate the state transition diagram from the Boolean functions. Learning from interpretation transition (LFIT) tackles the inverse problem: infer the Boolean function from state transitions. In a Boolean network, the value of nodes can be updated synchronously or asynchronously. In a *synchronous Boolean network*, all nodes are updated at the same

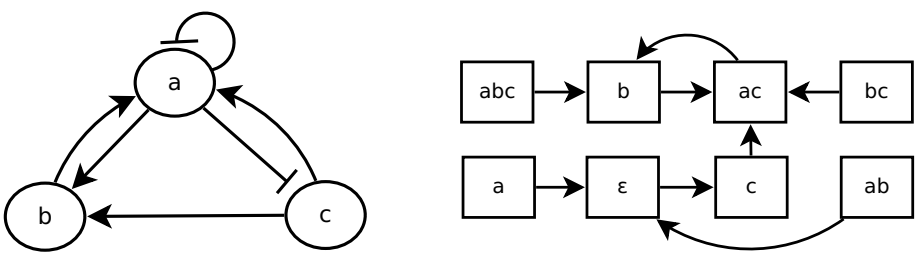


FIGURE 2 | A Boolean network B_1 (left) and its state transition diagram (right).

time. The successive sequence of states during an execution, called *trajectory* of a Boolean network, is deterministic in a synchronous Boolean network. In an *asynchronous Boolean network*, a node may not be updated at a time, so that its state transitions can be non-deterministic. In this paper, we deal only with synchronous ones.

2.2. LOGIC PROGRAMING

In this subsection, we recall some preliminaries of logic programing. We consider a propositional language L that is built from a finite set of propositional constants $p, q, r, \ldots$ and the logical connectives $\neg$, $\wedge$ and $\leftarrow$. A propositional constant p is also called an *atom* and $\neg p$ is negation of p. p and $\neg p$ are called *literals*.

In this paper, we consider a *logic program* (simply called a program) as a set of *rules* of the form

$$p \leftarrow p_1 \wedge \cdots \wedge p_m \wedge \neg p_{m+1} \wedge \cdots \wedge \neg p_n \quad (1)$$

where p and p_i's are atoms ($n \geq m \geq 1$). For any rule R of the form (1), the atom p is called the *head* of R and is denoted as $h(R)$, and the conjunction to the right of $\leftarrow$ is called the *body* of R. We represent the set of literals in the body of R of the form (1) as $b(R) = \{p_1, \ldots, p_m, \neg p_{m+1}, \ldots, \neg p_n\}$, and the atoms appearing in the body of R positively and negatively as $b^+(R) = \{p_1, \ldots, p_m\}$ and $b^-(R) = \{p_{m+1}, \ldots, p_n\}$, respectively. A rule R of the form (1) is interpreted as follows: $h(R)$ is true if all elements of $b^+(R)$ are true and none of the elements of $b^-(R)$ is true. When $b^+(R) = b^-(R) = \varnothing$, the rule is called a *fact rule*. The rule (1) is a *Horn clause* iff m = n.

Definition 1 (Herbrand base): the Herbrand Base of a program P, denoted by $\mathcal{B}$, is the set of all atoms in the language of P.

Definition 2 (Interpretation): let $\mathcal{B}$ be the Herbrand Base of a logic program P. An *interpretation* is a subset of $\mathcal{B}$. If an interpretation is the empty set, it is denoted by ϵ.

Definition 3 (Model): an interpretation I is a *model* of a program P if $b^+(R) \subseteq I$ and $b^-(R) \cap I = \emptyset$ imply $h(R) \in I$ for every rule R in P.

For a logic program P and an interpretation I, the *immediate consequence operator* (or T_P *operator*) (Apt et al., 1988) is the mapping $T_P : 2^{\mathcal{B}} \rightarrow 2^{\mathcal{B}}$:

$$T_P(I) = \left\{ h(R) \mid R \in P,\ b^+(R) \subseteq I,\ b^-(R) \cap I = \varnothing \right\}. \quad (2)$$

In the rest of this paper, we represent the state transitions of a logic program P as a set of pairs of interpretations $(I, T_P(I))$.

Definition 4 (Consistency): let R be a rule and (I, J) be a state transition. R is *consistent* with (I, J) iff $b^+(R) \subseteq I$ and $b^-(R) \cap I = \emptyset$ imply $h(R) \in J$. Let E be a set of state transitions, R is *consistent* with E if R is consistent with all state transitions of E. A logic program P is *consistent* with E if all rules of P are *consistent* with E.

Definition 5 (Subsumption): let R_1 and R_2 be two rules. If $h(R_1) = h(R_2)$ and $b(R_1) \subseteq b(R_2)$ then R_1 *subsumes* R_2. Let P be a logic program and R be a rule. If there exists a rule $R' \in P$ that subsumes R then P *subsumes* R.

We say that a rule R_1 is *more general* than another rule R_2 if R_1 subsumes R_2.

Example 2: let R_1 and R_2 be the two following rules: $R_1 = (a \leftarrow b)$, $R_2 = (a \leftarrow a \wedge b)$, R_1 subsumes R_2 because $(b(R_1) = \{b\}) \subset (b(R_2) = \{a, b\})$. When R_1 appears in a logic program P, R_2 is useless for P, because whenever R_2 can be applied, R_1 can be applied.

2.3. LEARNING FROM INTERPRETATION TRANSITIONS

LF1T (Inoue et al., 2014) is an *any time algorithm* that takes a set of one-step state transitions E as input. These one-step state transitions can be considered as positive examples. From these transitions, the algorithm learns a logic program P that represents the dynamics of E. To perform this learning process, we can iteratively consider one-step transitions. When the state transition diagram in **Figure 2** is given as input to *LF1T*, it can learn the Boolean network B_1.

In *LF1T*, the set of all atoms $\mathcal{B}$ is assumed to be finite. In the input E, a state transition is represented by a pair of interpretations (subset of $\mathcal{B}$). The output of *LF1T* is a logic program that realizes all state transitions of E.

Learning from 1-step transitions (LF1T)

Input: $E \subseteq 2^{\mathcal{B}} \times 2^{\mathcal{B}}$: (positive) examples/observations.
Output: A logic program P such that $J = T_P(I)$ holds for any $(I, J) \in E$.

To build a logic program with *LF1T*, we use a bottom-up method that generates hypotheses by *specialization* from the most general rules that are fact rules, until the logic program is consistent with all input state transitions. Learning by specialization ensures to output the most general valid hypothesis (Ribeiro and Inoue, 2014). Here, the notion of prime implicant is used to define minimality of logic programs. We consider that the logic program learned by **LF1T** is minimal if the body of each rule constitutes a prime implicant to infer the head.

Definition 6 (Prime implicant condition): let R be a rule and E be a set of state transitions such that R is consistent with E. $b(R)$ is a prime implicant condition of $h(R)$ for E if there is no rule R' such that $b(R') \subset b(R)$ and R' is consistent with E. Let P be a logic program such that $P \cup \{R\} \equiv P$: all models of $P \cup \{R\}$ are models of P and vice versa. $b(R)$ is a prime implicant condition of $h(R)$ for P if there is no rule R' such that $P \cup \{R'\} \equiv P$ and $b(R') \subset b(R)$.

For the sake of simplicity, according to Definition 3, we will call R a minimal rule of E (resp. P) if $b(R)$ is a *prime implicant condition* of $h(R)$ for E (resp. P). For any atom p, the most general minimal rule is the rule with an empty body ($p \leftarrow$) that states that the variable is always true in the next state, i.e., a fact.

Example 3: Let R_1, R_2, and R_3 be three rules and E be the set of state transitions of **Figure 2** as follows: $R_1 = a \leftarrow a \wedge b \wedge c$, $R_2 = a \leftarrow a \wedge b$, $R_3 = a \leftarrow b$. The only rule more general than R_3 is $R' = a$, but R' is not consistent with $(a, \epsilon) \in E$ so that R_3 is a minimal rule for E. Since R_3 subsumes both R_1 and R_2, they are not minimal rules of E. Let P be the logic program $\{a \leftarrow b, b \leftarrow a \wedge c, c \leftarrow \neg a\}$. R_3 is a minimal rule of P because P realizes E and R_3 is minimal for E.

In *Inductive Logic Programing*, refinement operators usually add a set of literals to the body of a rule to make it more specific (Muggleton and De Raedt, 1994). It is a way to revise the current knowledge to make it consistent with new information.

Similarly, in this algorithm, when a rule is not consistent with the observations, we refine it by adding literals into its body.

Definition 7 (Minimal specialization): let R_1 and R_2 be two rules such that $h(R_1) = h(R_2)$ and R_1 subsumes R_2. The minimal specialization $ms(R_1, R_2)$ of R_1 over R_2 is

$$ms(R_1, R_2) = \{h(R1) \leftarrow b(R1) \wedge \neg l_i | l_i \in b(R2) \setminus b(R1)\}$$

Minimal specialization can be used on a rule R to avoid the subsumption of another rule with a minimal reduction of the generality of R. By extension, minimal specialization can be used on the rules of a logic program P to avoid the subsumption of a rule with a minimal reduction of the generality of P. Let P be a logic program, R be a rule and S be the set of all rules of P that subsume R. The minimal specialization $ms(P, R)$ of P by R is as follows:

$$ms(P, R) = (P \setminus S) \cup (\bigcup_{R_P \in S} ms(R_P, R))$$

LF1T starts with an initial logic program $P = \{p \leftarrow | \ p \in \mathcal{B}\}$. Then **LF1T** iteratively analyzes each transition $(I, J) \in E$. For each variable A that **does not appear** in J, **LF1T** infers an **anti-rule** R_A^I:

$$R_A^I = A \leftarrow \bigwedge_{B_i \in I} B_i \wedge \bigwedge_{C_j \in (\mathcal{B} \setminus I)} \neg C_j$$

A rule of P that subsumes such an anti-rule is not consistent with the transitions of E and must be revised. The idea is to use minimal specialization to make P consistent with the new transitions (I, J) by avoiding the subsumption of all anti-rules R_A^I inferred from (I, J). After minimal specialization, P becomes consistent with the new transition while remaining consistent with all previously analyzed transitions. When all transitions of E have been analyzed, **LF1T** outputs the rules of the system that realize E.

3. LEARNING MARKOV(K) SYSTEMS

In order to learn Markov(k) when $k > 1$, we need to extend the **LF1T**. To achieve this goal, we introduce the **LFkT** algorithm. While only its essence was presented in Ribeiro et al. (2014), we formalize, in this section, the corresponding ideas and, in the next section, illustrate its merits on biological case studies taken from the literature.

3.1. FORMALIZATION

Definition 8 (Timed Herbrand base): let P be a logic program. Let $\mathcal{B}$ be the Herbrand base of P and k be a natural number. The timed Herbrand base of P (with period k) denoted by $\mathcal{B}_k$, is as follows:

$$\mathcal{B}_k = \bigcup_{i=1}^{k} \{v_{t-i} | v \in \mathcal{B}\}$$

where t is a constant term, which represents the current time step.

According to Definition 1, given a propositional atom v, v_j is a new propositional atom for each $j = t - i$ $(0 \leq i \leq k)$. A Markov(k) system can then be interpreted as a logic program as follows.

Definition 9 (Markov(k) system): let P be a logic program, $\mathcal{B}$ be the Herbrand base of P and $\mathcal{B}_k$ be the timed Herbrand base of P with period k. A *Markov(k) system* S with respect to P is a logic program where for all rules $R \in S$, $h(R) \in \mathcal{B}$ and all atoms appearing in $b(R)$ belong to $\mathcal{B}_k$.

In a Markov(k) system S, the atoms that appear in the body of the rules represent the value of the atoms that appear in the head, but at previous time steps. In a context of modeling gene regulatory networks, these latter atoms represent the concentration of the interacting genes. This concentration is abstracted as a Boolean value modeling the fact that it is lower or greater than a threshold.

Example 4: Let R_1 and R_2 be two rules, $R_1 = a \leftarrow b_{t-1} \wedge b_{t-2}$, $R_2 = b \leftarrow a_{t-2} \wedge \neg b_{t-2}$. The logic program $S = \{R_1, R_2\}$ is a Markov(2) system, i.e., the state of the system depends on the two previous states. The value of a is true at time step t only if b was true at $t-1$ and $t-2$. The value of b is true at time step t only if a was true at $t-2$ and b was false at $t-2$. The atoms that appear in the head of the rules of S is $\{a, b\}$. $\mathcal{B}_1$ represents these atoms from time step $t-1$: $\mathcal{B}_1 = \{a_{t-1}, b_{t-1}\}$ and $\mathcal{B}_2$ represents these atoms from time step $t-2$: $\mathcal{B}_2 = \{a_{t-1}, b_{t-1}, a_{t-2}, b_{t-2}\}$.

In the following definitions, we refer to $\mathbb{N}$ as the set of all natural numbers.

Definition 10 (Trace of execution): let B be the atoms that appear in the head of the rules of a Markov(k) system S. A *trace of execution* T is a finite sequence of states of S: $T = (S_0, \ldots, S_n)$, $n \geq 1$, $\forall i \in N$, $i \leq n$, $S_i \in 2^B$. For all $j \in \mathbb{N}$, we define:

$$prev\,(i, j, T) = \begin{cases} \varnothing & \text{if } i = 0 \text{ or } j = 0, \\ (S_{i-j-1}, \ldots, S_{i-1}) & \text{if } j + 1 \leq i \\ (S_0, \ldots, S_{i-1}) & \text{otherwise.} \end{cases}$$

We also define $prev(i, T) = prev(i, n, T)$ and $next(i', T) = S_{i'+1}$, $i' \in \mathbb{N}$, $i' < n$.

We denote by $|T|$ the size of the trace that is the number of elements of the sequence. A sub-trace of size m of a trace of execution T is a sub-sequence of consecutive states of T of size m, where $m \in \mathbb{N}$, $1 < m \leq |T|$. In the following, we will also denote $T = (S_0, \ldots, S_n)$ as $T = S_0 \rightarrow \ldots \rightarrow S_n$.

Definition 11 (Consistent traces): let $T = (S_0, \ldots, S_n)$ and $T' = (S'_0, \ldots, S'_m)$ be two traces of execution. T and T' are *k-consistent*, with $k \in \mathbb{N}$, iff $\forall i, j \in \mathbb{N}$, $i < n$, $j < m$, $S_i = S_j$ and $next(i, T) \neq next(j, T')$ imply $prev(i, k, T) \neq prev(j, k, T')$. T and T' are said *consistent* iff they are max(n, m) consistent.

As shown in **Figure 3**, a Markov(k) system may seem nondeterministic when it is represented by a state transition diagram (right part of the figure). That is because such state transition diagram only represents 1-step transitions. In this example, the transition from the state b is not Markov(1): the next state can be either a, b or $\in$. But it can be Markov(2), because all traces of size 2 of **Figure 3** are consistent.

Definition 12 (k-step interpretation transitions): let P be a logic program, $\mathcal{B}$ be the Herbrand base of P and $\mathcal{B}_k$ be the timed Herbrand base of P with period k. Let S be a Markov(k') system w.r.t P, $k' \geq k$. A *k-step interpretation transition* is a pair of interpretations (I, J) where $J \subseteq \mathcal{B}_k$ and $J \subseteq \mathcal{B}$.

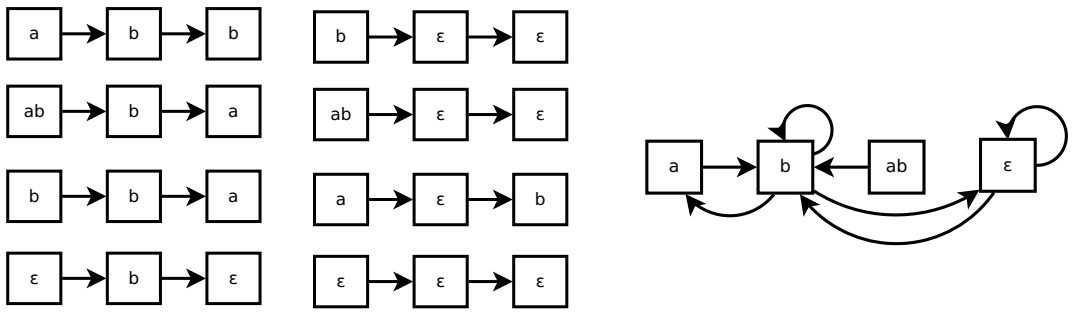


FIGURE 3 | Eight traces of executions of the system of Example 4 (left) and the corresponding state transitions diagram (right).

Example 5: The trace $ab \rightarrow b \rightarrow a$ can be interpreted in the following three ways:

- $(a_{t-2}b_{t-2}b_{t-1},\ a)$: the 2-step interpretation transition that corresponds to the full trace $ab \rightarrow b \rightarrow a$.
- $(a_{t-1}b_{t-1},\ b)$: the 1-step interpretation transition corresponding to the sub-trace $ab \rightarrow b$.
- $(b_{t-1},\ a)$: the 1-step interpretation transition that corresponds to the sub-trace $b \rightarrow a$.

Definition 13 (Extended consistency): let R be a rule and (I, J) be a k-step interpretation transition. R is *consistent* with (I, J) iff $b^+(R) \subseteq I$ and $b^-(R) \cap I = \varnothing$ imply $h(R) \in J$. Let T be a sequence of state transitions, R is *consistent* with T if it is consistent with every k-step interpretation transitions of T. Let O be a set of sequences of state transitions, R is *consistent* with O if R is consistent with all $T' \in O$.

3.2. ALGORITHM

Here, we briefly summarize the essence of LFkT. Because of the lack of space, the details of the algorithm, its pseudo-code, and the proofs of correctness are given as Supplementary Material. We refer to the pseudo-code of the appendix as follows: (algo.N l.x-y) for Algorithm N, line x to y. **LFkT** is an algorithm that can learn the dynamics of a Markov(k) system from its traces of executions. **LFkT** takes a set of traces of executions O as input, where each trace is a sequence of state transitions. If all traces are consistent, the algorithm outputs a logic program P that realizes all transitions of O. The learned influences can be at most k-step relations, where k is the size of the longest trace of O. The main idea is to extract n-step interpretation transitions, $1 \leq n \leq k$, from the traces of executions of the system. Transforming the traces into pairs of interpretations allows us to use minimal specialization (Ribeiro and Inoue, 2014) to iteratively learn the dynamics of the system.

LFkT:

- **Input**: A set of traces of execution E of a Markov(k) system S.
- Step 1: Initialize k logic programs with facts rules.
- Step 2: Convert the input traces of executions into interpretation transitions.
- Step 3: Revise iteratively the logic programs by all interpretation transitions using minimal specialization.
- Step 4: Merge all logic programs into one.
- **Output**: The rules of S, which generated E.

The idea of the algorithm is to start with the most general rules (algo.1 l.6-10) and use specialization to make them consistent with the input observations (algo.2). The algorithm analyzes each interpretation transition one by one and revises the learned rules when they are not consistent (algo.1 l.13-23). In the following, we will call an n-step rule any rule from the logic program learned from n-step transitions.

After analyzing all interpretation transitions, the programs that have been learned are merged into a unique logic program (algo.1 l.24-29). This operation ensures that the rules outputted are consistent with all observations. It can be checked by comparing each rule with other logic programs. If an n-step rule R is more general than an n'-step rules R', $n' < n$, then R is not consistent with the observations from which R' has been learned. To avoid this case, we can remove n-step rules that have no variable of the form v_{t-n}. Indeed, if such rules are consistent with the observations, then they should also have been learned from $(n-1)$-step rules. Finally, **LFkT** outputs a logic program that realizes all consistent traces of execution of O.

4. EVALUATION AND BIOLOGICAL CASE STUDY

In the previous subsections, we have illustrated step by step how the **LFkT** algorithm is able to learn Markov(k) systems. To illustrate the merits of our work, we now apply this approach to the analysis of the yeast cell cycle dataset from Spellman et al. (1998) and Cho et al. (1998), which have been previously analyzed in Li et al. (2006). In this paper, Li et al. tackle the inference of gene regulatory networks from temporal gene expression data. The originality of their work lies in the fact they consider delayed correlations between genes. The methodology can capture gene regulations that are delayed of k time units. The limits of the approach is that the authors only consider pairwise overlaps of expression levels shifted in time relative to each other. Another limit of the approach is that it is not able to make a distinction between a causal gene–gene regulation and the scenarios where two genes, A and B, are being co-regulated by a third gene C: do we have A that regulates B that regulates C, or is it a co-operation between A and B that regulates C?

Here, starting from a set of different traces coming from the yeast cell cycle system, we have performed various experiments

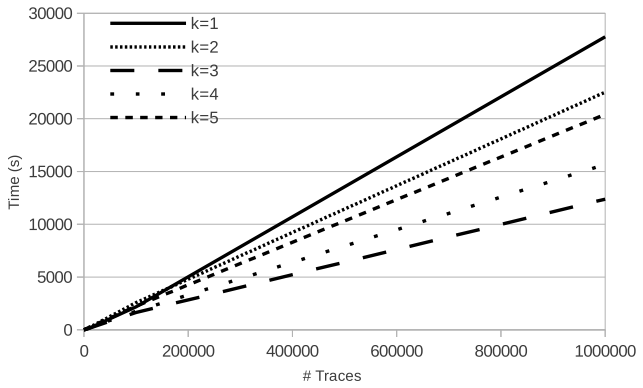


FIGURE 4 | LFkT run time varying the input size (number of traces).

where we have tuned the number of traces that have been considered on the one hand, the value of k (i.e., the number of time steps representing the memory of the system) on the other hand.

Figure 4 shows the evolution of run time of learning with **LFkT** on the five Boolean networks of the yeast cell cycle proposed by Li et al. (2006). These fives programs are, respectively, Markov(1) to Markov(5). In these experiments, for each Boolean network, the number of variables is 16 and the length of traces in input is five states. The five Boolean networks have been implemented as a logic program using Answer Set Programing (Brewka et al., 2011). The source code of these programs is given as Supplementary Material. Traces of executions of these programs have been computed using the answer set solver `clasp` (Gebser et al., 2012). All experiments are run with a C++ implementation of **LFkT** on a processor Intel Xeon (X5650, 2.67GHz) with 12 GB of RAM. The main purpose of these experiments is to assess the efficiency of our approach, i.e., how many traces **LFkT** can handle for a given k. Complete output of LFkT for these experiments is accessible as textfile at http://tony.research.free.fr/paper/Frontier/output.zip.

In the first table of **Figure 4**, the evolution of run time from 10 to 1,000,000 traces (which is arbitrary chosen as upper bound of the scalability of the experiments) shows that, in practice, learning with **LFkT** is linear in the number of traces when the number of variables is fixed. Results show that the algorithm can handle more than one million of traces in less than 10 h. Since each trace is a sequence of five state transitions, when learning the Markov(5) system, each trace can be decomposed into 15 interpretation transitions (one 5-step, two 4-step, three 3-step, four 2-step, and five 1-step). Learning the Markov(5) program from one million traces of executions of size five requires the processing of 15 million of interpretation transitions. Learning the Markov(4) to Markov(1) programs requires to process, respectively, 14 million, 12 million, 9 million, and 5 million of interpretation transitions. Intuitively one could expect that learning the Markov(2) system to take significantly more time than learning the Markov(1) system. But each program is different, i.e., the Markov(2) program is not an extension of the Markov(1) program with 2-step rules. That is why run time is not always larger for a larger k: learning time also depends on the rules that are learned. In this experiment, the best run time is obtained with the Markov(3) program. We cannot say that the rules of this program are simpler than the others, but they are simpler to learn for the algorithm. In the second table, we observe that the number of rules learned for the Markov(3) program is significantly smaller than for the others. It means that the algorithm needs to compare less rules for each traces analysis, which can explain the speed up.

In this benchmark, in order to be faithful to the biological experiments presented by Li et al. (2006), we considered $k = 5$ as a maximum. But our algorithm succeeds in processing larger memory effects. On some random dummy examples (accessible at the above mentioned URL), we were able to learn Markov(7) systems with the following performances: we can learn 10 traces in 2.8 s, 100 traces in 27 s, 1,000 traces in 249 s, 10,000 traces in 3,621 s, 100,000 traces in 39,973 s, and 1,000,000 traces in 441,270 s. Even if the computation time increases, it should be kept in mind that our method is designed to allow successive refinements of a model about its memory effect. These results show that such an approach is tractable even with a large number of input traces.

5. CONCLUSION AND FUTURE WORK

5.1. SUMMARY OF THE CONTRIBUTION

To understand the memory effect involved in some interactions between biological components, it is necessary to include delayed influences in the model. In this paper, we proposed a logical method to learn such models from state transition systems. We designed an approach to learn Boolean networks with delayed influences. We have given a step by step explanation of this methodology, and illustrated its merits on a biological benchmark coming from a real-life case study.

5.2. FURTHER WORKS

Further works aim at adapting the approach developed in the paper to the kind of data produced by biologists. This requires connecting through various biological databases in order to extract real time series data, and subsequently explore and use them to learn gene regulatory networks. On account of the noise inherent to biological data, the ability to either perform an efficient discretization of the data or to include the notion of noise inside the modeling framework is fundamental. We will thus have to discuss the discretization procedure and the robustness of our modeling against noisy data and compare it to existing approaches, like the Bayesian ones (Barker et al., 2011).

Regarding the model, we consider extending the methodology to asynchronous semantics. Garg et al. (2008) addressed the differences and complementarity of synchronous and asynchronous semantics to model regulatory networks and identify attractors. The authors focus on attractors, which are central to gene regulation. Previous studies about attractors with synchronous semantics [by Melkman et al. (2010) and Akutsu et al. (2011)] and asynchronous semantics [by Bernot et al. (2004) and Fauré et al. (2006)] showed that different updating rules result in different attractors. The benefits of the synchronous model are

to be computationally tractable, while classical state space exploration algorithms fail on asynchronous ones. Yet, the synchronous modeling relies on one quite heavy assumption: all genes can make a transition simultaneously and need an equivalent amount of time to change their expression level. Even if this is not realistic from a biological point of view, it is usually sufficient as the exact kinetics and order of transformations are generally unknown. The asynchronous semantics, however, helps to capture more realistic behaviors. That is why we plan to extend our approach to asynchronous semantics.

Finally, we will also address multi-valued networks that may be useful to capture behaviors that cannot be summarized through a pure Boolean framework.

AUTHOR CONTRIBUTIONS

Tony Ribeiro: formalization of the problem; design, implementation, description, and pseudo-code of the algorithm; design, implementation, run, and discussion of experiments. Morgan Magnin: state of the art, introduction, biological background, case study, and conclusion. Katsumi Inoue: supervision of the work; formalization of the logic programing and learning from interpretation transition approach background. Chiaki Sakama: formalization of the logic programing and learning from interpretation transition approach background.

ACKNOWLEDGMENTS

This work is supported in part by the 2014-2015 JSPS Challenging Exploratory Research, "Learning Cellular Automata Represented as Logic Programs." Morgan Magnin has further been supported by JSPS Fellowship grant.

SUPPLEMENTARY MATERIAL

The Supplementary Material for this article can be found online at http://www.frontiersin.org/Journal/10.3389/fbioe.2014.00081/abstract

REFERENCES

Abou-Jaoudé, W., Ouattara, D. A., and Kaufman, M. (2009). From structure to dynamics: frequency tuning in the p53-mdm2 network: I. logical approach. *J. Theor. Biol.* 258, 561–577. doi:10.1016/j.jtbi.2009.02.005

Akutsu, T., Kuhara, S., Maruyama, O., and Miyano, S. (2003). Identification of genetic networks by strategic gene disruptions and gene overexpressions under a Boolean model. *Theor. Comp. Sci.* 298, 235–251. doi:10.1016/S0304-3975(02)00425-5

Akutsu, T., Melkman, A. A., Tamura, T., and Yamamoto, M. (2011). Determining a singleton attractor of a Boolean network with nested canalyzing functions. *J. Comput. Biol.* 18, 1275–1290. doi:10.1089/cmb.2010.0281

Apt, K. R., Blair, H. A., and Walker, A. (1988). "Foundations of deductive databases and logic programming," in *Towards a Theory of Declarative Knowledge.* ed. J. Minker (San Francisco, CA: Morgan Kaufmann Publishers Inc.), 89–148.

Barker, N. A., Myers, C. J., and Kuwahara, H. (2011). Learning genetic regulatory network connectivity from time series data. *IEEE/ACM Trans. Comput. Biol. Bioinform.* 8, 152–165. doi:10.1109/TCBB.2009.48

Bernot, G., Comet, J.-P., Richard, A., and Guespin, J. (2004). Application of formal methods to biological regulatory networks: extending Thomas' asynchronous logical approach with temporal logic. *J. Theor. Biol.* 229, 339–347. doi:10.1016/j.jtbi.2004.04.003

Brewka, G., Eiter, T., and Truszczynski, M. (2011). Answer set programming at a glance. *Commun. ACM* 54, 92–103. doi:10.1145/2043174.2043195

Chaouiya, C. (2007). Petri net modelling of biological networks. *Brief. Bioinform.* 8, 210–219. doi:10.1093/bib/bbm029

Cho, R. J., Campbell, M. J., Winzeler, E. A., Steinmetz, L., Conway, A., Wodicka, L., et al. (1998). A genome-wide transcriptional analysis of the mitotic cell cycle. *Mol. Cell* 2, 65–73. doi:10.1016/S1097-2765(00)80114-8

Chueh, T.-H., and Lu, H. H.-S. (2012). Inference of biological pathway from gene expression profiles by time delay Boolean networks. *PLoS ONE* 7:e42095. doi:10.1371/journal.pone.0042095

Comet, J.-P., Bernot, G., Das, A., Diener, F., Massot, C., and Cessieux, A. (2012). Simplified models for the mammalian circadian clock. *Procedia Comput. Sci.* 11, 127–138. doi:10.1016/j.procs.2012.09.014

Comet, J.-P., Fromentin, J., Bernot, G., and Roux, O. (2010). "A formal model for gene regulatory networks with time delays," in *Computational Systems-Biology and Bioinformatics.* eds C. Jonathan, O. Yew-Soon, and C. Sung-Bae (Springer), 1–13.

Fauré, A., Naldi, A., Chaouiya, C., and Thieffry, D. (2006). Dynamical analysis of a generic Boolean model for the control of the mammalian cell cycle. *Bioinformatics* 22, e124–e131. doi:10.1093/bioinformatics/btl210

Garg, A., Di Cara, A., Xenarios, I., Mendoza, L., and De Micheli, G. (2008). Synchronous versus asynchronous modeling of gene regulatory networks. *Bioinformatics* 24, 1917–1925. doi:10.1093/bioinformatics/btn336

Gebser, M., Kaminski, R., Kaufmann, B., and Schaub, T. (2012). "Answer set solving in practice," in *Synthesis Lectures on Artificial Intelligence and Machine Learning,* Vol. 6. eds R. Kaminski, B. Kaufmann (Morgan and Claypool Publishers), 1–238.

Ghanbarnejad, F. (2011). *Perturbations in Boolean Networks.*

Inoue, K. (2011). "Logic programming for Boolean networks," in *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence, IJCAI'11* (Barcelona: AAAI Press), 924–930.

Inoue, K., Ribeiro, T., and Sakama, C. (2014). Learning from interpretation transition. *Mach. Learn.* 94, 51–79. doi:10.1007/s10994-013-5353-8

Inoue, K., and Sakama, C. (2012). "Oscillating behavior of logic programs," in *Correct Reasoning.* eds E. Esra, L. Joohyung, L. Yuliya, and P. David (Springer), 345–362.

Kauffman, S. A. (1969). Metabolic stability and epigenesis in randomly constructed genetic nets. *J. Theor. Biol.* 22, 437–467. doi:10.1016/0022-5193(69)90015-0

Klamt, S., Saez-Rodriguez, J., Lindquist, J. A., Simeoni, L., and Gilles, E. D. (2006). A methodology for the structural and functional analysis of signaling and regulatory networks. *BMC Bioinformatics* 7:56. doi:10.1186/1471-2105-7-56

Koh, C., Wu, F.-X., Selvaraj, G., and Kusalik, A. J. (2009). Using a state-space model and location analysis to infer time-delayed regulatory networks. *EURASIP J. Bioinform. Syst. Biol.* 2009, 14. doi:10.1155/2009/484601

Koksal, A. S., Pu, Y., Srivastava, S., Bodik, R., Fisher, J., and Piterman, N. (2013). Synthesis of biological models from mutation experiments. *ACM SIGPLAN Notices* 48, 469–482.

Lähdesmäki, H., Shmulevich, I., and Yli-Harja, O. (2003). On learning gene regulatory networks under the Boolean network model. *Mach. Learn.* 52, 147–167. doi:10.1023/A:1023905711304

Li, X., Rao, S., Jiang, W., Li, C., Xiao, Y., Guo, Z., et al. (2006). Discovery of time-delayed gene regulatory networks based on temporal gene expression profiling. *BMC Bioinformatics* 7:13. doi:10.1186/1471-2105-7-13

Liu, T.-F., Sung, W.-K., and Mittal, A. (2004). "Learning multi-time delay gene network using Bayesian network framework," in *16th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2004)* (Boca Raton: IEEE), 640–645.

Lopes, M., and Bontempi, G. (2013). Experimental assessment of static and dynamic algorithms for gene regulation inference from time series expression data. *Front. Genet.* 4:303. doi:10.3389/fgene.2013.00303

Mangan, S., and Alon, U. (2003). Structure and function of the feed-forward loop network motif. *Proc. Natl. Acad. Sci. U.S.A.* 100, 11980–11985. doi:10.1073/pnas.2133841100

Melkman, A. A., Tamura, T., and Akutsu, T. (2010). Determining a singleton attractor of an and/or Boolean network in o (n1. 587) time. *Inf. Process. Lett.* 110, 565–569. doi:10.1016/j.ipl.2010.05.001

Muggleton, S., and De Raedt, L. (1994). Inductive logic programming: Theory and methods. *J. Log. Program.* 19, 629–679. doi:10.1016/0743-1066(94)90035-3

Ribeiro, T., and Inoue, K. (2014). "Learning prime implicant conditions from interpretation transition," in *The 24th International Conference on Inductive Logic Programming.* Available at: http://tony.research.free.fr/paper/ILP2014long

Ribeiro, T., Magnin, M., and Inoue, K. (2014). "Learning delayed influence of dynamical systems from interpretation transition," in *The 24th International Conference on Inductive Logic Programming.* Available at: http://tony.research.free.fr/paper/ILP2014short

Siebert, H., and Bockmayr, A. (2006). "Incorporating time delays into the logical analysis of gene regulatory networks," in *Computational Methods in Systems Biology*. ed. P. Corrado (Springer), 169–183.

Silvescu, A., and Honavar, V. (2001). Temporal Boolean network models of genetic networks and their inference from gene expression time series. *Complex Syst.* 13, 61–78. doi:10.1186/1752-0509-5-61

Soinov, L. A., Krestyaninova, M. A., and Brazma, A. (2003). Towards reconstruction of gene networks from expression data by supervised learning. *Genome Biol.* 4, 6. doi:10.1186/gb-2003-4-10-341

Spellman, P. T., Sherlock, G., Zhang, M. Q., Iyer, V. R., Anders, K., Eisen, M. B., et al. (1998). Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization. *Mol. Biol. Cell* 9, 3273–3297. doi:10.1091/mbc.9.12.3273

Van Emden, M. H., and Kowalski, R. A. (1976). The semantics of predicate logic as a programming language. *J. Altern. Complement. Med.* 23, 733–742. doi:10.1145/321978.321991

Zhang, Z.-Y. (2008). *Time Series Segmentation for Gene Regulatory Process with Time-Window-Extension Technique.* 198–203.

Conflict of Interest Statement: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Received: 30 July 2014; accepted: 13 December 2014; published online: 16 January 2015.
Citation: Ribeiro T, Magnin M, Inoue K and Sakama C (2015) Learning delayed influences of biological systems. Front. Bioeng. Biotechnol. ***2****:81. doi: 10.3389/fbioe.2014.00081*
This article was submitted to Bioinformatics and Computational Biology, a section of the journal Frontiers in Bioengineering and Biotechnology.

frontiers in Bioengineering and Biotechnology

ORIGINAL RESEARCH
published: 04 September 2015
doi: 10.3389/fbioe.2015.00131

Designing experiments to discriminate families of logic models

*Santiago Videla[1,2,3,4], Irina Konokotina[5], Leonidas G. Alexopoulos[6], Julio Saez-Rodriguez[7], Torsten Schaub[3], Anne Siegel[1,2] and Carito Guziolowski[5]**

[1] UMR 6074 IRISA, CNRS, Campus de Beaulieu, Rennes, France, [2] Dyliss project, INRIA, Campus de Beaulieu, Rennes, France, [3] Institut für Informatik, Universität Potsdam, Potsdam, Germany, [4] LBSI, Fundación Instituto Leloir, CONICET, Buenos Aires, Argentina, [5] IRCCyN UMR CNRS 6597, École Centrale de Nantes, Nantes, France, [6] Department of Mechanical Engineering, National Technical University of Athens, Athens, Greece, [7] European Molecular Biology Laboratory, European Bioinformatics Institute, Hinxton, UK*

Logic models of signaling pathways are a promising way of building effective *in silico* functional models of a cell, in particular of signaling pathways. The automated learning of Boolean logic models describing signaling pathways can be achieved by training to phosphoproteomics data, which is particularly useful if it is measured upon different combinations of perturbations in a high-throughput fashion. However, in practice, the number and type of allowed perturbations are not exhaustive. Moreover, experimental data are unavoidably subjected to noise. As a result, the learning process results in a family of feasible logical networks rather than in a single model. This family is composed of logic models implementing different internal wirings for the system and therefore the predictions of experiments from this family may present a significant level of variability, and hence uncertainty. In this paper, we introduce a method based on Answer Set Programming to propose an optimal experimental design that aims to narrow down the variability (in terms of input–output behaviors) within families of logical models learned from experimental data. We study how the fitness with respect to the data can be improved after an optimal selection of signaling perturbations and how we learn optimal logic models with minimal number of experiments. The methods are applied on signaling pathways in human liver cells and phosphoproteomics experimental data. Using 25% of the experiments, we obtained logical models with fitness scores (mean square error) 15% close to the ones obtained using all experiments, illustrating the impact that our approach can have on the design of experiments for efficient model calibration.

Keywords: experimental design, Boolean logic models, phosphoproteomic, answer set programming, signaling networks

OPEN ACCESS

Edited by:
David A. Rosenblueth, Universidad Nacional Autónoma de México, Mexico

Reviewed by:
Jérôme Feret, INRIA, France
Hannes Klarner, Freie Universität Berlin, Germany

****Correspondence:***
Carito Guziolowski, IRCCyN UMR CNRS 6597, École Centrale de Nantes, 1 rue de la Noë, Nantes 44321, France
carito.guziolowski@irccyn.ec-nantes.fr

Specialty section:
This article was submitted to Bioinformatics and Computational Biology, a section of the journal Frontiers in Bioengineering and Biotechnology

Received: *05 July 2015*
Accepted: *17 August 2015*
Published: *04 September 2015*

Citation:
Videla S, Konokotina I, Alexopoulos LG, Saez-Rodriguez J, Schaub T, Siegel A and Guziolowski C (2015) Designing experiments to discriminate families of logic models. Front. Bioeng. Biotechnol. 3:131. doi: 10.3389/fbioe.2015.00131

1. Introduction

The recent development of high-throughput experimental technologies allows us to observe different cellular parts under multiple situations. This information is of great value to generate and validate computational models of the molecular processes happening within cells.

Thanks to their simplicity, qualitative approaches allow us to model larger-scale biological systems than quantitative methods. Among these approaches, logic models are able to capture interesting and relevant behaviors in the cell (Morris et al., 2010; Mbodj et al., 2013). We have previously proposed to generate logic models by training a prior knowledge network to phosphoproteomics data. Importantly, due to factors, such as the sparsity and the uncertainty of experimental measurements,

there are often multiple models that cannot be distinguished with the data at hand, that is, the model is non-identifiable, requiring to consider a set (a family) of logic models (Saez-Rodriguez et al., 2009; Guziolowski et al., 2013). In this paper, we propose to specialize the (possibly many) logic behaviors of this family by using an efficient strategy for experiment design, that is, an optimal selection of signaling perturbations to discriminate models at hand.

The experimental design problem consists of finding the most informative experiments in order to identify more accurate models (Kreutz and Timmer, 2009). On the one hand, in the context of quantitative models, this problem has been approached via methods for both parameter estimation and model discrimination (Kremling et al., 2004; Vatcheva et al., 2005; Mélykúti et al., 2010; Busetto et al., 2013; Stegmaier et al., 2013; Meyer et al., 2014). On the other hand, for qualitative models, fewer methods have been proposed (Ideker et al., 2000; Yeang et al., 2005; Barrett and Palsson, 2006; Szczurek et al., 2008; Sparkes et al., 2010; Atias et al., 2014). An optimal experimental design can be applied to either: (*i*) experimental setup selection, that is the optimal choice of species to perturb and measure, or (*ii*) perturbations selection, where one perturbation indicates which species will be perturbed in one experiment.

In this work, we focus on experimental design for selecting an optimal signaling perturbation set by considering the experimental setup fixed and a set of initial measurements from low combinatorial (i.e., single stimulus or inhibitor species) perturbations. We use training algorithms to identify a family of Boolean models explaining the data according to a prior knowledge network. This family needs to be discriminated by measuring the effect of additional perturbations. For this, we propose a new method that finds optimal set of signaling perturbations satisfying the following criteria: (*i*) it contains a minimal number of perturbations to discriminate all pairs of models in a family of Boolean networks, (*ii*) such perturbations maximize the pairwise differences of models' predictions, and (*iii*) they are subject to technologically inspired constraints, such as the minimization of experimental perturbations cost.

Compared to previous contributions in the context of logical models, our work presents certain differences and similarities. In general, in previous methods, the optimality criterion for a selection of perturbations is given by means of the so-called Shannon entropy (Shannon, 1948). In this context, the Shannon entropy provides a measure of the expected information gained in performing a specific experiment. Intuitively, the higher the Shannon entropy, the higher the ability of an experimental perturbation to distinguish between rival models (Ideker et al., 2000; Szczurek et al., 2008; Atias et al., 2014). In contrast, in our work, the main optimality criterion consists of maximizing the sum of pairwise differences over Boolean models' output. The intuition behind this criterion is to increase the chances to discriminate a pair of models despite the experimental noise. Nonetheless, it is worth noting that some pairs of models could be better discriminated than others. Thus, in principle, if one aims at having a more uniform pairwise discrimination, an entropy-based design criterion would be more appropriate. However, approaches based on the Shannon entropy must resign to exhaustiveness due to computational scalability. Maximizing the sum of pairwise differences was already proposed by Mélykúti et al. (2010) for the discrimination of ODEs models and, recently, the same idea has been used in the context of Boolean logic models (Atias et al., 2014). However, in contrast to our approach, the method introduced by Atias et al. (2014) aims at finding only one perturbation maximizing the number of differences in the output of the pair of models, which differs the most from each other. Therefore, in general, it does not guarantee that other pairs of models will be discriminated as well. More generally, except for Szczurek et al. (2008), previous approaches proposed assays composed of one perturbation. Therefore, only after the proposed perturbation has been carried out in the laboratory and models have been (partially) discriminated, another perturbation can be designed. In contrast, but similarly to Szczurek et al. (2008), we find the smallest number of perturbations to optimally discriminate all pairs of models at once. This approach is tailored to high-throughput technologies that are designed to measure the effects of tens of perturbations in a single run.

We provide a precise characterization of the combinatorial problem related to the optimal selection of signaling perturbations, together with an Answer Set Programming (Gebser et al., 2012) based solution to this problem included within the open source python package **caspo**, which is freely available for download[1]. We applied our method to two case studies using *in silico* and real phosphoproteomics datasets to measure the impact of our approach in a real setting. We show that optimal logic models with few input–output behaviors can be learned by combining a set of low combinatorial perturbations with a minimal set of greater combinatorial perturbations. In the artificially generated data, we obtained that phosphoproteomics measurements from 64 low combinatorial perturbations can be enriched with 10 combinatorial perturbations (from the 1630 possible) to identify a family of logic models with a fitness quality (mean square error) equal to one of the golden standard logic model used to generate the data. In the real dataset, we obtained that phosphoproteomics measurements from 12 low combinatorial perturbations can be enriched with 31 combinatorial perturbations (from the 120 available) to identify a family of logic models with a fitness quality at a 15% distance from the fitness of logical models explaining optimally all 120 responses to the perturbations considered.

2. Materials and Methods

2.1. Background

In this paper, we are interested in the discrimination of models based on synchronous Boolean networks (Kauffman, 1969). Importantly, we restrict ourselves to BNs describing models of immediate-early response as introduced in Saez-Rodriguez et al. (2009). Since we focus on fast (early) events, it is assumed that oscillation or multi-stability caused by feedback-loops (Remy et al., 2008) cannot happen until a second phase of signal propagation occurring at a slower time scale. Therefore, BNs with feedback-loops are not considered (Macnamara et al., 2012).

[1] http://bioasp.github.io/caspo/

Several related methods within this framework were published in the last few years in order to learn BNs from a prior knowledge network (PKN) and a phosphoproteomics dataset (Mitsos et al., 2009; Saez-Rodriguez et al., 2009; Guziolowski et al., 2013; Sharan and Karp, 2013; Videla et al., 2014). A PKN is a signed and directed graph describing causal relations among a set V of nodes representing biological species. An *experimental setup* is defined by three subsets of V, namely, possible *stimuli* (V_S), possible *inhibitors* (V_I), and *measured* species (V_M). A *signaling perturbation* is a combination of present/absent stimuli and inhibitors. Then, a phosphoproteomics dataset provides phosphorylation activities (in this context, immediate-early responses) of a set of measured species or readouts under several signaling perturbations. Note that any signaling perturbation is described by an n-dimensional Boolean vector, i.e., $p \in \mathbb{B}^n$, where $n = |V_S| + |V_I|$ and $\mathbb{B} = \{0,1\}$. More precisely, if the j^{th} position in p is assigned to 1 (resp. 0), the corresponding stimulus or inhibitor is said to be present (resp. absent) in the experimental perturbation p.

In general, aforecited methods for learning BNs explore the space of models compatible with the topology given by the PKN aiming at the minimization of two criteria, namely, the difference between data and model predictions, and the model size. On the one hand, the difference between data and model predictions is measured by means of the Mean Squared Error (MSE). On the other hand, the size of a BN is defined as the sum of its formulas' length. Further, due to the inherent noise in experimental data, we are interested not only on optimal but also nearly optimal BNs. That is, BNs having MSE and size within given tolerances with respect to the corresponding minimal values. In this context, it has been shown that the exhaustive enumeration of nearly optimal BNs explaining phosphoproteomics dataset with respect to a PKN leads to a large number of them (Guziolowski et al., 2013). Nonetheless, it often happens that for all measured species, several BNs describe exactly the same response to every possible signaling perturbation. In such a case, we say that those BNs describe the same *input–output behavior*. For example, in Guziolowski et al. (2013), several thousands of nearly optimal BNs described only 91 distinct responses. Concretely, the input–output behavior of a BN is described by a "truth table" whose entries are all possible signaling perturbations, and whose outputs are the corresponding Boolean vector responses. Therefore, we can see input–output behaviors merely as functions of the form β: $\mathbb{B}^n \to \mathbb{B}^m$ where $n = |V_S| + |V_I|$ and $m = |V_M|$. Notice that, for a given set of BNs, we can identify a canonical set of input–output behaviors $\boldsymbol{B}$ containing exactly one *representative* BN for each behavior.

2.2. Discriminating Input–Output Behaviors

In this section, we introduce our method to discriminate input–output behaviors in a pairwise fashion. We assume that we are given a set $\boldsymbol{B}$ of input–output behaviors. For instance, this set may result from the learning of BNs from given PKN and phosphoproteomics dataset. In what follows, we denote by $\boldsymbol{D}$ a selection of signaling perturbations. In theory, all combinatorial perturbations of stimuli and inhibitors could be considered. Nonetheless, in order to consider the limitation of current technology, in general, we restrict ourselves to perturbations having at most s stimuli and i inhibitors, with $0 \leq s \leq |V_S|$ and $0 \leq i \leq |V_I|$. Then, we denote with $\boldsymbol{P}$ the set of such possible signaling perturbations. Notably, the total number of perturbations is given by:

$$\sum_{j=0}^{s} \binom{|V_S|}{j} \times \sum_{j=0}^{i} \binom{|V_I|}{j}$$

where $\binom{n}{m}$ denotes the binomial coefficient, i.e., $\frac{n!}{m!(n-m)!}$. Alternatively, $\boldsymbol{P}$ could be defined by the user providing any fixed list of feasible perturbations. Next, our method has three main steps: (1) find the minimum number k of signaling perturbations in $\boldsymbol{P}$ to discriminate every pair of input–output behaviors in $\boldsymbol{B}$, (2) find all sets of exactly k signaling perturbations in $\boldsymbol{P}$ maximizing the sum of pairwise differences of input–output behaviors in $\boldsymbol{B}$, and (3) minimize the complexity of the experiments by minimizing the number of present stimuli and inhibitors in the set of selected perturbations. In what follows, we give more details and mathematical definitions for each of these steps. In addition, we introduce a parameter k_{max} describing the maximum number of perturbations that can be performed simultaneously.

2.2.1. Step 1: Required Signaling Perturbations to Discriminate all Behaviors Pairwise

Usually, several perturbations must be performed in order to discriminate among every pair of behaviors. However, in order to minimize experimental costs, one would like to perform as few perturbations as possible. Therefore, our first criterion consists of finding the minimum number of perturbations, which allow us to discriminate among every pair of input–output behaviors. To be more precise, we aim at finding the smallest $k \in (0, k_{max})$ such that there exists a set $\boldsymbol{D}$ having k perturbations $p \in \boldsymbol{P}$ satisfying:

$$\left(\forall \beta, \beta' \in \boldsymbol{B} :: (\exists p \in \boldsymbol{D} :: \beta(p) \neq \beta'(p))\right). \tag{1}$$

Let us denote with $\mathcal{D}_{(k,s,i)}$ the set of all $\boldsymbol{D} \subseteq \boldsymbol{P}$ with $|\boldsymbol{D}| = k$ and satisfying (1). It is worth noting that we restrict our search to at most s stimuli, i inhibitors, and k_{max} perturbations. Therefore, there may be cases where does not exists $\boldsymbol{D}$ discriminating all input–output behaviors pairwise. For such cases, we relax the constraint of full pairwise discrimination and define $\mathcal{D}_{(k,s,i)}$ as before but setting $k = k_{max}$ and without requiring the satisfaction of (1). That is, some but not all pairs of input–output behaviors are discriminated.

2.2.2. Step 2: Maximizing Differences Over Measured Species

Once we have identified that k signaling perturbations are required to discriminate between all input–output behaviors in $\boldsymbol{B}$ (or alternatively, $k = k_{max}$), the next question is how to select among all possible sets $\boldsymbol{D} \in \mathcal{D}_{(k,s,i)}$. Then, we define the *differences* (Θ_{diff}) generated by a set $\boldsymbol{D} \in \mathcal{D}_{(k,s,i)}$ over the family of input–output behaviors $\boldsymbol{B}$ as:

$$\Theta_{diff}(\boldsymbol{B}, \boldsymbol{D}) = \sum_{\beta, \beta' \in \boldsymbol{B}} \sum_{p \in \boldsymbol{D}} \mathcal{H}(\beta(p), \beta'(p)) \tag{2}$$

where $\mathcal{H}$ denotes the Hamming distance over Boolean vectors, i.e., the number of positions at which the corresponding vectors

values are different. Our second criterion consists of finding all sets of k perturbations $\boldsymbol{D} \in \mathcal{D}_{(k,s,i)}$ such that the function Θ_{diff} is maximized,

$$D^*_{(k,s,i)} = \underset{\boldsymbol{D} \in \mathcal{D}_{(k,s,i)}}{\arg\max}\, \Theta_{diff}(\boldsymbol{B}, \boldsymbol{D})\,. \tag{3}$$

2.2.3. Step 3: Minimizing the Complexity of Experiments

The complexity of any signaling perturbation is essentially related to the number of present stimuli and inhibitors in it. Thus, in this step, we aim at finding the simplest sets of perturbations among all $\boldsymbol{D}^* \in \mathcal{D}^*_{(k,s,i)}$. Toward this end, we define two functions counting the number of stimuli (Θ_{V_S}) and inhibitors (Θ_{V_I}) being present in a given set of signaling perturbations. More precisely, let us recall that every perturbation p is a Boolean vector such that, if the j^{th} position in p is assigned to 1 (resp. 0), the corresponding stimulus or inhibitor is said to be present (resp. absent) in p. Thus, for the set $U = V_S$ or $U = V_I$ of either stimuli or inhibitors, we can define Θ_U as,

$$\forall \boldsymbol{D}^* \in \mathcal{D}^*_{(k,s,i)}, \quad \Theta_U\left(\boldsymbol{D}^*\right) = \sum_{p \in \boldsymbol{D}^*} \sum_{u_j \in U} p_j \tag{4}$$

where p_j denotes the j^{th} position in p corresponding to either a stimulus if $U = V_S$, or an inhibitor if $U = V_I$. Finally, we consider two additional optimization criteria in lexicographic order (Marler and Arora, 2004) aiming at the identification of the simplest $\boldsymbol{D}^* \in \mathcal{D}^*_{(k,s,i)}$, and we define the family of optimal sets of signaling perturbations $\boldsymbol{D_{opt}} \in \mathcal{D}_{opt}$ as follows:

$$\mathcal{D}_{opt} = \underset{\boldsymbol{D}^* \in \mathcal{D}^*_{(k,s,i)}}{\arg\min}\, \left(\Theta_{V_S}\left(\boldsymbol{D}^*\right), \Theta_{V_I}\left(\boldsymbol{D}^*\right)\right). \tag{5}$$

Notice that we minimize first Θ_{V_S} and then, with lower priority Θ_{V_I}, but this is an arbitrary choice, which can be revisited.

2.3. Experimental Design Powered by Answer Set Programming

The method described in Section 2 is implemented in the publicly available python package **caspo**. Our software strongly relies on a form of logic programming known as Answer Set Programming (ASP) (Gebser et al., 2012). ASP provides a declarative framework for modeling knowledge-intense combinatorial (optimization) problems. Moreover, state-of-the-art ASP solvers offer powerful implementations. In our context, the ASP logic program is satisfiable for positive integers *k, s, i* if there exists a selection of k signaling perturbations in $\boldsymbol{P}$ satisfying (1). Then, the solving consists of considering, starting from $k = 1$, increasing values for k until the ASP logic program is satisfiable. Once the solver finds the smallest k or reaches k_{max}, it proceeds to solve the multi-objective optimization problem in lexicographic order: first, by maximizing the pairwise differences over the Boolean models' outputs as defined in (3), and then by minimizing the complexity of experimental perturbations, as defined in (5). It is worth noting that, thanks to the declarativeness and elaboration tolerance of ASP, it is straightforward to consider additional constraints for specific use cases.

2.4. The Loop for Learning and Discriminating Input–Output Behaviors

In what follows we assume the existence of a method for learning nearly optimal BNs and their corresponding set of input–output behaviors. Further, such a method must be parametrized specifying allowed tolerances with respect to optimal fitness and size. Also, we assume an implementation of the method described in Section 2. In our case, we rely on the python package **caspo**, which implements both methods providing an unified framework. In order to evaluate our method in a systematic way, we have implemented the workflow shown in **Figure 1**[2]. For a more detailed description, we refer the reader to pseudo-code algorithms (Algorithm S1 and S2) provided in Supplementary Material.

We start by learning strictly optimal input–output behaviors from a given PKN and initial dataset. Then, the workflow follows a "cautious" strategy in the sense that it will try to discriminate among input–output behaviors as soon as we find more than one. Every time we discriminate among a set of input–output behaviors, an optimal set of signaling perturbations is proposed. Then, both the set of perturbations and the corresponding measurements obtained after performing the experiments are added to the dataset used for learning and the workflow starts over. Notably, in our simulations, measurements are extracted automatically from either artificial or real datasets available beforehand. Meanwhile, in real case studies, measurements would be provided by concrete wet experiments.

Importantly, there are cases when the learning method returns a single optimal input–output behavior. In such cases, the workflow explores nearly optimal behaviors by considering a range of tolerances, first over the optimal model size and then over the optimal MSE. Extending the discrimination procedure to nearly optimal behaviors allows ensuring that the complete workflow is robust to noise in data. Nonetheless, after considering certain ranges of tolerances on both size and fitness to data, there could be only one input–output behavior. In such a case, we interpret the behavior at hand to be robust enough and the workflow terminates. Otherwise, the workflow has two additional stop conditions: (1) when all proposed signaling perturbations for discrimination are already present in the dataset used for learning; (2) when the number of experiments in the dataset reaches a given maximum number of allowed experiments.

3. Results

3.1. Experimental Design on Artificial and Real Case Studies

We evaluate our approach using the workflow described in Section 4 for real-world signaling pathways in human liver cells, and both artificial and real phosphoproteomics datasets. At every loop iteration, we compute two metrics over the learned input–output behaviors: (1) the *learning* MSE, which is computed with respect to the dataset used for learning, and (2) the *testing* MSE, which is computed with respect to the complete space of signaling perturbations under consideration (either artificial or real datasets available beforehand).

[2] An implementation is publicly available at http://github.com/svidela/sbloopy

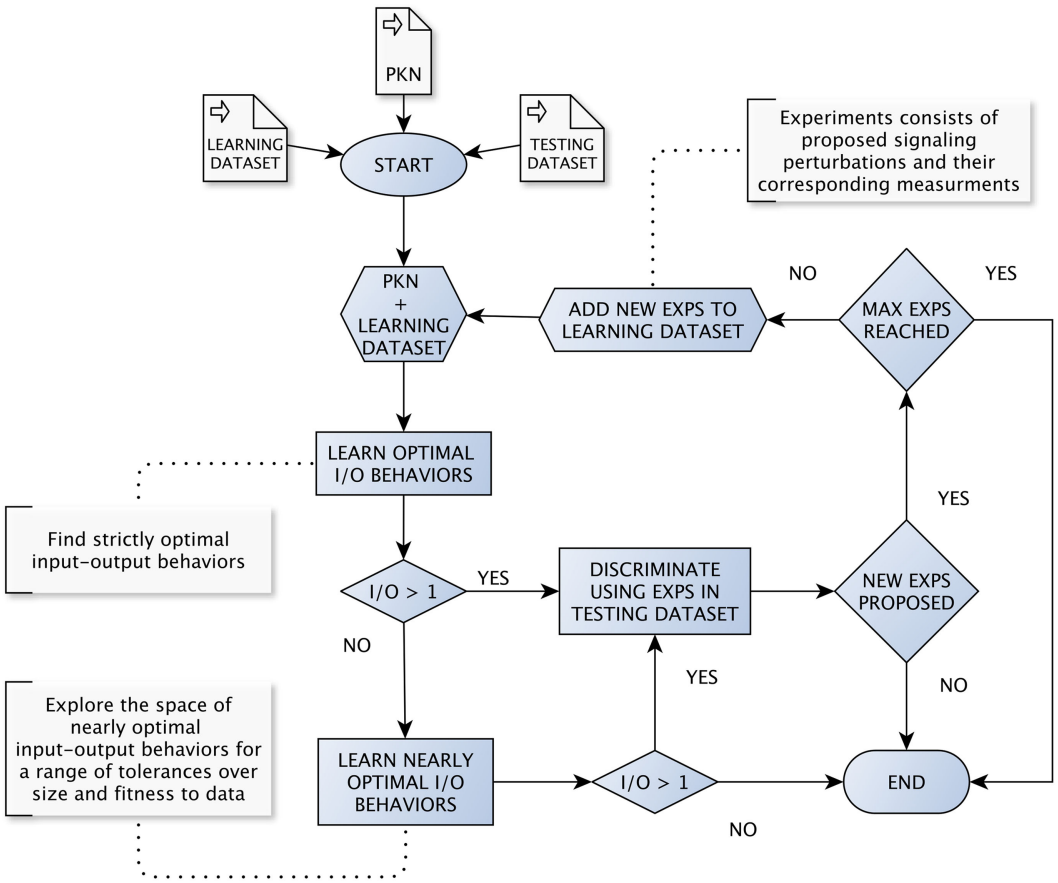


FIGURE 1 | The loop for learning and discriminating input–output behaviors. The loop starts by learning optimal input–output behaviors from a given PKN and initial dataset. Then, we try to discriminate among learned input–output behaviors as soon as we find more than one. Every time we discriminate among a set of behaviors, an optimal set of signaling perturbations is proposed. Next, both the set of perturbations and the corresponding measurements are added to the dataset used for learning and the loop starts over. When the learning method returns a single optimal input–output behavior, the workflow explores nearly optimal behaviors by considering a range of tolerances, first over the optimum model size and then over the optimum MSE.

3.1.1. Artificial Case Studies

The PKN was introduced in Saez-Rodriguez et al. (2009) and here we use a variation that we used also in Guziolowski et al. (2013). Further, to motivate our study, we considered the experimental setup (choice of stimuli, inhibitors, and measured species) from a publicly available phosphoproteomics dataset (Alexopoulos et al., 2010). It contains 7 stimuli, 7 inhibitors, and 15 readouts. Using the PKN, we have generated 100 random BNs as our *gold standards*. We require that every gold standard has size between 28 and 32, and between 2 and 4 AND gates. Next, for each gold standard, an artificial Boolean dataset is generated by performing the simulation of every possible signaling perturbation over the network. That is, each artificial dataset consists of 2^{14} signaling perturbations with their corresponding output measurements. Moreover, toward more realistic phosphoproteomics datasets, we add random noise to Boolean outputs using the distribution $Beta(\alpha = 1, \beta = 5)$. In this context, the loop starts with a dataset of size 64 having all perturbations (and the corresponding artificial measurements) including all combinations of 0 or 1 stimulus with 0 or 1 inhibitor. For the following iterations, optimal sets of signaling perturbations are chosen among all combinations of 0–3 stimuli with 0–2 inhibitors. The maximum number of perturbations to discriminate a given set of behaviors was set to 5, and the maximum number of experiments allowed in the dataset used for learning was set to 80. Additionally, for each gold standard dataset, starting from the same 64 initial datasets, we performed 50 random selections of 10 and 16 experiments. These experiments were added to the initial datasets and we learned BNs from randomly selected experiments. Finally, we computed the testing MSE of this family of BNs with respect to the total 2^{14} experiments.

3.1.2. Real Case Study

To validate this approach on a real phosphoproteomic dataset, we used a larger PKN than in the *in silico* dataset. The reason being that 2 phosphoproteomics datasets were available for this PKN: a low combinatorial one referred to as *screening*, where only 1 stimulus was perturbed per experiment, and the *follow-up*, which had greater combinatorial perturbations. The PKN and datasets were introduced in Melas et al. (2012). The PKN was constructed from several sources of information; it was pruned using the *screening* dataset to keep only the signaling pathways

that show a significant response on specific cells. The *follow-up* dataset had 120 combinatorial signaling perturbations and was used to learn optimal BNs fitting the data. The experimental setup for this dataset consisted of 12 stimuli, 3 inhibitors, and 16 readouts. In this context, the loop starts with a dataset having the 12 responsive experiments from the screening dataset. Then, in the following iterations, optimal sets of perturbations are chosen among the available experiments in the follow-up dataset. It is worth noting that at each iteration, there may be several optimal sets of perturbations to discriminate behaviors at hand. Thus, we executed the loop 30 times and at each iteration, one among all optimal sets of signaling perturbations was randomly chosen. The maximum number of perturbations to discriminate a given set of behaviors was set to 5, and the maximum number of experiments allowed in the dataset used for learning was set to 50. Additionally, starting from the same 12 initial datasets, we performed 30 random selections of 20 and 38 experiments. These experiments were added to the initial datasets and we learned BNs from randomly selected experiments. Finally, we computed the testing MSE of this family of BNs with respect to the total 120 experiments.

In **Figure 2**, we show the evolution of the learning MSE for 100 artificial datasets and the 30 real data executions. In **Figure 3**, we show the evolution of the testing MSE for the same artificial and real datasets. For the artificial dataset, we observe that the learning MSE remains constant independent from the number of experiments used; while for the real dataset, it slightly increases with the number of experiments. For the real case, we observe a significative difference in the learning MSE of the logic models learned from a low combinatorial set of experiments (screening data) compared to the MSE of those learned from a more combinatorial follow-up dataset (**Figure 2B**). For both datasets, we observe that the testing MSE converges to the optimal MSE obtained when using the full available datasets. For the artificial case, the testing MSE converges exactly to the optimal MSE (0.047) after 10 experiments; a random selection of experiments is far from reaching this MSE value. For the real case, it converges to an MSE (0.149) at a 15% distance from optimal MSE after 31 experiments; a random selection of experiments shows comparable results. In contrast to a random selection, the proposed method guarantees selecting perturbations that can propose networks

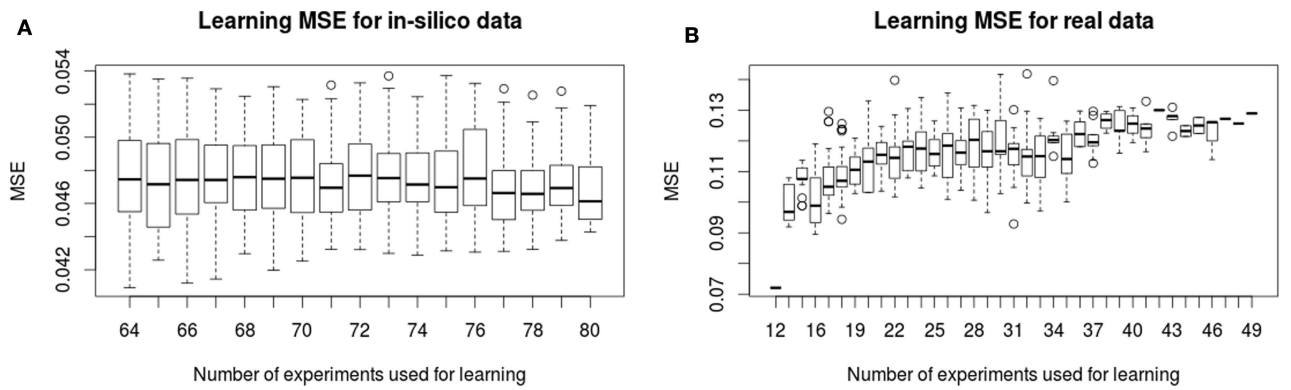


FIGURE 2 | Learning MSE for *in silico* (A) and real (B) phosphoproteomic datasets. The *X*-axis shows the number of experiments (optimal signaling perturbations and measurements associated) used for learning at each iteration. The *Y*-axis shows the learning MSE obtained at each iteration, it represents the quality of the learned models with respect to the experiments used in the learning step.

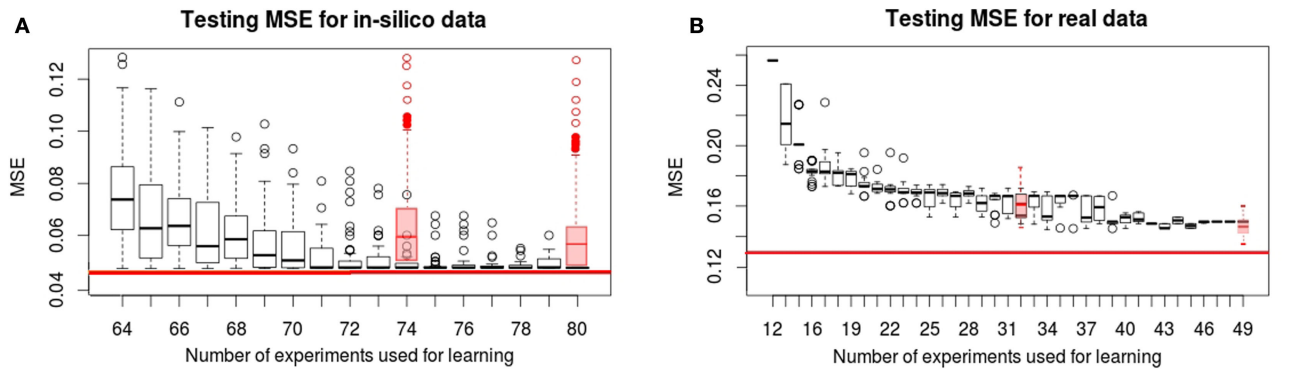


FIGURE 3 | Testing MSE for *in silico* (A) and real (B) phosphoproteomic datasets. The red line represents the optimal MSE learned using the full available experimental datasets (2^{14} experiments for *in silico* and 120 for real datasets). The *X*-axis shows the number of experiments (optimal signaling perturbations and measurements associated) used for learning at each iteration. The *Y*-axis shows the testing MSE obtained at each iteration, it represents the quality of the learned models with respect to the full experimental dataset at each iteration. Red boxplots are the results obtained when the set of signaling perturbations was composed of randomly selected experiments of size 74 or 80 for the *in silico* case, and 32 or 49 for the real case.

with few input–output behaviors: in the real and *in silico* executions, learned logic models had two to eight behaviors after optimal experimental design. One run of the workflow for the artificial case studies proposes 5–16 (on average 11.5) signaling perturbations, while one run of the workflow for real case studies proposes 7–37 (on average 18.9) optimal signaling perturbations. This shows a space of input–output behaviors more difficult to discriminate, therefore requiring more signaling perturbations in the real case study. In 80% of the artificial benchmarks, the loop terminated because all optimal experimental designs were already proposed, in 13%, because the number of 80 allowed experiments was reached, and in 7%, when the search space considered by exploring all size and fitness tolerance range yielded only one input–output behavior. For the real case, in 42% of cases, the loop terminated because all optimal experimental designs were already proposed, in 16%, because behaviors were indistinguishable with the available 120 perturbations, in 6% because the number of 50 allowed experiments was exceeded, and in 36%, because the timeout of 48 h was reached.

3.2. Proposing Experiments to Discriminate Input–Output Behaviors

Using the real-case PKN and the complete follow-up dataset (120 experiments), we explored the space of nearly optimal BNs by setting 0.2% of tolerance with respect to the minimum MSE. By doing this, we found 35208 BNs describing 32 logical input–output behaviors. Notably, in regards of the available experimental observations and their intrinsic uncertainty, such behaviors explained the data equally well. Next, we identified 6558 optimal sets of signaling perturbations, having 0–3 stimuli combined with 0–2 inhibitors, in order to discriminate among the 32 input–output behaviors. Each optimal set consists of 9 signaling perturbations yielding 3378 pairwise differences. In **Figure 4A**, we show one example of optimal signaling perturbations. Next, we looked at which specific measured species generated differences (**Figure 4B**). On the one hand, for one measured species, viz., *CREB*, we generated pairwise differences with eight of the nine proposed signaling perturbations. On the other hand, for all other measured species, we generated pairwise differences with at most two out of the nine signaling perturbations. Moreover, for three measured species, viz., *ERK*, *MAP2K*1, and *rps6Ka*1, we generated pairwise differences with only one experimental perturbation (#7).

4. Discussion

The main result of this paper is that the experimental design loop combining learning and discriminating steps shows a fast convergence of the testing MSE (computed with respect to the complete space of signaling perturbations under consideration) to an approximation of its optimal value: in the real case study, based on a very small initial screening dataset (12 perturbations), 30 well-chosen perturbations are enough to learn input–output behaviors whose fitness with respect to the total *follow-up* perturbations is 15% greater than the optimal MSE. This confirms that follow-up phosphoproteomics assays can be highly redundant and should be designed carefully.

4.1. Testing MSE Non-Monotonic Evolution

In artificial case studies, the learning MSE (computed with respect to the dataset used for learning) remains somehow constant when new perturbations are added to the dataset. This suggests that the 64 perturbations from the initial dataset may be enough to constrain the training method in a part of the search space, which is close to an optimal Boolean network. Then, introducing sub-optimality searches in the loop allows us to explore efficiently the search space of Boolean networks around such an optimum. On the contrary, the learning MSE for the real case study appears to be very heterogeneous at each iteration of the proposed workflow. The best models optimizing the fitness to the 12 screening data (single stimulus) are at a very small distance (0.07), suggesting that the Boolean networks explaining properly the data should be easy to identify. However, as soon as the observations from additional perturbations (each consisting of a combination of different stimulus and inhibitor species) is added to the dataset, the learning fitness increases to (0.11–0.15) showing a significant variability. This suggests that the best models optimizing each dataset are placed in different parts of the search space and that the training dataset is not robust to small variations. Altogether, values for testing MSE evidence that the discriminative method should be always applied iteratively: after some iterations, it may appear that, although a family of optimal BNs has been totally discriminated, applying a step of discrimination for a closer model to the optimal BNs finally allows to identify BNs with a better fitting. Concretely, our analysis strongly suggests that not only the 120 *follow-up* perturbations are redundant, but also that additional experiments to the 120 at hand are needed to improve the robustness of the BNs identification process. Finally, the validation of the method when using random selection is difficult to evaluate for the real case, since our search space of optimal signaling perturbation was constrained to the 120 available experiments. While in this case the performance was not better than random, we have shown in artificial cases how our method is significantly better than random in larger space of experiments.

4.2. Introduction of Sub-Optimality Criteria in the Learning Step

The above mentioned behavior of the learning MSE confirms that the space of optimal logic models returned by training procedures is very sensitive to the dataset under consideration. That is, it may constantly change when observations of new perturbations are being considered (see the toy example provided in Supplementary Material). Relaxing the tolerance of optimality in our learning procedure allows us in many cases (75% of artificial case studies and 40% of real case studies) to learn new perturbations that will decrease the testing MSE at each iteration (see **Figure 5**). On the contrary, in other cases, it heavily altered the space of learned logical models yielding a larger MSE. In these cases, however, the MSE of learned logical networks was always improved in a later step. Whereas the cycle of learning and experimental design for artificial case studies follows homogeneous trajectories in all 100 cases (the number of iterations was on average 6.45 with $\sigma = 2.3$, the number of experiments selected was on average 11.5 with $\sigma = 3.2$), the cycle for real case studies shows more variability in the 30 considered cases (the number of iterations was on average 10.4

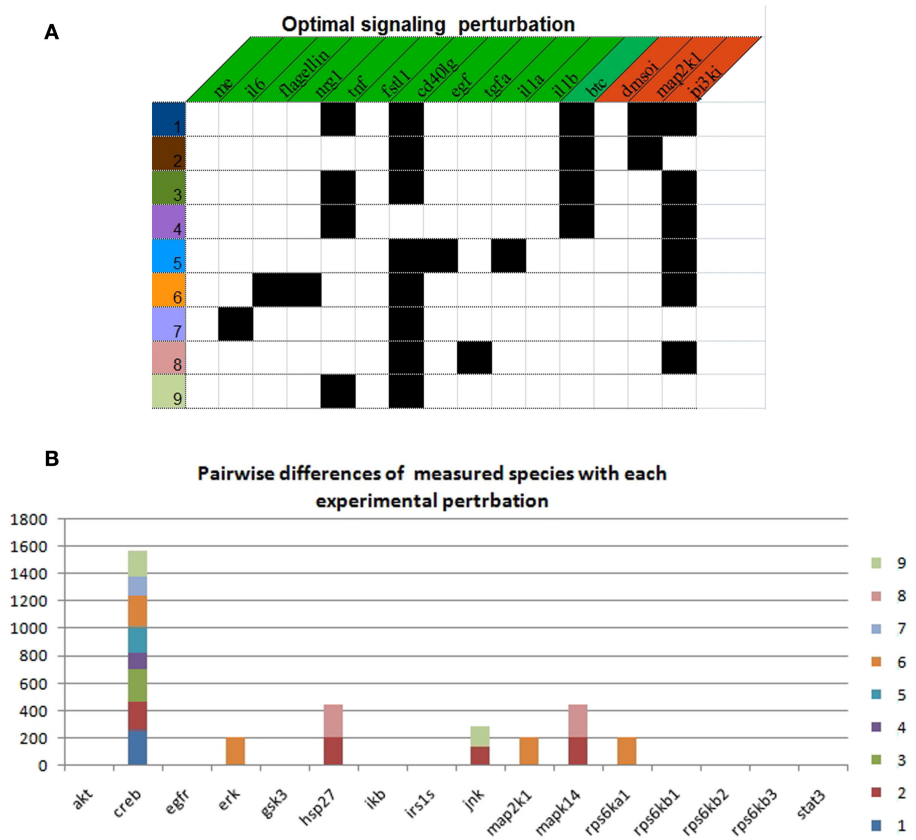


FIGURE 4 | Optimal experimental design to discriminate between 32 input–output behaviors. **(A)** Description of each experimental perturbation. Black squares indicate the presence of the corresponding stimulus (green header) or inhibitor (red header). **(B)** Number of pairwise differences by measured species with each experimental perturbation.

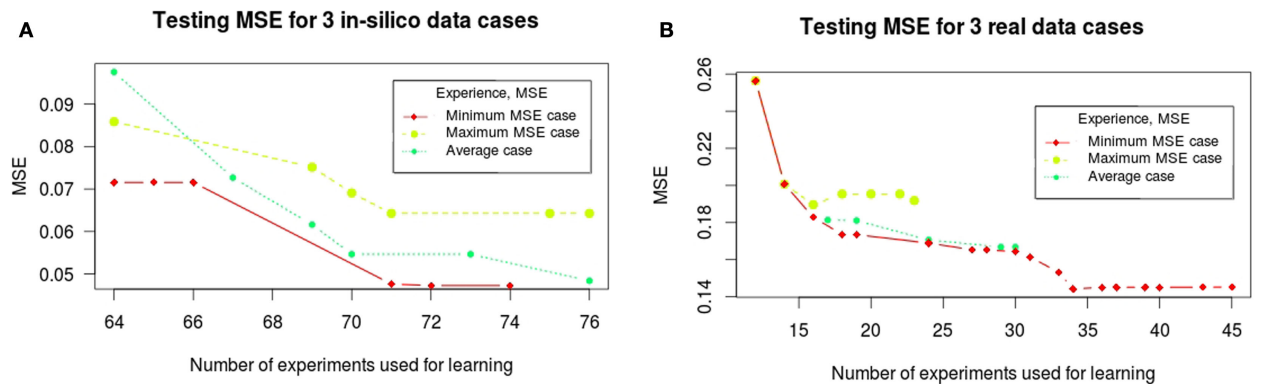


FIGURE 5 | Trajectories of the testing MSE for three significative cases for *in silico* (A) and real (B) phosphoproteomic datasets: the case where a maximum testing MSE was found, where an average testing MSE was found, and when the minimal testing MSE was found. The *X*-axis shows the number of experiments (optimal signaling perturbations and measurements associated) used for learning at each iteration. The *Y*-axis shows the testing MSE obtained at each iteration; it represents the quality of the learned models with respect to the full experimental dataset at each iteration.

with $\sigma = 4.5$, the number of experiments selected was on average 18.9 with $\sigma = 8.1$). In **Figure 5**, we show the trajectories of this cycle for three significative cases in both datasets: the case where a maximum testing MSE was found, where an average testing MSE was found, and when the minimal testing MSE was found.

4.3. Technological Constraints

The technological criteria used in this work focused on minimizing the number of perturbed species; however, many other criteria could be taken into account. For example, we could assign weights to stimuli and inhibitors in order to describe the cost

of each perturbation and minimize the required budget. Also, if certain stimuli and/or inhibitors are not compatible with each other, we could consider additional constraints in order to avoid such combinations. Finally, a constraint can be added to reduce the *variability* on the selection of inhibitors, since inhibitions require additional control experiments. Adding such criteria may be useful given the fact that we often found many optimal sets of signaling perturbations that would allow to discriminate a family of Boolean networks equally well. In addition, when full pairwise discrimination of input–output behaviors is not possible, we could define an objective function in order to maximize the number of discriminated pairs using a fixed number of perturbations. Interestingly, an important feature of the computational method adopted, that is, Answer Set Programming, is to easily allow for modifications of the constraints over the search space. Thus, the framework that we propose in the tool caspo is intentionally rather generic and should be adapted to the characteristics of the signaling system, which is studied.

Supplementary Material

The Supplementary Material for this article can be found online at http://journal.frontiersin.org/article/10.3389/fbioe.2015.00131

References

Alexopoulos, L. G., Saez-Rodriguez, J., Cosgrove, B., Lauffenburger, D. A., and Sorger, P. (2010). Networks inferred from biochemical data reveal profound differences in toll-like receptor and inflammatory signaling between normal and transformed hepatocytes. *Mol. Cell. Proteomics* 9, 1849–1865. doi:10.1074/mcp.M110.000406

Atias, N., Gershenzon, M., Labazin, K., and Sharan, R. (2014). Experimental design schemes for learning Boolean network models. *Bioinformatics* 30, i445–i452. doi:10.1093/bioinformatics/btu451

Barrett, C. L., and Palsson, B. Ø (2006). Iterative reconstruction of transcriptional regulatory networks: an algorithmic approach. *PLoS Comput. Biol.* 2:e52. doi:10.1371/journal.pcbi.0020052

Busetto, A. G., Hauser, A., Krummenacher, G., Sunnåker, M., Dimopoulos, S., Ong, C. S., et al. (2013). Near-optimal experimental design for model selection in systems biology. *Bioinformatics* 29, 2625–2632. doi:10.1093/bioinformatics/btt436

Gebser, M., Kaminski, R., Kaufmann, B., and Schaub, T. (2012). *Answer Set Solving in Practice. Synthesis Lectures on Artificial Intelligence and Machine Learning*. Morgan and Claypool Publishers.

Guziolowski, C., Videla, S., Eduati, F., Thiele, S., Cokelaer, T., Siegel, A., et al. (2013). Exhaustively characterizing feasible logic models of a signaling network using answer set programming. *Bioinformatics* 29, 2320–2326. doi:10.1093/bioinformatics/btt393

Ideker, T. E., Thorsson, V., and Karp, R. M. (2000). "Discovery of regulatory interactions through perturbation: inference and experimental design," in *Pacific Symposium on Biocomputing*, Vol. 5, eds R. B. Altman, A. K. Dunker, L. Hunter, and T. E. Klein 305–316.

Kauffman, S. (1969). Metabolic stability and epigenesis in randomly constructed genetic nets. *J. Theor. Biol.* 22, 437–467. doi:10.1016/0022-5193(69)90015-0

Kremling, A., Fischer, S., Gadkar, K., Doyle, F., Sauter, T., Bullinger, E., et al. (2004). A benchmark for methods in reverse engineering and model discrimination: problem formulation and solutions. *Genome Res.* 14, 1773–1785. doi:10.1101/gr.1226004

Kreutz, C., and Timmer, J. (2009). Systems biology: experimental design. *FEBS J.* 276, 923–942. doi:10.1111/j.1742-4658.2008.06843.x

Macnamara, A., Terfve, C., Henriques, D., Bernabé, B. P., and Saez-Rodriguez, J. (2012). State-time spectrum of signal transduction logic models. *Phys. Biol.* 9, 045003. doi:10.1088/1478-3975/9/4/045003

Marler, R. T., and Arora, J. S. (2004). Survey of multi-objective optimization methods for engineering. *Struct. Multidiscipl. Optim.* 26, 369–395. doi:10.1007/s00158-003-0368-6

Mbodj, A., Junion, G., Brun, C., Furlong, E. E., and Thieffry, D. (2013). Logical modelling of *Drosophila* signalling pathways. *Mol. Biosyst.* 9, 2248–2258. doi:10.1039/c3mb70187e

Melas, I. N., Mitsos, A., Messinis, D. E., Weiss, T. S., Saez Rodriguez, J., and Alexopoulos, L. G. (2012). Construction of large signaling pathways using an adaptive perturbation approach with phosphoproteomic data. *Mol. Biosyst.* 8, 1571–1584. doi:10.1039/c2mb05482e

Mélykúti, B., August, E., Papachristodoulou, A., and El-Samad, H. (2010). Discriminating between rival biochemical network models: three approaches to optimal experiment design. *BMC Syst. Biol.* 4:38. doi:10.1186/1752-0509-4-38

Meyer, P., Cokelaer, T., Chandran, D., Kim, K. H., Loh, P.-R., Tucker, G., et al. (2014). Network topology and parameter estimation: from experimental design methods to gene regulatory network kinetics using a community based approach. *BMC Syst. Biol.* 8:13. doi:10.1186/1752-0509-8-13

Mitsos, A., Melas, I., Siminelakis, P., Chairakaki, A., Saez-Rodriguez, J., and Alexopoulos, L. G. (2009). Identifying drug effects via pathway alterations using an integer linear programming optimization formulation on phosphoproteomic data. *PLoS Comput. Biol.* 5:e1000591. doi:10.1371/journal.pcbi.1000591

Morris, M., Saez-Rodriguez, J., Sorger, P., and Lauffenburger, D. A. (2010). Logic-based models for the analysis of cell signaling networks. *Biochemistry* 49, 3216–3224. doi:10.1021/bi902202q

Remy, E., Ruet, P., and Thieffry, D. (2008). Graphic requirements for multistability and attractive cycles in a Boolean dynamical framework. *Adv. Appl. Math.* 41, 335–350. doi:10.1016/j.aam.2007.11.003

Saez-Rodriguez, J., Alexopoulos, L. G., Epperlein, J., Samaga, R., Lauffenburger, D. A., Klamt, S., et al. (2009). Discrete logic modelling as a means to link protein signalling networks with functional analysis of mammalian signal transduction. *Mol. Syst. Biol.* 5, 331. doi:10.1038/msb.2009.87

Shannon, C. E. (1948). A mathematical theory of communication. *Bell Syst. Tech. J.* 27, 379–423. doi:10.1002/j.1538-7305.1948.tb00917.x

Sharan, R., and Karp, R. M. (2013). Reconstructing Boolean models of signaling. *J. Comput. Biol.* 20, 249–257. doi:10.1089/cmb.2012.0241

Sparkes, A., Aubrey, W., Byrne, E., Clare, A., Khan, M. N., Liakata, M., et al. (2010). Towards robot scientists for autonomous scientific discovery. *Autom. Exp.* 2, 1. doi:10.1186/1759-4499-2-1

Stegmaier, J., Skanda, D., and Lebiedz, D. (2013). Robust optimal design of experiments for model discrimination using an interactive software tool. *PLoS ONE* 8:e55723. doi:10.1371/journal.pone.0055723

Szczurek, E., Gat-Viks, I., Tiuryn, J., and Vingron, M. (2008). Elucidating regulatory mechanisms downstream of a signaling pathway using informative experiments. *Mol. Syst. Biol.* 5, 287–287. doi:10.1038/msb.2009.45

Vatcheva, I., de Jong, H., Bernard, O., and Mars, N. J. I. (2005). Experiment selection for the discrimination of semi-quantitative models of dynamical systems. *Artif. Intell.* 170, 472–506. doi:10.1016/j.artint.2005.11.001

Videla, S., Guziolowski, C., Eduati, F., Thiele, S., Gebser, M., Nicolas, J., et al. (2014). Learning Boolean logic models of signaling networks with ASP. *Theor. Comput. Sci.* doi:10.1016/j.tcs.2014.06.022

Yeang, C., Mak, H. C., McCuine, S., Workman, C., Jaakkola, T., and Ideker, T. E. (2005). Validation and refinement of gene-regulatory pathways on a network of physical interactions. *Genome Biol.* 6, R62. doi:10.1186/gb-2005-6-7-r62

Conflict of Interest Statement: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

frontiers in
BIOENGINEERING AND BIOTECHNOLOGY

REVIEW ARTICLE
published: 19 December 2014
doi: 10.3389/fbioe.2014.00075

Toward synthesizing executable models in biology

***Jasmin Fisher*[1,2]*, *Nir Piterman*[3] *and Rastislav Bodik*[4]**

[1] *Microsoft Research, Cambridge, UK*
[2] *Department of Biochemistry, University of Cambridge, Cambridge, UK*
[3] *Department of Computer Science, University of Leicester, Leicester, UK*
[4] *Electrical Engineering and Computer Science, University of California Berkeley, Berkeley, CA, USA*

Edited by:
David A. Rosenblueth, Universidad Nacional Autónoma de México, Mexico

Reviewed by:
Hongying Dai, Children's Mercy Hospital, USA
Hannes Klarner, Freie Universität Berlin, Germany

***Correspondence:**
Jasmin Fisher, Microsoft Research Cambridge, 21 Station Road, Cambridge CB1 2FB, UK
e-mail: jasmin.fisher@microsoft.com, jf416@cam.ac.uk

Over the last decade, executable models of biological behaviors have repeatedly provided new scientific discoveries, uncovered novel insights, and directed new experimental avenues. These models are computer programs whose execution mechanistically simulates aspects of the cell's behaviors. If the observed behavior of the program agrees with the observed biological behavior, then the program explains the phenomena. This approach has proven beneficial for gaining new biological insights and directing new experimental avenues. One advantage of this approach is that techniques for analysis of computer programs can be applied to the analysis of executable models. For example, one can confirm that a model agrees with experiments for all possible executions of the model (corresponding to all environmental conditions), even if there are a huge number of executions. Various formal methods have been adapted for this context, for example, model checking or symbolic analysis of state spaces. To avoid manual construction of executable models, one can apply synthesis, a method to produce programs automatically from high-level specifications. In the context of biological modeling, synthesis would correspond to extracting executable models from experimental data. We survey recent results about the usage of the techniques underlying synthesis of computer programs for the inference of biological models from experimental data. We describe synthesis of biological models from curated mutation experiment data, inferring network connectivity models from phosphoproteomic data, and synthesis of Boolean networks from gene expression data. While much work has been done on automated analysis of similar datasets using machine learning and artificial intelligence, using synthesis techniques provides new opportunities such as efficient computation of disambiguating experiments, as well as the ability to produce different kinds of models automatically from biological data.

Keywords: executable biology, synthesis, verification, Boolean networks, signaling pathways

EXECUTABLE BIOLOGY

Investigating phenomena through the scientific method is an iterative process of hypothesis-driven experimentation. We observe the world around us, experiment with it, and, based on the experimental data, come up with hypotheses trying to explain how the systems that we study actually work (**Figure 1A**). These hypotheses lead to new predictions that then need to be tested in the real world. In biology, working hypotheses are referred to as mechanistic models aiming to provide a mechanistic explanation for observed phenomena. *Executable Biology* is an emerging field focused on the construction of such mechanistic models as executable computer programs. The basic construct of these computer programs (or computational models) is a state-machine, which relates different states to one another by defining how given certain events (e.g., a molecular signal), one state is transformed into another (Fisher and Henzinger, 2007). The components composing such a state-machine represent biological entities, such as cells, proteins, or genes that react to events involving neighboring components by state transformations. These state-machines can then be composed together to form complex computational models representing biological behaviors. As opposed to quantitative mathematical models such as stochastic and dynamic models, computational models are qualitative, as they explain the cause of observed phenomena. A major advantage of qualitative models is that different models can be used to describe the same biological phenomena at different levels of detail (abstraction), and that the different levels can be formally related to one another. For example, models can represent the molecular level, or, at a higher level of abstraction, they may represent the cellular level.

Computational models can be used to test different mechanistic hypotheses. Since the computational model represents a hypothetical mechanism that results in the experimental data, when we execute the model we can formally check whether a possible outcome of the mechanism is consistent with the data. Due to the non-deterministic nature of biological models, it is impossible to exhaustively test that all possible executions of a model conform to the data. Model-checking, on the other hand, is a technique that systematically analyzes all possible outcomes of a computational model without executing them one by one (Clarke et al., 1999). Hence, if model-checking verifies that all possible outcomes of our computational model agree with the experimental data, and that all the experimental observations can be reproduced by the model,

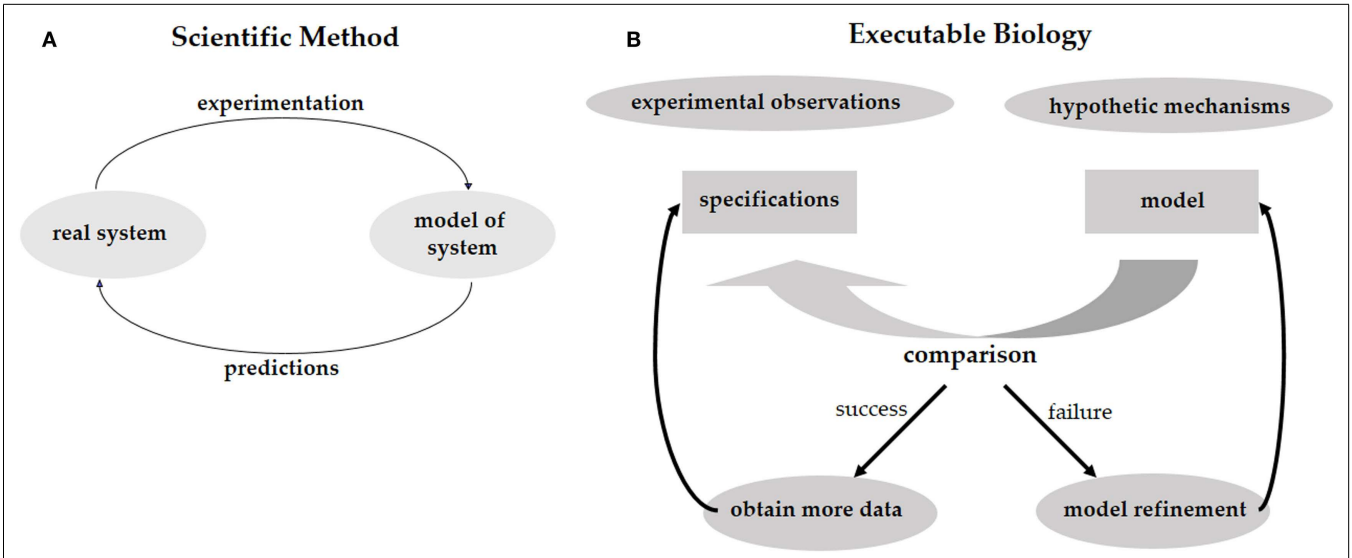


FIGURE 1 | (A) The scientific method calls for the elaboration of a predictive model of the system under study. The model should reproduce the existing experimental results and should be predictive regarding future experiments. By performing these experiments we validate the model, or refine it to a better model that captures more facts about the system. **(B)** The same process for executable biology calls for the elaboration of a model in the form of a computer program. The model is compared with specifications obtained from experimental observations. Failure to reproduce the experimental results leads to a refinement of the model. Predictions are used to guide further experimentation.

then we have a guarantee that the model is realistic and represents a mechanism that fits and explains the data. In case some of the experimental data cannot be reproduced by the model then we know that the hypothesis is wrong. We then need to refine the model until it produces the additional outcomes. Furthermore, if some of outcomes of the computational model disagree with the experimental data then the mechanistic hypothesis represented by the model may be wrong and we would need to revise the model so it will only produce outcomes that are supported by the data. In this case, the refinement of the model will offer new predictions suggesting additional experiments in order to validate the mechanistic hypothesis represented by the model. Executable biology is therefore an interplay between collecting data in experiments and constructing executable models that capture a mechanistic understanding of how a particular system works. By executing these models under different conditions that correspond to the experimental data and comparing the outcomes to the experimental observations, we can identify inconsistencies between hypothetical mechanisms and the actual experimental observations. Similar to the scientific method, this iterative process leads to new hypotheses, which serve to refine the mechanistic model and then need to be validated experimentally (**Figure 1B**).

Instead of constructing executable mechanistic models manually, one can extract such models automatically from experimental data using a technique called synthesis. Program synthesis is a method used to extract computer programs from their high-level specification. In biology, this concept is extremely appealing, as we would like to avoid the laborious manual process of model construction, which is prone to conscious and unconscious biases and errors, and replace it with an automatic process to synthesize the model directly from the data. Obviously, such a process could yield many different models explaining the same data set, in which case another interesting point would be to identify a way to differentiate between the different models. This could be in the form of an experiment that could either verify or falsify a particular hypothetical model. Hence, automatically synthesizing models of biological programs from experimental data has tremendous advantages over manually constructed biological programs. Usage of synthesis could lead to significant advantages in terms of time and labor to produce models, in terms of our confidence in the inferred features of models, and in terms of the next steps to take to decide between multiple possible models.

MODELING METHODOLOGY

We now present the methodology of executable modeling, describing its steps as a workflow that a biologist might follow when developing and analyzing an executable model. We accompany the explanation with the details of a running example. We exemplify the process through a developmental model of a fragment of the *C. elegans* vulval precursor cells (VPC) system. We will model the lateral signaling mechanism that six adjacent VPC use to collectively determine their fate. The description here follows closely the elaboration of the synthesis process described in (Koksal et al., 2013).

CHOOSE A SUITABLE ABSTRACTION LEVEL

Based on the biological question, we choose the level of abstraction, including the biological entities and their possible values (states). The abstraction, coupled with assumptions given by the biologist, defines a space of possible models, and the role of synthesis will be to find models in this candidate space.

For example, we want to develop a model that explains how VPCs coordinate to determine their fate. Specifically, we are interested in (i) identifying which pairwise protein interactions are

involved in the coordination; (ii) how the cells gain resilience against "time conflicts," which arise when two cells happen to signal each other simultaneously; and (iii) how a cell avoids "self-signaling," which arises when a cell C1 signals its neighbors to take fate 2° – how does C1 ensure that it does not listen to its own signal and incorrectly enter fate 2°? These three modeling questions already dictate which entities and mechanisms must be preserved by the modeling abstraction and which might be abstracted away: the model must have an object per protein to uncover the effect of protein interactions. Similarly, there will be an object per cell, to enable modeling of inter-cell communication. Each such cell model will be composed of multiple proteins that will interact with proteins inside the cell as well as proteins (receptors) on neighboring cells. Finally, to model the effects of different times when signaling happens, our models will, in some fashion, need to model the progress of time.

Figure 2A (top right) shows the model of our cell comprises seven proteins and a decision circuit that models how the fates are determined. The figure also shows how six (identical) cells are connected into a system with the anchor cell (AC) and two

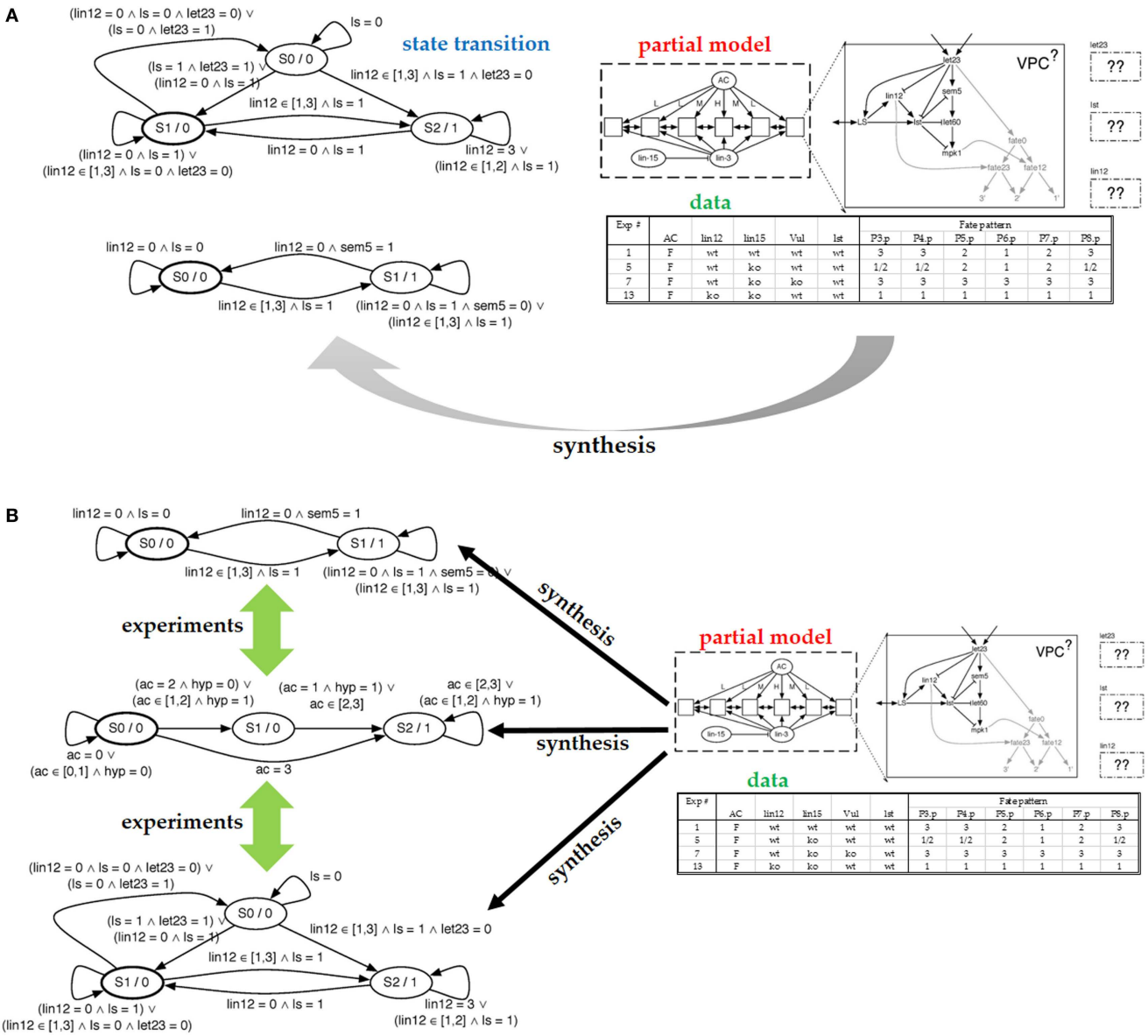


FIGURE 2 | (A) Partial model submitted to the synthesizer and the resulting state machine produced by the synthesizer. On the right, we see the structure of a cell with the components that comprise it. We see the configuration of the six cells and the communication allowed between them. Finally, we include some of the experimental data used to specify which models are correct. On the left, we see the resulting state transition diagrams produced for let23 (top) and lst (bottom). **(B)** In the case that the synthesis engine can produce multiple possible models that explain the data, we can ask for experiments that distinguish between the different models. Such experiments are expressed in terms of the experimental setting created for the synthesis effort.

external proteins communicating the interaction between cells (inter-cellular signaling). To define the model as an executable program, we discretize both time and protein levels. The model will execute in a fixed number of steps (10–15 steps in our setting). Protein concentrations will use a small number of levels (2–5). Discretization levels are set by the biologist. Protein behavior is modeled with a protein response function, which is specific to each protein. The model updates the protein level in each step. The protein response function reads the current level of the protein as well as the levels of incoming proteins, and computes the next state of the protein. The protein response function thus forms a state machine, with one state for each discretized concentration level, and transitions are predicates that test levels of incoming proteins. **Figure 2A** (left) shows two protein response functions.

To summarize the model semantics, each protein has a state (concentration) to model how the protein concentration evolves over time, which allows for control of when one protein triggers another. Cells are networks of proteins in which all proteins take their step simultaneously (synchronously); there is no need to execute proteins of a cell in arbitrary order, because we are only interested in time-sensitivity between cells. The six cells take steps one after another, controlled by the scheduler that models the different rates of progress of the cells. The role of the scheduler will be explained in the next subsection.

COLLECT DATA AND EXPRESS PRIOR INFORMATION AS A PARTIAL MODEL

There are two common methods for narrowing the space of possible models. First, one can state biological "certainties," such as which proteins are believed to participate in the system under study. Second, one can add the observed biological behaviors that the model needs to reproduce.

Existing biological knowledge will be used to construct an initial model, called a *partial* model, which defines the space of candidate models. We can think of the partial model as a parametric model whose parameters are determined during model synthesis. The partial model may include known biological entities (e.g., proteins), their possible states and interactions. For example, in the model in **Figure 2A**, the previous biological knowledge included the interactions between the entities and the encapsulation of entities to cells, based on previous data on the behaviors of the entities, the cells, and the VPC system in general. In addition, the structure of the protein response functions was decided based on beliefs regarding the sensing capabilities of the proteins and the assumption that all involved entities are represented in the model. One might also fix the type of interaction between the entities (inducing or inhibiting) and possibly restrict the number of arguments to protein response functions based on the number of active sites of the protein. These prior-belief restrictions define the "structure" of the candidate space and narrow down the search process.

The experimental data are used to test correctness of models in the candidate model space. The experimental data have to be mapped to the model level: the environmental conditions are viewed as the inputs to the model while the experimental observations are viewed as model outputs or intermediate states. In the VPC example, each experiment is a mutation-phenotype pair, where the mutation is the input to the computational model, while the phenotype is the output from executing the model. In more detail, the mutations either knock out a gene or constitutively turn it on. For each mutation, the experimental data records the fate taken by each of the six cells. There are three possible fates, and in each experiment, all cells are identically mutated. **Figure 2A** (bottom right) shows a portion of the table that maps mutations to fates. Our model will execute by first reading the particular mutation, which changes the behavior of the model. The model is then executed for a predetermined number of time steps, covering the period during which the cells coordinate. When the model terminates, it outputs the six fates.

Another possible example of model output is the time series of a certain combination of entity values. In all cases, the experiments are translated to the same constraint language that is used to express the structural restriction on the models.

Because a model can have multiple executions, we distinguish between allowable behavior and required behavior, using both types of behaviors to decide whether a particular model is correct. In **Figure 2A**, note that for the second mutation in the mutations-to-fates table, the experiments have observed multiple fates. Presumably, this stochasticity in the cell is due to the loss of "synchronization" between cells caused by the mutation. A correct model will need to be able to reproduce each of the observed fates – they represent the allowable behavior. The required behavior is that each of these alternative fates must be reproducible. That is, there must be an execution of the model that produces one observed fate and another execution that produces the other fate. To endow our model with the ability to reproduce this stochastic behavior, we make the model non-deterministic. We view non-determinism as an abstraction of stochasticity, in that our model will not tell us the probability with which each fate can be reached, only whether it can be reached. The benefit of using non-determinism is that it is not necessary to use randomness to make the model behave stochastically. To make the model non-deterministic, it will suffice to control how the six cells interleave their steps; we call this interleaving a *schedule*. The model includes a scheduler that can non-deterministically select one of the possible schedules.

VERIFICATION OF MODELS

The model we have described so far is not specific to synthesis. In fact, one can develop the model entirely manually. It will be desirable to verify this model, which would mean to ensure that for each mutation, the model produces the indicated fates no matter which schedule is selected by the scheduler, and that all alternative fates are produced by some schedule. This verification is performed without explicitly enumerating all schedules, as there are too many. Instead, the model is translated into logical constraints which are supplied to a solver, which in turn is asked to find a schedule that fails to produce the indicated fates. If the solver proves that no such schedule exists, the model has been verified.

SYNTHESIZE AND ENUMERATE ALTERNATIVE MODELS

It is usually tedious to manually develop a complete model that verifies against the experiments. To employ synthesis, we ask the synthesizer to complete the partial model into a verifiable model. At the technical level, the synthesis works as a search process. The

partial model defines a space of candidate models. Each candidate corresponds to one possible completion of the partial model. Most of the candidate models are typically incorrect (i.e., they disagree with the experimental data), and the synthesizer needs to find a candidate that is correct. As in verification, the search is formulated as solving of a set of constraints. The translation to constraints is performed by a tool (a special compiler) that accepts any partial model and produces logical constraints.

Usually, this will be formed as a type of constraint satisfaction problem, and it is the role of a given solver to search for a solution for it. The translation back and forth between the constraint language and the models and their potential behavior is the main programing task in this endeavor. If the solver manages to find a possible solution then this is a potential model. If the synthesizer fails to find possible models, the prior knowledge and assumptions have to be reconsidered in order to enlarge the search space, for example, by adding additional protein interactions. Having synthesis algorithms produce useful information for such enlargement is an ongoing research topic.

In the VPC case study, the laborious aspect of model development is to write protein response functions that collectively behave as the real cell. Therefore, these response functions will be produced by the synthesizer. These functions will describe how proteins respond to suppression and activation, informally, how long it takes for proteins to become activated. If no such function can be found, we assume that the model is missing an interaction, and the prior knowledge needs to be revised, as done by Fisher et al. (2007). **Figure 2A** (top right) shows the partial model supplied to the synthesizer; the response functions for three proteins need to be synthesized. Given the table with experimental data and the number of steps to be taken, the synthesizer completes the partial model with protein response functions. Two of the synthesized response functions are shown on the left. This model is correct in that it can be verified as described above.

ANALYSIS OF THE SPACE OF FEASIBLE MODELS

The synthesizer has discovered that multiple models match the data. It is therefore natural to ask whether these models represent alternative match the data. It is therefore natural to ask whether these models represent alternative explanations of how the cells behave. Given how we posed the problem, these models are equivalent, because we have fixed the interaction network and solved for transfer functions. Hence, our models will differ only in their protein responses, which we typically do not consider to be different explanations. To arrive at an alternative explanation, we pose to the synthesizer a different partial model (with a different set of interactions) and ask whether response functions exist for that model. An alternative technique is to give the synthesizer a partial model with a superset of interactions and then read out the synthesized response functions: if a function ignores an incoming protein, then we can remove that incoming edge, arriving at another model. In **Figure 2B**, we show alternative models synthesized from our partial model.

COMPUTE ADDITIONAL EXPERIMENTS TO CONDUCT

If alternative models exist, it is because we do not possess sufficient experiments to narrow down the candidate space to a single model. To rule out some alternative models, we can ask the synthesizer to compute additional experiments for which the measurable result differs between the alternative models. In the VPC case study, this is done by searching the space of mutations (for which experiments have not yet been performed), looking for a mutation such that at least two alternative models differ in their fate outcomes. Performing this experiment and adding its result to those that guide the synthesis process is guaranteed to rule out some of the models. If no such experiment exists, then, from the point of view of the existing experimental system, it is impossible to distinguish between the alternative models, and additional experimental methods need to be considered in order to facilitate ruling out some of these models.

Minimizing the number of experiments

Assume that you want to rerun the experiments, for example, to increase your confidence in the measurements. Do you need to perform all the experiments or is it sufficient to redo a small subset of experiments? Indeed, prior knowledge may make some experiments unnecessary. Alternatively, one or more experiments may collectively make some other experiments superfluous, because no new knowledge useful to model inference is present in the latter set of experiments. This problem is again posed as a search over a sufficient set of experiments that will infer the same set of plausible models as the full set of experiments. In our case study, we were able to reduce the number of experiments from 48 to 4.

DEVELOPING MODELS WITH SYNTHESIS

We now aim to generalize some considerations discussed above and present questions that need to be answered in order to apply synthesis effectively in biological domains. We follow the structure of the previous section and revisit the issues that were highlighted there.

CHOOSE A SUITABLE ABSTRACTION LEVEL

The first question that needs answering is whether the biological question that we have in mind can be helped by synthesis. At the current level, successful applications of synthesis are restricted to constructing discrete models at a relatively low level of detail[1]. For example, models of continuous evolution of protein networks that match time series expression levels are more suited for other methods. The synthesis techniques we talk about here are based on constraint solving. Such techniques are more appropriate when the values of entities can be represented by discrete values and their changes over time are abstracted to talk about "what happens next," ignoring the detail of "when exactly" it happens. This dictates a relatively high level of abstraction and questions that relate to, for example, possible interactions, causality, and nature of interaction. The kind of models that can be produced are, for example, Boolean networks (Kauffman, 1969), state transition diagrams (Efroni et al., 2007), and Petri-nets (Bonzanni et al., 2013). Such programs will generally manipulate variables ranging over discrete domains and changing by transition rules that set a next

[1]We consider work for parameter estimation of networks of differential equations as belonging to a different class of applications. Indeed, the technique it relies on is completely different to those used here. For a survey of available techniques for parameter estimation we refer the reader to Jianyong et al. (2012).

value based on the current value of an entity and those affecting it. The stress on current and next is intentional. It marks a clear difference between the mathematical models that talk about the transformation of values over time and the computational models that we have in mind here.

SYNTHESIZE AND ENUMERATE ALTERNATIVE MODELS

One can look for differences or commonalities between all correct models leading to further biological experimentation. For example, one can look on the options synthesized for a certain part of the system. If the set of options is very restricted, this means that the existing knowledge and experiments lead to a good understanding of the structure of this part of the system. If the number of options is very large and potentially contradictory then additional information about this part is missing. In particular, summarizations of the entire space of correct models can give us information on what is impossible (if no model uses such a feature) and what must be true (if all models have such a feature). One of the questions that will need resolution in order to make synthesis more applicable in the biological domain is how to better classify the set of potential models and what kinds of questions can be asked about them, which could also be useful for manual elaboration of models.

COMPUTE ADDITIONAL EXPERIMENTS TO CONDUCT

Similar techniques to those applied to produce the model resulting from synthesis can be used to search the space of experiments. In particular, adding information about the cost of experiments and their feasibility could narrow down the space of possible experiments and suggest that the "easiest" experiments to perform that would still give information valuable in ruling out potential models.

OVER FITTING TO EXPERIMENTS

One of the issues in parameter estimation techniques is that of over fitting models to noisy and unreliable data, leading to models that are too restrictive. We stress that the approach to tackle over fitting within the context of synthesis must be different. Here, the technique itself is structured so as to determine a space of possible models. It does not make sense to synthesize with only some of the information in hand and then test the resulting models with additional data. Indeed, the step that could lead to over fitting is in the definition of the space of possible models and not in the partition of the space between "correct" and "incorrect" models. First, since the technique can declare that a certain space does not contain "correct" models, the risk of over fitting is somewhat reduced. Over fitting may result in models that are not realistic but can still explain all the observed phenomena. The predictions of such models should lead to the identification of the errors in the definition of the search space. Second, the technique is geared toward the production of multiple models and not the "best fitting model." The resilience to over fitting should be part of the definition of the space of possible models and the type of correspondence between the models and the experimental results. Removal of some of the experimental results and testing them at a later stage, essentially will lead to either of the two answers that we would have reached in the first place: a narrower space of "correct" solutions or the non-existence of a "correct" solution.

ADDITIONAL EXAMPLES

We explore a few more examples of the usage of synthesis techniques as described above.

SYNTHESIZING BOOLEAN NETWORKS FROM GENE EXPRESSION EXPERIMENTS

Another example of synthesis is the following application to the extraction of a Boolean network from experimental data (Guziolowski et al., 2013). The existing biological knowledge consists of the connections between the different biological entities along with their directions. That is, the authors assume that they know which proteins interact and, for every interaction, whether the interaction is positive or negative. This information is summarized in the form of a directed and annotated graph $(G = V, E)$, where V is a set of nodes, and $E \subseteq V \times V$ is the set of edges. The annotation of edges with $+$ and $-$ signs is given separately.

The assumption on the structure of the model is that it is a Boolean network. That is, every biological entity corresponds to a variable that is either on or off (0 or 1), and there are rules that govern the changes in values of these entities according to the values of the entities that have an edge to them. In particular, the function that sets the value of an entity is a Boolean function that includes all the entities that affect the entity we are interested in and where the sign of every interaction is respected. So, an entity that affects positively cannot have an inverse effect with every other possible combination of the other inputs and vice versa. It is well known that such networks stabilize according to these rules. Thus, there are certain states (assignments of values to all the variables) in which updates produce no change.

The experimental framework assumes some inputs $I \subseteq V$, which are biological entities that can be affected by experiments, and some outputs $O \subseteq V$, which are biological entities that can be measured. Thus, the set of experiments that can be done on this network are to set the values of the inputs (by mutation or other intervention) and to measure the values of the outputs (e.g., phosphorylation values). Outputs are discretized to a number of levels that correspond to noise level in the measurements. In this particular case, the output is discretized to 100 levels. It is assumed that the output values that are associated with a certain input correspond to the stability point of the network when the inputs are set to the experimental value.

The utility of the network is measured by the sum of the square distance of the measured outputs from the output of the network. For a specific experiment e and specific output $o \in O$, we write $\theta_{e,o}$ as the value of this output in this experiment. Similarly, given a candidate Boolean network, the stability value of a certain output under the same experiment is denoted $\rho_{e,o}$. The distance of a specific experiment is $d_e = 1/m \sum_o (\rho_{e,o} - \theta_{e,o})^2$, where $m = |O|$. That is, the average of the square of the distance from the network prediction and the actual experimental results over all outputs. The overall distance of the network is the average of the distances overall experiments, that is $d = 1/n \sum_e d_e$, where n is the number of available experiments. The network utility is to minimize the distance of the network from experiments and at the same time minimize the size of the network (as measured by the size of the functions that govern changes in variable values).

The authors of this research then pose this question as a search question. Find the network that best matches the experimental data with the minimal size. The initial search yields 16 models that result from allowing two possible functions for four entities ($2^4 = 16$ – not very surprising as searching for a minimum often does). But relaxing the requirements to within a distance of 10% from the possible minimum to take noise into account produces about 10,000 possible networks.

The main strength of this analysis technique is that it is now possible to analyze all these 10,000 networks simultaneously and derive conclusions from their commonalities. For example, for some entities the same functions occurred in *all* the models. Functions that do not appear in even a single model are overruled. And some entities had a very small number of possible functions. In addition, the analysis can extract whether it is possible to perform experiments within the given experimental framework that will distinguish between models. That is, design an experiment over the given inputs so that the two different networks will have different values for the given outputs, and would thus be measurable by a given experiment. Every two models that cannot be distinguished by the experiments are considered equivalent from the point of view of the experimental setting and they found that there are 91 such classes of models. They proceed to propose experiments that will overrule some of these model classes.

The work of Sharan and Karp (2013) uses different underlying techniques for searching; however, their definition of the problem is quite similar to the setting above. As in the previous work, they search for Boolean networks. Their assumptions regarding the knowledge about possible interactions is weaker, in that they do not assume that they know the directions of interaction and also accept if interactions do not have an inhibition/activation annotation. As a result, they allow more general functions for the next state function of individual entities in the Boolean network. They use the same measurement for the distance between the experimental results and the network behavior. Finally, the underlying solving techniques are through integer linear programing (ILP) and not constraint (or Boolean) solving as we mostly do here.

A similar application of synthesis to extract Boolean networks is in (Dunn et al., 2014). The authors again search for a Boolean network that matches a given experimental setting. Here, the assumptions are somewhat different, leading to some different choices. The relevant biological data correspond to possible connections between the entities but this time without directionality and without the label of activation/inhibition. Accordingly, it is the role of the synthesis engine to find the exact connections as well as the way that entities affect each other. The update functions for the entities are restricted to a small set of possible functions. The last choice significantly narrows the search space. The treatment of the experimental data is also different. Here, the authors do not assume the existence of inputs but rather search for an execution of the network from a given initial state corresponding to the experimental setting to a final state corresponding to the measurements. This time the experimental data are made Boolean by the authors and the matching between an experimental measurement and the state of the Boolean network has to be exact. As before, the authors summarize all possible models in the space that match the experimental data. They draw conclusions regarding common features of all these models and experimentally verify some of their predictions.

FUTURE PROSPECTS AND OPEN PROBLEMS

In all the cases discussed previously, executions of models were considered bounded and of a certain length. It would be interesting to lift this restriction and relieve the modeler from the need to make this decision. Techniques that support such synthesis efforts are in general more complicated, and it would be very interesting to see them adapted for the biological context (Vardi, 2008; Kupferman, 2012).

The level of abstraction we discussed above is very convenient for synthesis. We assume that genes have discrete levels of expression. It would be very interesting to devise techniques that produce models at various levels of abstraction, for example, accompany the transfer functions supplied above by molecular interaction models that could give rise to the same behavior.

In the detailed case study presented in an earlier section, the model reproduced stochastic behavior with non-determinism based on Boolean logic: a particular fate could either happen or not happen. Logical modeling was sufficient in that case study, because both the inputs to the model (mutations) and the outputs (fates) were discrete values. In modeling situations where the data are quantitative and noisy, the modeling may need to prevent discretization and may require stochastic reasoning with probabilistic distributions, requiring that we change our underlying reasoning engine from a logical one to a probabilistic one [see e.g., Fränzle et al. (2008)]. Genomic high-throughput data falls into this category. Much more work is needed to make this modeling transition.

When one considers synthesis, the most important part of the synthesizer is the compiler that translates a partial model into constraints. The construction of the compiler can be laborious, especially if the model has advanced semantics, as it did in our detailed case study in an earlier section. To simplify the process of creating the compiler, it may be possible to rely on the recently developed symbolic virtual machine (Torlak and Bodik, 2014), which allows one to define the modeling language in a simple way, by writing a so-called interpreter. The symbolic virtual machine produces the compiler automatically from the interpreter. As we have suggested, making such a tool easier for domain experts to use is required and far from accomplished. The efforts described above rely on extensive collaborations between biologists and computer scientists. Gaining more experience in synthesis at a level that will allow the creation of custom level tools that can be used for general synthesis projects, rather than being custom made for a certain synthesis effort is a very ambitious goal. Making such tools usable by domain experts (biologists) is a further challenge. This also implies that at this stage, potential users of the technique cannot rely on existing tools and must invest in the development of synthesis engines for their own needs.

Further case studies should also consider the size boundary applied in the case studies described earlier. The current limit of such techniques is applications to systems with a few tens of proteins (working on standard desktop computers). Scaling the techniques applied above to hundreds of proteins is a major challenge that will require improvements to the underlying solvers as

well as finding more efficient ways to encode the specific synthesis questions arising from biology in better ways.

CONCLUDING REMARKS

In recent years, we have seen an increase in the usage of executable biology for modeling various biological systems and phenomena. Advantages, such as the ability to automatically check whether a model adheres to requirements arising from biological data and to answer further queries about the model, make this set of techniques more applicable. In computer science, the research on verification of models has led to work on automatic synthesis of models from their high-level descriptions. Here, we give a short survey of this technique and how it can be used for biological modeling. We summarize some of the main instances where synthesis has been applied to the production biological models and how this extra power gives further insights into the model in question. Finally, we also discuss some of the future developments needed in order to make this technique more applicable for biological research.

REFERENCES

Bonzanni, N., Garg, A., Feenstra, K. A., Schutte, J., Kinston, S., Miranda-Saavedra, D., et al. (2013). Hard-wired heterogeneity in blood stem cells revealed using a dynamic regulatory network model. *Bioinformatics* 29, i80–i88. doi:10.1093/bioinformatics/btt243

Clarke, E. M., Grumberg, O., and Peled, D. (1999). *Model Checking*. Cambridge, MA: MIT Press.

Dunn, S. J., Martello, G., Yordanov, B., Emmott, S., and Smith, A. G. (2014). Defining an essential transcription factor program for naive pluripotency. *Science* 344, 1156–1160. doi:10.1126/science.1248882

Efroni, S., Harel, D., and Cohen, I. R. (2007). Emergent dynamics of thymocyte development and lineage determination. *PLoS Comput. Biol.* 3:e13. doi:10.1371/journal.pcbi.0030013

Fisher, J., and Henzinger, T. A. (2007). Executable cell biology. *Nat. Biotechnol.* 25, 1239–1249. doi:10.1038/nbt1356

Fisher, J., Piterman, N., Hajnal, A., and Henzinger, T. A. (2007). Predictive modeling of signaling crosstalk during *C. elegans* vulval development. *PLoS Comput. Biol.* 3:e92. doi:10.1371/journal.pcbi.0030092

Fränzle, M., Hermanns, H., and Teige, T. (2008). "Stochastic satisfiability modulo theory: a novel technique for the analysis of probabilistic hybrid systems," in *Hybrid Systems: Computation and Control*, eds M. Egerstedt and B. Mishra (Berlin: Springer), 172–186.

Guziolowski, C., Videla, S., Eduati, F., Thiele, S., Cokelaer, T., Siegel, A., et al. (2013). Exhaustively characterizing feasible logic models of a signaling network using answer set programming. *Bioinformatics* 29, 2320–2326. doi:10.1093/bioinformatics/btt393

Jianyong, S., Garibaldi, J. M., and Hodgman, C. (2012). Parameter estimation using metaheuristics in systems biology: a comprehensive review. *IEEE/ACM Trans. Comput. Biol. Bioinform.* 9, 185–202. doi:10.1109/tcbb.2011.63

Kauffman, S. A. (1969). Metabolic stability and epigenesis in randomly constructed genetic nets. *J. Theor. Biol.* 22, 437–467. doi:10.1016/0022-5193(69)90015-0

Koksal, A. S., Pu, Y., Srivastava, S., Bodik, R., Fisher, J., and Piterman, N. (2013). "Synthesis of biological models from mutation experiments," in *Proceedings of the 40th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages* (Rome: ACM), 469–482.

Kupferman, O. (2012). "Recent challenges and ideas in temporal synthesis," in *SOFSEM 2012: Theory and Practice of Computer Science*, eds M. Bieliková, G. Friedrich, G. Gottlob, S. Katzenbeisser, and G. Turán (Berlin: Springer), 88–98.

Sharan, R., and Karp, R. M. (2013). Reconstructing Boolean models of signaling. *J. Comput. Biol.* 20, 249–257. doi:10.1089/cmb.2012.0241

Torlak, E., and Bodik, R. (2014). "A lightweight symbolic virtual machine for solver-aided host languages," in *Proceedings of the 35th ACM SIGPLAN Conference on Programming Language Design and Implementation* (Edinburgh: ACM), 530–541. doi:10.1145/2594291.2594340

Vardi, M. (2008). "From verification to synthesis," in *Verified Software: Theories, Tools, Experiments*, eds N. Shankar and J. Woodcock (Berlin: Springer), 2–2.

Conflict of Interest Statement: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Received: 11 August 2014; accepted: 05 December 2014; published online: 19 December 2014.

*Citation: Fisher J, Piterman N and Bodik R (2014) Toward synthesizing executable models in biology. Front. Bioeng. Biotechnol. **2**:75. doi: 10.3389/fbioe.2014.00075*

This article was submitted to Bioinformatics and Computational Biology, a section of the journal Frontiers in Bioengineering and Biotechnology.

frontiers in Genetics

REVIEW
published: 10 September 2015
doi: 10.3389/fgene.2015.00285

A survey about methods dedicated to epistasis detection

Clément Niel[1]*, *Christine Sinoquet*[2], *Christian Dina*[3] and *Ghislain Rocheleau*[4]

[1] Computer Science Institute of Nantes-Atlantic (Lina), Centre National de la Recherche Scientifique UMR 6241, Ecole Polytechnique de l'Université de Nantes, Nantes, France, [2] Computer Science Institute of Nantes-Atlantic (Lina), Centre National de la Recherche Scientifique UMR 6241, University of Nantes, Nantes, France, [3] Institut du Thorax, Institut National de la Santé et de la Recherche Médicale UMR 1087, Centre National de la Recherche Scientifique UMR 6291, University of Nantes, Nantes, France, [4] European Genomic Institute for Diabetes FR3508, Centre National de la Recherche Scientifique UMR 8199, Lille 2 University, Lille, France

During the past decade, findings of genome-wide association studies (GWAS) improved our knowledge and understanding of disease genetics. To date, thousands of SNPs have been associated with diseases and other complex traits. Statistical analysis typically looks for association between a phenotype and a SNP taken individually via single-locus tests. However, geneticists admit this is an oversimplified approach to tackle the complexity of underlying biological mechanisms. Interaction between SNPs, namely epistasis, must be considered. Unfortunately, epistasis detection gives rise to analytic challenges since analyzing every SNP combination is at present impractical at a genome-wide scale. In this review, we will present the main strategies recently proposed to detect epistatic interactions, along with their operating principle. Some of these methods are exhaustive, such as multifactor dimensionality reduction, likelihood ratio-based tests or receiver operating characteristic curve analysis; some are non-exhaustive, such as machine learning techniques (random forests, Bayesian networks) or combinatorial optimization approaches (ant colony optimization, computational evolution system).

Keywords: epistasis detection, genome-wide association study, complex disease, biological data mining, feature selection

OPEN ACCESS

Edited by:
David A. Rosenblueth, Universidad Nacional Autónoma de México, Mexico

Reviewed by:
Jingyi Jessica Li, University of California, Los Angeles, USA
Xiaodan Fan, The Chinese University of Hong Kong, Hong Kong

***Correspondence:**
Clément Niel, Computer Science Institute of Nantes-Atlantic (Lina), Centre National de la Recherche Scientifique UMR 6241, Ecole Polytechnique de l'Université de Nantes, Rue Christian Pauc, BP 50609, 44306 Nantes, France
clement.niel@univ-nantes.fr

Specialty section:
This article was submitted to Bioinformatics and Computational Biology, a section of the journal Frontiers in Genetics

Received: *13 May 2015*
Accepted: *27 August 2015*
Published: *10 September 2015*

Citation:
Niel C, Sinoquet C, Dina C and Rocheleau G (2015) A survey about methods dedicated to epistasis detection. Front. Genet. 6:285. doi: 10.3389/fgene.2015.00285

Introduction

Genome-wide association studies (GWAS) have generated huge datasets in the past 8 years in order to find association between genetic polymorphisms and phenotypes. Individual risk prediction based on those discoveries was promising. Nevertheless, genetic architecture of complex diseases, such as type II diabetes, is still largely misunderstood (Vassy et al., 2014). Indeed, gene-environment and gene-gene interactions must be considered to better understand etiology of such phenotypes. In other words, various joint effects of genetic variations, namely epistasis, are likely to partly determine the disease state (Mackay and Moore, 2014). While common genome-wide association analysis checks for potential SNP-disease associations in a one-SNP-at-a-time fashion, looking for all potential epistatic interactions in such datasets will quickly result in combinatorial overload. This is why classical GWAS often left behind the daunting task of epistasis detection.

Several strategies came up to overcome the epistasis intricacy. After a first section dealing with epistasis generalities, we will present in this review the main categories of methods dedicated to epistasis detection. These methods are classified as follows. First, some exhaustive approaches for searching significant genetic marker combinations will be introduced. As some of these, like

Multi-Dimensional Reduction (MDR), are not manageable at a genome-wide scale, we will next turn our attention to filtering strategies which aim at reducing the size of the dataset, thereby decreasing the size of the search space. A final section will deal with machine learning and data mining techniques. This review does not intend to provide an exhaustive list of all software programs designed to find epistatic interactions, but rather to give an overview of the main categories of strategies put forward in the last 5 years.

Background—Epistasis

During the past decade GWAS have played a central role in the discovery of genotype-phenotype associations. In GWAS analyses, geneticists rely on DNA polymorphism markers to detect these associations. One of the most popular classes of genetic markers, Single Nucleotide Polymorphism (SNP), allows comparison of allelic frequencies between a sample of cases ascertained for a disease and a sample of controls. In the standard approach, SNPs are tested one by one for statistical association with the disease (Hirschhorn, 2009). Genetic variants are considered to have independent effects on the phenotype. As a result, only additive effects are considered under this approach. This kind of analysis has been widely used for years, but results are often not as appealing as expected. Indeed, with the "one locus at a time" strategy, only a little part of the genetic variance explains the phenotype, the remaining part being referred to "missing heritability" (Maher, 2008; Manolio et al., 2009).

It has been commonly admitted that missing heritability is partly due to genetic variants showing effects when they interact with one or more other variants (Eichler et al., 2010). Epistasis refers to the combinatorial effect of one or more genetic variants (**Figure 1**). These effects might interactively contribute besides existing marginal effects or they can also exist in absence of any marginal effect. In the last case, traditional statistical parametric methods will likely miss those interactions owing to the inflexibility of parametric models (Culverhouse et al., 2002; McKinney et al., 2006). For instance, in complex diseases like asthma (Howard et al., 2002), diabetes (Cho et al., 2004) or hypertension, additive genetic variation involves many SNPs, among which a vast majority have very small effect sizes (odds ratio less than 1.2, see **Box 1**) (Ritchie, 2015). As complex traits are poorly explained by additive models, one expects gene-environment or gene-gene interactions to substantially contribute to the genetics of these diseases.

Thus, epistasis detection has become an important field of research in human genetics: more complex models are studied nowadays, where combinations of genetic variants are examined for association with a trait. From a biological point of view, it seems unlikely that some phenotypes are only driven by genetic variants acting independently. For instance, large and complex networks of gene-gene and protein-protein interactions are well known in systems biology for their high connectivity, density and resistance to variation (Boone et al., 2007). Moreover, it has been observed that consequences of induced mutations are greatly variable in different genetic backgrounds (Mackay, 2014). Once aware of all this, it seems inconsistent to see gene-gene interactions as rare events.

Biological Epistasis and Statistical Epistasis

First, it is essential to distinguish biological epistasis (also called functional epistasis) from statistical epistasis (Cordell, 2002). The term biological epistasis was coined by Bateson (1909). In its original definition, it only involved allele effect at one locus concealed by the effect of another allele at a second locus. This can be seen as a broadening of the dominance concept at an inter-loci level. A more recent definition also allows genetic variant effects to be enhanced by effects of other genetic variants (Siemiatycki and Thomas, 1981). Generally, speaking, an epistatic effect exists when the effect of an allele at a genetic variant depends either on the presence or absence of another genetic variant.

On the other hand, statistical epistasis refers to the departure from additive effects of genetic variants at different loci with regard to their global contribution to the phenotype (Wang et al., 2010a). This definition was proposed by Fisher (1918). One relies on this definition when one wants to detect epistatic interactions with computational methods. Ultimately, the goal consists in interpreting interactions found to be statistically relevant in order to get closer to their biological definition and to apprehend the underlying functional mechanisms. This last step is undoubtedly the more difficult one (Moore and Williams, 2005) and is often disregarded.

A recent concrete example of epistasis has been described by Gertz et al. (2010), where three SNPs were shown to be involved in an epistatic interaction in yeast *Saccharomyces cerevisiae* (**Figure 2**). In the following, italic characters refer to the gene while normal characters refer to the corresponding protein. One SNP is located in the promoter region of *RME1* which encodes a transcription factor repressing the transcription of *IME1*, a gene coding for a transcription factor which promotes sporulation. State of this SNP influences the production rate of RME1. The second SNP is located in the promoter region of *IME1*. Its state affects the binding specificity of RME1-*IME1*. The third SNP lies in the coding region of *IME1* and its state conditions the binding specificity of IME1-kinase, which is the active form of IME1. Gertz and coworkers showed that the allele combination of these SNPs have a non-additive effect on the RME1-*IME1* binding and on the sporulation efficiency. Consequently, sporulation efficiency is partly ruled by epistasis. Many other cases of epistasis have been evidenced recently (Smith et al., 2014; Ellis et al., 2015; Huang et al., 2015; Liu et al., 2015; Matsubara et al., 2015).

Origin of Epistasis: an Evolutionary Point of View

Canalization is a theory proposed by Waddington (1942). It is based on a generally admitted assumption: natural selection maintains the majority of a population into a healthy condition. Thus, in response to genetic and environmental variations, phenotypic modifications are buffered. This is especially true for vital physiological levels, such as blood glucose or blood pressure. To this end, evolution has favored complex robust systems resistant to variations (Moore and Williams, 2009). A compelling argument in favor of this hypothesis is the redundancy rate in biological networks. This feature is well known in systems biology

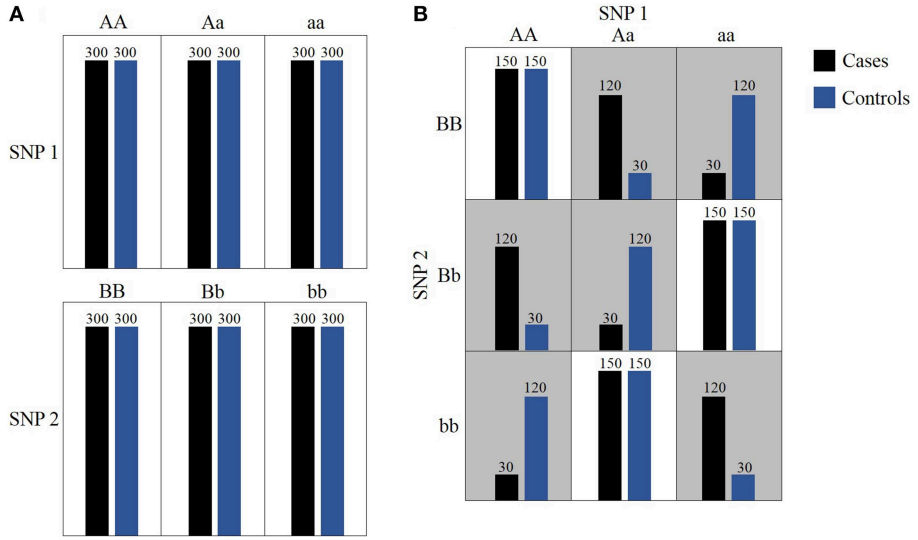


FIGURE 1 | Toy example of epistasis. (A) Neither SNP 1 nor SNP 2 presents a marginal effect. **(B)** In gray cells, allele combinations between SNP 1 and SNP 2 induce statistically significant epistatic effect on the phenotype distribution.

BOX 1 | Logistic regression and odds ratios.

A logistic regression model is a statistical model that depicts the relationship between a linear combination of variables (e.g., SNPs in a GWAS) and a binary trait, the disease phenotype (i.e., affected/unaffected status). The probability p of being affected is expressed in the log scale as:

$$\log\left(\frac{p}{1-p}\right) = \alpha + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2$$

where x_1 and x_2 each correspond to the at-risk genetic variants, $x_1 x_2$ accounts for the interaction between them, and β_i are parameters being estimated from the data.

Odds ratios are highly related to logistic regression models. Indeed, $\exp(\beta_x)$ is an estimate of the odds ratio between the outcome and predictor variable x when values of other predictor variables are fixed. This is interesting because interpretation of odds ratios is intuitive. An odds is a measure related to probabilities. If an event has some non-null probability to occur in a particular experiment, odds for this event can be viewed has the ratio of the number of events to the number of non-events if the experiment were repeated multiple times. Thus, high odds correspond to high probability for this event, and *vice versa*. Given a probability p of occurrence for this event, an odds is defined as follows: $Odds = \frac{proportion\ of\ success}{proportion\ of\ failure} = \frac{p}{1-p}$.

An odds ratio (OR) is then simply the ratio of two odds. It evaluates association between disease occurrence and predictor variables. As such, this measure is closely related to statistical independence: if two variables (in the example below, SNP genotype and disease status) are statistically independent, their OR reduces to 1. Note that an OR not equal to 1 does not necessarily imply a statistically significant association.

Table 1 | Example of 2 × 3 frequency table to compute an allelic odds ratio.

		SNP genotype		
		AA	**Aa**	**aa**
Disease status	**Affected**	a	b	c
	Unaffected	d	e	f

Based on **Table 1** above, the odds ratio might be calculated using $OR = \frac{(2*a+b)/(2*d+e)}{(2*c+b)/(2*f+e)}$, assuming allele A is the at-risk allele. This OR is also called the allelic odds ratio (Sasieni, 1997).

where protein-protein interaction and gene-gene interaction networks exhibit redundant pathways making them resistant to variations (e.g., to deletion of a network node). A disease state would then be due to accumulation of mutations in the genetic network such that its robustness is outstripped. Therefore, all these network interactions are likely to involve epistatic effects. Canalization theory thus explains why so many variants only provide small contributions to the phenotype (Moore, 2003).

Challenges in Epistasis Detection

Challenges in epistasis detection are threefold. The first one is statistical. Statistical methods traditionally used in univariate SNP-phenotype associations are not adequate to find epistasis.

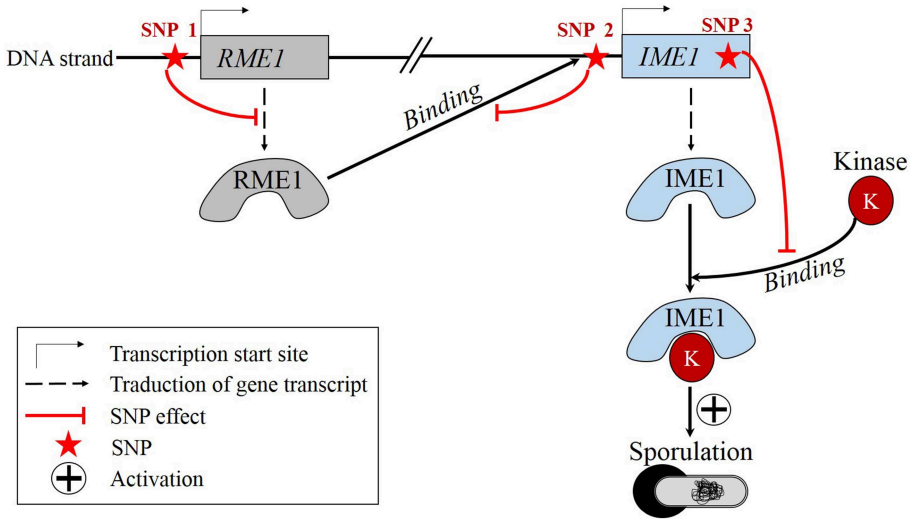


FIGURE 2 | Real example of epistasis: *S. cerevisiae* sporulation is regulated by epistatic effects among three SNPs. State of SNP 1 modulates the production rate of RME1. State of SNP 2 influences the binding specificity of RME1. State of SNP 3 conditions the binding specificity of IME1-kinase.

Finding epistatic interactions is a typical case of the *large p, small n* problem (Johnstone and Titterington, 2009). In practice, the aim is to balance the false-positive rate—produced by the astronomic number of tests performed—and the false-negative rate—a consequence of applying too much stringent significance thresholds. Moreover, SNPs involved in epistatic interactions may have very low minor allele frequencies (MAFs) whereas the number of variants to be tested might be huge. As a result, data is often sparse, leading to the so-called *curse of dimensionality*. The second challenge is computational. Though the overall complexity is linear with the number of individuals in the studied population, it becomes exponential when the interaction order increases. In 2-way interactions, this complexity corresponds to quadratic complexity. The number of combinations to be tested within a dataset containing 1 million SNPs is tremendous: 5×10^{11} pairwise interactions, 1.7×10^{17} 3-way interactions, 4.2×10^{22} 4-way interactions, 8.3×10^{27} 5-way interactions, and so on (Ritchie, 2015). Hence, an exhaustive search of epistatic interactions of order 3 or more would lead to a computational burden too prohibitive. Finally, the third challenge is the interpretation of the analytical results. To interpret statistical results biologically is not straightforward, for statistical interaction does not automatically entails interaction at the biological or mechanistic level (Cordell, 2002).

Exhaustive Search for Epistasis

In this section, we will discuss strategies of detection that exhaustively test all combinations of variants. Exhaustive search has been proposed to circumvent the local optimality problem, a drawback of heuristic techniques. Most exhaustive methods are designed to detect only pairwise interactions and those directed at higher order detection are simply not scalable. Despite their shortcomings, traditional parametric regression methods serve as a foundation in the field, as emphasized in the following subsection. Then, we will present a strategy derived from such regression methods and designed to be faster than traditional methods. Finally, we will discuss two model-free approaches.

Parametric Regression Methods

Traditionally, the most common framework for exploring GWAS data is parametric regression models. A parametric algorithm has a fixed number of parameters that has to be estimated from the data, and relies on strong assumptions about the probability distribution generating the data. This class of algorithms makes accurate predictions when those assumptions are sufficiently close to reality, but performs badly when proved incorrect. Logistic regression (see **Box 1**) has been widely used as a parametric method for exhaustive search of interactions in association analysis. For example, software PLINK (Purcell et al., 2007) has implemented logistic regression models to detect epistasis. But, in high dimensional data, parameter estimation is a costly and non-accurate procedure that introduces large standard errors because sample sizes are too small compared to genome-wide data size. As a consequence, many false positives are generated when dealing with such data. To overcome this problem, *p*-values are usually corrected with Bonferroni multiple-test correction (see **Box 2**). This correction being overly conservative, only interactions with very strong effects will be detected and many other interactions will be missed. Hence, the logistic regression strategy has been widely portrayed as unsuitable for handling genome-wide datasets (Cordell, 2009; Moore and Williams, 2009; Steen, 2012). Highly related to standard regression methods, penalized regression techniques, such as the LASSO (least absolute shrinkage and selection operator) or SCAD (smoothly clipped absolute deviation) gained some popularity to detect SNP-SNP interactions. However, those techniques are restricted to two-way interactions and are still

BOX 2 | Bonferroni correction.

Problem - Hypothesis-based statistical tests (e.g., *t*-test) are subject to false positive inflation when multiple tests are performed. For example, at a traditional 5% threshold set for statistical significance, there is a 5% chance to falsely reject the null hypothesis. Hence, if this test is performed 100 times when the null hypothesis is in fact true, and 5 tests are found to be statistically significant, then all 5 represent false positive associations. In this case, it is said that the risk is high and uncontrolled. This issue is known as the *problem of multiple tests*.

Answer - Bonferroni correction is applied to properly adjust the type I error rate. It consists in dividing the significance threshold by the total number of tests performed. For instance, if a study involves testing for 100 000 hypotheses at a desired global 5%significance level, the corrected significance level for each test is set at $\frac{0.05}{100\,000} = 5 \times 10^{-7}$.

Shortcoming - This method tends to reject non-null hypotheses due to its conservativeness. This conservative feature is also a shortcoming. It becomes inaccurate because it only favors strongly significant associations. As a result, many true positive associations will be missed (i.e., creating false negatives), thereby leading to a loss in statistical power.

prone to inflated false positive rate. Moreover, they are too computationally intensive to exhaustively search through all the pairwise interaction search space. In that case, feature selection techniques are required (further discussed in Section Two-stage Approach: Filters to Obtain Reduced Search Space). The interested reader is referred to Gou et al. (2014) for a recent detailed application of penalized regression-based approach for epistasis detection.

Bitwise Representation of Data and Likelihood Ratio-based Testing

We will introduce the Boolean operation-based testing and screening (BOOST) software program to exemplify this section. Designed to be fast, BOOST runs an exhaustive analysis of all potential pairwise SNP-SNP interactions (Wan et al., 2010). The main feature of BOOST is to build contingency tables and use them to calculate log-likelihood ratios for evaluating interaction effects. For two SNPs, a contingency table is a 3×3 matrix displaying the frequency distribution of all nine possible genotypes (**Figure 1B**). However, computing all potential contingency tables at a genome-wide scale is a time-consuming process. In fact, there are as many contingency tables as there are pairwise interactions to test (see Section Challenges in Epistasis Detection). In order to boost the procedure in terms of time and space efficiency, GWAS data is first transformed in a binary way. In usual data representation, each row symbolizes a SNP and each column symbolizes a subject (**Figure 3A**). In binary representation, each SNP is depicted by three rows, each of them describing the genotype status (i.e., 0, 1, or 2), and two columns depict cases and controls subjects respectively (**Figure 3B**). Each table cell contains a bit string where each bit represents one subject and its genotype: 1 if it corresponds to the genotype status encoded by the current row, 0 otherwise. Even if the binary matrix seems three times larger than the usual one, its space usage is smaller because one bit is an eighth of a byte, and bytes are the usual units (i.e., non binary) used for storing information. That representation also sticks closer to machine-language, which means that building a contingency table from it only involves fast bitwise (i.e., Boolean) operations.

Once contingency tables are constructed, the program is ready to test for pairwise interactions. The way to detect epistasis complies with Fisher's epistasis definition (see Section Biological Epistasis and Statistical Epistasis) since authors look for a difference between the independent effect model (i.e., marginal effects) and the model which includes both marginal and interaction effects. In other words, for each SNP pair, BOOST tests for a departure from the linear additive model. Under the assumption of equivalence between a logistic regression model and its corresponding log-linear model (Agresti, 2002), this departure is expressed in terms of log-likelihoods. However, the traditional log-likelihood of marginal effect model is constructed via computationally costly iterations that are not tractable at a genome-wide scale. Hence, authors use a non-iterative approximation of the log-likelihood ratio called Kirkwood superposition approximation (KSA) (Matsuda, 2000). On the basis of contingency tables, all pairwise interactions are tested with this indulgent KSA. As it is an approximation, too many false positives are deemed significant with respect to a threshold specified by the user. Therefore, after this first quick screening phase, interaction effects of the selected SNP pairs are again evaluated in a second phase. The number of SNP pairs is supposed to be reduced enough during the first phase in such a way that evaluation of interaction effects via a classical log-likelihood ratio on the remaining pairs is now affordable. Finally, significance of evaluated effects is assessed with a χ^2 test. One could say that the use of the χ^2 statistic discredits the method with the following argument: testing interaction effects of a SNP that shows high marginal effect with a χ^2 statistics may lead to evidence of a statistically significant epistatic effect while that perceived signal could solely be due to noise induced by high marginal effect. For instance, the latter issue has been reported in 2013 by Goudey and coworkers in their result section (Goudey et al., 2013). As a consequence, this phenomenon could favor the selection of many false positive interactions that have little to no epistatic effect. However, even if BOOST uses the χ^2 statistics to ultimately assess significance of epistatic interactions, tested SNP pairs already show significant association with a log-likelihood difference between the model which does not consider interactions (reduced model) and the model that does consider them (full model).

This approach is faster than its contemporary Bayesian method BEAM (see Section Bayesian Networks) and shows comparative power of detection. A year later, an even faster version that relies on graphic processing units (GPU) instead of central processing units (CPU) was developed. However, an important shortcoming arises because BOOST heavily relies on contingency table construction: low minor allele frequencies (MAF) generate sparse contingency tables, which hampers the detection power of BOOST. Indeed, in each cell of the contingency table, a minimal number of individuals is required so

A

	Controls			Cases		
	I1	I2	I3	I4	I5	I6
SNP 1	0	**1**	**1**	2	2	0
SNP 2	0	1	2	2	0	2
SNP 3	1	1	0	1	**2**	**2**

B

		Controls			Cases		
		I1	I2	I3	I4	I5	I6
SNP 1	= 0	1	0	0	0	0	1
	= 1	0	**1**	**1**	0	0	0
	= 2	0	0	0	1	1	0
SNP 2	= 0	1	0	0	0	1	0
	= 1	0	1	0	0	0	0
	= 2	0	0	1	1	0	1
SNP 3	= 0	0	0	1	0	0	0
	= 1	1	1	0	1	0	0
	= 2	0	0	0	0	**1**	**1**

FIGURE 3 | Representations of GWAS data. (A) Classical representation: cell (i, j) corresponds to status of SNP i for individual j. **(B)** Binary representation: cell (i, j) corresponds to the true (1) or false (0) assertion that a SNP i has a specific value (0, 1, or 2) for individual j. For ease of comprehension, the link between these two representions is highlighted in gray.

that the χ^2 test is statistically valid. But when contingency tables are sparse, this requirement is not met, thus leading to failure of epistatic interactions detection. Despite the fact that nearly all true positives are detected (i.e., the detection power is high), BOOST is sensitive to type I errors (Yoshida and Koike, 2011). Finally, a notable shortcoming is that the method only analyzes pairwise interactions and no higher order interactions.

ROC Curve Analysis

Goudey et al. introduced the genome-wide interaction search (GWIS) model-free approach in 2013 with the purpose of pairwise epistasis detection (Goudey et al., 2013). While BOOST compares a difference in segregation between two regression models, GWIS tests the difference in segregation power between a SNP pair and the corresponding SNPs taken individually. GWIS is not based on regression analysis, but exploits receiver operating characteristic (ROC) curves to test the discrimination power of SNP pairs. A ROC curve plots the true positive rate (i.e., sensitivity) against the false positive rate (i.e., 1 – specificity) of a classification model. In the context of GWAS, a ROC curve represents the performance of some model designed in classifying individuals according to their affected or unaffected status. For each pair of SNPs, GWIS considers three classification models and builds the respective ROC curves: two for each SNP taken individually, and one for the SNP pair. When the ROC curve corresponding to a SNP pair lies over the other two curves corresponding to individual SNPs, the SNP pair is said to have better prediction power than SNPs taken individually. The next question is to assess if the departure in prediction power between these classification models is significant. To answer this question, Goudey et al. proposed a model-free hypothesis test called *difference in sensitivity and specificity* (DSS). The goal is to quantify the gain in sensitivity and specificity of a ROC curve over another one (Goudey et al., 2013). It seems important to the authors to perform exhaustive search rather than heuristics, in order to avoid being trapped in local optima, then missing significant pairs. GWIS is also designed to be fast (e.g., faster than BOOST) and to scale up to datasets containing millions of SNPs.

The BOOST and GWIS strategies are designed to run exhaustive genome-wide fast scans of epistatic interactions. However, they are restricted to the detection of interacting SNP pairs, which is a substantial limitation. All epistatic models assuming interaction with order greater than two will be missed by these two methods. In the next section, we present a technique that overcomes this problem and exhaustively looks for higher order epistasis.

A Full Combinatorial Approach

Multifactor Dimensionality Reduction (MDR) is now a reference in the epistasis detection field. No parameters are estimated (i.e., nonparametric) and no assumptions are made on the genetic model (i.e., model-free) under this supervised classification approach. This strategy could detect interactions even when independent main effects are inexistent (Ritchie et al., 2001, 2003; Hahn et al., 2003). It is not constrained to identification of pairwise interactions but also searches for higher order interactions (Moore et al., 2006).

First, MDR partitions the dataset for cross-validation. By default, nine tenths of the dataset (training set) is used to build the model and the remaining tenth (testing set) is used to evaluate this model. The model is built following the steps presented in **Figure 4**. For an interaction order specified by the user, the corresponding number of SNPs is drawn (**Figure 4A**). Genotype combination counts are then distributed into a contingency table (**Figure 4B**). For instance, in a two-SNP biallelic interaction model, the nine possible two-locus genotype combinations are allotted into their respective table cells. For a three-SNP interaction model, twenty-seven table cells would

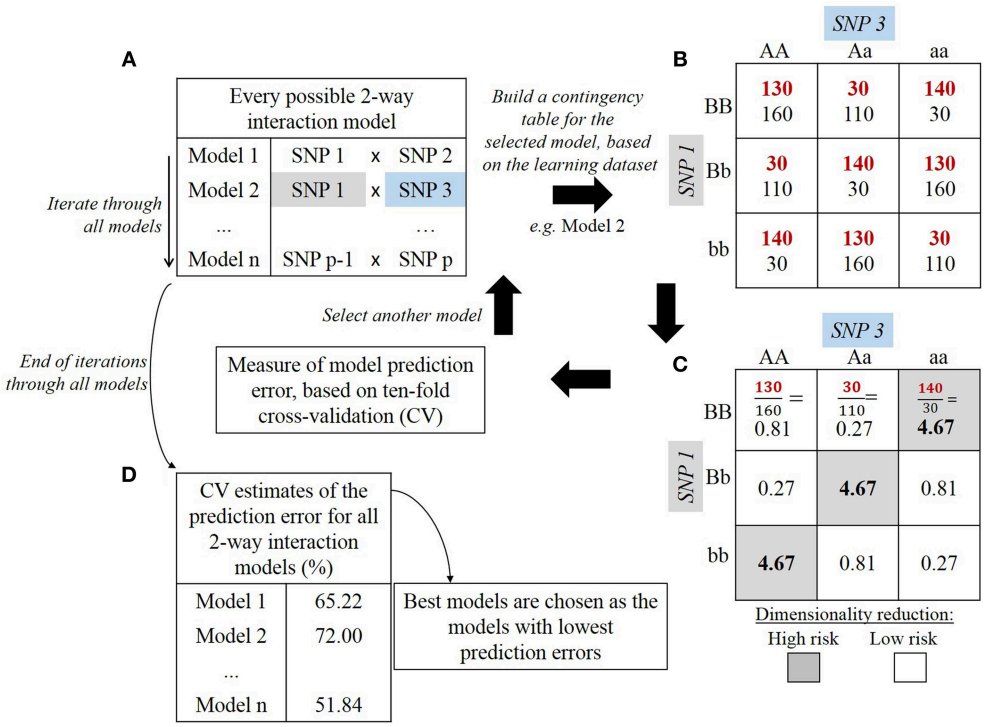


FIGURE 4 | Steps of multifactor dimensionality reduction (MDR) algorithm: example of 2-way interaction model. Description of one iteration of the cross-validation process. In **(A)**, a SNP combination is drawn among all potential SNP combinations. In **(B)**, numbers in red denote counts for cases whereas numbers in black denote counts for controls. In **(C)**, each cell displays the ratio of cases over controls. In **(D)**, the prediction error is estimated over the 10 iterations.

be needed. Then, the count of cases and controls is reported for each genotype combination and each cell is evaluated with the following ratio: $\frac{number\ of\ cases\ sharing\ this\ genotype\ combination}{number\ of\ controls\ sharing\ this\ genotype\ combination}$ (**Figure 4C**). This way, each genotype combination is classified either as high-risk if the above ratio lies beyond a specified threshold (e.g., 1.0), or as low-risk if it lies below that threshold (De et al., 2014). The classification model is then formed by merging cells marked high-risk in one group and all cells marked low-risk in another group. This explains why that method refers to "Dimensionality Reduction": starting with a problem where dimensionality equals the chosen interaction order, only one dimension remains in the end with high-risk and low-risk values. These steps are repeated for every possible combination of SNPs at a given interaction order, and each combination results in one prediction model. A 10-fold cross-validation process allows to assess the quality of such models. In other words, for each of the 10 iterations of the cross-validation, the models are trained to discriminate between low-risk and high-risk groups through the learning step (on nine tenths of the data). The proportion of ill-classified affected and unaffected individuals is evaluated on the testing set (one tenth of the data). Finally, the prediction error of each model is estimated over the 10 iterations (**Figure 4D**). The top best models over the 10-fold cross-validation are retained.

As the main feature of MDR is to reduce the data dimension, it can easily be combined with other classification methods (Moore and Andrews, 2015). This flexibility is also a good point to emphasize because since 2006, many extensions of MDR have been proposed so that it is applicable to quantitative traits (Gui et al., 2013). Besides, other variants of the MDR algorithm have been proposed that rely on parallel implementations to boost MDR computing time performance (Bush et al., 2006), to handle missing data (Namkung et al., 2009), or to implement permutation tests (Greene et al., 2009a). However, MDR remains a brute-force search algorithm that induces a prohibitive computational burden when the number of SNPs to analyze exceeds several hundreds. This lack of scalability is its most critical shortcoming in a genome-wide analysis context.

Most exhaustive strategies cannot afford screens of higher order interaction space search since they are not designed to scale up (Taylor and Ehrenreich, 2015). Even the aforementioned GWIS method is restricted to pairwise interaction detection. Exhaustive methods allowing exploration of higher order interactions, like MDR, cannot handle a genome-wide analysis and are constrained to several hundreds of SNPs. To overcome this shortcoming, a common technique is to preprocess data, reducing the entire SNP set to a smaller subgroup that has a tractable size for exhaustive higher order genetic interaction analysis. However, the type of filter is also important. Choosing a marginal-effect dependent filter would be indeed counterproductive with a method like MDR which is most effective in detecting interactions showing pure epistatic effects.

Two-stage Approach: Filters to Obtain Reduced Search Space

To address the computational burden issue, the overarching goal of some methods is to restrict the analysis to a small subset of candidate markers so that the exhaustive investigation of the remaining combinations is computationally tractable, even for higher order interactions. One approach is to conduct a single SNP-SNP analysis to keep only SNPs with significant marginal effects. SNP combinations are then tested among the remaining marker subset. For example, this strategy has been used in combination with stepwise logistic regression to pre-select a small fraction of SNPs (e.g., pre-determined 10%) based on single-SNP associations significance, before testing for interactions between the selected markers (Marchini et al., 2005). But such filtering leads to an obvious bias where epistatic interactions exclusively induced by combinatorial effects (i.e., with no marginal effect) are not picked up. Nevertheless, there are other ways to reduce the number of SNP combinations down to an informative subgroup. There also exists data mining and data integration techniques dedicated to filter and score downsized genetic variant sets, where null marginal effect is not a rejection condition. We will illustrate each technique in the next two subsections.

Filtering Based on Data Mining Techniques

We will illustrate this category with the ReliefF method. ReliefF approach consists in learning informative features from the dataset without any *a priori* knowledge (Robnik-Šikonja and Kononenko, 2003). The algorithm computes a proximity measure between individuals on the basis of genome-wide genetic similarity. The goal is to evaluate the quality of genetic variants according to how well their values distinguish individuals near to each other.

The algorithm is quite simple (**Figure 5**). For each individual (noted *I*), the procedure determines the nearest individuals (i.e., neighbors) sharing the same phenotype (set noted *S* for *same*), and also the nearest individuals that show up the opposite phenotype (set noted *O* for *opposite*). If *I* and *S* show different values for a marker, then this variant discriminates individuals having the same phenotype, thus decreasing its importance. On the contrary, if *I* and *O* show different values for a marker, this variant discriminates individuals having different phenotypes, thereby its importance is increased. These steps are then repeated over a predefined number of individuals. Moore and coworkers showed in 2007 that ReliefF algorithm is scalable (Moore and White, 2007).

The popularity of ReliefF gave rise to several variations (Kononenko, 1994) that we will quickly present below. RReliefF (Regressional ReliefF) was designed to study quantitative traits like eQTL epistasis (Huang et al., 2013). When applied to a genome-wide dataset, noisy genetic markers may be attributed too much weight, hence inflating their importance estimates. To alleviate this problem, TuRF (Tuned ReliefF) proposed to eliminate from the SNPs set considered for epistasis detection, SNPs with no or very low importance. These SNPs rarely discriminate individuals from their neighbors having a different phenotype (Moore and White, 2007). Importance of remaining SNPs is then re-estimated, without considering these noisy SNPs. Results are encouraging since TuRF power of detection is identical to or better than ReliefF. ECRF (Evaporate Cooling ReliefF) also attempts to solve the noisy variable problem (McKinney et al, 2007). It significantly outperforms ReliefF for detecting epistasis. Its algorithm combines information theory and ReliefF. In ReliefF and its above extensions, the user-defined number of nearest individuals to consider (i.e., *S* and *O*) is usually fixed at 10. Using such a predefined number may be considered as a selection bias since the information coded in the data is not fully exploited. To tackle this issue, SURF (Spatially Uniform ReliefF) proposes to take into account all neighbors within a given distance rather than a fixed number of neighbors (Greene et al., 2009b). SURF generally takes into consideration much more neighbors than ReliefF, labeling 25–50% of all individuals as neighbors. So when applied to a GWAS dataset, SURF has higher power of detection than ReliefF, albeit this may become a cumbersome procedure. A latest variation, SURF∗ (Greene et al., 2010), also considers information of farthest individuals to build importance scores. In terms of detection power of epistatic interactions, the performance of TuRF and ReliefF has been compared in Moore and White (2007). ECRF has also been compared to ReliefF in McKinney et al (2007). Finally, SURF has been compared to both ReliefF and TuRF in Greene et al. (2009b). However, ECRF and SURF have not been compared to each other, as well as ECRF and TuRF. ECRF and TuRF show improved performance over ReliefF, whereas SURF and SURF* show improved performance over both ReliefF and TuRF.

Filtering Based on Data-integration Techniques

Another research area advocates the use of knowledge from external databases, in order to select SNP groups that are relevant to the phenotype of interest (Grady et al., 2011). Even if this approach is hindered by a lack of epistasis understanding in complex organisms, it avoids the black box effect of data mining techniques that may hamper the interpretation of underlying biological mechanisms.

One way to do that is to query information in online public protein-protein interaction databases like IntAct (Kerrien et al., 2012), BioGRID (Chatr-Aryamontri et al., 2015), STRING (Franceschini et al., 2013) or ChEMBL (Willighagen et al., 2013). It is then possible to narrow all SNPs down to a reduced list of markers located in genes that encode for proteins involved in relevant interactions. When markers are mapped to an interacting gene pair, tests are exhaustively conducted on interactions between each SNP of the first gene against each SNP of the second gene. Unfortunately, one would probably fail at discovering new biological models by selecting SNPs in such a direct way. A more promising strategy is to come up with a score for each SNP (Ritchie, 2015), based on assessed relative importance of the proteins encoded by the genomic region encompassing the SNP. Novel findings are within reach by running a prioritization scheme rather than a strict removal (Pattin and Moore, 2008).

Resorting to pathways is also interesting. For instance, this approach has already been applied with information drawn

ReliefF algorithm

```
Input: GWAS dataset
       n: number of closest individuals to consider (with respect to the genotypic data)
       t: number of iteration

Output: Importances of all SNPs of GWAS dataset

For i=1 to t:
       Randomly select an individual I
       Find the n nearest neighbors S
       Find the n nearest neighbors O
       For each SNP in the GWAS data:
              Decrease SNP importance each time the SNP genotype differs between
                     I and a neighbor in S
              Increase SNP importance each time the SNP genotype differs between
                     I and a neighbor in O
       End for
End for
```

FIGURE 5 | ReliefF algorithm.

from pathways involved in lipid synthesis (Ma et al., 2015), by including evidence from public databases like KEGG Pathway (Kanehisa et al., 2012), Reactome (Croft et al., 2014) or BioCarta (Nishimura, 2001). For a pathway of interest, one first looks at the involved genes, and then maps SNPs to these genes. The technique is similar to the above protein interaction-guided analysis. But there is a bias as certain pathways are more deeply studied than others: genes (and SNPs therein) involved in a very well-known pathway may be given more weight than those involved in a less studied one. Instead of relying on guidance restricted to pathways or to protein-protein interactions, the comprehensive knowledge approach (Pendergrass et al., 2013a) is more global as it exploits pathways, protein interactions, gene expression, gene ontology, etc. As appealing as this approach might be, it is not currently possible to accurately evaluate results found by this strategy because implementing pathway simulations is not a trivial task. This would require a tool designed to simulate pathways and protein-protein interaction networks, and then simulate GWAS data where several SNPs are involved in these networks. Such a tool does not exist yet. Therefore, for this kind of filter based on comprehensive knowledge, we cannot properly and objectively assess its scientific relevance.

One software program worth mentioning is Biofilter. It gathers information from 13 databases (Pendergrass et al., 2013a), which contain experimental evidence of interaction, pathway or ontological similarity relationships. On the basis of biological plausibility, Biofilter models interactions that will be tested irrespective of the marginal effects. So it creates polygenic models, thanks to gene-disease and gene-gene connection knowledge (Pendergrass et al., 2013b). The statistical and computational challenges are also addressed since not all combinations of interactions are examined. Statistical relevance is based on the statement that the more two genes are involved in a relationship, the more likely they are to share an important biological link (Bush et al., 2009).

Although data-integration techniques yield meaningful and biologically relevant results, exploiting external information sources like pathways or protein-protein interaction networks is controversial. Online databases are incomplete and so is our understanding of biological pathways. Thus, making use of them to build filters would in most cases results in a flawed analysis. Moore and Hill recently recommended (Moore and Hill, 2015) to combine both the *biased* approach (from a biologist point of view) based on expert knowledge, and computational approaches solely driven by GWAS data (neither immune to bias from a statistician point of view). Similarly to computational exhaustive methods, this combined approach is taking advantage of artificial intelligence methods, which we discuss in the next section.

Non-exhaustive Searches Enhanced by Artificial Intelligence

Machine learning and combinatorial optimization represent alternatives to parametric statistical methods for detecting combinations of variants that are associated with a phenotype. Machine learning methods build non-parametric models to compile information further used for epistatic detection. Combinatorial optimization techniques consider a search space of solutions (i.e., combinations of potentially interacting SNPs) and browse through this space to find the more relevant combinations. Heuristics are commonly used in these algorithms, especially when dealing with genome-wide datasets in search of higher order genetic interactions. Identification of classification variables and interactions between them which allows outcome prediction is a well-known hurdle addressed by the machine learning and data mining fields of artificial intelligence (Cordell, 2009). In such non-parametric models, precautions must be taken to avoid overfitting (see **Box 3**). It has to be noted that if the model complexity of the underlying genetic mechanisms is too high compared to the sample size, using non-parametric methods

BOX 3 | Overfitting.

The aim of machine learning is to explain a system by learning a model with a training dataset. But dataset's particularities result in an overly tuned model adjusted for very specific features (Leinweber, 2007). In other words, overfitting happens when the training stage gives too much importance to the noise within data. Overfitting is detected when a simpler and more accurate model exists. However, identifying what to ignore in the overfitting model is a non-trivial task. Overfitting typically arises when model complexity is too high compared to the size of the training data. In practice, cross-validation possibly combined with pruning is used to avoid overfitting.

may not be affordable. In this case, parametric methods are the only practical alternative, assuming that the model assumptions are not severely violated.

A majority of these heuristics test for associations of variants allowing interactions, rather than testing for interactions themselves. The distinction lies in the following: besides SNPs involved in epistatic interactions, a model representing associations allowing for interactions also includes SNPs which have marginal effects. Therefore, although it is not a straight proof of epistasis, it is nonetheless an examination of polygenic models. Thus, if such procedures heavily rely on marginal effects for association findings, they will detect multiple SNPs with independent effect. But if they do not rely on marginal effects, they will also consider epistatic interactions.

With regard to machine learning techniques, we will first take a look at random forests and their variants, then move on to Bayesian network-based strategies. As for combinatorial optimization strategies, ant colony optimization and computational evolution system approaches will be presented.

Random Forests and their Variants

A tree-based algorithm generates a tree where each tree-node represents a predictor variable and a path designates a sequence of predictor variables from the root to the leaves of the tree. When the tree is constructed from GWAS data, each node represents a SNP. A basic tree-growing algorithm is deterministic in that each step looks for the predictor variable that optimally segregates the population. So a grown tree is a classifier which represents a SNP set allowing prediction of the phenotype of interest. This approach can handle SNPs that are associated in a non-linear way, dealing with interactions encoded in a hierarchical fashion between layers of the tree. A notable shortcoming of tree-based methods is that they are quite dependent of marginal effects. At the beginning of the tree learning step, the algorithm looks for a single SNP that well discriminates cases from controls. In practice, this is equivalent to looking for SNPs with high marginal effects.

Random forests were designed to avoid bias generated by growing a single tree. The random forest strategy creates multiple—generally thousands—classification or regression trees (e.g., CART) in order to apply an ensemble procedure. An ensemble procedure aggregates the predictions of all trees to produce a powerful and robust prediction tool (Breiman, 2001). The SNP set output is defined as the most important variable set of the random forest (to be further explained in this section). Although growing a random forest is a relatively computationally intensive procedure, it has been evaluated as a good strategy for detecting the most predictive SNPs in large-scale association studies (Bureau et al., 2005) and was applied to GWAS several times in the last 5 years with epiForest (Jiang et al., 2009), random Jungle (Schwarz et al., 2010) and SNPInterforest (Yoshida and Koike, 2011).

A classification tree is grown using the following steps (Jiang et al., 2009). First, a bootstrap sampling is performed from the GWAS dataset comprised of N individuals and M SNPs. It consists in randomly selecting, with replacement, N individuals from the N original individuals. Individuals not drawn are called out-of-the-bag (OOB) individuals. So a new dataset and an OOB set are created for each grown tree. Then a random feature selection is applied to construct each node of the tree. To do so, instead of considering all variables from the initial GWAS dataset, a random subset of variables is picked out without replacement. A recursive data splitting procedure is next executed, such that a parent node results in two child nodes given a rule that leads to a better discrimination of the current set of individuals (from the parent node) with regards to the disease status. This discrimination score is measured as a goodness of split or a decrease in impurity Δi. The tree is then grown up to its largest extent. These previous steps are repeated until a forest is built (**Figure 6**).

For each node, a so-called variable importance is assessed to evaluate its contribution to the trait either individually or via multi-way interactions with other predictor markers. In other words, variable importance represents weight approximating the causal effect of a predictor variable. There are several ways to measure variable importance (Schwarz et al., 2010). One is the *Gini importance*, a second one is the *permutation importance*, and a third one is the *conditional variable importance*, based on permutation importance. The conditional variable importance seems to be more suitable when applied to genetic data while the other two are biased in presence of linkage disequilibrium (correlation between SNPs) (Strobl et al., 2008). Compared to the original random forest construction, algorithms readjusted for epistasis detection include multiple SNPs at each tree-node during tree building (Botta et al., 2014). It is intended to detect SNP combinations even when marginal effects are very weak or inexistent (Yoshida and Koike, 2011). The readjusted method is less sensitive to SNPs presenting little marginal effects than an exhaustive approach like MDR. However, even if random forests reveal associations potentially pointing at interactions, they cannot make a distinction between a scenario of interacting SNPs and a scenario of several independent SNPs additively contributing to the phenotype. As a result, random forests are lacking clear interpretation.

More recently, another tree assembling software program was developed: GWGGI (Wei and Lu, 2014). It differs from the previous methods in two points. First, it uses a tree-growing algorithm which is more computationally efficient (Lu et al., 2012): the standard variable selection procedure is replaced with

A **Random forest algorithm**

```
Input: GWAS dataset with M SNPs to analyze
       t: number of trees to build

Output: An ensemble of trees

For i=1 to t:

      - Select a bootstrap sample from the GWAS dataset

      - Grow a tree from the bootstrapped data,
        by repeating the following steps for
        each terminal node of the tree:

            1. Select a random subset of variables
               from the bootstrap sample

            2. Find the variable that best splits
               the data into homogeneous groups

            3. Split the terminal node into two child nodes

End for
```

B **Toy example of the three steps to execute to grow a tree**

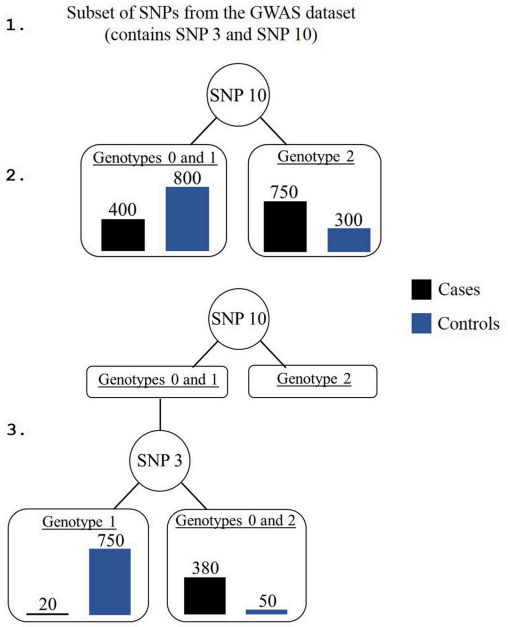


FIGURE 6 | Random forest algorithm. (A) Algorithm of a random forest procedure. **(B)** An example of the three steps needed to grow one tree.

a forward algorithm. The principle of a forward algorithm is to take into account previously selected variables. The novel variable identified is the one, when added to the previous set of variables, allowing for the most accurate prediction. Secondly, the GWGGI algorithm relies on likelihood ratios and the Mann-Whitney statistics to assess the predictors' importance in order to facilitate the statistical significance assessment of selected association models. Since each tree can be considered as a multi-locus genotype model, each individual is confronted to each grown tree and a likelihood ratio is generated: $LR_i^t = \frac{P(G_i^t|D)}{P(G_i^t|\overline{D})}$ where G_i^t is the genotype of individual i mapped t, and D (*resp.*$\overline{D}$) is the control status (*resp.* case status). Then for each individual, all likelihood ratios are assembled into a unique one by averaging the total number of trees. Finally, a U-statistic is constructed with comparisons between assembled likelihood ratios of cases vs. controls in order to evaluate the joint association of the selected SNPs with the phenotype (Wei et al., 2013). The U-statistic is calculated in the following way: $U = \frac{\sum_{i=1}^{N\_cases}\sum_{j=1}^{N\_controls}\psi(LR_i, LR_j)}{N\_cases * N\_controls}$. The ψ function is a kernel function defined as:

$$\psi\left(LR_i, LR_j\right) = \begin{cases} 1 & if\ LR_i > LR_j \\ 0.5 & if\ LR_i = LR_j \\ 0 & if\ LR_i < LR_j \end{cases}$$

The null hypothesis states that there is no association between the selected SNPs and the phenotype.

Bayesian Networks

Bayesian networks provide a compact representation of dependencies between variables. A Bayesian network consists of two components: a graphical one and a probabilistic one. In the former—directed acyclic graph (DAG)—variables are represented by nodes and dependencies between them are represented by directed edges. The probabilistic component of a Bayesian network associates a probability distribution with each node of the DAG, thus accounting for uncertainty. A Bayesian network encodes the Markov property: each variable is independent of its non-descendants, given its parents in the DAG. The governing theorem of a Bayesian network is the following. Let X, Y, and Z be variables of the Bayesian network. If $P(X|Y, Z) = P(X|Y)$, then X is conditionally independent of Z, given Y (noted $X \perp Z|Y$). When applied to genetic data, variables are typically SNPs and phenotypic values. Bayesian networks offer an appealing and intuitive way to capture relationships existing between genetic markers and disease status. The structure learning of a Bayesian network amounts to a model selection problem. Because this learning is an NP-hard problem (Chickering et al., 2004), specific techniques have to be used to reduce the computational burden.

A famous Bayesian network-based software program called BEAM (Bayesian Epistasis Association Mapping) (Zhang and Liu, 2007) is often used as a Bayesian-based reference for performance comparisons. BEAM relies on a Markov Chain Monte Carlo (MCMC) algorithm to test iteratively each marker, conditional on the current status of other markers. For each marker, the algorithm outputs its posterior probability of association with disease. Markers are then distributed into three groups: group 0 for markers unlinked with the phenotype, group 1 for SNPs that contribute independently to the phenotype (additive model) and group 2 for SNPs that influence the disease risk given particular allele combinations (epistasis model). After

that partitioning phase, a B-statistic is used to further filter detected SNP groups. When the BEAM method was originally published, the B-statistic was a new alternative to the usual χ^2 test of association between a phenotype and a set of SNPs. A detailed explanation of its computation would require a much deeper presentation of BEAM, which is not the aim of this section. The interested reader is referred to Zhang and Liu (2007) for a comprehensive explanation of how to build a B-statistic. Although the B-statistic enables to get rid of expensive permutation tests, MCMC iterations make this method inadequate when handling datasets containing more than 500,000 genetic markers, which is now commonplace in GWAS studies.

More recently, Han et al. (2012) also worked with Bayesian networks to capture SNP-disease associations with EpiBN. As these authors consider that SNPs are causal with respect to the phenotype, the Bayesian network built here is composed of two layers: one layer with the phenotype as a unique node, connected to parent nodes of the phenotype in the second layer which represents the SNPs associated with the phenotype. Edges between nodes representing SNPs can exist, thus allowing detection of interactions between genetic variants in the model. Instead of a MCMC-based algorithm, they use a Branch-and-Bound iterative procedure to learn the structure of the Bayesian network. At each iteration, the algorithm adds, deletes or reverses an edge. Then a score function is called to find the best network structure evolution since the previous iteration. The network is iteratively constructed and at each iteration, the current network structure goodness is assessed with a score function. The goal is to maximize this score. The score function is made of two terms that indicate how well the current structure fits the data—on the basis of a maximum likelihood ratio—and how complex the Bayesian network is. In Han et al. (2012), it has been shown through multiple simulations that the EpiBN software program seems to outperform BEAM in interactions detection power.

A different but not less appealing Bayesian strategy is the Markov blanket-based method. It allows discovery of SNPs in the local pathway of the phenotype, also referred to as "local causal SNPs" (Alekseyenko et al., 2011). In the context of GWAS, this strategy is used to avoid the time-consuming training processes like tree-growing of random forests or structure learning of a full Bayesian network. The principle is to find a minimal set of variables that completely shield the disease status from all other variables, thus resulting in a local Bayesian network fraction that borders the phenotype node in the graph: this set is defined as the Markov blanket. In other words, each SNP will be statistically independent of the case-control status when conditioned on the SNPs forming the Markov Blanket. A Markov blanket-based strategy can be applied for causal findings because the Markov Blanket contains direct causal variables (i.e., parent nodes), direct effect variables (i.e., child nodes), and direct causal variables of direct effect variables (i.e., spouses) (**Figure 7A**).

With the goal of finding a minimal SNP set, this strategy is expected to minimize the number of false positives. Besides its classification accuracy, this strategy has been put forward for its compactness (Aliferis et al., 2010a). Moreover, the Markov blanket-based strategy has proved to properly address the combinatorial hurdle raised by epistasis analysis at the GWAS scale (Aliferis et al., 2010b). The Markov blanket construction algorithm will generally go through two stages (**Figure 7B**). The first one, called "forward phase," adds new relevant variables to the candidate Markov blanket (noted *canMB*). In practice, this stage consists in finding the SNP X which is the most associated with the phenotype, given *canMB* (e.g., tested with a G^2 test, which is a subclass of likelihood-ratio tests and is similar to a χ^2 test, McDonald, 2014), and including X in *canMB* if X is dependent of the phenotype, given *canMB* (e.g., if the G^2 statistics is lower than some user-specified threshold): $\neg(X \textit{ Phenotype} \mid \textit{canMB}) \Rightarrow \textit{add } X \textit{ in canMB}$. This operation is repeated until *canMB* no longer changes from one iteration to the other. The second phase, called "backward phase," aims at removing false positives that were included in the previous step. To achieve it, each SNP of the candidate Markov blanket is checked. A SNP Y is detected as a false positive if it is independent of the phenotype given a SNP subset of *canMB*. Three implementations of this approach were recently developed: DASSO-MB (Han et al., 2010), TIE* (Alekseyenko et al., 2011; Statnikov et al., 2013) and IMBED (Yanlan and Jiawei, 2012), and all proved to be more sample-efficient than BEAM, i.e., less samples are needed to reach the same power of detection as BEAM. In DASSO-MB (Han et al., 2010, Han and coworkers postulate that, in epistatic interaction studies, only causal SNPs are sought, and consequently only parent nodes of the phenotype have to be detected. Hence, DASSO-MB represents a more specific application of the Markov Blanket approach. Considering a set of 19 SNPs already known to be associated with rheumatoid arthritis, an application of TIE* (Target Information Equivalency) showed that a Markov blanket-based approach could make the whole SNP set independent of the phenotype when conditioned on three other SNPs identified in the Markov blanket (Alekseyenko et al., 2011). In other words, the reported SNP set does not provide any predictive information about the disease status beyond that brought by the three SNPs identified with the Markov blanket.

The bias of this approach is that the first SNP added to the candidate Markov blanket is picked on the basis of a univariate test. So the detection of marker combinations when marginal effects are slight or nonexistent is still a major obstacle (Han and Chen, 2011). Markov blanket-based strategies also heavily rely on the *faithfulness* assumption, defined with respect to the sample, as follows: every conditional independence in the Bayesian network also exists in the probability distribution of the variables. In practice, this hypothesis is rarely met in GWAS.

Ant Colony Optimization

Ants communicate with each other through pheromone levels to find the optimal path leading to food. If an ant finds a shorter path, it will produce and increase the pheromone concentration along this path. Other ants will more likely follow that path showing increased pheromone concentrations, thereby creating a positive feedback to find the best path to food. In 2010, AntEpiSeeker algorithm (Wang et al., 2010b) was derived from the generic ant colony optimization (Dorigo and Gambardella, 1997) (ACO) algorithm. AntEpiSeeker performs the search of

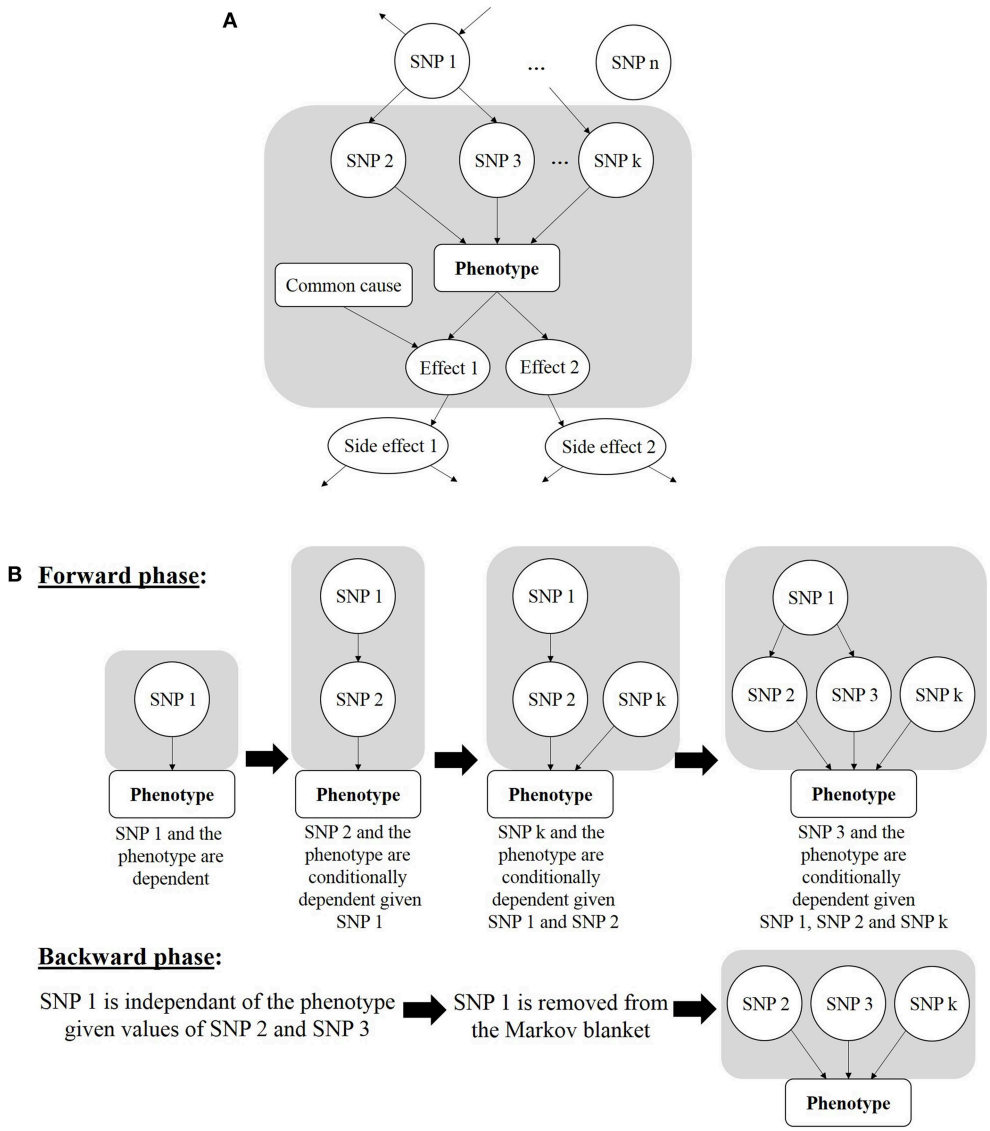


FIGURE 7 | (A) Markov blanket of a phenotype, in gray area. It is made of the parents (SNP 2, SNP 3, and SNP k), of the children (Effect 1 and Effect 2) and of the spouses (Common cause of Effect 1 with respect to the Phenotype). **(B)** Two stages of Markov blanket learning. For ease of reading, the Markov blanket is reduced to parents from **(A)**.

multiple groups of SNPs associated with the disease in parallel. The algorithm is an iterative procedure where artificial ants cooperate at each iteration to update knowledge about the propensity of SNPs to be related to the disease (**Figure 8**). From a computational point of view, ants represent SNP sets that have potential epistatic effects, and a pheromone concentration is a weight evaluated by epistatic interaction significance of the selected set of SNPs. Communication between ants is mimicked by a probability distribution function (PDF) shared by all ants. The PDF is a function describing the probability of selecting a specific SNP at a specific iteration. This probability depends on the pheromone concentration for this SNP at this iteration, and on another factor which allows to weight SNPs according to expert knowledge drawn from additional biological data. At each iteration, multiple SNPs are picked up, depending on the PDF, to build each ant. Then a χ^2 test is used as a score function to measure the association between an ant and the phenotype. Results are used to update the PDF for the next iteration. Once highly suspected sets of SNPs are assembled, AntEpiSeeker conducts a second analysis stage: an exhaustive search of epistasis within each built ant is performed, as well as within the set of SNPs that have the highest pheromone levels. The ant colony strategy was also exploited more recently in MACOED (Jing and Shen, 2015).

The positive feedback effect represents an interesting feature of the algorithm. Unfortunately, many parameters require fine tuning, like the number of iterations, the order of interactions, the number of SNPs in each ant, or the evaporation rate of

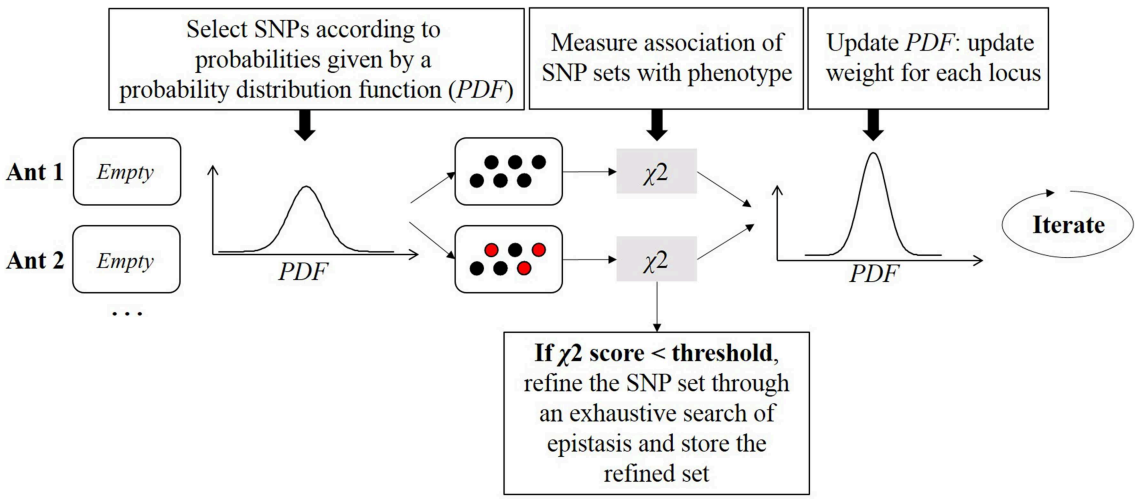


FIGURE 8 | Ant colony optimization procedure. For each ant, multiple SNPs are drawn. The probability distribution function (PDF) gives the probability of each SNP to be drawn. Once an ant is filled with a SNP set, joint association of this SNP set with the phenotype is evaluated with a χ^2 test. For each ant, the PDF is updated according to *p*-values resulting from χ^2 tests, such that SNPs efficiently classifying individuals will have a higher probability of being drawn in the next iteration.

pheromones which is an ingredient of the update function of the PDF. Those parameters must be estimated a priori, which is considered as a limitation of this algorithm.

Computational Evolution System

The algorithm behind the Computational Evolution System (CES) is an original strategy based on natural selection and Darwinian evolution. The goal is to grow a computer program from several basic building blocks, similar to a DNA strand emerging from a composition of the four basic nucleotides. This program tries to reproduce the natural evolution process underlying complex real biological systems. The first question is what the building blocks are, whenever one wants to build such a computer program. The answer is non-trivial and is decisive in epistasis analysis when trying to avoid dependence to marginal effects. In a recent application of CES (Moore and Hill, 2015), the building blocks were defined as basic functions involving SNPs. A basic function is an operator (add, delete, and copy) aggregating SNPs in combinations, and the resulting composition of building blocks is called a solution. In other words, a solution can be perceived as a set composed of various elements, where each element is a function dealing with genetic polymorphisms. A solution is thus a classifier designed to predict the case-control status of an individual given its genotype.

A CES is governed by a pyramidal architecture where each level is probabilistically controlled by its upper layer. The lowest level is a two-dimensional grid of solutions where each solution is a list of building blocks. The second level is a grid of solution operators influencing the lower layer. Each cell consists of a combination of add, delete, and copy operators having a given probability of being executed. Attributes can be added, deleted or copied either randomly or using expert knowledge. A third level of computation is used to introduce changes in execution probabilities of the latter operators. A last level controls the variation rate of the third layer. Uncertainty is injected in this architecture in order to mimic a realistic natural evolution system. As a result, there is high flexibility in model creation based on CES.

The stage during which all solutions are modified is called a generation. From one generation to the next, accuracy of each solution is modified as follows: an operator is drawn according to the execution probability distribution; this operator is then applied to each solution. It has to be noted that the initialization of the CES grid of solutions is either random or guided with expert knowledge. This last option has been highly recommended (Greene et al., 2009a; Payne et al., 2010). The accuracy of each solution is assessed in the following way. Each solution is applied to case and control individuals to obtain two distinct score distributions: one for cases and one for controls. A threshold is determined as the arithmetic mean between the medians of the two distributions. Then individuals are predicted to be cases or controls given this threshold. The solution accuracy is computed afterwards as an error ratio between predicted and actual status. Once one knows how to compare solutions, one can select the optimal solution which maximizes the prediction accuracy. The solution is selected among all generations (e.g., 1000 generations) in the following way. Each solution occupies a lattice position in the two-dimensional grid and competes with its neighborhood composed of eight adjacent solutions. Within this neighborhood, the solution with the highest accuracy is selected to replace the central position of that neighborhood.

This approach is interesting in that it allows modeling of complex interactions with few hypotheses. It also has the capability to use expert knowledge, and is well suited for parallelization. However, the computational complexity of the CES strategy precludes a direct analysis of GWAS data with hundreds of thousands SNPs. Such datasets will require a preprocessing step with filtering methods introduced in Section Two-stage Approach: Filters to Obtain Reduced Search Space.

TABLE 2 | Summary table of strategies reviewed to detect epistasis along with representative software programs and datasets applications.

Strategy		Software program	Exhaustivity	Pairwise-restricted	Dataset	# SNIPs	# Individuals	Runtime		References
								Sequential	Parallel	
Bitwise operations and Likelihood ratio tests		BOOST	Yes	Yes	WTCCC–multiple diseases	459,019	5000	NA	23 h (4 CPUs)	Wan et al., 2010
ROC curve analysis		GWIS	Yes	Yes	WTCCC–multiple diseases	459,019	5000	60 h	10.9 h (4 CPUs)	Goudey et al., 2013
Combinatorial		MDR	Yes	No	Simulated	50	500	36 min	NA	Moore et al., 2006
Random forest		Random jungle	No	No	Crohn's Disease	275,153	1003	12.7 h	0.53 h (40 CPUs)	Strobl et al., 2008
		SnpInterforest	No	No	WTCCC - RA	10,000	3500	98 h	NA	Yoshida and Koike, 2011
		GWGGI–TAMW	No	No	WTCCC - CAD	459,019	4864	10 h	NA	Wei and Lu, 2014
		GWGGI–LRMW	No	No	WTCCC - CAD	459,019	4864	3.5 h	NA	Wei and Lu, 2014
Bayesian		BEAM	No	No	AMD	47,727	3500	8 days	NA	Zhang and Liu, 2007
		epiBN	No	No	AMD	96,933	146	NA	NA	Han et al., 2012
	Markov blanket	Dasso-MB	No	No	AMD	91 495	14G	NA	NA	Han et al., 2010
		FEPI-MB	No	No	Simulated	500	4000	0.5 s	NA	Han et al., 2011
		IMBED	No	No	AMD	96,933	146	NA	NA	Yanlan and Jiawei, 2012
		TIE*	No	No	NARAC	490,073	2 044	NA	NA	Statnikov et al., 2013
Ant colony optimization		AntEpiSeeker	No	No	WTCCC – RA	332,831	3503	NA	5 days (2 CPUs)	Wang et al., 2010b
Computational evolution system		CES	No	No	Prostate cancer	219	2286	NA	NA	Moore and Hill, 2015

Runtimes were not always available (NA) and are indicated for simulated datasets when no real data application is available. The notation "WTCCC—multiple diseases" stands for "WTCCC—Bipolar Disorder (BD), Coronary Artery Disease (CAD), Crohn's Disease (CD), Hypertension (HT), Rheumatoid Arthritis (RA), Type 1 Diabetes (T1D), and Type 2 Diabetes (T2D).

Discussion

While an exhaustive epistasis analysis has become a quite straightforward task for SNP pairs, higher order interactions search in an exhaustive way is not conceivable at the moment. In this paper, we reviewed main current strategies for epistasis detection: exhaustive ones based on brute-force approach, filtering ones aiming at reducing genome-wide SNP set size, and different machine learning and combinatorial optimization procedures to find SNP associations yielding the best classification power. **Table 2** summarizes categories of methods described in this paper and gives representative software programs illustrating each category. In particular, this table highlights characteristics of the largest GWAS dataset analyzed using each software program. Runtimes are indicated, when available, for sequential and parallel versions of each program, for information about scalability.

Despite efforts for developing novel methods dedicated to epistasis detection, genetic variance of complex traits is weakly explained by detected epistatic interactions. This may be due to low detection power of pure and strict epistatic interactions for many of these methods. Much remains to be done to improve power of detection using model-free searches. For instance, the TURF method (see Section Filtering Based on Data Mining Techniques) which excludes SNPs with low predictive power, prior to performing epistasis detection, could be extended to other strategies like random forests, thereby improving detection of epistasis.

Precision of association measure estimates between epistatic interactions and phenotypes can be enhanced by increasing the $\frac{number\ of\ samples}{number\ of\ SNPs}$ ratio. First, increasing the sample size is a way to improve power of epistasis detection. Federating data from laboratories in the context of meta-analysis is a widespread approach, though subject to biases due to heterogeneity of laboratory practices. Second, reducing the number of SNPs to analyze might improve the statistical power under a given hypothesis. For instance, such a reduction of the search space size is possible thanks to systematic methods, like using significant pairwise interactions as a prior basis for the search of higher order interactions. Regarding data integration approaches, biological expert knowledge based-filters are often proposed to guide epistasis analysis. Being a biased approach, it is recommended to run at the same time a procedure without any *a priori* knowledge (Ritchie, 2015). Although development of epistasis detection methods is growing, many methods are hampered in presence of genetic heterogeneity or incomplete penetrance. Random forest-based techniques have been described to efficiently deal with genetic heterogeneity because data is split in different subsets in early stages of the algorithm (Koo et al., 2013). Besides, some of the existing software programs, like BEAM, will soon become unsuitable to GWAS datasets which will keep growing in size so that several millions of SNPs will be the rule rather than the exception. On the other hand, such a huge number of SNP might increase power of existing strategies tailored to handle massive datasets.

An interesting fact rarely discussed in literature describing the strategies reviewed in this survey is the confusing boundary between epistasis and linkage disequilibrium. Because linkage disequilibrium is by definition a phenomenon involving dependence between genetic variants, its frontier with epistatic interactions may be blurred since the aforementioned software programs are designed to detect SNPs that jointly affect the phenotype. This issue is particularly acute for case-only approaches. For standard case/control studies, if estimation of linkage disequilibrium within controls provides the same result as within cases, then the observed linkage disequilibrium does not originate from epistatic interactions.

Development of simulation models dealing with epistasis is also an active research area (Moore et al., 2015). Even if some authors already use various simulation models to estimate efficiency of their algorithms (Beam et al., 2014), these simulation tools lack the complexity of genetic mechanisms observed in real data. For instance, simulation models used in most software programs introduced in the previous sections only generate pairwise epistatic interactions. As a consequence, strategies dealing with higher order interaction detection are not confronted to simulation scenarios involving those types of interactions. Hopefully, such a gap will certainly be filled in the future.

With regard to evaluating association strength several authors rely on *p*-values to sort the best candidate SNPs. However, *p*-values alone do not allow any straightforward statement about the association strength. A *p*-value only estimates the probability of having observed the value of the test statistic under the null hypothesis (i.e., there is no association between the tested SNP and the phenotype) (du Prel et al., 2009). Odds ratio combined with confidence intervals are also widely used measures in GWAS reports.

The need for scalable and powerful strategy to detect SNP-SNP interactions is clearly unmet today. This is especially true for detection of higher order interactions. Massive testing of SNPs combinations should no longer be a tedious task, but rather a routine operation in a GWAS analysis workflow.

Conclusion

Currently, no strategy to detect epistasis stands out: all must strike balance between time efficiency and detection power. However, different techniques are available to reduce running times. Some authors improved time efficiency through parallelization of their strategies, e.g., random forests, ant colony optimization and approaches based on computational evolution. Other authors implemented versions of their software programs which use graphic processing units (GPU) instead of traditional central processing units (CPU).

Acknowledgments

CN is supported by the Regional Bioinformatics Research project GRIOTE granted by the Pays de la Loire region on the one hand, and the European Genomic Institute for Diabetes (EGID) Labex (Lille) on the other hand. GR's work is supported by a Chair in Biostatistics jointly sponsored by the Centre National de la Recherche Scientifique and Lille 2 University. We also thank two anonymous reviewers for very helpful comments and valuable improvement of the manuscript.

References

Agresti, A. (2002). *Categorical Data Analysis, 2nd Edn*. Hoboken, NJ: John Wiley & Sons, Inc.

Alekseyenko, A. V., Lytkin, N. I., Ai, J., Ding, B., Padyukov, L., Aliferis, C. F., et al. (2011). Causal graph-based analysis of genome-wide association data in rheumatoid arthritis. *Biol. Direct*. 6:25. doi: 10.1186/1745-6150-6-25

Aliferis, C. F., Statnikov, A., Tsamardinos, I., Mani, S., and Koutsoukos, X. D. (2010a). Local causal and markov blanket induction for causal discovery and feature selection for classification part I: algorithms and empirical evaluation. *J. Mach. Learn. Res.* 11, 171–234.

Aliferis, C. F., Statnikov, A., Tsamardinos, I., Mani, S., and Koutsoukos, X. D. (2010b). Local Causal and markov blanket induction for causal discovery and feature selection for classification part II: analysis and extensions. *J. Mach. Learn. Res.* 11, 235-284.

Bateson, W. (1909). *Mendel's Principles of Heredity*. Cambridge, UK: Cambridge University Press.

Beam, A. L., Motsinger-Reif, A., and Doyle, J. (2014). Bayesian neural networks for detecting epistasis in genetic association studies. *BMC Bioinformat*. 15:368. doi: 10.1186/s12859-014-0368-0

Boone, C., Bussey, H., and Andrews, B. J. (2007). Exploring genetic interactions and networks with yeast. *Nat. Rev. Genet*. 8, 437–449. doi: 10.1038/nrg2085

Botta, V., Louppe, G., Geurts, P., and Wehenkel, L. (2014). Exploiting SNP Correlations within Random Forest for genome-wide association studies. *PLoS ONE* 9:e93379. doi: 10.1371/journal.pone.0093379

Breiman, L. (2001). Random forests. *Mach. Learn*. 45, 5–32. doi: 10.1023/A:1010933404324

Bureau, A., Dupuis, J., Falls, K., Lunetta, K. L., Hayward, B., Keith, T. P., et al. (2005). Identifying SNPs predictive of phenotype using random forests. *Genet. Epidemiol*. 28, 171–182. doi: 10.1002/gepi.20041

Bush, W. S., Dudek, S. M., and Ritchie, M. D. (2006). Parallel multifactor dimensionality reduction: a tool for the large-scale analysis of gene-gene interactions. *Bioinformatics* 22, 2173–2174. doi: 10.1093/bioinformatics/btl347

Bush, W. S., Dudek, S. M., and Ritchie, M. D. (2009). Biofilter: a knowledge-integration system for the multi-locus analysis of genome-wide association studies. *Pac. Symp. Biocomput*. 368–379. doi: 10.1142/9789812836939_0035

Chatr-Aryamontri, A., Breitkreutz, B. J., Oughtred, R., Boucher, L., Heinicke, S., Chen, D., et al. (2015). The BioGRID interaction database: 2015 update. *Nucleic Acids Res*. 43, D470–D478. doi: 10.1093/nar/gku1204

Chickering, D. M., Heckerman, D., and Meek, C. (2004). Large-sample learning of Bayesian Networks is NP-Hard. *J. Mach. Learn. Res*. 5, 1287–1330.

Cho, Y. M., Ritchie, M. D., Moore, J. H., Park, J. Y., Lee, K.-U., Shin, H. D., et al. (2004). Multifactor-dimensionality reduction shows a two-locus interaction associated with Type 2 diabetes mellitus. *Diabetologia* 47, 549–554. doi: 10.1007/s00125-003-1321-3

Cordell, H. J. (2002). Epistasis: what it means, what it doesn't mean, and statistical methods to detect it in humans. *Hum. Mol. Genet*. 11, 2463–2468. doi: 10.1093/hmg/11.20.2463

Croft, D., Mundo, A. F., Haw, R., Milacic, M., Weiser, J., Wu, G., et al. (2014). The Reactome pathway knowledgebase. *Nucleic Acids Res*. 42, D472–D477. doi: 10.1093/nar/gkt1102

Culverhouse, R., Suarez, B. K., Lin, J., and Reich, T. (2002). A Perspective on Epistasis: limits of models displaying no main effect. *Am. J. Hum. Genet*. 70, 461–471. doi: 10.1086/338759

De, R., Bush, W. S., and Moore, J. H. (2014). Bioinformatics challenges in genome-wide association studies (GWAS). *Methods Mol. Biol*. 1168, 63–81. doi: 10.1007/978-1-4939-0847-9_5

Dorigo, M., and Gambardella, L. M. (1997). Ant colonies for the travelling salesman problem. *Biosystems* 43, 73–81. doi: 10.1016/S0303-2647(97)01708-5

Eichler, E. E., Flint, J., Gibson, G., Kong, A., Leal, S. M., Moore, J. H., et al. (2010). Missing heritability and strategies for finding the underlying causes of complex disease. *Nat. Rev. Genet*. 11, 446–450. doi: 10.1038/nrg2809

Ellis, J. A., Scurrah, K. J., Li, Y. R., Ponsonby, A. L., Chavez, R. A., Pezic, A., et al. (2015). Epistasis amongst PTPN2 and genes of the vitamin D pathway contributes to risk of juvenile idiopathic arthritis. *J. Steroid Biochem. Mol. Biol*. 145, 113–120. doi: 10.1016/j.jsbmb.2014.10.012

Fisher, R. A. (1918). The correlation between relatives on the supposition of Mendelian inheritance. *Trans. R. Soc. Edin*. 52, 399–433. doi: 10.1017/S0080456800012163

Franceschini, A., Szklarczyk, D., Frankild, S., Kuhn, M., Simonovic, M., Roth, A., et al. (2013). STRING v9.1: protein-protein interaction networks, with increased coverage and integration. *Nucleic Acids Res*. 41, D808–D815. doi: 10.1093/nar/gks1094

Gertz, J., Gerke, J. P., and Cohen, B. A. (2010). Epistasis in a quantitative trait captured by a molecular model of transcription factor interactions. *Theor. Popul. Biol*. 77, 1–5. doi: 10.1016/j.tpb.2009.10.002

Gou, J., Zhao, Y., Wei, Y., Wu, C., Zhang, R., Qiu, Y., et al. (2014). Stability SCAD: a powerful approach to detect interactions in large-scale genomic study. *BMC Bioinformatics*. 15:62. doi: 10.1186/1471-2105-15-62

Goudey, B., Rawlinson, D., Wang, Q., Shi, F., Ferra, H., Campbell, R. M., et al. (2013). GWIS–model-free, fast and exhaustive search for epistatic interactions in case-control GWAS. *BMC Genomics*. 13 (Suppl. 3):S10. doi: 10.1186/1471-2164-14-S3-S10

Grady, B. J., Torstenson, E. S., McLaren, P. J., DE Bakker, P. I., Haas, D. W., Robbins, G. K., et al. (2011). Use of biological knowledge to inform the analysis of gene-gene interactions involved in modulating virologic failure with efavirenz-containing treatment regimens in ART-naïve ACTG clinical trials participants. *Pac. Symp. Biocomput*. 253–264.

Greene, C. S., Hill, D. P., and Moore, J. H. (2009a). Environmental sensing of expert knowledge in a computational evolution system for complex problem solving in human genetics. *Genet. Evolut. Comput*. 19–36. doi: 10.1007/978-1-4419-1626-6_2

Greene, C. S., Himmelstein, D. S., Kiralis, J., and Moore, J. H. (2010). The informative extremes: using both nearest and farthest individuals can improve relief algorithms in the domain of human genetics. *Evolut. Comput. Mach. Learn. Data Min. Bioinform*. 6023, 182–193. doi: 10.1007/978-3-642-12211-8_16

Greene, C. S., Penrod, N. M., Kiralis, J., and Moore, J. H. (2009b). Spatially uniform relieff (SURF) for computationally-efficient filtering of gene-gene interactions. *BioData Min.* 2:5. doi: 10.1186/1756-0381-2-5

Gui, J., Moore, J. H., Williams, S. M., Andrews, P., Hillege, H. L., van der Harst, P., et al. (2013). A simple and computationally efficient approach to multifactor dimensionality reduction analysis of gene-gene interactions for quantitative traits. *PLoS ONE* 8:e66545. doi: 10.1371/journal.pone.0066545

Hahn, L. W., Ritchie, M. D., and Moore, J. H. (2003). Multifactor dimensionality reduction software for detecting gene-gene and gene-environment interactions. *Bioinformatics* 19, 376–382. doi: 10.1093/bioinformatics/btf869

Han, B., and Chen, X. W. (2011). bNEAT: a Bayesian network method for detecting epistatic interactions in genome-wide association studies. *BMC Genomics* 12 (Suppl. 2):S9. doi: 10.1186/1471-2164-12-S2-S9

Han, B., Chen, X.-W., and Talebizadeh, Z. (2011). FEPI-MB: identifying SNPs-disease association using a Markov Blanket-based approach. *BMC Bioinform.* 12 (Suppl. 12):S3. doi: 10.1186/1471-2105-12-S12-S3

Han, B., Chen, X. W., Talebizadeh, Z., and Xu, H. (2012). Genetic studies of complex human diseases: characterizing SNP-disease associations using Bayesian networks. *BMC Syst Biol.* 6 (Suppl. 3):S14. doi: 10.1186/1752-0509-6-S3-S14

Han, B., Park, M., and Chen, X. W. (2010). A Markov blanket-based method for detecting causal SNPs in GWAS. *BMC Bioinform*. 11 (Suppl. 3):S5. doi: 10.1186/1471-2105-11-S3-S5

Cordell, H. J. (2009). Detecting gene-gene interactions that underlie human diseases. *Nat. Rev. Genet*. 10, 392–404. doi: 10.1038/nrg2579

Hirschhorn, J. N. (2009). Genomewide association studies–illuminating biologic pathways. *N. Engl. J. Med*. 360, 1699–1701. doi: 10.1056/NEJMp0808934

Howard, T. D., Koppelman, G. H., Xu, J., Zheng, S. L., Postma, D. S., Meyers, D. A., et al. (2002). Gene-gene interaction in Asthma: IL4RA and IL13 in a Dutch population with Asthma. *Am. J. Hum. Genet*. 70, 230–236. doi: 10.1086/338242

Huang, C. H., Pei, J. C., Luo, D. Z., Chen, C., Chen, Y. W., and Lai, W. S. (2015). Investigation of gene effects and epistatic interactions between Akt1 and neuregulin 1 in the regulation of behavioral phenotypes and social functions in genetic mouse models of schizophrenia. *Front. Behav. Neurosci*. 8:455. doi: 10.3389/fnbeh.2014.00455

Huang, Y., Wuchty, S., and Przytycka, T. M. (2013). eQTL Epistasis - challenges and computational approaches. *Front. Genet*. 4:51. doi: 10.3389/fgene.2013.00051

Jiang, R., Tang, W., Wu, X., and Fu, W. (2009). A random forest approach to the detection of epistatic interactions in case-control studies. *BMC Bioinform*. 10:S65. doi: 10.1186/1471-2105-10-S1-S65

Jing, P. J., and Shen, H. B. (2015). MACOED: a multi-objective ant colony optimization algorithm for SNP epistasis detection in genome-wide association studies. *Bioinformatics* 31, 634–641. doi: 10.1093/bioinformatics/btu702

Johnstone, I. M., and Titterington, D. M. (2009). Statistical challenges of high-dimensional data. *Philos. Trans. A. Math. Phys. Eng. Sci.* 367, 4237–4253. doi: 10.1098/rsta.2009.0159

Kanehisa, M., Goto, S., Sato, Y., Furumichi, M., and Tanabe, M. (2012). KEGG for integration and interpretation of large-scale molecular data sets. *Nucleic Acids Res.* 40, D109–D114. doi: 10.1093/nar/gkr988

Kerrien, S., Aranda, B., Breuza, L., Bridge, A., Broackes-Carter, F., Chen, C., et al. (2012). The IntAct molecular interaction database in 2012. *Nucleic Acids Res.* 40, D841–D846. doi: 10.1093/nar/gkr1088

Kononenko, I. (1994). Estimating attributes: analysis and extensions of RELIEF. *Lect. Notes Comp. Sci.* 784, 171–182. doi: 10.1007/3-540-57868-4_57

Koo, C. L., Liew, M. J., Mohamad, M. S., and Salleh, A. H. (2013). A Review for detecting gene-gene interactions using machine learning methods in genetic epidemiology. *Biomed. Res. Int.* 2013:432375. doi: 10.1155/2013/432375

Leinweber, D. J. (2007). Stupid data miner tricks:overfitting the S&P 500. *J. Invest.* 16, 15–22. doi: 10.3905/joi.2007.681820

Liu, J., Martin-Yken, H., Bigey, F., Dequin, S., François, J. M., and Capp, J. P. (2015). Natural yeast promoter variants reveal epistasis in the generation of transcriptional-mediated noise and its potential benefit in stressful conditions. *Genome Biol. Evol.* 7, 969–984. doi: 10.1093/gbe/evv047

Lu, Q., Wei, C., Ye, C., Li, M., and Elston, R. C. (2012). A likelihood ratio-based Mann-Whitney approach finds novel replicable joint gene action for type 2 diabetes. *Genet. Epidemiol.* 36, 583–593. doi: 10.1002/gepi.21651

Ma, L., Keinan, A., and Clark, A. G. (2015). Biological knowledge-driven analysis of epistasis in human GWAS with application to lipid traits. *Methods Mol. Biol.* 1253, 35–45. doi: 10.1007/978-1-4939-2155-3_3

Mackay, T. F. (2014). Epistasis and quantitative traits: using model organisms to study gene-gene interactions. *Nat. Rev. Genet.* 15, 22–33. doi: 10.1038/nrg3627

Mackay, T. F., and Moore, J. H. (2014). Why epistasis is important for tackling complex human disease genetics. *Genome Med.* 6, 42. doi: 10.1186/gm561

Maher, B. (2008). Personal genomes: the case of the missing heritability. *Nature* 456, 18–21. doi: 10.1038/456018a

Manolio, T. A., Collins, F. S., Cox, N. J., Goldstein, D. B., Hindorff, L. A., Hunter, D. J., et al. (2009). Finding the missing heritability of complex diseases. *Nature* 461, 747–753. doi: 10.1038/nature08494

Marchini, J., Donnelly, P., and Cardon, L. R. (2005). Genome-wide strategies for detecting multiple loci that influence complex diseases. *Nat. Genet.* 37, 413–417. doi: 10.1038/ng1537

Matsubara, K., Yamamoto, E., Mizobuchi, R., Yonemaru, J., Yamamoto, T., Kato, H., et al. (2015). Hybrid breakdown caused by epistasis-based recessive incompatibility in a cross of rice (Oryza sativa L.). *J. Hered.* 106, 113–122. doi: 10.1093/jhered/esu065

Matsuda, H. (2000). Physical nature of higher-order mutual information: intrinsic correlations and frustration. *Phys. Rev. E.* 62, 3096–3102. doi: 10.1103/PhysRevE.62.3096

McDonald, J. H. (2014). *Handbook of Biological Statistics, 3rd Edn.* Baltimore, MD: Sparky House Publishing.

McKinney, B. A., Reif, D. M., Ritchie, M. D., and Moore, J. H. (2006). Machine learning for detecting gene-gene interactions. *Appl. Bioinformat.* 5, 77–88. doi: 10.2165/00822942-200605020-00002

McKinney, B. A., Reif, D. M., White, B. C., Crowe, J. E. Jr., and Moore, J. H. (2007). Evaporative cooling feature selection for genotypic data involving interactions. *Bioinformatics* 23, 2113–2120. doi: 10.1093/bioinformatics/btm317

Moore, J. H. (2003). The ubiquitous nature of epistasis in determining susceptibility to common human diseases. *Hum. Hered.* 56, 73–82. doi: 10.1159/000073735

Moore, J. H., Amos, R., Kiralis, J., and Andrews, P. C. (2015). Heuristic identification of biological architectures for simulating complex hierarchical genetic interactions. *Genet Epidemiol.* 39, 25–34. doi: 10.1002/gepi.21865

Moore, J. H., and Andrews, P. C. (2015). Epistasis analysis using multifactor dimensionality reduction. *Methods Mol. Biol.* 1253, 301–314. doi: 10.1007/978-1-4939-2155-3_16

Moore, J. H., Gilbert, J. C., Tsai, C. T., Chiang, F. T., Holden, T., Barney, N., et al. (2006). A flexible computational framework for detecting, characterizing, and interpreting statistical patterns of epistasis in genetic studies of human disease susceptibility. *J. Theor. Biol.* 241, 252–261. doi: 10.1016/j.jtbi.2005.11.036

Moore, J. H., and Hill, D. P. (2015). Epistasis analysis using artificial intelligence. *Methods Mol. Biol.* 1253, 327–346. doi: 10.1007/978-1-4939-2155-3_18

Moore, J. H., and White, B. C. (2007). Tuning ReliefF for genome-wide genetic analysis. *Evol. Comput. Mach. Learn. Data Min. Bioinformat.* 4447, 166–175. doi: 10.1007/978-3-540-71783-6_16

Moore, J. H., and Williams, S. M. (2005). Traversing the conceptual divide between biological and statistical epistasis: systems biology and a more modern synthesis. *Bioessays* 27, 637–646. doi: 10.1002/bies.20236

Moore, J. H., and Williams, S. M. (2009). Epistasis and its implications for personal genetics. *Am. J. Hum. Genet.* 85, 309–320. doi: 10.1016/j.ajhg.2009.08.006

Namkung, J., Elston, R. C., Yang, J. M., and Park, T. (2009). Identification of gene-gene interactions in the presence of missing data using the multifactor dimensionality reduction method. *Genet Epidemiol.* 33, 646–656. doi: 10.1002/gepi.20416

Nishimura, D. (2001). BioCarta. *Biotech Softw. Internet Rep.* 2, 117–120. doi: 10.1089/152791601750294344

Pattin, K. A., and Moore, J. H. (2008). Exploiting the proteome to improve the genome-wide genetic analysis of epistasis in common human diseases. *Hum. Genet.* 124, 19–29. doi: 10.1007/s00439-008-0522-8

Payne, J. L., Greene, C. S., Hill, D. P., and Moore, J. H. (2010). Sensible initialization of a computational evolution system using expert knowledge for epistasis analysis in human genetics. *Exploitation Link. Learn. Evol. Algorithms* 3, 215–226. doi: 10.1007/978-3-642-12834-9_10

Pendergrass, S. A., Frase, A., Wallace, J., Wolfe, D., Katiyar, N., Moore, C., et al. (2013a). Genomic analyses with biofilter 2.0: knowledge driven filtering, annotation, and model development. *Bio. Data Min.* 6:25. doi: 10.1186/1756-0381-6-25

Pendergrass, S. A., Verma, S. S., Holzinger, E. R., Moore, C. B., Wallace, J., Dudek, S. M., et al. (2013b). Next-generation analysis of cataracts: determining knowledge driven gene-gene interactions using Biofilter, and gene-environment interactions using the PhenX Toolkit. *Pac. Symp. Biocomput.* 147–58. doi: 10.1142/9789814447973_0015

du Prel, J.-B., Hommel, G., Röhrig, B., and Blettner, M. (2009). Confidence interaval or *p*-value? *Dtsch. Arztebl. Int.* 106, 335–339. doi: 10.3238/arztebl.2009.0335

Purcell, S., Neale, B., Todd-Brown, K., Thomas, L., Ferreira, M. A., Bender, D., et al. (2007). PLINK: a tool set for whole-genome association and population-based linkage analyses. *Am. J. Hum. Genet.* 81, 559–575. doi: 10.1086/519795

Ritchie, M. D. (2015). Finding the epistasis needles in the genome-wide haystack. *Methods Mol. Biol.* 1253, 19–33. doi: 10.1007/978-1-4939-2155-3_2

Ritchie, M. D., Hahn, L. W., and Moore, J. H. (2003). Power of multifactor dimensionality reduction for detecting gene-gene interactions in the presence of genotyping error, missing data, phenocopy, and genetic heterogeneity. *Genet. Epidemiol.* 24, 150–157. doi: 10.1002/gepi.10218

Ritchie, M. D., Hahn, L. W., Roodi, N., Bailey, L. R., Dupont, W. D., Parl, F. F., et al. (2001). Multifactor-dimensionality reduction reveals high-order interactions among estrogen-metabolism genes in sporadic breast cancer. *Am. J. Hum. Genet.* 69, 138–147. doi: 10.1086/321276

Robnik-Šikonja, M., and Kononenko, I. (2003). Theoretical and empirical analysis of ReliefF and RReliefF. *Mach. Learn.* 53, 23–69. doi: 10.1023/A:1025667309714

Sasieni, P. D. (1997). From genotypes to genes: doubling the sample size. *Biometrics* 53, 1253–1261. doi: 10.2307/2533494

Schwarz, D. F., König, I. R., and Ziegler, A. (2010). On safari to Random Jungle: a fast implementation of Random Forests for high-dimensional data. *Bioinformatics* 26, 1752–1758. doi: 10.1093/bioinformatics/btq257

Siemiatycki, J., and Thomas, D. C. (1981). Biological models and statistical interactions: an example from multistage carcinogenesis. *Int. J. Epidemiol.* 10, 383–387. doi: 10.1093/ije/10.4.383

Smith, S. B., Reenilä, I., Männistö, P. T., Slade, G. D., Maixner, W., Diatchenko, L., et al. (2014). Epistasis between polymorphisms in COMT, ESR1, and GCH1 influences COMT enzyme activity and pain. *Pain* 155, 2390–2399. doi: 10.1016/j.pain.2014.09.009

Statnikov, A., Lytkin NI, Lemeire, J., and Aliferis, C. F. (2013). Algorithms for discovery of multiple markov boundaries. *J. Mach. Learn. Res.* 14, 499–566.

Steen, K. V. (2012). Travelling the world of gene-gene interactions. *Brief Bioinform.* 13, 1–19. doi: 10.1093/bib/bbr012

Strobl, C., Boulesteix, A.-L., Kneib, T., Augustin, T., and Zeileis, A. (2008). Conditional variable importance for random forests. *BMC Bioinformat.* 9:307. doi: 10.1186/1471-2105-9-307

Taylor, M. B., and Ehrenreich, I. M. (2015). Higher-order genetic interactions and their contribution to complex traits. *Trends Genet.* 31, 34–40. doi: 10.1016/j.tig.2014.09.001

Vassy, J. L., Hivert, M. F., Porneala, B., Dauriz, M., Florez, J. C., Dupuis, J., et al. (2014). Polygenic type 2 diabetes prediction at the limit of common variant detection. *Diabetes* 63, 2172–2182. doi: 10.2337/db13-1663

Waddington, C. H. (1942). Canalization of development and the inheritance of acquired characters. *Nature* 150, 563–565. doi: 10.1038/150563a0

Wan, X., Yang, C., Yang, Q., Xue, H., Fan, X., Tang, N. L., et al. (2010). BOOST: a fast approach to detecting gene-gene interactions in genome-wide case-control studies. *Am. J. Hum. Genet.* 87, 325–340. doi: 10.1016/j.ajhg.2010.07.021

Wang, X., Elston, R. C., and Zhu, X. (2010a). The meaning of interaction. *Hum. Hered.* 70, 269–277. doi: 10.1159/000321967

Wang, Y., Liu, X., Robbins, K., and Rekaya, R. (2010b). AntEpiSeeker: detecting epistatic interactions for case-control studies using a two-stage ant colony optimization algorithm. *BMC Res. Notes* 3:117. doi: 10.1186/1756-0500-3-117

Wei, C., and Lu, Q. (2014). GWGGI: software for genome-wide gene-gene interaction analysis. *BMC Genet.* 15:101. doi: 10.1186/s12863-014-0101-z

Wei, C., Schaid, D. J., and Lu, Q. (2013). Trees Assembling Mann-Whitney approach for detecting genome-wide joint association among low-marginal-effect loci. *Genet. Epidemiol.* 37, 84–91. doi: 10.1002/gepi.21693

Willighagen, E. L., Waagmeester, A., Spjuth, O., Ansell, P., Williams, A. J., Tkachenko, V., et al. (2013). The ChEMBL database as linked open data. *J. Cheminform.* 5:23. doi: 10.1186/1758-2946-5-23

Yanlan, L., and Jiawei, L. (2012). An improved markov blanket approach to detect SNPs-Disease Associations in case-control studies. *Int. J. Digit. Content Technol. Appl.* 6, 278–286. doi: 10.4156/jdcta.vol6.issue15.32

Yoshida, M., and Koike, A. (2011). SNPInterForest: a new method for detecting epistatic interactions. *BMC Bioinformat.* 12:469. doi: 10.1186/1471-2105-12-469

Zhang, Y., and Liu, J. S. (2007). Bayesian inference of epistatic interactions in case-control studies. *Nat. Genet.* 39, 1167–1173. doi: 10.1038/ng2110

Conflict of Interest Statement: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

frontiers in
BIOENGINEERING AND BIOTECHNOLOGY

TECHNOLOGY REPORT
published: 19 August 2014
doi: 10.3389/fbioe.2014.00027

Systems biology of cancer: a challenging expedition for clinical and quantitative biologists

Ilya Korsunsky[1]*, Kathleen McGovern[2], Tom LaGatta[3], Loes Olde Loohuis[4], Terri Grosso-Applewhite[4], Nancy Griffeth[5] and Bud Mishra[1,3]

[1] *Department of Computer Science, Courant Institute, New York University, New York, NY, USA*
[2] *Department of Mathematics and Statistics, Hunter College, City University of New York, New York, NY, USA*
[3] *Department of Mathematics, Courant Institute, New York University, New York, NY, USA*
[4] *Department of Computer Science, The Graduate Center, City University of New York, New York, NY, USA*
[5] *Department of Mathematics and Computer Science, Lehman College, City University of New York, New York, NY, USA*

Edited by:
David A. Rosenblueth, Universidad Nacional Autónoma de México, Mexico

Reviewed by:
Jérôme Feret, Institut National en Informatique et Automatique, France
Fabio Passetti, Instituto Nacional de Câncer, Brazil

***Correspondence:**
Ilya Korsunsky, Department of Computer Science, Courant Institute, New York University, 715 Broadway Room 1012, New York, NY 10003, USA
e-mail: ilya.korsunsky@gmail.com

A systems-biology approach to complex disease (such as cancer) is now complementing traditional experience-based approaches, which have typically been invasive and expensive. The rapid progress in biomedical knowledge is enabling the targeting of disease with therapies that are precise, proactive, preventive, and personalized. In this paper, we summarize and classify models of systems biology and model checking tools, which have been used to great success in computational biology and related fields. We demonstrate how these models and tools have been used to study some of the twelve biochemical pathways implicated in but not unique to pancreatic cancer, and conclude that the resulting mechanistic models will need to be further enhanced by various abstraction techniques to interpret phenomenological models of cancer progression.

Keywords: molecular modeling, new algorithms, dynamical models, multi-scale modeling, cancer pathway modeling

1. INTRODUCTION

The defeat of cancer was envisioned, somewhat optimistically, after just a few years of research starting with extensive genomic and transcriptomic data collection. Such portrayal of the future might have been inspired by on-going research that has focused on characterizing cancer as a disease of the genome and has galvanized massive data-collection projects, such as the ICGC (International Cancer Genome Consortium) (Zhang et al., 2011) and TCGA (The Cancer Genome Atlas): an atlas "to systematically explore the entire spectrum of genomic changes involved in more than 20 types of human cancer" (TCGA, 2013). Such projects have provided an impetus for developing genomics and bioinformatics tools to study genomic aberrations, driver mutations, loss of heterozygosity, copy number fluctuations, epigenomic modifications, and identifications of classes of oncogenes and tumor-suppressor genes. However, the recent focus has begun to shift to a much more amorphous and dynamic model of cancer, as it has become apparent that a better characterization of the disease must also include the evolution of cancer phenotypes in a heterogeneous population of cells, whose individual types and states need to be understood from single-cell measurements of DNA and RNA, at the very least. The picture of natural somatic evolution of cancer, emerging from recent studies, is quite complex: cancer is driven by numerous pathways, by interactions among multiple heterogeneous subpopulations, the immune system and the microenvironment, and also, by intricate "signaling games" played among cancer stem and progenitor cells, further tempered by metabolic constraints. To treat cancer as a "disease of the phenome," cancer systems biology research will need to analyze and model complexities of both cell-autonomous and cell-population-level processes.

Consequently, models of cancer evolution may need to deal with state-space trajectories of thousands of rapidly evolving cell-types in a heterogeneous tumor population. The experimental setup to harvest and feed the data to such an algorithm is challenging: it is not yet possible to routinely sample multiple single tumor-cells (either *in situ* or circulating) from a single human patient at multiple stages of their natural progression (unperturbed by any therapy). Our approach may circumvent this problem by using computational systems biology to simulate this progression on phenomenological and mechanistic models.

Primary challenges for cancer systems biologists, as corroborated (Reya et al., 2001; Jordan et al., 2006; Shackleton et al., 2009; Marjanovic et al., 2013) by prominent research biologists, are as follows: (1) The nature and origin of heterogeneity in cancer are not well understood. (2) Cancer stem cells, their interactions with the stroma (normal cells) and the roles they play in the population, especially in choreographing cancer progression are computationally complex and require sophisticated algorithms and modeling techniques. (3) Disentangling how and which cell-autonomous processes manifest at the population level require new analysis tools. Succinctly generating hypotheses and efficiently correlating them to experimental data require highly sophisticated algorithms, which will very likely involve multiple levels of abstraction, composition of qualitative and quantitative models, and symbolic model checking tools that rely on notions of simulation and bisimulation (exact or approximate). These new challenges in modeling

and analysis will spur on new research in theoretical computer science. The possible approaches to these challenges are discussed further with illustrative examples.

The paper intends to motivate a disparate group of researchers from multiple disciplines to attack a problem that has not only remained undefeated despite a decades-long all-consuming war against cancer but also has recently revealed new complexities, against which our arsenal has no effective weapons. We wish to inspire game theorists, control engineers, and computer scientists to modify their traditional tools to tame and contain cancer as in many other chronic diseases. We wish to encourage system biologists, bioinformaticists, and oncologists to familiarize themselves with the newer and more powerful tools that rely on abstraction and meta-analysis to overcome the challenges posed by heterogeneity and temporality.

In what follows we focus on the new algorithmic strategies developed to address heterogeneity and temporality as well as other future challenges and obstacles: we start with a summary of classes of models (stochastic, differential, finite-state models, hierarchical, rule-based, and multi-scale) and computational tools (based on execution, simulation, bisimulation, abstraction, composition, and model checking) that are being actively developed by computer scientists. We discuss how these models and tools can be applied to cancer using examples of some of the biochemical pathways implicated in pancreatic cancer (e.g., TGF-β signaling). We also identify critical gaps in the currently available toolkits and future research directions.

The most common form of pancreatic cancer, pancreatic ductal adenocarcinoma (PDAC), is still one of the least understood and most difficult to diagnose and treat of cancers. A central question to ameliorating these difficulties is to identify the genetics drivers behind the origins and progression of PADC. Although PADC is known (Delpu et al., 2011) to arise from 3 different types of precursor lesions, pancreatic intraepithelial neoplasia (PanIN), intraductal papillary mucinous neoplasms (IPMN), and mucinous cystic neoplasms (MCN), the genetic events that characterize the lesions and the transition from lesion to tumor are unknown. It is well accepted that while particular genomic events drive tumorigenesis, it is the change in cellular function caused by that event that is selected for through somatic evolution. Intracellular signaling pathways are common targets of these events. Because of their well understood relations to cellular function, pathways are more consistent and regular markers of tumorigenesis. To better understand which pathways are affected in PDAC, Jones et al. (2008) examined several candidate pathways and found 12 primary ones most common in PDAC tumor samples. In particular, they implicated the pathways associated with apoptosis, DNA damage control, regulation of the G1/S transition in the cell cycle, hedgehog signaling, homophilic cell adhesion, integrin signaling, c-Jun N-terminal kinase signaling, KRAS signaling, regulation of invasion, small GTPase-dependent kinase signaling, TGF-β signaling, and Wnt/Notch signaling. A better understanding of these pathways, how they interact, and how they are affected in PDAC will lead to better clinical diagnosis and intervention.

The rest of the paper is organized as follows. Section 2 summarizes models and tools currently used to represent and analyze dynamical systems in systems biology. Section 3 discusses the need for novel tools to deal with the influx of new personalized data. In Sections 2 and 3, we also turn to several systems biological examples, all related to cancer, which we have explored in the context of a National Science Foundation Expedition-in-Computing project. Our team focused and developed systems for model checking, robustness analysis, multi-scale analysis, etc., which have played a strong role in improving our understanding of the pancreatic cancer phenotypes. Our starting point was with the twelve pathways identified by Jones et al. (2008), described above. We describe a few examples using these pathways to motivate the use of the new modeling and analytical tools described above and the additional use of techniques and tools for abstracting, combining, and otherwise manipulating models. We discuss the biological significance of each example, followed by a brief explanation of the results obtained from the application of the chosen tool. Lastly, Section 4 concludes with a discussion on how the new class of tools we propose will affect biological modeling and clinical practice in cancer.

2. MODELS AND TOOLS FOR CELL-AUTONOMOUS DYNAMIC PROCESSES

Despite their apparent variety, all computational models of dynamic systems are just abstract, succinct, and formal representations of reality; their form almost always consists of two components: state, which describes the most relevant parts of the configuration of the system at some time, and flow, which describes how the configuration will change in the near future. Usually, we will prefer models with succinct state-space description, but only to the extent that this need for succinctness does not introduce unacceptable distortion in the dynamic behavior of the model. Within a framework comprising such models, researchers have developed powerful tools to compare, translate, and combine formal models of disparate types. Two important weapons in a systems biologist's arsenal are the processes of abstraction and composition: abstraction facilitates translations among representations, as needed, while composition enables construction of complex, multi-scale, and systems-level models built from simpler component structures.

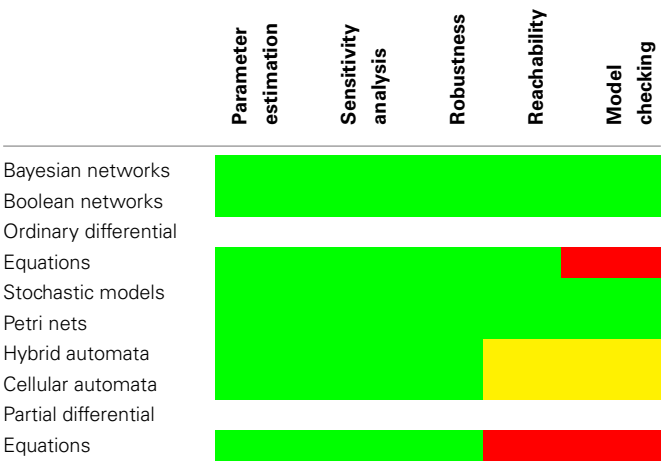
Analytical tools comprise the other half of the toolkit. They allow for the examination of model properties beyond basic simulation. However, tool applicability is inherently limited by the fact that a specific tool might have been developed originally for use in one specific class of models. **Table 1A** provides a sense of the compatibility of some key analytical tools for a broad variety of model classes. To construct this table, we relied on an extensive literature survey of each model class and tool (White, 1977; Dytham, 1995; Bengtsson et al., 1996; Henzinger et al., 1997; Cozman, 1997; Ghosh and Tomlin, 2001; Alur et al., 2001; Bandini et al., 2001; Barton and Lee, 2002; Sutner, 2002, 2009; Wang et al., 2002; Antoniotti et al., 2003a,b; Shmulevich et al., 2003; Ghosh et al., 2003; Friedman and Koller, 2003; Lincoln and Tiwari, 2004; Janes et al., 2004; Friedman, 2004; Li and Chan, 2004; Kwiatkowska et al., 2004; Ihekwaba et al., 2004; Hagiya et al., 2004; Das et al., 2004; Pe'er, 2005; Fauré et al., 2006; Reeves et al., 2006; Langmead et al., 2006; Kim et al., 2006; Chaouiya, 2007; Saez-Rodriguez et al., 2007; Fränzle and Herde, 2007; Narasimhan and Biswas, 2007; Wilkinson,

2007; Sandmann, 2007; Mukherjee and Speed, 2008; Clarke et al., 2008; Sandmann and Wolf, 2008; Ryu et al., 2008; Donaldson and Gilbert, 2008; Figueirêdo et al., 2008; Yuceer et al., 2008; Qian and Dougherty, 2009; Tatyana et al., 2009; Wartlick et al., 2009; Sobie, 2009; Langmead, 2009; Didier et al., 2009; Müssel et al., 2010; Sarkar and Sobie, 2010; Campagna and Piazza, 2010; Bortolussi and Policriti, 2010; Garmaroudi et al., 2010; Yang and Lin, 2010; Donzé et al., 2010; Gunawardena, 2010; Vikram et al., 2010; Kobayashi and Hiraishi, 2010, 2011; Gong et al., 2010, 2011a,b,c; Dimitrova et al., 2011; Grosu et al., 2011; Alfieri et al., 2011; Fischer and Kaiser, 2011; Aldinucci et al., 2011; Brim et al., 2011; Sarkar et al., 2012; Horvath, 2012; Iyengar et al., 2012). In this table, each row represents a class of models. From top to bottom, the models range over Bayesian networks, Boolean networks, ordinary differential equations (ODEs), stochastic models, Petri nets, hybrid automata, cellular automata, and partial differential equations (PDEs), each with differing notions of states (discrete, continuous, hybrid, etc.) and flows (transition, evolution, dynamics, etc.). In addition, we chose these models to represent a broad range of model features, including deterministic, non-deterministic, spatial, non-spatial, continuous, discrete, temporal, and logical. Each column, on the other hand, represents a tool. From left to right (in order of increasing complexity), they encompass: parameter estimation, sensitivity analysis, reachability analysis, and model checking of properties describable in propositional temporal logic.

Each entry represents the availability of the tool for the model class. Red implies that the tool is unavailable or inapplicable. Yellow denotes limited applicability. Green denotes wide-spread applicability across models in that class. For obvious reasons, the simpler tools generally have a wider range of applicability than do the complex ones. The most complex tools have proven difficult to adapt to novel circumstances, thus motivating the use of abstraction to broaden their range of applicability. Traditionally, model abstraction has been used to create models that are structurally simpler, but that have the advantage of facilitating rapid analysis by efficient algorithms and provide easily comprehensible explanations of properties and counter-examples. **Table 1B** provides examples of implementations of these tools.

Table 1 | Tools tables. **(A)** A table of references for the use of each analytical tool in each model type, where available. The colors denote availability, as specified in the legend. **(B)** A (non-exhaustive) table of available implementations of analytical tools described in the previous sections.

(A)



(B)

Analytical tool	Resource
Parameter estimation	Simbiology
	JSim, Polynome, and PyMorph
	PARES
Sensitivity analysis	MATLAB – systems biology toolbox
	Simbiology
Robustness analysis	MATLAB – systems biology toolbox
	R sensitivity package
	BIOCHAM
Reachability analysis	MATLAB – robust control toolbox
	PROD, TReX, and RAMAS
Model checking	SMV, HyTECH, and HySAT
	UPPAAL, PRISM, and NuSMV

Available.
Unavailable.
Partially available.

2.1. EXAMPLES

Abstraction provides simplification. For examples, models and analyses that can be constructed and performed, we briefly summarize the findings of two studies on some of the 12 pathways implicated in pancreatic cancer. More details are included in Sections 2.1.1 and 2.1.2, and for more information on the tools used, see Section 2.2.

The first study (Gong et al., 2011c) uses a Boolean circuit as an abstraction of several interacting pathways, including MDM2, P53, NFκB, and HMGB1, and performs symbolic model checking on the resulting abstract circuit. Among many findings, it confirmed, as expected, that P53 can induce the transcription of MDM2, while MDM2 is a negative regulator of P53, and that NFκB's activation is not a necessary checkpoint that the cancer cell must go through to achieve both proliferation and immortality. Other local analyses related to such abstraction involve: reachability analysis, local and global robustness analysis, parameter identification, and analysis of their sensitivity, etc.

The second study investigates a published model of extrinsically induced apoptosis (Albeck et al., 2008) using parameter sensitivity analysis based on a popular statistical tool called partial least squares regression. The analysis reveals 6 enzymatic reactions that contribute substantially to the time it takes the cell to commit to apoptosis from the initial ligand binding event. Interestingly, all 6 reactions occur prior to the permeabilization of the membrane, confirming the accepted theory that permeabilization is the non-reversible step that commits the cell to apoptosis.

2.1.1. *Model checking Boolean model of pancreatic cancer pathways*

While there is a plethora of chemical reagents in a cell, which, in principle, can react with one another, most of these reactions do

not happen under normal physiological conditions (temperature, pH, etc.). Instead, they are tightly regulated by reaction-specific enzymes and the genes that code for them. Thus, gene regulatory networks are characterized by sharp transitions, in which some subset of reactions is turned on, while the others turned off. Thomas et al. have used Boolean models to describe and analyze this behavior of gene regulatory networks (Thomas, 1991, 1998; Bornholdt, 2008), and have shown that it can be well approximated by asynchronous Boolean networks, in which genes are represented as nodes and the regulation by wiring.

A recent study used model checking of a Boolean model of the HMGB1 pathway to verify several experimentally observed behaviors of cancer cells and to suggest further hypotheses for experimental study (Gong et al., 2011c). **Figure 1** shows a circuit diagram representation of this Boolean network. One analytical result was that over-expression of HMGB1 would increase proliferation and decrease apoptosis. This has been experimentally observed, as reported in Kang et al. (2009). Another analytical result is that once the protein Cyclin E is activated by the HMGB1 pathway, and DNA synthesis has commenced, the cell will continue to proliferate, and thus be relatively independent of external controls. This has been identified by Weinberg and Hanahan (2000) as one of the hallmarks of cancer. Another analytical result, which NFκB oscillates after release of HMGB1, had been observed by Hoffmann et al. (2002). Some additional analytical results suggest that P53 can induce the transcription of MDM2, while MDM2

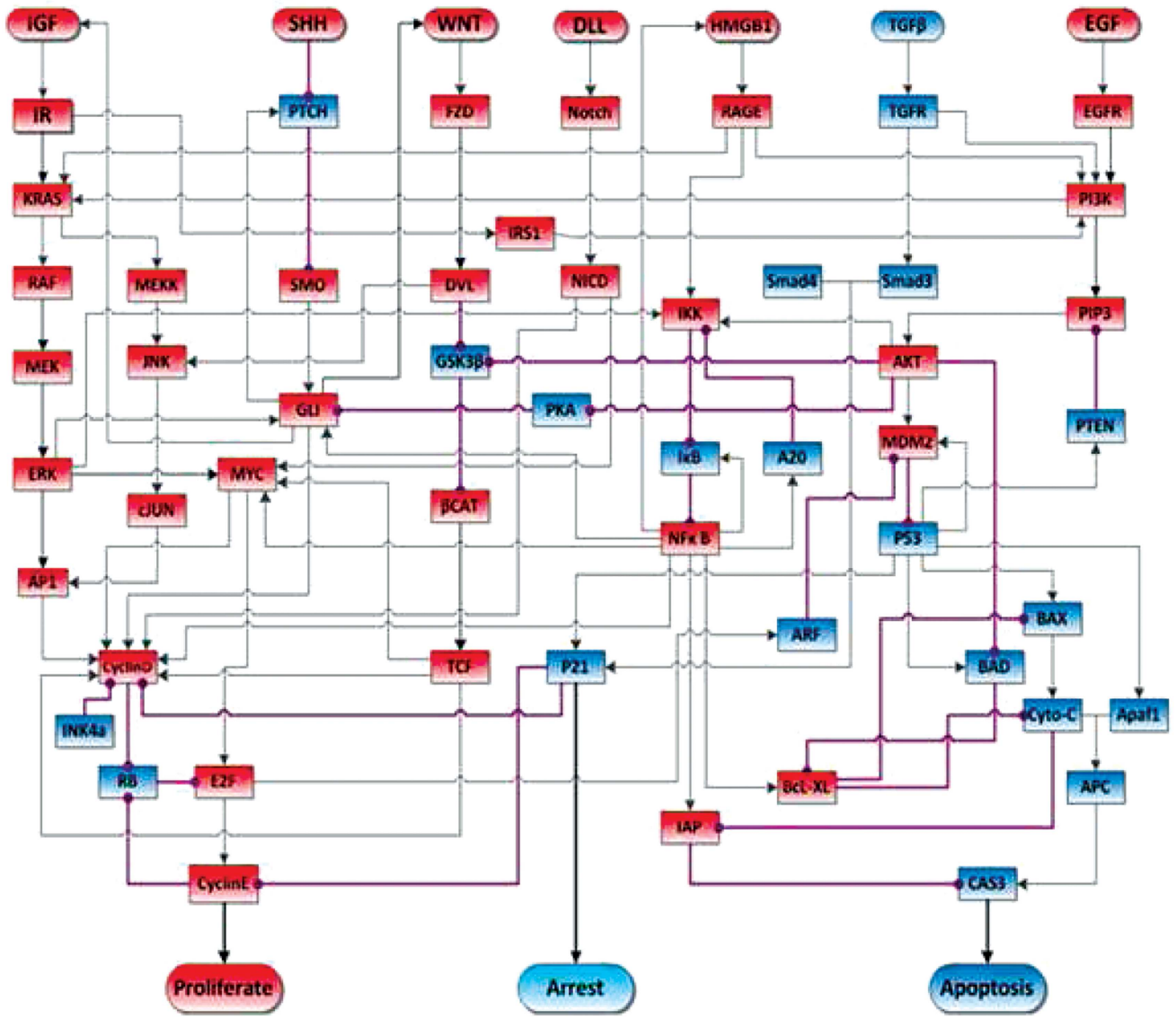


FIGURE 1 | Schematic view of signal transduction in the pancreatic cancer model. Blue nodes represent tumor-suppressor proteins, red nodes represent oncoproteins/lipids. Arrow represents protein activation, circle-headed arrow represents deactivation. The acronyms in each rectangular node stand for signal transduction proteins. The rounded rectangular nodes on the top of the figure stand for ligands that activate the pathways. Finally, the rounded nodes at the bottom stand for a cell behavior activated by the connected effector proteins. Figure adapted from Gong et al. (2011c).

is a negative regulator of P53, and that NFκB's activation is not a necessary checkpoint that the cancer cell should go through on the path to proliferation and immortality.

These results show that model checking can be a powerful tool for the understanding of biological behaviors, just as it has been a powerful tool for understanding complex electronic circuits. Over the past three decades, as the complexity of the engineered circuits have approached that of the natural biological systems, the engineering community had to develop design automation tools built upon powerful algorithms for circuit validation and model checking, first introduced by Clarke and Mishra (1984). Model checking has now become standard protocol for validating electronic circuits.

2.1.2. Sensitivity analysis of TRAIL-induced apoptosis ODE model

A partial least squares regression (PLSR) on a well established model for TRAIL-induced apoptosis (Albeck et al., 2008) established the key reactions responsible for the time it takes for the effector protein cPARP to attain its half saturation. **Figure 2A** represents the reaction network. Each reaction is depicted in the

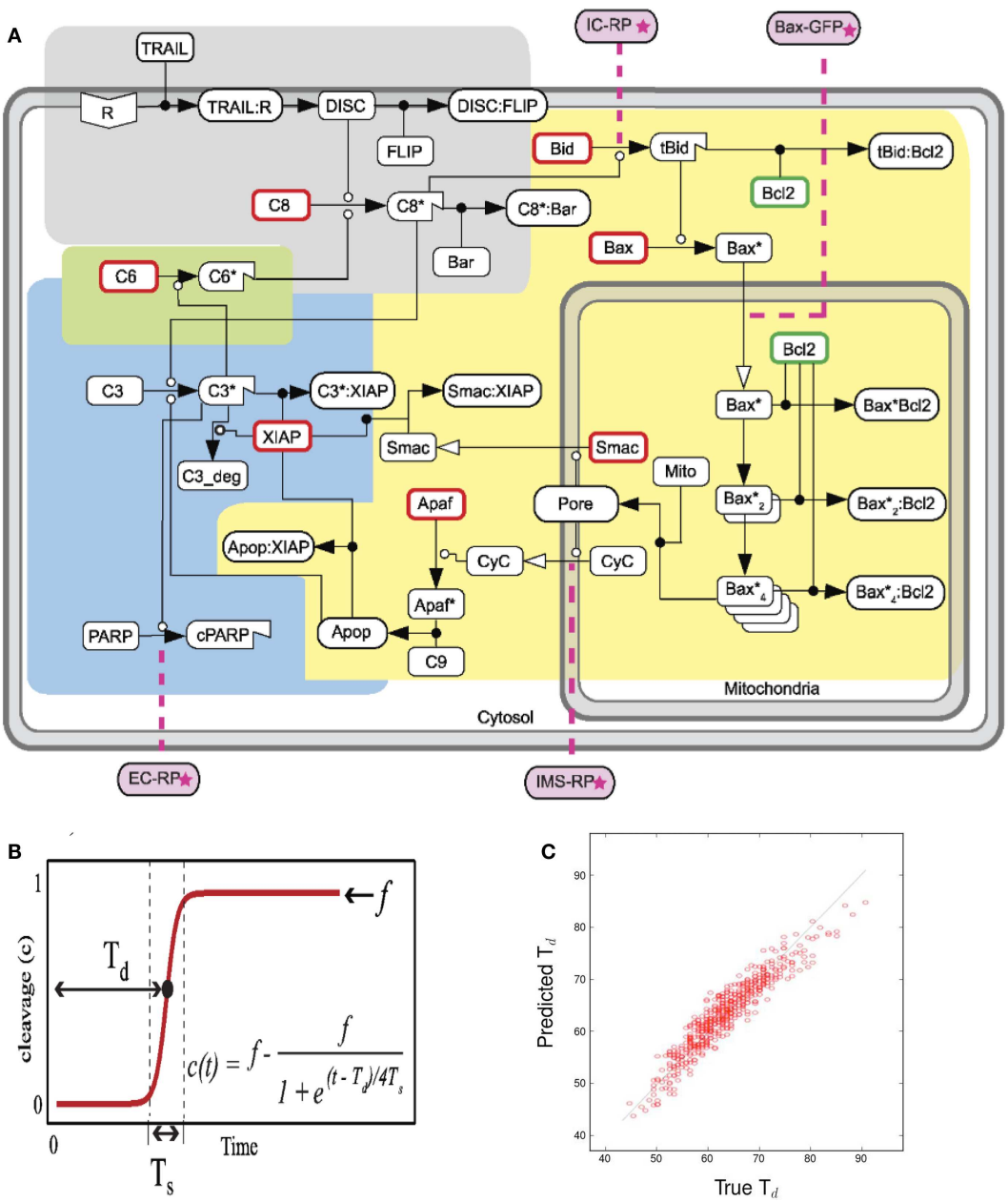


FIGURE 2 | Figures accompanying the sensitivity analysis. **(A)** Schematic of the extrinsic apoptosis reaction model. Each color represents a functional pathway. Adapted from Albeck et al. (2008). **(B)** Cleavage of PARP to cPARP in response to TRAIL induction. Adapted from Albeck et al. (2008). **(C)** Predicted vs. True T_d. Quality of linear regression measured by R^2 value.

compartment (membrane, cytoplasm, or mitochondria) in which it takes place.

Programed cell death, or apoptosis, is crucial in the development and maintenance of a multi-cellular organism, but also provides a critical ingredient to the character of cancer progression, its dominant phenotypes and heterogeneity. Once certain apoptotic proteins are triggered in a cell, whether from intrinsic or environmental signals, a normal cell commits to a program that results in its eventual cell death. Changes in the cell's ability to respond to apoptotic signals and the timing behind its response can cause major disturbances in cellular population homeostasis. It is important to understand how robust this response is to genetic mutations.

For this purpose, we analyzed extrinsic apoptosis signal transduction pathway models using tools designed for sensitivity analysis, to identify the key proteins that may be rate-limiting. Rate-limiting proteins are postulated to have the greatest effect on the apoptotic response, and thus suggest important mutations responsible for diseases in which this response is diminished.

In the ODE model, cleavage of the effector protein PARP into cPARP is the indicator of apoptosis. T_d, the time from ligand-receptor binding to the point at which half of all PARP is cleaved, represents the response time of the cell to the apoptosis-inducing signal. **Figure 2B** shows the typical dynamics of PARP cleavage as well as how to estimate T_d.

Our sensitivity analysis of T_d to all the kinetic rate parameters is based on a linear regression that illustrates the promise of this approach, as indicated by the regression results in **Figure 2C**.

The reactions with the most impact on T_d are described in **Table 2**. We found that the most important reactions are those that precede the permeabilization of the mitochondrial membrane. These results suggest that the flood of mitochondrial proteins into the cytoplasm is difficult to control, and that the most effective drugs would target reactions upstream in the cascade.

Table 2 | Reactions discovered to affect T_d the most in sensitivity analysis.

Reaction	Description
$L + R \underset{k_{-1}}{\overset{k_1}{\rightleftharpoons}} L : R \overset{\kappa_1}{\rightarrow} R^*$	Ligand-receptor binding and unbinding and receptor activation
$R^* + C8 \overset{k_3}{\rightarrow} R^* : C8$	Caspase-8 binding to active receptor
$C8^* + Bar \overset{k_4}{\rightarrow} C8^* : Bar$	Caspase-8 binding to Bar
$C8^* + Bid \overset{k_{10}}{\rightarrow} C8^* : Bid$	Caspase-8 binding to Bid
$Bid + Bcl2 \underset{k_{-11}}{\overset{k_{11}}{\rightleftharpoons}} Bid : Bcl2$	Bid binding and unbinding to Bcl2
$tBid + Bax \overset{k_{12}}{\rightarrow} tBid : Bax$	Activated Bid binding to Bax

2.2. REVIEW OF FIRST GENERATION TOOLS

This section includes a brief description of types of models that have been used to simulate dynamic systems in biology as well as the types of tools that have been used to analyze these models.

2.2.1. Model descriptions

2.2.1.1. Rule-based model. Rule-based models provide a concise way to specify highly complex, parameterized interaction networks between agents (e.g., molecules). The user needs only to encode the possible behaviors of complex molecules and the modeling software automatically generates an ODE (Ordinary Differential Equations) or CTMC (Continuous Time Markov Chains) model to simulate directly. Agent interactions can be aggregated into macroscopic behaviors, capturing temporal changes to statistical properties only, and abstracting away the details of the rules.

2.2.1.2. Dynamic Bayesian network. These models represent the joint distribution of all variables in the system over time (a global time). The network (represented graphically) arises from a factorization of this joint distribution into conditional distributions through the application of Bayes' rule. An edge in the network graph represents a conditional dependence between two variables. Conditional dependences may change over time, so that this is a time-varying graph.

2.2.1.3. Boolean network. This model is characterized by the fact that each variable can only take one of two values, usually on/off or high/low. Boolean networks are commonly used to model gene regulatory networks, in which genes are considered on or off at any given time.

2.2.1.4. Ordinary differential equations (ODEs). Each variable in this model is characterized by an ordinary differential equation that describes how its rate of production and decay are governed by the concentrations of the ensemble of molecules. Such a system of equations is particularly useful for modeling a large biochemical reaction system, in which the average concentrations of each molecule type can be described through mass action dynamics.

2.2.1.5. Continuous time Markov chain (CTMC). This class of stochastic models considers objects as stochastic Markov processes, in which state changes are probabilistic rather than deterministic. Markov processes have no memory, that is, the probability of any given state change depends only on the current state, and not on the history of the states.

These models are useful for capturing the dynamics of small reactive systems, in which small stochastic fluctuations have large effects.

2.2.1.6. Petri network. Historically, Petri Nets (PNs) were developed to model chemical reactions, but have been used extensively to reason about resource sharing in concurrent systems (in computer science). Thus, as they are capable of describing variables and consumption/production transformations among variables in terms of a simple bipartite graph, they have been used in describing biological processes involving small number of molecules. This basic formulation has been further extended to include various features that arise in systems biology, such as continuous and hybrid dynamics, stochastic fluctuations, and a notion of real time.

2.2.1.7. Hybrid automata. In a hybrid automata model, system dynamics are continuous in the short term but in the longer term may switch between discrete modes. Hybrid automata can also simplify a system of complex non-linear equations into several simpler interacting components.

2.2.1.8. Cellular automata. Cellular automata (CA) are spatially and temporally discrete models, whose dynamics are controlled by a set of rules, based on the state of the site and those in its neighborhood. Cellular automata are especially useful in modeling spatial processes such as morphological evolution of tumor growth or cell migration.

2.2.1.9. Partial differential equations. PDEs are a widely studied topic in mathematics and generally describe the continuous dynamics of some variables with respect to 2 or more other variables. In systems biology, PDEs are most commonly used to model system dynamics over time and space. They are thus useful for the same kinds of systems as cellular automata.

2.2.2. Tool descriptions

2.2.2.1. Parameter estimation. The dynamical behavior of a model is dependent on all parameter choices, incorporating numerical parameters to topological ones (e.g., the structure of a network).

Experimental measurements are frequently unavailable for important parameters of a model, and expensive to obtain. Parameter estimation tools are available for all types of models to approximate parameters correctly in model construction. Two general approaches to this tool have emerged. One is based on matching model behavior to numerical data, and the other is based on matching it to higher level descriptions in temporal logic. Parameter estimation often results in a range of possible parameter values that allow the model to reproduce the desired specifications. The width of these ranges depends on sensitivity and robustness.

2.2.2.2. Sensitivity analysis. Parameter sensitivity is the degree to which small changes in a parameter's value affect the overall model behavior. Sensitivity analysis assigns a numerical sensitivity score to each parameter. In a molecular interaction network, these scores yield insight into the relative importance of some molecules in function of the circuit. For instance, a high sensitivity of cell growth to a particular protein may suggest the protein's roll as an oncoprotein.

2.2.2.3. Robustness analysis. In contrast to the sensitivity analysis, robustness probes the system with large perturbations in the parameter values. Instead of identifying the role of key parameters in the model behavior, robustness tests the conditions under which the model reliably produces the same output. This insight is crucial in drug discovery, in that it identifies the targets needed to alter the model's output to produce a significantly different behavior.

2.2.2.4. Reachability analysis. The combinatorial state space of a model can be enormous and each of these states can have different biological significance. Reachability analysis aims to quantify the states that are reachable via an execution of the model, given an initial set of conditions. This analysis stems from graph theory, in which the states of a system are modeled as discrete nodes and the dynamics as edge transitions between the nodes. Therefore, for continuous models, a pre-processing discretization step is necessary to transform it into a discrete model. Biologically, this tool is very powerful at predicting the ability of a cell model to reach unfavorable phenotypes. However, a major challenge is to define states that are biologically meaningful.

2.2.2.5. Model checking. Model checking concisely characterizes all possible behaviors of the model with properties in a high-level, expressive language called temporal logic. Such properties include cycles, temporal precedence, and steady state. Like reachability analysis, model checking performs an exhaustive search on the state space of a model, and therefore, relies on a discretization of the state space. Traditional model checking is geared toward efficiently searching large, finite graphs with deterministic transitions. However, biological systems introduce stochastic complex networks, which we model using infinite graphs and probabilistic transitions. To deal with these new challenges, model checking has been recently expanded to include time-bounds to analyze infinite graphs and probabilities and statistical sampling to analyze graphs with probabilistic transitions. The statistical sampling used in model checking employs Monte Carlo sampling, which is a family of algorithms to efficiently sample from a probability distribution that is usually difficult to sample directly.

2.2.2.6. Causal analysis. Large quantitative models offer a rich representation of the dynamics of a system. However, within all the details of the model, it may be difficult to derive a qualitative understanding of a particular event. For instance, a model of intracellular signaling in cancer may include multiple intersecting pathways and thousands of reagents, but it may not be clear, which reagents and reactions are responsible for the activation of NFκB.

Structural causal analysis has emerged in several fields as a way to answer such qualitative questions in systems whose dynamics consist of discrete events (Nielsen et al., 1981; Danos et al., 2007, 2012; Paulevé et al., 2013). In this analysis, the user identifies a particular outcome of interest and the analysis infers the sequence of events leading up to that outcome or a set of events without which the outcome would not occur. Given these sequences or sets of events, the user can focus on those parts of the model that include the relevant events. For instance, we may be interested in which pathway activations led to the transcription of a particular gene. Causal analysis can identify, which pathways directly led to the transcription in the model, even if the user has no initial hypothesis.

The interpretations of causality discussed here are specific to the systems in which they are implemented. Other notions of causality plays a vital role in systems biology and related fields of machine learning (Pearl, 2000; Kleinberg and Hripcsak, 2011) and statistical inference (Loes et al., 2013), with its roots deep in the philosophical foundations of science (Hume, 1902; Cartwright, 2004). For the sake of limiting the scope, we refrain from delving deeper into the various notions and applications of causality.

2.2.2.7. Model reduction. Model reduction (MR) simplifies a model in such a way that the model is more tractable to represent and execute and less prone to overfitting from too many parameters, while the relevant dynamics of the model remain unperturbed. For demonstration, we consider two examples. The first (Feret et al., 2009; Danos et al., 2010) considers a rule-based model of intracellular signaling that would produce an intractably large system of ODEs with the typical semantics. Instead, the authors compute a set of coarse grained variables, called fragments, from the original set of all possible reagents, according to the interactions between the rules. Unlike that of the original set of molecular species, the system of ODEs for these fragments is compact and tractable. In a particular implementation of their method, the authors compute the fragments for a large model of EGFR signaling that consists of 71 rules and 18,051,984,143,555,729,567 molecular species. The model reduction results in only 175,988 fragments, making it possible to construct a feasible system of ODEs to compute the dynamics of these rules. Moreover, the reduction is proven correct (Danos et al., 2010), in that it does not change the quantitative dynamics of the original model.

The second work (Radulescu et al., 2012) focuses on the use of tropical geometry (TG) for model reduction of networks of biochemical reactions, as represented by a system of differential equations. TG has been used in modeling Algebraic Differential Equations that often appear in the study of normal and aberrant biochemical pathways. TG can informally be described as a piece-wise linear or skeletonized version of algebraic geometry, which has been widely applied in enumerative algebraic geometry in the past and more recently, in computational systems biology for model reduction. Thus, TG's most prominent applications are in obtaining "good" time scale separation in a biochemical reaction network. Its applications are ideal when in the dynamics of certain species, there is a dominant reaction whose effect overshadows that of the rest – not uncommon in an enzymatic reaction. In such a situation, TG can approximate the dynamics of a particular species by only its dominant reaction, until that dominant reaction changes. Tropicalization exploits this idea by simplifying the polynomials that define the rates in the ODE system. Namely, it turns the polynomial into a sum through a log transform and then chooses the largest term by transforming the sum into a max operator. This step reduces the polynomial to a piece-wise smooth function, with fewer parameters but almost identical behavior.

3. MODELS AND TOOLS FOR HETEROGENEOUS POPULATION DYNAMICS

3.1. ABSTRACTION OF ODE MODEL TO TIMED AUTOMATON

To reiterate, model abstraction is a process that simplifies a model in such a way that preserves almost all properties that need to be examined. Such simplifications at multiple scales may play a critical role in modeling a heterogeneous population of cells in a tumor.

We illustrate this approach with an example, highly relevant to cancer: we abstract an ODE model of a bistable switch that controls the G1/S transition in the cell cycle. The key molecules and their interaction leads to a high-level description of the ODE model as portrayed in **Figure 3A**. The two positive feedback loops governing their interactions lead to two stable states and hysteresis in the transitions between the states. The latter property blocks the circuit from transitioning to the G1 phase once it is in the S phase. The value of the growth signals ranges from 0 (no growth signal) to 2 (full saturation).

Our goals in constructing this abstract model are to identify the steady states of the model, as these are likely to be biologically significant, and to characterize the types of transitions among them. In this example, the resulting abstract model shown in **Figure 3B** is a two-state model that captures the two steady states of the detailed, mechanistic model. The transition paths are described by a distribution over the time taken by a transition between a pair of states and by the concentration that modulates a certain transition. For instance, in the presence of a high concentration of input signal, the transition from G1 to S phase is marked by a timing distribution centered on a smaller time (see **Figure 3C**). Notice that both the bistability and hysteresis, the two most important properties of the mechanistic model, are preserved in the abstract model. On the other hand, the exact concentrations of all the molecules in the system are abstracted away. The construction of this simple model was achieved through iterative sampling and simulation, but more complex models may require more advanced techniques, such as those studied in transition state theory and transition path theory (Vanden-Eijnden, 2006).

We performed the abstraction by statistically sampling traces of the concrete ODE model. Each trace began at a stable state perturbed by changing the growth signal and ended when the reagent concentrations reached steady state again. From this procedure, the result of each trace was a starting concrete state, an ending concrete state, and the transition time to get from one to the other. The abstract states were identified by performing k-means clustering on all the starting and ending states of the trace samples, with increasing numbers of clusters. We chose the result that produced the smallest variance of reagent concentrations within clusters, while minimizing the number of clusters. To compute the transition times between some abstract states A and B, for instance, we first labeled the beginning and ending state of each trace sample with the closest abstract states, respectively. Then we considered all traces that started in abstract state A and ended in abstract state B, at a particular growth signal value, and used the transition times of these samples to compute statistics[1] on the transition time between abstract states A and B, at the same growth signal value.

The gain we have achieved by abstracting a simple ODE model into a simpler discrete state model may not be clear in the context of analyzing just a single cell. However, assured that the abstraction is correct, we can use the approximate dynamics to model each cell in a population of a very large number of cells. Note that such an analysis for large mechanistic models would be intractable for realistic cell populations. Instead of modeling the detailed biochemical interactions within a cell, we view each cell as a strategic agent, interacting stochastically with other cells and its own microenvironment. This game theoretic perspective may illuminate emergent behaviors of the population that were impossible to observe in the single-cell simulations.

[1] Namely, mean and variance were used to approximate the distribution of transition time.

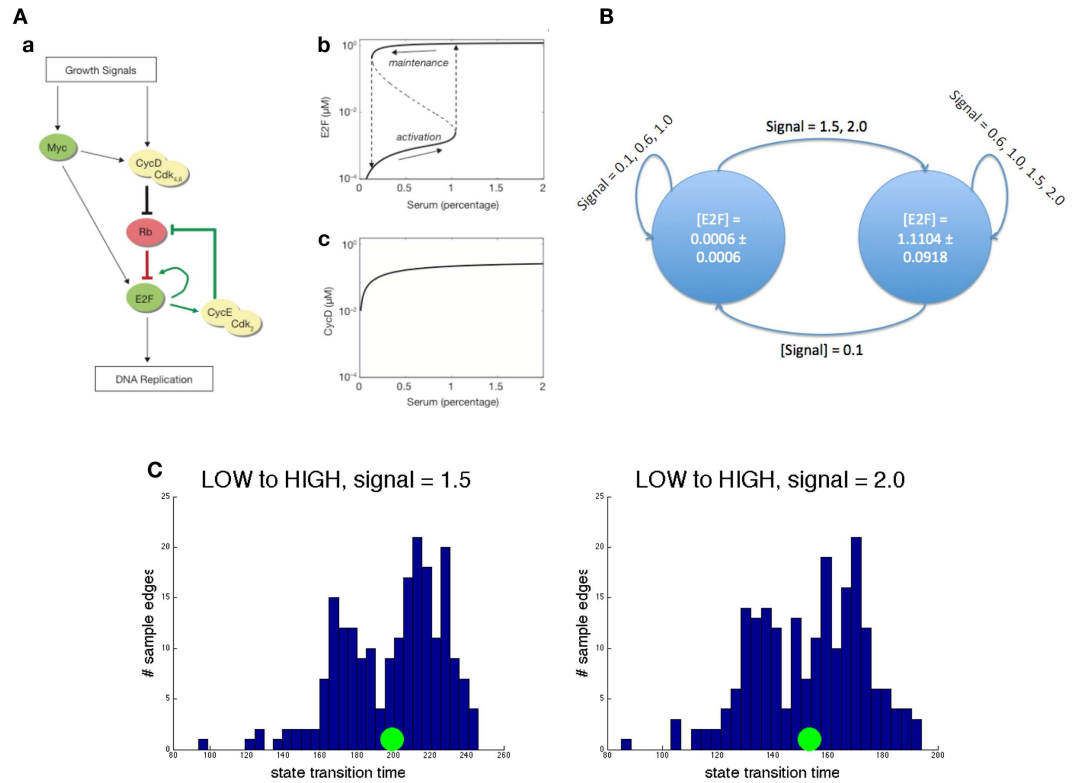


FIGURE 3 | Abstraction example. **(A)** A concrete ODE model. Left: circuit diagram. Pointed arrows denote activation while flat head arrows denote inhibition. Right: demonstration of hysteresis in E2F concentration and the lack of it in cyclin D concentration. **(B)** Its reduced abstraction model. Each state is actually characterized by statistics on all species in the original model. Here, the mean and SD of E2F is used for brevity. Each edge is marked by the growth signal values (mean and SD) that cause that transition. **(C)** Two representative edge distributions. The left panel shows the distribution of transition time from the low E2F state to the high one when the growth signal value is 1.5. The right panel shows the same transition for a growth signal value of 2.0. In both, the green circle represents the mean. Notice that the distributions look similar but the mean transition time decreases substantially for the higher input.

This simple example raises many questions about the nature of models, their relationships to one another, and the possibility of constructing composite models out of modular ones. While we observed that the abstract model above captures two key dynamical properties of the original model, are there guarantees about other dynamical information that we may have lost? For instance, was there a rare but important third state that could produce large population-level effects? It is imperative to formally describe the similarity and distance between these two models, which ostensibly represent the same biological system. Finally, how exactly would we construct a composite model from these abstract models to capture their biochemical and mechanical interactions, which are not specified in the single-cell models?

3.1.1. Formal definition of the abstract model

In this section, we provide a formal definition of the model. This section is meant for readers with a computational background who are interested in the formal details of the model. Reader can safely omit this section and refer instead to the informal description given earlier in the paper.

The formal definition of the timed discrete state abstract model follows.

$$\textbf{Model } M =< S, E, I >$$
$$\textbf{Abstract States } S = (s_1, s_2, \ldots s_n)$$
$$\textbf{Concrete State } s = (p_1, p_2, \ldots p_k), p_i \in \mathbb{R}^{nspecies}$$
$$\textbf{Edges } E = S \times S \times I \rightarrow \Delta(\mathbb{R})$$
$$\textbf{Input Values } I = (in_1, in_2, \ldots in_{ninput}), in_i \in \mathbb{R}$$

The model is a 3-tuple of a set of abstract states, a set of input values, and a set of edges. Each abstract state is currently characterized by a set of clustered concrete model states, although in the future, abstract states would be more succinctly described using some distribution over the ODE network state. An edge is a map from one abstract state (i.e., start state), another abstract state (i.e., end state), and an input (e.g., extracellular signal) to a probability function over the time. Simply, it estimates the time it takes to get from state 1 to state 2 given some input. The set of inputs is the set of all possible inputs to the network, as described in the edge definition.

3.2. COMPOSITION OF LIVER MODEL WITH AGENT BASED POPULATION MODEL

This section illustrates how composition of models allows exploration of the interaction between two or more disparate systems. The goal of the study described here was to determine the optimal dosing schedule for treatment with Taxol, which is a chemotherapeutic drug against many forms of cancer. The main result was an optimal schedule, which would avoid liver damage while eliminating cancer cells.

The problem was to model both liver toxicity in the presence of Taxol and also a population of cells in homeostasis (e.g., a tumor in a specific "cancer hallmark" state). Since the systems are not independent and are modeled with entirely different techniques, their simultaneous simulation is non-trivial. The liver model was constructed from the literature (Holmes et al., 1991; Rahman et al., 1994; Tamura et al., 1995; Manzano et al., 1996; Guengerich and Johnson, 1997) as a system of ODEs (depicted in **Figure 4A**) and the population as an agent based system, in which the cells signal one another to commit apoptosis or divide, determined by the population size.

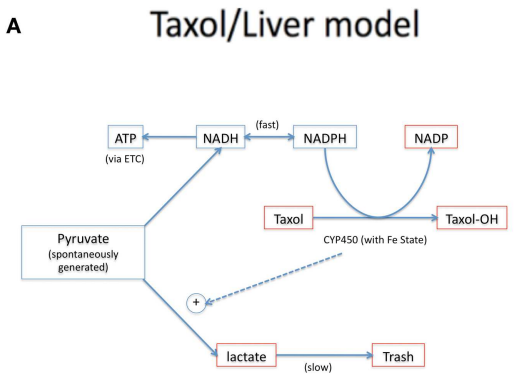
The composition consisted of a KMC-like (kinetic Monte Carlo) simulation algorithm, in which the population model took discrete steps and the liver model was simulated continuously between the steps. Both models shared a common variable, tracking the concentration of Taxol in the organism. At the end of each model's simulation "step," the model continuously updated the global concentration of Taxol.

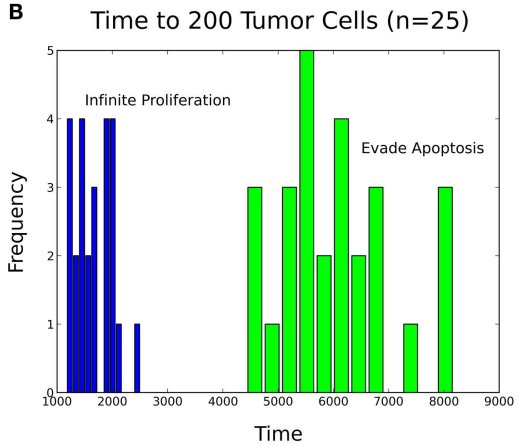
Deregulated growth in the population model was simulated by allowing one cell to either adopt a strategy of constitutive proliferation or evasion of apoptosis, which was then passed on to its offspring. The time it took for the mutant cell to produce 200 offspring is summarized in **Figure 4B**. Taxol is modeled as a diffusing agent that kills a cell when it tries to proliferate, thus targeting both mutant and wild type cells.

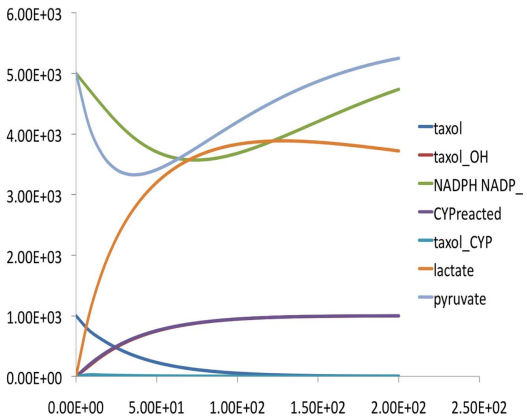
In the liver model, Taxol is metabolized and causes the build up of lactate, the main source of Taxol based liver toxicity. A sample trace of this metabolism is depicted in **Figure 4C**. The four possible effects of different dosing schedules for Taxol are depicted in **Figure 4D**. We discovered that it was possible to produce the optimal (3rd) effect in this model.

3.3. NEXT GENERATION OF DYNAMIC MODELS

To receive the best treatment and diagnosis, most cancer patients are willing to undergo invasive procedures to sample tumor







D

Cancer Cells	Normal Cells	Liver
Dead	Dead	Dead
Dead	Alive	Dead
Dead	Alive	Alive
Alive	Alive	Alive

FIGURE 4 | Figures accompanying the composition model. **(A)** Wiring diagram of liver model. **(B)** Time to tumor for different phenotypic aberrations. **(C)** Sample trace of ODE liver model simulation. **(D)** Four possible steady-state outcomes of composite model simulation, with different Taxol delivery schedules.

and metastatic tissue. It will soon be possible to augment the data from these traditional means with non-invasively collected high-coverage DNA and RNA sequencing data, coupled to single-molecule and single-cell analysis with increasingly finer temporal granularity, using next-generation sequencing (NGS) technologies (Wigler, 2012). With such tools, we can study the diversity of individual cells in populations of cells.

Temporal models of dynamic biological processes, multi-scale and multi-level abstractions, and the analytical tools based on statistics and temporal logic could provide much sharper tools to address the challenges that these data pose.

A powerful approach is to build statistical analysis tools upon simple phenomenological models – in which the data themselves are viewed dynamically in terms of "snapshots in the temporal chain of events," each event coordinated collectively by different cell-types in different cell-states (Ramakrishnan et al., 2010). With these logical analyses, inferred temporal logic invariants reveal various causal linkages between events that were earlier indistinguishable from mere correlations – recorded by the data and redescribed by the phenomenological models (Kleinberg and Mishra, 2009). The next steps in system biology's progress in the biomedical arena would be improving our current understanding of mechanisms described by pathways, metabolic processes, signaling, etc., and in seeking to intervene in the components of these mechanisms to modify the system's behavior (Olde Loohuis et al., 2014).

Success of such a program hinges on how we address the following questions (many of them partially solved):

(1) When can two models be considered "the same?"
(2) When can one model be considered an abstraction of another?
(3) When can one model be considered to approximate another model?
(4) How can several models be combined to provide larger models, either containing multiple subsystems or at multiple scales?

3.4. MULTI-SCALE MODELS

Computer science research has addressed several of these questions. Model equivalence provides tools such as simulation and bisimulation for defining and algorithmically testing whether two models represent the same trajectories of events. Model approximation extends these tools by allowing essentially equivalent models to be slightly different due to stochasticity or granularity. Model composition provides tools for combining disparate models both accurately and efficiently, by considering the models' relevant interactions and independencies, respectively. This is tightly related to hierarchy and decomposition, which provide structures to efficiently represent, store, and execute composite models. Finally, evolution, while not inherently a computer science concept, is essential to understanding and modeling population effects.

Section 3.2 provides as example of a multi-scale, composite model of a tumor cell population, liver metabolism, and the simultaneous effects of Taxol treatment on both. Conceptually, this example helps illustrate the notion of combining two disparate types of models to study the emergent properties of the larger system. Practically, this model can serve as the basis to study the effects of various chemo-therapeutic dosage regimens, such as metronomic therapy, on the tumor and other organ systems.

3.5. REVIEW OF NEXT-GENERATION TOOLS

We take the time here to illustrate hypothetical sequences of abstractions and to describe the types of tools that will be necessary in analyzing large scale dynamical models in modern systems biology.

3.5.1. Illustration of model abstractions

Systems biology aims to describe large systems instead of isolated parts. It would be impractical to attempt to attain this goal with one model type, because different types of models lend themselves to modeling different types of systems, at different scales. To illustrate how the proposed approach permits a variety of modeling techniques to be applied to a single problem, we use a sequence of abstractions in which we can view the same system in many different ways. We start with a rule-based specification of a reactive biochemical system, which can be executed in a variety of ways.

For instance, the specification can be transformed into an executable model that is either deterministic or probabilistic, as illustrated by the left (deterministic) and right (stochastic) sides of **Figure 5**.

First, in order to model the system using the sequence of deterministic models on the left-hand side of **Figure 5**, we start by assuming mass action kinetics. This permits tracking the average behavior of the chemical species using an ordinary differential equations model (see Section 2.1.2 for an example). This conversion is standard and well documented for rule-based models (Blinov et al., 2004; Danos et al., 2010). If the ODE dynamics exhibit sharp transitions among several regimes, each of which can be described by simpler ODE models, we abstract the ODE model into a hybrid automaton (HA). The HA contains discrete modes, each of whose dynamics is modeled by a simpler ODE model. This transformation has been defined and used in Alfieri et al. (2011), Grosu et al. (2011), and Noel et al. (2011). Alternatively, the ODE dynamics may be very steep. That is, molecular concentrations are either high or low but do not dwell in the intermediate states for long. In this case, the ODE model can be transformed into a Boolean network, in which there are activating edges from x_1 to x_2 if $\frac{dx_2}{dt}$ is positively related to the concentration of x_1 and inhibitory edges if it is negatively related.

Next, consider the probabilistic side of the figure. Again starting with a rule-based model, it is appropriate to use a probabilistic model if the concentrations of species are low and stochastic effects could have significant effects on the overall dynamics. In this case, we transform the rule-based model into a stochastic model that simulates sampling from the chemical master equation (Danos et al., 2007; Smith et al., 2012) through a set of reactions and reaction rates.

Under the assumptions of a well-mixed and homogeneous system, this model can be simulated as a CTMC using the kinetic Monte Carlo (KMC) algorithm. To improve efficiency, at some cost to accuracy, we can transform this stochastic model into a dynamic Bayesian network (DBN). Through careful sampling, we

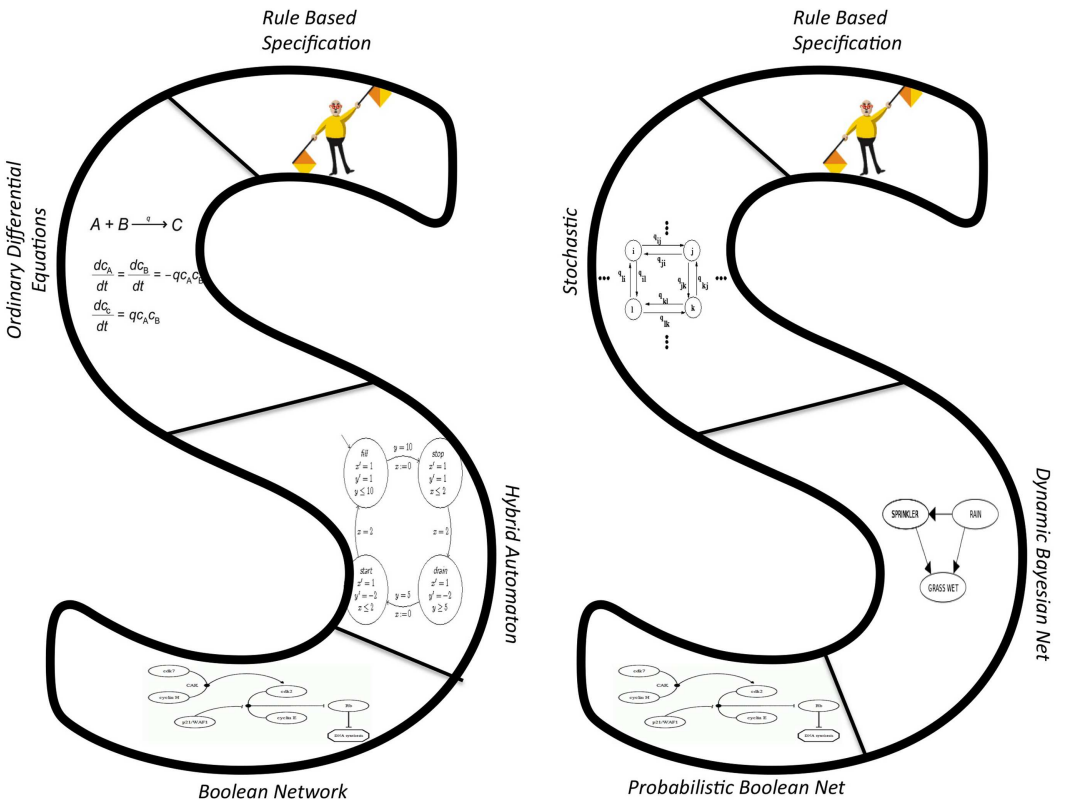


FIGURE 5 | Sequences of abstractions from rule-based specifications to deterministic (right) and stochastic (left) models.

can then find the distribution of reagent concentrations varying over time that is formalized by the DBN. If we further find that variable values tend to vacillate between a range of high and low values, we can model the DBN as a probabilistic Boolean network (PBN). Lähdesmäki et al. (2006) have explored the relationship between these two models and showed how they can represent similar systems.

3.5.2. Tools description

3.5.2.1. Model equivalence. When can we consider two models to be the same, so that we can justify substituting one kind of models by another? In what sense are they to be considered equivalent? What does this mean if models are stochastic – do they produce just the same aggregate results, such as averages, or must distributions be the same?

A very powerful concept for deterministic models is that of bisimulation (Desharnais et al., 2004; Danos et al., 2006), which was first developed in the context of reasoning about complex computational systems, such as an operating system. A bisimulation defines an equivalence between two models in terms of the simulation events (see **Figure 6**). Two models are thus equivalent if they can exhibit identical sequences of events for all possible simulations.

These ideas are usable even when the two models are of disparate types. To make this precise, define a trajectory as a set of states/observations produced by the simulation of a model. Two

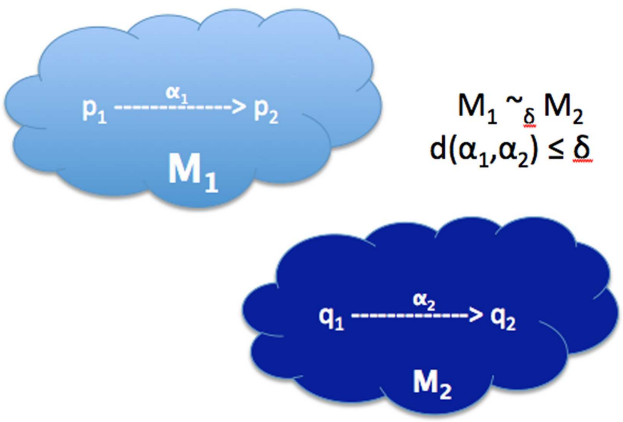


FIGURE 6 | Approximate bisimulation equivalence. α_1 and α_2 are trajectories of simulations in $\mathcal{M}_1$ and $\mathcal{M}_2$, respectively. $d(\alpha_1, \alpha_2)$ is the distance metric between the trajectories.

models ($\mathcal{M}_1$ and $\mathcal{M}_2$) are bisimilar ($\mathcal{M}_1 \sim \mathcal{M}_2$), if for every simulation in $\mathcal{M}_1$, there is a simulation in $\mathcal{M}_2$ that produces an equivalent trajectory, and vice versa. For this situation, a notion of bisimulation is required that can be used to ask if a model of apoptosis in one organism may be bisimilar to an analogous model in another organisms, even though the states in the two distinct

organisms are described in terms of behavior of two different sets of genes, related by gene-orthology.

3.5.2.2. Model approximations. Bisimulation equivalence is often too strong a constraint, and often approximate bisimulation equivalence (ABE) is sufficient for applications (Girard and Pappas, 2005). In ABE, we assume that the simulation trajectories for both models $\mathcal{M}_1$ and $\mathcal{M}_2$ lie in a single metric space (X, d). The models $\mathcal{M}_1$ and $\mathcal{M}_2$ are said to be approximately bisimulation equivalent up to precision δ if the corresponding simulation outputs are individually separated by distance at most δ. In this case, we write $\mathcal{M}_1 \sim_\delta \mathcal{M}_2$.

3.5.2.3. Model compositions. Given a pair of models of interacting systems, we may wish to create a model that captures the essence of the combined system. Although, intuitively this is a rather simple concept, a good formal definition is difficult, as state-reachability and temporal dynamics interact in a complex way. One approach that has been used works by first defining a composition operation using a suitable heuristic and then showing that the resulting model is a "good" approximation of the real system. In a typical definition, the state of the composite model is described by a combination of the variables in its children's states. If these variables do not overlap, the simulation of the composite model is trivial: the sub-models run in parallel, and the composite is their Cartesian product. When they share variables (e.g., crosstalk in a signaling network), parallel simulation may fail, as the flow of one may depend on variables in the other. A naïve approach would simulate both for ε time, implicitly assuming that the variables change only infinitesimally, update the flows of both, and repeat – which, however, is infeasible for continuous flows, as ε would have to approach 0 for accurate results; discrete flows are less problematic.

Consider three types of dependencies between a variable and a flow in different models.

I. A close interaction: a small change in the variable causes a significant change in the flow.
II. A remote interaction: a large change in the variable is required to cause a significant change in the flow.
III. No (empty) interaction: no amount of change in the variable will affect the flow.

Clearly, no interactions would result in the trivial composition. The presence of one close interaction creates the "ε dilemma" discussed above. Thus, any partition that introduces such "ε dilemmas" is to be minimized. On the other hand, if all interactions between models were remote, we could define a guard condition for each interaction that is triggered when a variable changes sufficiently to require an update in its corresponding flow. The guard conditions constitute a set of discrete, timed events that are typically simulated using kinetic Monte Carlo.

3.5.2.4. Hierarchy and decomposition. We envision a large systems biology model as a hierarchical combination of smaller models. Thus, one can formulate the hierarchy as a tree structure (see **Figure 7**). The leaves (blue) represent atomic models that are well-defined outside of the compositional framework. The root (green) represents the full meta-model, and the other internal nodes (red) are partial-compositions of other models. Each node in this tree represents a complete executable model, defined by a state and a flow (see Section 2.2).

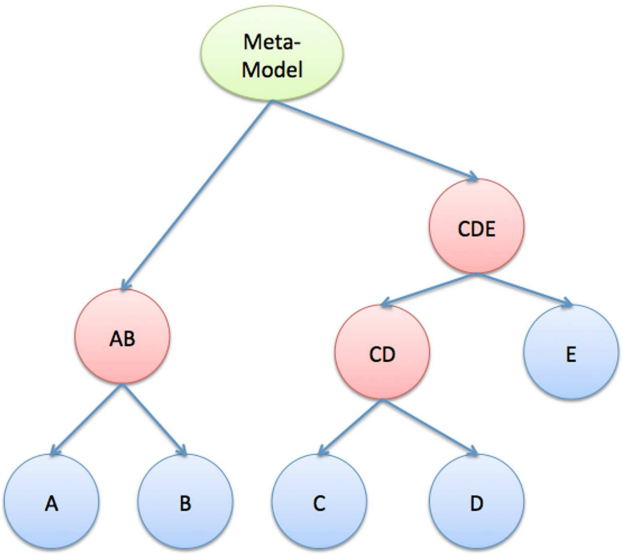


FIGURE 7 | Hierarchical composition. Tree structure that describes the hierarchical relationships between atomic models, the meta-model, and partial-composition models.

Clearly, to ensure that we can efficiently and accurately simulate such a large systems biology model, it is often required that it has a modular structure (**Figure 7**), in which intra-modular dynamics can be of types I, II, or III, but inter-modular dynamics can only be of type II and III. This requirement is not as stringent as it may seem at first; multi-cellular biological systems are naturally organized in this way. Intracellular dynamics are separated from one another by cell membranes but connected via slower acting, intercellular signals. Solid organs have their own internal dynamics and share "variables" via hormones and neuro-transmitters. Even gene and protein interactions in regulatory and metabolic systems can be decomposed into pathways that interact with each other through weak cross-talks (see **Figure 1** for an example of crosstalk interaction between pathways).

To this end, we propose not only a formal structure in which to specify and simulate multi-scale models in systems biology but also a philosophy of modularity that follows the structures established by nature.

3.5.2.5. Evolution. While "proximate" explanations in biology can be presented using mechanistic models of the kind we have described earlier, "ultimate" explanations are impossible except in the light of evolution, where the dynamics is to be understood in terms of multiple strategic agents. One powerful use of abstraction – built from approximations and compositions – is in allowing a translation from mechanistic models, in which the internal state is described in great details, to strategic models, in

which the input and output behavior is characterized in terms of some less detailed internal states (phenomenological states). This shift allows us to connect mechanistic models to a burgeoning class of systems biology models that are based on game theory.

4. DISCUSSION

From a pragmatic perspective, the study of cancer should aim to exploit patient data at all levels in drug discovery and therapy design. Analysis of data in the quantity currently available, with granularity at the level of a specific cell, requires more refined techniques than have been previously available. However, recent developments in modeling suggest that systems biology is primed to take the lead in this investigation, which necessitates the incorporation of large amounts of data into integrated models of multiple simultaneous processes operating at different scales.

Specifically, therapy design requires accurate, tractable progression models that track the evolution of pathway activity and genomic alterations that characterize various stages of the disease over time. To this end, we need rigorous notions of abstraction that allow us to retain detailed pathway information in simpler models. Therapy design must also take into account the toxicity of chemotherapy and budget constraints (e.g., the ones imposed by the monetary cost incurred by the healthcare system). Our approach requires integration among highly disparate models, and to this end, we need a rigorous way to simulate models simultaneously at different scales.

Finally, modern analytical tools will play a crucial role in the construction and application of these abstractions, hierarchical composite models. For instance, we need model checking to systematically characterize cancer phenotypes in terms of temporal properties. Also, sensitivity analysis is indispensable for identifying the key targets of signaling networks for drug discovery.

In summary, we need ways to simulate and analyze models efficiently. We also need to formalize model abstraction and to characterize its properties. These problems have been studied extensively in computational research, such as rate-distortion theory and bisimulation equivalence, and could now meaningfully be adapted to meet the needs of biological systems. Most importantly, we need a means to personalize complex heterogeneous models to patients, in order to devise the most effective therapies for each patient.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under Grant No. 0926200.

REFERENCES

Albeck, J. G., Burke, J. M., Spencer, S. L., Lauffenburger, D. A., and Sorger, P. K. (2008). Modeling a snap-action, variable-delay switch controlling extrinsic cell death. *PLoS Biol.* 6:e299. doi:10.1371/journal.pbio.0060299

Aldinucci, M., Bracciali, A., Lio, P., Sorathiya, A., and Torquati, M. (2011). "StochKit-FF: efficient systems biology on multicore architectures," in *Euro-Par 2010 Parallel Processing Workshops*, eds M. R. Guarracino, F. Vivien, J. L. Träff, et al. (Berlin, Heidelberg: Springer), 167–175.

Alfieri, R., Bartocci, E., Merelli, E., and Milanesi, L. (2011). Modeling the cell cycle: from deterministic models to hybrid systems. *Biosystems* 105, 34–40. doi:10.1016/j.biosystems.2011.03.002

Alur, R., Belta, C., Ivancic, F., Kumar, V., Mintz, M., Pappas, G. J., et al. (2001). "Hybrid modeling and simulation of biomolecular networks," in *Hybrid Systems: Computation and Control*, eds M. D. Di Benedetto and A. Sangiovanni-Vincentelli (Berlin, Heidelberg: Springer), 19–32.

Antoniotti, M., Mishra, B., Piazza, C., Policriti, A., and Simeoni, M. (2003a). "Modeling cellular behavior with hybrid automata: bisimulation and collapsing," in *Computational Methods in Systems Biology*, ed. C. Priami (Berlin, Heidelberg: Springer), 57–74.

Antoniotti, M., Policriti, A., Ugel, N., and Mishra, B. (2003b). Model building and model checking for biochemical processes. *Cell Biochem. Biophys.* 38, 271–286. doi:10.1385/CBB:38:3:271

Bandini, S., Mauri, G., and Serra, R. (2001). Cellular automata: from a theoretical parallel computational model to its application to complex systems. *Parallel Comput.* 27, 539–553. doi:10.1016/S0167-8191(00)00109-5

Barton, P. I., and Lee, C. K. (2002). Modeling, simulation, sensitivity analysis, and optimization of hybrid systems. *ACM Trans. Model. Comput. Simul.* 12, 256–289. doi:10.1145/643120.643122

Bengtsson, J., Larsen, K., Larsson, F., Pettersson, P., and Yi, W. (1996). *UPPAALÑa Tool Suite for Automatic Verification of Real-Time Systems.* Berlin, Heidelberg: Springer.

Blinov, M. L., Faeder, J. R., Goldstein, B., and Hlavacek, W. S. (2004). Bionetgen: software for rule-based modeling of signal transduction based on the interactions of molecular domains. *Bioinformatics* 20, 3289–3291. doi:10.1093/bioinformatics/bth378

Bornholdt, S. (2008). Boolean network models of cellular regulation: prospects and limitations. *J. R. Soc. Interface* 5(Suppl. 1), S85–S94. doi:10.1098/rsif.2008.0132.focus

Bortolussi, L., and Policriti, A. (2010). Hybrid dynamics of stochastic programs. *Theor. Comp. Sci.* 411, 2052–2077. doi:10.1016/j.tcs.2010.02.008

Brim, L., Fabriková, J., Drazan, S., and Safranek, D. (2011). "Reachability in biochemical dynamical systems by quantitative discrete approximation," in *Proceedings of Third International Workshop on Computational Models for Cell Processes, CompMod 2011* (EPTCS), Vol. 67, 97–112.

Campagna, D., and Piazza, C. (2010). Hybrid automata, reachability, and systems biology. *Theor. Comp. Sci.* 411, 2037–2051. doi:10.1016/j.tcs.2009.12.015

Cartwright, N. (2004). Causation: one word, many things. *Philos. Sci.* 71, 805–820. doi:10.1086/426771

Chaouiya, C. (2007). Petri net modelling of biological networks. *Brief. Bioinformatics* 8, 210–219. doi:10.1093/bib/bbm029

Clarke, E., and Mishra, B. (1984). "Automatic verification of asynchronous circuits," in *Logics of Programs*, eds E. Clarke and D. Kozen (Berlin, Heidelberg: Springer), 101–115.

Clarke, E. M., Faeder, J. R., Langmead, C. J., Harris, L. A., Jha, S. K., and Legay, A. (2008). "Statistical model checking in biolab: applications to the automated analysis of T-cell receptor signaling pathway," in *Computational Methods in Systems Biology*, eds M. Heiner and A. M. Uhrmacher (Berlin, Heidelberg: Springer), 231–250.

Cozman, F. (1997). "Robustness analysis of Bayesian networks with local convex sets of distributions," in *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence* (San Fransisco: Morgan Kaufmann Publishers Inc), 108–115.

Danos, V., Desharnais, J., Laviolette, F., and Panangaden, P. (2006). Bisimulation and cocongruence for probabilistic systems. *Inform. Comput.* 204, 503–523. doi:10.1016/j.ic.2005.02.004

Danos, V., Feret, J., Fontana, W., Harmer, R., Hayman, J., Krivine, J., et al. (2012). Graphs, rewriting and pathway reconstruction for rule-based models. *FSTTCS* 18, 276–288. doi:10.4230/LIPIcs.FSTTCS.2012.276

Danos, V., Feret, J., Fontana, W., Harmer, R., and Krivine, J. (2007). "Rule-based modelling of cellular signalling," in *CONCUR 2007-Concurrency Theory*, eds L. Caires and V. T. Vasconcelos (Berlin, Heidelberg: Springer), 17–41.

Danos, V., Feret, J., Fontana, W., Harmer, R., and Krivine, J. (2010). "Abstracting the differential semantics of rule-based models: exact and automated model reduction," in *Logic in Computer Science (LICS), 2010 25th Annual IEEE Symposium on* (New York: IEEE), 362–381.

Das, S., Sikdar, B. K., and Chaudhuri, P. P. (2004). "Characterization of reachable/nonreachable cellular automata states," in *Cellular Automata*, eds P. M. A. Sloot, B. Chopard and A. G. Hoekstra (Berlin, Heidelberg: Springer), 813–822.

Delpu, Y., Hanoun, N., Lulka, H., Sicard, F., Selves, J., Buscail, L., et al. (2011). Genetic and epigenetic alterations in pancreatic carcinogenesis. *Curr. Genomics* 12, 15. doi:10.2174/138920211794520132

Desharnais, J., Gupta, V., Jagadeesan, R., and Panangaden, P. (2004). Metrics for labelled markov processes. *Theor. Comp. Sci.* 318, 323–354. doi:10.1016/j.tcs.2003.09.013

Didier, F., Henzinger, T. A., Mateescu, M., and Wolf, V. (2009). "Approximation of event probabilities in noisy cellular processes," in *Computational Methods in Systems Biology*, eds P. Degano and R. Gorrieri (Berlin, Heidelberg: Springer), 173–188.

Dimitrova, E., García-Puente, L. D., Hinkelmann, F., Jarrah, A. S., Laubenbacher, R., Stigler, B., et al. (2011). Parameter estimation for Boolean models of biological networks. *Theor. Comp. Sci.* 412, 2816–2826. doi:10.1016/j.jtbi.2010.10.003

Donaldson, R., and Gilbert, D. (2008). "A model checking approach to the parameter estimation of biochemical pathways," in *Computational Methods in Systems Biology*, eds M. Heiner and A. M. Uhrmacher (Berlin, Heidelberg: Springer), 269–287.

Donzé, A., Clermont, G., and Langmead, C. J. (2010). Parameter synthesis in nonlinear dynamical systems: application to systems biology. *J. Comput. Biol.* 17, 325–336. doi:10.1089/cmb.2009.0172

Dytham, C. (1995). *The Effect of Habitat Destruction Pattern on Species Persistence: A Cellular Model*. Hoboken: Oikos, 340–344.

Fauré, A., Naldi, A., Chaouiya, C., and Thieffry, D. (2006). Dynamical analysis of a generic Boolean model for the control of the mammalian cell cycle. *Bioinformatics* 22, e124–e131. doi:10.1093/bioinformatics/btl210

Feret, J., Danos, V., Krivine, J., Harmer, R., and Fontana, W. (2009). Internal coarse-graining of molecular systems. *Proc. Natl. Acad. Sci. U.S.A.* 106, 6453–6458. doi:10.1073/pnas.0809908106

Figueirêdo, P., Coutinho, S., and Zorzenon dos Santos, R. (2008). Robustness of a cellular automata model for the HIV infection. *Physica A* 387, 6545–6552. doi:10.1016/j.physa.2008.07.011

Fischer, D., and Kaiser, Ł (2011). "Model checking the quantitative μ-calculus on linear hybrid systems," in *Automata, Languages and Programming*, eds L. Aceto, M. Henzinger, and J. Sgall (Berlin, Heidelberg: Springer), 404–415.

Fränzle, M., and Herde, C. (2007). Hysat: an efficient proof engine for bounded model checking of hybrid systems. *Formal Methods Syst. Design* 30, 179–198. doi:10.1007/s10703-006-0031-0

Friedman, N. (2004). Inferring cellular networks using probabilistic graphical models. *Science* 303, 799–805. doi:10.1126/science.1094068

Friedman, N., and Koller, D. (2003). Being Bayesian about network structure. a Bayesian approach to structure discovery in Bayesian networks. *Mach. Learn.* 50, 95–125. doi:10.1023/A:1020249912095

Garmaroudi, F. S., Marchant, D., Si, X., Khalili, A., Bashashati, A., Wong, B. W., et al. (2010). Pairwise network mechanisms in the host signaling response to coxsackievirus b3 infection. *Proc. Natl. Acad. Sci. U.S.A.* 107, 17053–17058. doi:10.1073/pnas.1006478107

Ghosh, R., Tiwari, A., and Tomlin, C. (2003). "Automated symbolic reachability analysis; with application to delta-notch signaling automata," in *Hybrid Systems: Computation and Control*, eds O. Maler and A. Pnueli (Berlin, Heidelberg: Springer), 233–248.

Ghosh, R., and Tomlin, C. J. (2001). "Lateral inhibition through delta-notch signaling: a piecewise affine hybrid model," in *Hybrid Systems: Computation and Control*, eds M. D. Di Benedetto and A. Sangiovanni-Vincentelli (Berlin, Heidelberg: Springer), 232–246.

Girard, A., and Pappas, G. J. (2005). "Approximate bisimulations for nonlinear dynamical systems," in *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC'05. 44th IEEE Conference on* (New York: IEEE), 684–689.

Gong, H., Wang, Q., Zuliani, P., Faeder, J. R., Lotze, M., and Clarke, E. (2011a). *Symbolic Model Checking of Signaling Pathways in Pancreatic Cancer*. Winona, MN: BICoB, 245.

Gong, H., Zuliani, P., and Clarke, E. M. (2011b). "Model checking of a diabetes-cancer model," in *AIP Conference Proceedings*, Vol. 1371, (Melville, NY: AIP Publishing), 234.

Gong, H., Zuliani, P., Wang, Q., and Clarke, E. M. (2011c). "Formal analysis for logical models of pancreatic cancer," in *Decision and Control and European Control Conference (CDC-ECC), 2011 50th IEEE Conference on* (New York: IEEE), 4855–4860.

Gong, H., Zuliani, P., Komuravelli, A., Faeder, J. R., and Clarke, E. M. (2010). Analysis and verification of the hmgb1 signaling pathway. *BMC Bioinformatics* 11(Suppl. 7):S10. doi:10.1186/1471-2105-11-S7-S10

Grosu, R., Batt, G., Fenton, F. H., Glimm, J., Le Guernic, C., Smolka, S. A., et al. (2011). "From cardiac cells to genetic regulatory networks," in *Computer Aided Verification*, eds G. Gopalakrishnan and S. Qadeer (Berlin, Heidelberg: Springer), 396–411.

Guengerich, F. P., and Johnson, W. W. (1997). Kinetics of ferric cytochrome p450 reduction by nadph-cytochrome p450 reductase: rapid reduction in the absence of substrate and variations among cytochrome p450 systems. *Biochemistry* 36, 14741–14750. doi:10.1021/bi9719399

Gunawardena, J. (2010). Models in systems biology: the parameter problem and the meanings of robustness. *Elem. Comput. Syst. Biol.* 1, 21–48. doi:10.1002/9780470556757.ch2

Hagiya, M., Takahashi, K., Yamamoto, M., and Sato, T. (2004). "Analysis of synchronous and asynchronous cellular automata using abstraction by temporal logic," in *Functional and Logic Programming*, eds Y. Kameyama and P. J. Stuckey (Berlin, Heidelberg: Springer), 7–21.

Henzinger, T. A., Ho, P.-H., and Wong-Toi, H. (1997). "Hytech: a model checker for hybrid systems," in *Computer Aided Verification*, ed. O. Grumberg (Berlin, Heidelberg: Springer), 460–463.

Hoffmann, A., Levchenko, A., Scott, M. L., and Baltimore, D. (2002). The IκB-NF-κB signaling module: temporal control and selective gene activation. *Science* 298, 1241–1245. doi:10.1126/science.1071914

Holmes, F. A., Walters, R. S., Theriault, R. L., Buzdar, A. U., Frye, D. K., Hortobagyi, G. N., et al. (1991). Phase II trial of Taxol, an active drug in the treatment of metastatic breast cancer. *J. Natl. Cancer Inst.* 83, 1797–1805. doi:10.1093/jnci/83.24.1797-a

Horvath, A. (2012). *The Monte Carlo Em Method for the Parameter Estimation of Biological Models*. Amsterdam: MMB & DFT, 37.

Hume, D. (1902). *Enquiries Concerning the Human Understanding: And Concerning the Principles of Morals*. Gloucestershire: Clarendon Press.

Ihekwaba, A., Broomhead, D., Grimley, R., Benson, N., and Kell, D. (2004). Sensitivity analysis of parameters controlling oscillatory signalling in the NF-κB pathway: the roles of IKK and IκBα. *Syst. Biol.* 1, 93–103. doi:10.1049/sb:20045009

Iyengar, R., Zhao, S., Chung, S.-W., Mager, D. E., and Gallo, J. M. (2012). Merging systems biology with pharmacodynamics. *Sci. Transl. Med.* 4, 126s7. doi:10.1126/scitranslmed.3003563

Janes, K. A., Kelly, J. R., Gaudet, S., Albeck, J. G., Sorger, P. K., and Lauffenburger, D. A. (2004). Cue-signal-response analysis of TNF-induced apoptosis by partial least squares regression of dynamic multivariate data. *J. Comput. Biol.* 11, 544–561. doi:10.1089/cmb.2004.11.544

Jones, S., Zhang, X., Parsons, D. W., Lin, J. C.-H., Leary, R. J., Angenendt, P., et al. (2008). Core signaling pathways in human pancreatic cancers revealed by global genomic analyses. *Science* 321, 1801–1806. doi:10.1126/science.1164368

Jordan, C. T., Guzman, M. L., and Noble, M. (2006). Cancer stem cells. *N. Eng. J. Med.* 355, 1253–1261. doi:10.1056/NEJMra061808

Kang, R., Tang, D., Schapiro, N. E., Livesey, K. M., Farkas, A., Loughran, P., et al. (2009). The receptor for advanced glycation end products (rage) sustains autophagy and limits apoptosis, promoting pancreatic tumor cell survival. *Cell Death Differ.* 17, 666–676. doi:10.1038/cdd.2009.149

Kim, J., Bates, D. G., Postlethwaite, I., Ma, L., and Iglesias, P. A. (2006). Robustness analysis of biochemical network models. *IEE Proc. Syst. Biol.* 153, 96–104. doi:10.1049/ip-syb:20050024

Kleinberg, S., and Hripcsak, G. (2011). A review of causal inference for biomedical informatics. *J. Biomed. Inform.* 44, 1102–1112. doi:10.1016/j.jbi.2011.07.001

Kleinberg, S., and Mishra, B. (2009). "The temporal logic of causal structures," in *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence* (Corvalis, OR: AUAI Press), 303–312.

Kobayashi, K., and Hiraishi, K. (2010). "Reachability analysis of probabilistic Boolean networks using model checking," in *SICE Annual Conference 2010, Proceedings of* (New York: IEEE), 829–832.

Kobayashi, K., and Hiraishi, K. (2011). "A symbolic approach to probabilistic verification of Boolean networks," in *IECON 2011-37th Annual Conference on IEEE Industrial Electronics Society* (New York: IEEE), 3764–3769.

Kwiatkowska, M., Norman, G., and Parker, D. (2004). Probabilistic symbolic model checking with prism: a hybrid approach. *Int. J. Software Tools Technol. Trans.* 6, 128–142. doi:10.1007/s10009-004-0140-2

Lähdesmäki, H., Hautaniemi, S., Shmulevich, I., and Yli-Harja, O. (2006). Relationships between probabilistic Boolean networks and dynamic Bayesian networks as models of gene regulatory networks. *Signal Processing* 86, 814–834. doi:10.1016/j.sigpro.2005.06.008

Langmead, C. J. (2009). "Generalized queries and Bayesian statistical model checking in dynamic Bayesian networks: application to personalized medicine," in *8th Annual International Conference on Computational Systems Bioinformatics*. Woodside: Life Sciences Society, 201–212.

Langmead, C. J., Jha, S. K., and Clarke, E. M. (2006). *Temporal-logics as Query Languages for Dynamic Bayesian Networks: Application to D. Melanogaster Embryo Development*. Carnegie Mellon University.

Li, Z., and Chan, C. (2004). Inferring pathways and networks with a Bayesian framework. *FASEB J.* 18, 746–748. doi:10.1096/fj.03-0475fje

Lincoln, P., and Tiwari, A. (2004). "Symbolic systems biology: hybrid modeling and analysis of biological networks," in *Hybrid Systems: Computation and Control*, eds R. Alur and G. J. Pappas (Berlin, Heidelberg: Springer), 660–672.

Loes, O. L., Giulio, C., Alex, G., Daniele, R., Giancarlo, M., Marco, A., et al. (2013). Inferring causal models of cancer progression with a shrinkage estimator and probability raising. arXiv:1311.6293.

Manzano, A., Roig, T., Bermudez, J., and Bartrons, R. (1996). Effects of Taxol on isolated rat hepatocyte metabolism. *Am. J. Physiol.* 271, C1957–C1962.

Marjanovic, N. D., Weinberg, R. A., and Chaffer, C. L. (2013). Cell plasticity and heterogeneity in cancer. *Clin. Chem.* 59, 168–179. doi:10.1373/clinchem.2012.184655

Mukherjee, S., and Speed, T. P. (2008). Network inference using informative priors. *Proc. Natl. Acad. Sci. U.S.A.* 105, 14313–14318. doi:10.1073/pnas.0802272105

Müssel, C., Hopfensitz, M., and Kestler, H. A. (2010). BoolNetÑan R package for generation, reconstruction and analysis of Boolean networks. *Bioinformatics* 26, 1378–1380. doi:10.1093/bioinformatics/btq124

Narasimhan, S., and Biswas, G. (2007). Model-based diagnosis of hybrid systems. *IEEE Trans. Syst. Man Cybern. A: Syst. Hum.* 37, 348–361. doi:10.1109/TSMCA.2007.893487

Nielsen, M., Plotkin, G., and Winskel, G. (1981). Petri nets, event structures and domains, part I. *Theor. Comp. Sci.* 13, 85–108. doi:10.1016/0304-3975(81)90112-2

Noel, V., Vakulenko, S., and Radulescu, O. (2011). "Algorithm for identification of piecewise smooth hybrid systems: application to eukaryotic cell cycle regulation," in *Algorithms in Bioinformatics*, eds T. M. Przytycka and M.-F. Sagot (Berlin, Heidelberg: Springer), 225–236.

Olde Loohuis, L., Witzel, A., and Mishra, B. (2014). Cancer hybrid automata: model, beliefs and therapy. *J. Inform. Comput.* 236, 68–86. doi:10.1016/j.ic.2014.01.013

Paulevé, L., Andrieux, G., and Koeppl, H. (2013). "Under-approximating cut sets for reachability in large scale automata networks," in *Computer Aided Verification*, eds N. Sharygina and H. Veith (Berlin, Heidelberg: Springer), 69–84.

Pearl, J. (2000). *Causality: Models, Reasoning and Inference*, Vol. 29. Cambridge: Cambridge University Press.

Pe'er, D. (2005). Bayesian network analysis of signaling networks: a primer. *Sci. Signal.* 2005, l4. doi:10.1126/stke.2812005pl4

Qian, X., and Dougherty, E. R. (2009). On the long-run sensitivity of probabilistic Boolean networks. *J. Theor. Biol.* 257, 560–577. doi:10.1016/j.jtbi.2008.12.023

Radulescu, O., Gorban, A. N., Zinovyev, A., and Noel, V. (2012). Reduction of dynamical biochemical reaction networks in computational biology. *Front. Genet.* 3:131. doi:10.3389/fgene.2012.00131

Rahman, A., Korzekwa, K. R., Grogan, J., Gonzalez, F. J., and Harris, J. W. (1994). Selective biotransformation of Taxol to 6α-hydroxytaxol by human cytochrome p450 2c8. *Cancer Res.* 54, 5543–5546.

Ramakrishnan, N., Tadepalli, S., Watson, L. T., Helm, R. F., Antoniotti, M., and Mishra, B. (2010). Reverse engineering dynamic temporal models of biological processes and their relationships. *Proc. Natl. Acad. Sci. U.S.A.* 107, 12511–12516. doi:10.1073/pnas.1006283107

Reeves, G. T., Muratov, C. B., Schüpbach, T., and Shvartsman, S. Y. (2006). Quantitative models of developmental pattern formation. *Dev. Cell* 11, 289–300. doi:10.1016/j.devcel.2006.08.006

Reya, T., Morrison, S. J., Clarke, M. F., and Weissman, I. L. (2001). Stem cells, cancer, and cancer stem cells. *Nature* 414, 105–111. doi:10.1038/35102167

Ryu, S., Lin, S.-C., Ugel, N., Antoniotti, M., and Mishra, B. (2008). Mathematical modeling of the formation of apoptosome in intrinsic pathway of apoptosis. *Syst. Synth. Biol.* 2, 49–66. doi:10.1007/s11693-009-9022-y

Saez-Rodriguez, J., Simeoni, L., Lindquist, J. A., Hemenway, R., Bommhardt, U., Arndt, B., et al. (2007). A logical model provides insights into T cell receptor signaling. *PLoS Comput. Biol.* 3:e163. doi:10.1371/journal.pcbi.0030163

Sandmann, W. (2007). "Simultaneous stochastic simulation of multiple perturbations in biological network models," in *Computational Methods in Systems Biology*, eds M. Calder and S. Gilmore (Berlin, Heidelberg: Springer), 15–31.

Sandmann, W., and Wolf, V. (2008). "Computational probability for systems biology," in *Formal Methods in Systems Biology*, ed. J. Fisher (Berlin, Heidelberg: Springer), 33–47.

Sarkar, A. X., Christini, D. J., and Sobie, E. A. (2012). Exploiting mathematical models to illuminate electrophysiological variability between individuals. *J. Physiol.* 590, 2555–2567. doi:10.1113/jphysiol.2011.223313

Sarkar, A. X., and Sobie, E. A. (2010). Regression analysis for constraining free parameters in electrophysiological models of cardiac cells. *PLoS Comput. Biol.* 6:e1000914. doi:10.1371/journal.pcbi.1000914

Shackleton, M., Quintana, E., Fearon, E. R., and Morrison, S. J. (2009). Heterogeneity in cancer: cancer stem cells versus clonal evolution. *Cell* 138, 822–829. doi:10.1016/j.cell.2009.08.017

Shmulevich, I., Gluhovsky, I., Hashimoto, R. F., Dougherty, E. R., and Zhang, W. (2003). Steady-state analysis of genetic regulatory networks modelled by probabilistic Boolean networks. *Comp. Funct. Genomics* 4, 601–608. doi:10.1002/cfg.342

Smith, A. M., Xu, W., Sun, Y., Faeder, J. R., and Marai, G. E. (2012). Rulebender: integrated modeling, simulation and visualization for rule-based intracellular biochemistry. *BMC Bioinformatics* 13(Suppl. 8):S3. doi:10.1186/1471-2105-13-S8-S3

Sobie, E. A. (2009). Parameter sensitivity analysis in electrophysiological models using multivariable regression. *Biophys. J.* 96, 1264–1274. doi:10.1016/j.bpj.2008.10.056

Sutner, K. (2002). Cellular automata and intermediate reachability problems. *Fundamenta Informaticae* 52, 249–256.

Sutner, K. (2009). Model checking one-dimensional cellular automata. *J. Cell. Automata* 4, 213–224.

Tamura, T., Sasaki, Y., Nishiwaki, Y., and Saijo, N. (1995). Phase I study of paclitaxel by three-hour infusion: hypotension just after infusion is one of the major dose-limiting toxicities. *Cancer Sci.* 86, 1203–1209. doi:10.1111/j.1349-7006.1995.tb03316.x

Tatyana, L., Dirk, R., and Gennady, B. (2009). Distributed parameter identification for a label-structured cell population dynamics model using CFSE histogram time-series data. *J. Math. Biol.* 59, 581–603. doi:10.1007/s00285-008-0244-5

TCGA. (2013). *The Cancer Genome Atlas – Mission and Goal*. Available at: http://cancergenome.nih.gov/abouttcga/overview/missiongoal

Thomas, R. (1991). Regulatory networks seen as asynchronous automata: a logical description. *J. Theor. Biol.* 153, 1–23. doi:10.1016/j.biosystems.2011.06.006

Thomas, R. (1998). Laws for the dynamics of regulatory networks. *Int. J. Dev. Biol.* 42, 479–485.

Vanden-Eijnden, E. (2006). "Transition path theory," in *Computer Simulations in Condensed Matter Systems: From Materials to Chemical Biology*, Vol. 1, eds M. Ferrario, G. Ciccotti, and K. Binder (Berlin, Heidelberg: Springer), 453–493.

Vikram, V., Wadadekar, Y., Kembhavi, A. K., and Vijayagovindan, G. (2010). Pymorph: automated galaxy structural parameter estimation using python. *Mon. Not. R. Astron. Soc.* 409, 1379–1392. doi:10.1111/j.1365-2966.2010.17426.x

Wang, H., Rish, I., and Ma, S. (2002). Using sensitivity analysis for selective parameter update in Bayesian network learning. *Assoc. Adv. Artif. Intell.* Available from: http://www.aaai.org/Papers/Symposia/Spring/2002/SS-02-03/SS02-03-005.pdf

Wartlick, O., Kicheva, A., and González-Gaitán, M. (2009). Morphogen gradient formation. *Cold Spring Harb. Perspect. Biol.* 1, 1–22. doi:10.1101/cshperspect.a001255

Weinberg, R., and Hanahan, D. (2000). The hallmarks of cancer. *Cell* 100, 57–70. doi:10.1016/S0092-8674(00)81683-9

White, R. W. (1977). Dynamic central place theory: results of a simulation approach. *Geogr. Anal.* 9, 226–243. doi:10.1111/j.1538-4632.1977.tb00576.x

Wigler, M. (2012). Broad applications of single-cell nucleic acid analysis in biomedical research. *Genome Med.* 4, 79. doi:10.1186/gm380

Wilkinson, D. J. (2007). Bayesian methods in bioinformatics and computational systems biology. *Brief. Bioinformatics* 8, 109–116. doi:10.1093/bib/bbm007

Yang, Y., and Lin, H. (2010). "Reachability analysis based model validation in systems biology," in *Cybernetics and Intelligent Systems (CIS), 2010 IEEE Conference on* (New York: IEEE), 14–19.

Yuceer, M., Atasoy, I., and Berber, R. (2008). A software for parameter estimation in dynamic models. *Braz. J. Chem. Eng.* 25, 813–821. doi:10.1590/S0104-66322008000400018

Zhang, J., Baran, J., Cros, A., Guberman, J. M., Haider, S., Hsu, J., et al. (2011). International cancer genome consortium data portal – a one-stop shop for cancer genomics data. 2011:bar026. doi:10.1093/database/bar026

Conflict of Interest Statement: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Received: 27 April 2014; accepted: 18 July 2014; published online: 19 August 2014.
Citation: Korsunsky I, McGovern K, LaGatta T, Olde Loohuis L, Grosso-Applewhite T, Griffeth N and Mishra B (2014) Systems biology of cancer: a challenging expedition for clinical and quantitative biologists. Front. Bioeng. Biotechnol. ***2****:27. doi: 10.3389/fbioe.2014.00027*
This article was submitted to Bioinformatics and Computational Biology, a section of the journal Frontiers in Bioengineering and Biotechnology.

frontiers in Genetics

OPINION
published: 11 September 2015
doi: 10.3389/fgene.2015.00290

Normal vs. Malignant hematopoiesis: the complexity of acute leukemia through systems biology

Jennifer Enciso[1,2], *Luis Mendoza*[3†] *and Rosana Pelayo*[1*†]

[1] Oncology Research Unit, Mexican Institute for Social Security, Mexico City, Mexico, [2] Biochemistry Sciences Program, Universidad Nacional Autónoma de México, Mexico City, Mexico, [3] Departamento de Biología Molecular y Biotecnología, Instituto de Investigaciones Biomédicas, Universidad Nacional Autónoma de México, Mexico City, Mexico

Keywords: acute leukemia, early hematopoiesis, bone marrow, mathematical modeling, regulatory networks, systems biology

OPEN ACCESS

Edited by:
Tim Wilhelm Nattkemper, Bielefeld University, Germany

Reviewed by:
Xiaogang Wu, Institute for Systems Biology, USA
Seth Corey, Northwestern University, USA
Ingmar Glauche, Technische Universität Dresden, Germany

***Correspondence:**
Rosana Pelayo, rosanapelayo@gmail.com

† These authors have contributed equally to this work.

Specialty section:
This article was submitted to Bioinformatics and Computational Biology, a section of the journal Frontiers in Genetics

Received: *12 May 2015*
Accepted: *31 August 2015*
Published: *11 September 2015*

Citation:
Enciso J, Mendoza L and Pelayo R (2015) Normal vs. Malignant hematopoiesis: the complexity of acute leukemia through systems biology. Front. Genet. 6:290. doi: 10.3389/fgene.2015.00290

The Early Stages of Malignant Hematopoiesis: A Multi-cellular, Multi-compartment and Multi-factorial Challenging Study Model

Development of normal hematopoietic cells is an ordered multi-step process, tightly regulated by a complex network of intrinsic factors and microenvironmental cues that control cell fate decisions within the bone marrow (BM) (Pelayo et al., 2012; Purizaca et al., 2012; Boulais and Frenette, 2015). During malignant hematological disorders, including acute leukemias (AL), the uncontrolled differentiation of precursors of the lymphoid or myeloid series sustains tumor growth at the expense of normal blood cell production. Moreover, selection and dominance among leukemic clones occur while competing for niche resources and creating abnormal BM microenvironments that co-participate in the pathobiology of the disease (Colmone et al., 2008; Ayala et al., 2009; Purizaca et al., 2012; Kim et al., 2015; Vilchis-Ordoñez et al., 2015). Thus, due to the complexity and health impact of AL (Gupta et al., 2014), new strategies to better predict cell population dynamics according to genetics, microenvironmental and clinical heterogeneous contexts may contribute to understand its pathobiology and to guide strategies for decreasing overall mortality.

Mathematical modeling has emerged as a powerful tool in biomedical and health research because it enables the simulation of complex biological systems and the efficient generation of testable hypotheses. In recent years, leukemic cell dynamics has been addressed from the novel view of systems biology, resulting in helpful stochastic and deterministic models and providing clearer understanding of the disease by simplification of malignant clonal evolution processes (Vesely et al., 2011; Amir et al., 2013; Paguirigan et al., 2015). However, models fitted to experimental data must strike a balance between simplicity and reality, so that they can bring insights into clinical scenarios.

Here we discuss the importance and challenges of incorporating the BM microenvironment into AL modeling, as a key element that will control the interplay between cell populations and the selective pressure leading to leukemic or normal hematopoiesis progression. By developing integrative tools that better mimic and predict the behavior of heterogeneous and polyclonal cells in the context of abnormal microenvironments within leukemic bone marrow, we may learn about crucial mechanisms co-participating in the etiology and progression of the disease.

Normal vs. Leukemic Clones: Systems Biology in the Study of Acute Leukemia Complexity

Continuous dynamic modeling with differential equations (DEs) has been the most popular systems biology tool for the study of normal and leukemic hematopoiesis. This type of modeling is useful for the time evolving non-linear competition between normal and leukemic cell populations,

considering multiple compartments to simulate different maturation stages or multiclonal behavior (Catlin et al., 2005; Stiehl and Marciniak-Czochra, 2012; MacLean et al., 2013; Stiehl et al., 2014).

Of special interest, theoretical data suggests the existence of an initial "steady state" before the disease development, when co-existence of normal hematopoiesis with a limited number of pre-leukemic cells controls leukemia installation (Rubinow and Lebowitz, 1976; Stiehl and Marciniak-Czochra, 2012; Swaminathan et al., 2015). A sudden change in the homeostatic parameters may induce leukemic cell expansion leading to a progressive decrease of normal hematopoiesis, while perturbation of initial homeostatic state endows malignant cells with self-renewal and proliferation. Accordingly, the model by Rubinow and Lebowitz's on competition advantage of leukemia cells proposed a higher value of their equilibrium number that refers to the maximum population size that can be supported within the niche. If the stop-expansion signal for malignant progenitors is not delivered before the equilibrium number is reached, a signal activating the slow-down of normal cells promotes the expansion of the leukemic population. High equilibrium numbers in leukemic compartments could be biologically interpreted as independence from the microenvironment, unbalanced proliferation/apoptosis rates, and further accumulation of blasts.

Using a stochastic model to simulate stem cell decisions, Abkowitz and colleagues have analyzed the behavior of individual components (HSC) acting collectively within a dynamical complex context (clonal diversity plus heterogeneous surrounding microenvironment). By tracking HSC replication, the expansion of the hematopoietic system was apparent from birth to adolescence, when steady-state levels are reached. Stochastic modeling of replication kinetics has shown to be useful to predict cell rebounding upon hematopoietic transplantation or under emerging conditions (Catlin et al., 2005, 2011). In contrast, agent-based deterministic modeling of HSC organization in health and hierarchical-related diseases, like chronic myeloid leukemia, are powerful for simulating additional heterogeneity scenarios to be considered, i.e., aging, HSC-niche interaction and therapy outcomes (Glauche et al., 2011). Unlike CML, AL cells show apparent dependence on their own "leukemic niche" (Veiga et al., 2006; Colmone et al., 2008; Basak et al., 2010; Jacamo et al., 2014). Recent models suggest additional feedback mechanisms assuming both, the leukemic and normal cell interdependence on the same growth factors (Stiehl et al., 2014).

In addition to the normal vs. leukemic competition, increasing evidence of genetic diversity supports the multiclonal evolution of AL (Choi et al., 2007; van Delft et al., 2011; Amir et al., 2013). Strikingly, rather than as a consequence of new acquired mutations, relapse could be explained as a deterministic clonal selection where high proliferative cells are eliminated by chemotherapy, while distinct slow-cycling or self-renewing cells stay protected and may re-emerge when the competing clones (leukemic high-proliferating cells) and their negative feedback (normal hematopoietic cells) have been eliminated. Similar to deterministic models of chemotherapy-dependent clonal selection, the stochastic modeling by Kimmel and Corey drives to the conclusion on the co-existence of distinct clones and the extremely broad heterogeneity of cancer cells. However, the stochastic acquisition of mutations may provide theoretical evidence of the parallel evolving clones with unique proliferative potentials, and represent a suitable model for chronic chemotherapy-induced transition to secondary malignancy (Kimmel and Corey, 2013). Despite the fact that linear mutation structures can simplify the population dynamics, it is necessary to consider proliferation heterogeneity. Interestingly, the acquisition of *de novo* mutations is more probable during long treatment schemes (Lindsley et al., 2015).

Technological advances in RT-PCR, RNA-seq and mass cytometry methods for single cell analysis are providing highly specific clusterization of cell populations that allow the identification of experimentally unseen cell transition stages from the earliest steps of differentiation (Marco et al., 2014; Moignard et al., 2015). With new experimental models and molecular research progress, parameters and assumptions considered for the development of mathematical models, evolve to a more complex understanding of leukemogenesis. The view of two or more hematopoietic populations competing within compartments, plus the resulting regulation among compartments from the isolated feedback loops is too simplistic. Therefore, it is becoming of substantial importance to take into account additional intercellular interactions, including those with non-hematopoietic neighboring cells within the BM niches.

Modeling the Interplay Between Leukemia Cells and the Tumor Microenvironment

Tumor-microenvironment interplay is essential for the protection and progression of malignant cells, where a number of interactions mediated by integrins, cytokines and chemokines, extracellular matrix (ECM) proteins, and other molecules produced and expressed by niche cellular elements, may dictate the final fate decision (Raaijmakers, 2011). The recent multi-compartment model by Gerdes for T-cell lymphoma/leukemia, suggests that premalignant cells can get established in any available permissive niche, compensating their low affinity for specific interactions with an increased efficiency for resource utilization when compared to normal clones (Gerdes et al., 2013).

Closer to this multi-component interaction outlook has been the development of generic-cancer cell-automata models. This type of discrete modeling makes the evaluation of homogeneous or heterogeneous cell populations in a grid where every cell has a defined state and neighborhood possible. Strikingly, cell-automata modeling concede single-cell resolution and had become a very promising tool for the study of tissue development and tumors, including microenvironmental factors like ECM density and oxygen diffusion that control tumor size (Chen et al., 2014; Scott et al., 2014).

In spite of the power of these approaches, it is clear that the feedback existing between the BM microenvironmental components and the malignant cell decisions operates at a molecular level regulating intracellular pathways. How could we

mimic the complexity at cellular population and molecular levels at the same time? How could we address the multi-cellular system within systems complexity?

Simulation of One-cell Molecular Network Models with Multi-cellular Methods

Knowledge about the hematopoietic system has been benefited from the development of regulatory networks for early HSC differentiation, T lymphocyte development, plasticity and signaling, among others (Albert and Wang, 2009; Naldi et al., 2010; Martínez-Sosa and Mendoza, 2013; Tian and Smith-Miles, 2014). Considering that every computational simulation with a specific initial state of an intracellular network represents a single cell dynamic profile, to simulate a multi-cellular process we must simultaneously simulate as many networks as cells within the system (Wu et al., 2009). Accordingly, Mendoza proposed a virtual culture of Th cells that simulate differentiation of naive CD4+ T cells to Th1, Th2, Th17, and Treg subsets. In this model, each cell phenotype is defined by molecular patterns of activation, while the input for each virtual cell at

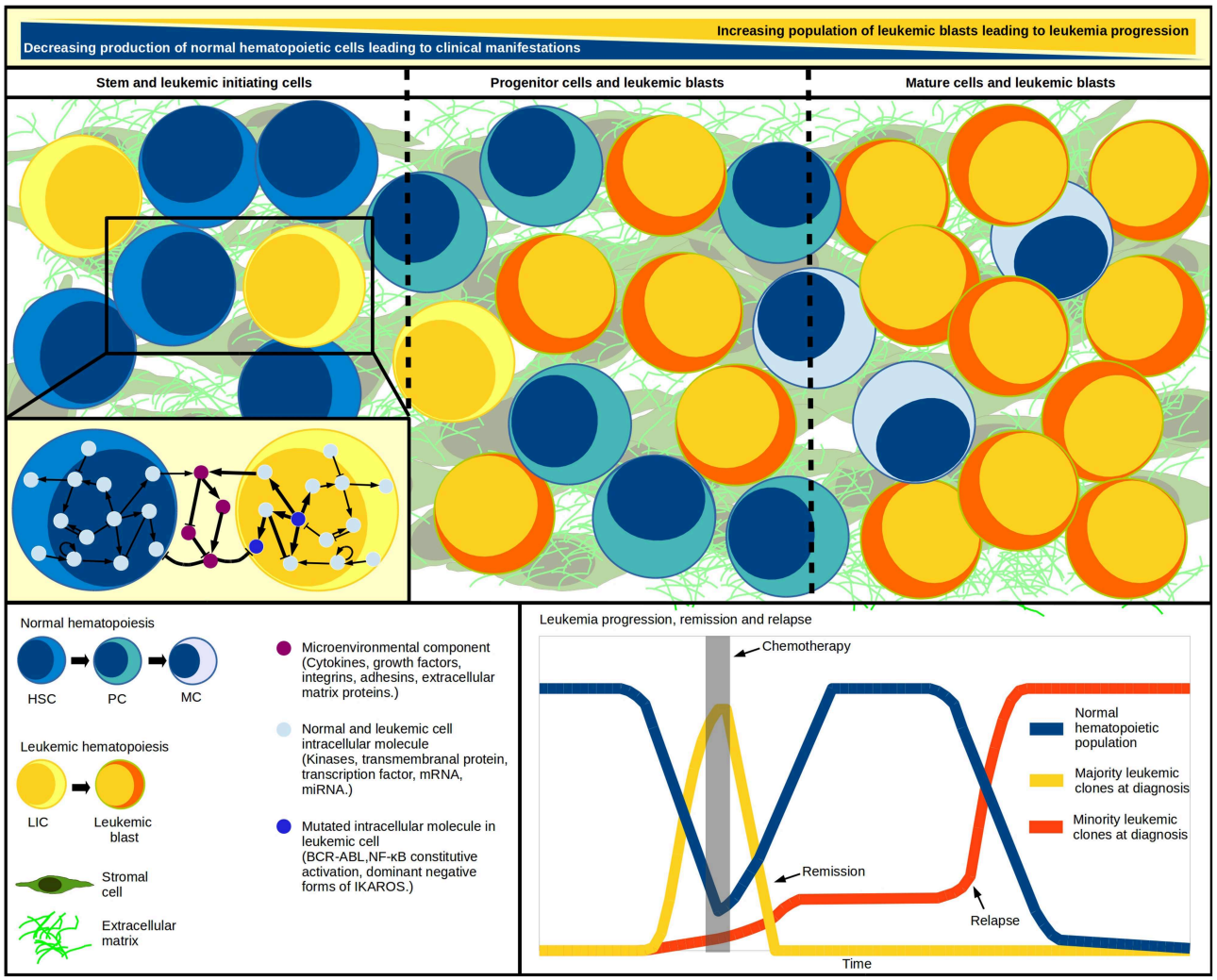


FIGURE 1 | Systems within a system. Leukemic initiation and progression is a tightly regulated competitive process, where at least three systems must work together: the normal hematopoietic differentiation, the leukemic cell production, and the hematopoietic microenvironment where malignant and normal cells competition takes place. As blast population increases, the normal cell abundance decreases. Cell population changes have been modeled through differential equations in multi-compartment continuous modeling, strategy that allows the representation of hematopoietic hierarchy with the assignation of different kinetic parameters values for cells within each compartment. The model mimics *in vivo* fundamental properties like quiescence in the stem-cells compartment and increasing proliferation in developing cells. Additional to regulatory feedback between normal and malignant hematopoietic populations, an abnormal microenvironment may play a crucial cooperating role in the inverse leukemic/normal relationship by disrupting the HSC-niche communication. The genetic diversity within the various leukemia-initiating cells and tumor cells highlights the multiclonal complexity of the disease, and suggests the existence of minor malignant clones—undetected at diagnosis—that become apparent upon chemotherapy and drive individuals to relapse. HSC, hematopoietic stem cell; HPC, hematopoietic progenitor cell; PC, precursor cell; MC, mature cell; LIC, leukemia initiating cell.

any time-step proceeds from the intercellular communication (Mendoza, 2013). More importantly, the dynamics of a given regulatory network respond to the concentration of regulatory cytokines produced by the cell itself and to neighbor signal intensities. Thus, applying tools like virtual cultures to malignant hematopoiesis may help to understand blast accumulation or the intercommunication between leukemia-initiating cells and an abnormal BM microenvironment (**Figure 1**). The recent demonstration of pro-inflammatory cytokines produced by ALL cells suggests that this condition may promote their own survival and account for the exhaustion of normal progenitor cells (Vilchis-Ordoñez et al., 2015). The pathological consequences of a pro-inflammatory microenvironment can be resumed in three potential principles: (a) leukemic cells showing aberrant expression of cytokines that perturb normal hematopoiesis, (b) mutated stromal cells favoring a permissive microenvironment for leukemia initiation, progression, and maintenance (Shalapour et al., 2010), or (c) normal hematopoietic cells responding to biological stress due to blast overcrowding by activating pro-inflammatory pathways. These three scenarios might act independently or sinergistically by means of positive feedback.

To solve this, hybrid models are also mathematical tools with great potential to model microenvironment-dependent systems, allowing the scaling to tridimensional modeling and the consideration of discrete decisions on cell processes like migration and proliferation (Anderson, 2005; Scott et al., 2014). Although these dedicated models have considered microenvironmental factors for solid tumor progression, they still miss the direct feedback existing between extracellular factors and the intra-cellular signaling pathways that regulate cell fate decisions. Of note, an intracellular view would allow modeling of constitutive or null activation of specific pathway mediators and analyzing the putative consequent effects on disease dynamics. Virtual cultures make this possible, but the very high computational requirements when modeling excessive number of cells may represent by now a weakness of the strategy.

For any of the discussed modeling approaches, the importance of a rigorous experimental validation of mathematical modeling for complex processes is high and has been limited by the experimental systems that are conventionally used to study human leukemogenesis. The combination of single-cell sequencing, 3-D organoid-like cultures and xenotransplantation would provide new information for malignant vs. normal cell discrimination and cell population dynamics within more natural microenvironmental structures. Furthermore, a proper validation of current and future investigations from the view of systems biology will benefit from longitudinal, prospective clinical studies.

To this extent, the use of "edge-technology" *in silico* strategies for multi-cellular (leukemic, hematopoietic, and stromal components), multi-compartment (differentiation stages), and agent-based (individual cells network) modeling of leukemia pathobiology is a promising tool for the study of feedback pathways in the searching of auxiliary strategies for leukemia treatment, normal hematopoiesis rebounding, and relapse delay. The construction of novel "systems within a system" integrative theoretical models (**Figure 1**) that better mimic and predict the behavior of the disease may transform our vision of malignant hematopoiesis and provide helpful platforms for new testable hypotheses.

Author Contributions

JE: Analysis of published data, discussion of the topic-related information, drafting, and writing the paper. LM and RP: Conception and design of the Opinion Article, analysis of published data, discussion of the related information, drafting, and writing the paper. Critical review of the intellectual content.

Acknowledgments

Our work is supported by the National Council of Science and Technology (CONACyT, Grant CB-2010-01-152695 to RP), the Mexican Institute for Social Security (IMSS, Grants FIS/IMSS/PROT/G13/1229, and FIS/IMSS/PROT/G14/1289 to RP), and Universidad Nacional Autónoma de México (UNAM, Grant UNAM-DGAPA-PAPIIT IN200514 to LM). JE was awarded by the PRODESI-IMSS Program and is scholarship holder from CONACyT and IMSS.

References

Amir, el-A. D., Davis, K. L., Tadmor, M. D., Simonds, E. F., Levine, J. H., Bendall, S. C., et al. (2013). viSNE enables visualization of high dimensional single-cell data and reveals phenotypic heterogeneity of leukemia. *Nat. Biotechnol.* 31, 545–552. doi: 10.1038/nbt.2594

Albert, R., and Wang, R. S. (2009). Discrete dynamic modeling of cellular signaling networks. *Methods Enzymol.* 467, 281–306. doi: 10.1016/S0076-6879(09)67011-7

Anderson, A. R. (2005). A hybrid mathematical model for solid tumour invasion: the importance of cell adhesion. *Math. Med. Biol.* 22, 163–186. doi: 10.1093/imammb/dqi005

Ayala, F., Dewar, R., Kieran, M., and Kalluri, R. (2009). Contribution of bone microenvironment to leukemogenesis and leukemia progression. *Leukemia* 23, 2233–2241. doi: 10.1038/leu.2009.175

Basak, P., Chatterjee, S., Das, P., Das, M., Pereira, J. A., Dutta, R. K., et al. (2010). Leukemic stromal hematopoietic microenvironment negatively regulates the normal hematopoiesis in mouse model of leukemia. *Chin. J. Cancer.* 29, 969–979. doi: 10.5732/cjc.010.10431

Boulais, P. E., and Frenette, P. S. (2015). Making sense of hematopoietic stem cell niches. *Blood* 125, 2621–2629. doi: 10.1182/blood-2014-09-570192

Catlin, S. N., Busque, L., Gale, R. E., Guttorp, P., and Abkowitz, J. L. (2011). The replication rate of human hematopoietic stem cells *in vivo*. *Blood* 117, 4460–4466. doi: 10.1182/blood-2010-08-303537

Catlin, S. N., Guttorp, P., and Abkowitz, J. L. (2005). The kinetics of clonal dominance in myeloproliferative disorders. *Blood* 106, 2688–2692. doi: 10.1182/blood-2005-03-1240

Chen, D., Jiao, Y., and Torquato, S. (2014). A cellular automaton model for tumor dormandy: emergence of a proliferative switch. *PLoS ONE* 9:e109934. doi: 10.1371/journal.pone.0109934

Choi, S., Henderson, M. J., Kwan, E., Beesley, A. H., Sutton, R., Bahar, A. Y., et al. (2007). Relapse in children with acute lymphoblastic leukemia involving selection of a preexisting drug-resistant subclone. *Blood* 110, 632–639. doi: 10.1182/blood-2007-01-067785

Colmone, A., Amorim, M., Pontier, A. L., Wang, S., Jablonski, E., and Sipkins, D. A. (2008). Leukemic cells create bone marrow niches that disrupt the behavior of normal hematopoietic progenitor cells. *Science* 322, 1861–1865. doi: 10.1126/science.1164390

Gerdes, S., Newrzela, S., Glauche, I., von Laer, D., Hansmann, M. L., and Roeder, I. (2013). Mathematical modeling of oncogenesis control in mature T-cell populations. *Front. Immunol.* 4:380. doi: 10.3389/fimmu.2013.00380

Glauche, I., Thielecke, L., and Roeder, I. (2011). Cellular aging leads to functional heterogeneity of hematopoietic stem cell: a modelling perspective. *Aging Cell* 10, 457–465. doi: 10.1111/j.1474-9726.2011.00692.x

Gupta, S., Rivera-Luna, R., Ribeiro, R. C., and Howard, S. C. (2014). Pediatric oncology as the next global child health priority: the need for national childhood cancer strategies in low- and middle-income countries. *PLoS Med.* 11:e1001656. doi: 10.1371/journal.pmed.1001656

Jacamo, R., Chen, Y., Wang, Z., Ma, W., Zhang, M., Spaeth, E. L., et al. (2014). Reciprocal leukemia-stroma VCAM-1/VLA-4-dependent activation of NF-kB mediates chemoresistance. *Blood* 123, 2691–2702. doi: 10.1182/blood-2013-06-511527

Kim, J. A., Shim, J. S., Lee, G. Y., Yim, H. W., Kim, T. M., Kim, M., et al. (2015). Microenvironmental remodeling as a parameter and prognostic factor of heterogeneous leukemogenesis in acute myelogenous leukemia. *Cancer Res.* 75, 2222–2231. doi: 10.1158/0008-5472.CAN-14-3379

Kimmel, M., and Corey, S. (2013). Stochastic hypothesis of transition from inborn neutropenia to AML: interactions of cell population dynamics and population genetics. *Front. Oncol.* 3:89. doi: 10.3389/fonc.2013.00089

Lindsley, R. C., Mar, B. G., Mazzola, E., Grauman, P. V., Shareef, S., Allen, S. L., et al. (2015). Acute myeloid leukemia ontogeny is defined by distinct somatic mutations. *Blood* 125, 1367–1376. doi: 10.1182/blood-2014-11-610543

MacLean, A. L., Lo Celso, C., and Stumpf, M. P. (2013). Population dynamics of normal and leukaemia stem cells in the haematopoietic stem cell niche show distinct regimes where leukaemia will be controlled. *J. R. Soc. Interface* 10:20120968. doi: 10.1098/rsif.2012.0968

Marco, E., Karp, R. L., Guo, G., Robson, P., Hart, A. H., Trippa, L., et al. (2014). Bifurcation analysis of single-cell gene expression data reveals epigenetic landscape. *Proc. Natl. Acad. Sci. U.S.A.* 111, E5643–E5650. doi: 10.1073/pnas.1408993111

Martínez-Sosa, P., and Mendoza, L. (2013). The regulatory network that controls the differentiation of T lymphocytes. *BioSystems* 113, 96–103. doi: 10.1016/j.biosystems.2013.05.007

Mendoza, L. (2013). A virtual culture of CD4+ T lymphocytes. *Bull. Math. Biol.* 75, 1012–1029. doi: 10.1007/s11538-013-9814-9

Moignard, V., Woodhouse, S., Haghverdi, L., Lilly, A. J., Tanaka, Y., Wilkinson, A. C., et al. (2015). Decoding the regulatory network of early blood development from single-cell gene expression measurements. *Nat. Biotechnol.* 33, 269–276. doi: 10.1038/nbt.3154

Naldi, A., Carneiro, J., Chaouiya, C., and Thieffry, D. (2010). Diversity and plasticity of Th cell types predicted from regulatory network modelling. *PLoS Comput. Biol.* 6:e1000912. doi: 10.1371/journal.pcbi.1000912

Paguirigan, A. L., Smith, J., Meshinchi, S., Carroll, M., Maley, C., and Radich, J. P. (2015). Single-cell genotyping demonstrates complex clonal diversity in acute myeloid leukemia. *Sci. Transl. Med.* 7:281re2. doi: 10.1126/scitranslmed.aaa0763

Pelayo, R., Dorantes-acosta, E., Vadillo, E., and Fuentes-pananá, E. (2012). "From HSC to B-lymphoid cells in normal and malignant hematopoiesis," in *Advances in Hematopoietic Stem Cell Research,* ed R. Pelayo (Croatia: InTech), 277–298.

Purizaca, J., Meza, I., and Pelayo, R. (2012). Early lymphoid development and microenvironmental cues in B-cell acute lymphoblastic leukemia. *Arch. Med. Res.* 43, 89–101. doi: 10.1016/j.arcmed.2012.03.005

Raaijmakers, M. H. (2011). Niche contributions to oncogenesis: emerging concepts and implications for the hematopoietic system. *Haematologica* 96, 1041–1048. doi: 10.3324/haematol.2010.028035

Rubinow, S. I., and Lebowitz, J. L. (1976). A mathematical model of the acute myeloblastic leukemic state in man. *Biophys. J.* 16, 897–910. doi: 10.1016/S0006-3495(76)85740-2

Scott, J. G., Hjelmeland, A. B., Chinnaiyan, P., Anderson, A. R., and Basanta, D. (2014). Microenvironmental variables must influence intrinsic phenotypic parameters of cancer stem cells to affect tumourigenicity. *PLoS Comput. Biol.* 10:e1003433. doi: 10.1371/journal.pcbi.1003433

Shalapour, S., Eckert, C., Seeger, K., Pfau, M., Prada, J., Henze, G., et al. (2010). Leukemia-associated genetic aberrations in mesenchymal stem cells of children with acute lymphoblastic leukemia. *J. Mol. Med.* 88, 249–265. doi: 10.1007/s00109-009-0583-8

Stiehl, T., Baran, N., Ho, A. D., and Marciniak-Czochra, A. (2014). Clonal selection and therapy resistance in acute leukaemias: mathematical modelling explains different proliferation patterns at diagnosis and relapse. *J. R. Soc. Interface* 11:20140079. doi: 10.1098/rsif.2014.0079

Stiehl, T., and Marciniak-Czochra, A. (2012). Mathematical modeling of leukemogenesis and cancer stem cell dynamics. *Math. Model. Nat. Phenomena* 7, 166–202. doi: 10.1051/mmnp/20127199

Swaminathan, S., Klemm, L., Park, E., Papaemmanuil, E., Ford, A., Kweon, S. M., et al. (2015). Mechanisms of clonal evolution in childhood acute lymphoblastic leukemia. *Nat. Immunol.* 16, 766–774. doi: 10.1038/ni.3160

Tian, T., and Smith-Miles, K. (2014). Mathematical modeling of GATA-switching for regulating the differentiation of hematopoietic stem cell. *BMC Syst. Biol.* 8(Suppl. 1):S8. doi: 10.1186/1752-0509-8-S1-S8

van Delft, F. W., Horsley, S., Colman, S., Anderson, K., Bateman, C., Kempski, H., et al. (2011). Clonal origins of relapse in ETV6-RUNX1 acute lymphoblastic leukemia. *Blood* 117, 6247–6254. doi: 10.1182/blood-2010-10-314674

Veiga, J. P., Costa, L. F., Sallan, S. E., Nadler, L. M., and Cardoso, A. A. (2006). Leukemia-stimulated bone marrow endothelium promotes leukemia cell survival. *Exp. Hematol.* 34, 610–621. doi: 10.1016/j.exphem.2006.01.013

Vesely, M. D., Kershaw, M. H., Schreiber, R. D., and Smyth, M. J. (2011). Natural innate and adaptive immunity to cancer. *Annu. Rev. Immunol.* 29, 235–271. doi: 10.1146/annurev-immunol-031210-101324

Vilchis-Ordoñez, A., Contreras-Quiroz, A., Vadillo, E., Dorantes-Acosta, E., Reyes-López, A., Quintela-Nuñez del Prado, H. M., et al. (2015). Bone marrow cells in acute lymphoblastic leukemia create a proinflammatory microenvironment influencing normal hematopoietic differentiation fates. *Biomed Res. Int.* 2015:386165. doi: 10.1155/2015/386165

Wu, M., Yang, X., and Chan, C. (2009). A dynamic analysis of IRS-PKR signaling in liver cells: a discrete modeling approach. *PLoS ONE* 4:e8040. doi: 10.1371/journal.pone.0008040

Conflict of Interest Statement: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Advantages of publishing in Frontiers

OPEN ACCESS
Articles are free to read, for greatest visibility

COLLABORATIVE PEER-REVIEW
Designed to be rigorous – yet also collaborative, fair and constructive

FAST PUBLICATION
Average 85 days from submission to publication (across all journals)

COPYRIGHT TO AUTHORS
No limit to article distribution and re-use

TRANSPARENT
Editors and reviewers acknowledged by name on published articles

SUPPORT
By our Swiss-based editorial team

IMPACT METRICS
Advanced metrics track your article's impact

GLOBAL SPREAD
5'100'000+ monthly article views and downloads

LOOP RESEARCH NETWORK
Our network increases readership for your article

Frontiers
EPFL Innovation Park, Building I • 1015 Lausanne • Switzerland
Tel +41 21 510 17 00 • Fax +41 21 510 17 01 • info@frontiersin.org
www.frontiersin.org

Find us on