# Z34Bio: An SMT-based Framework for Analyzing Biological Computation

Boyan Yordanov, Christoph M. Wintersteiger, Youssef Hamadi and Hillel Kugler

Microsoft Research, Cambridge, UK,
http://research.microsoft.com/z3-4biology

**Abstract**

The basic principles governing the development and function of living organisms remain only partially understood, despite significant progress in molecular and cellular biology and tremendous breakthroughs in experimental methods. The development of system-level, mechanistic, computational models has the potential to become a foundation for improving our understanding of natural biological systems, and for designing engineered biological systems with wide-ranging applications in nanomedicine, nanomaterials and computing. We describe Z34Bio (Z3 for Biology), a unified SMT-based framework for the automated analysis of natural and engineered biological systems. Z34Bio enables addressing important biological questions, and studying models more complex than previously possible. The framework provides a formalization of the semantics of several model classes used widely for biological systems, which we illustrate through the treatment of chemical reaction networks and Boolean networks. We present case-studies which we make available as SMT-LIB benchmarks, to enable comparison of different analysis techniques, and towards making this new domain accessible to the formal verification community.

## 1 Introduction

Many mechanisms and properties of biological systems remain only partially understood, thus limiting our comprehension of natural living systems and processes. Recently, advanced experimental techniques have enabled the rational design and construction of biological systems, delineating a branch of biology as an engineering discipline, with potential applications in nanomedicine, nanomaterials and computing. However, understanding the system-level behavior of organisms or designing ones with specific behavior remains a major challenge for the engineering and the reverse engineering of biological systems.

Computational modeling, together with methods enabling the automated analysis of realistic models for diverse biological queries, can help address these challenges and tackle important questions related to *biological computation* - the information processing within living organisms. Along this direction, we introduce Z34Bio (Z3 for Biology) as a framework that allows flexible and scalable analysis of biological models using Satisfiability Modulo Theories (SMT)-based procedures. The framework provides a formalization of the semantics of several widely used formalisms in biological modeling, which we illustrate through the treatment of chemical reaction networks (CRNs) and Boolean networks (BNs), as well as combinations thereof. These formalisms are useful for describing DNA computing circuits (as well as more general biochemical mechanisms within natural systems) and biological interaction networks such as gene regulation networks (GRNs). We formalize the semantics of CRNs and BNs as transition systems, which we represent and analyze symbolically using SMT to allow flexible and convenient encoding of (possibly infinite-state) biological models.

The richness of the various SMT logics also allows us to express a range of important biological properties that are not easily captured by other specification formalisms. For instance,

we are able to formalize and study certain mass-conservation properties and the effect of gene knockouts on system dynamics. The availability of efficient decision procedures for some SMT logics such as uninterpreted functions and bit vectors (UFBV) with quantifiers [25] provides a foundation for the analysis of such questions, even for large and complex systems. While the Z3 theorem prover [9] is used in Z34Bio, arbitrary SMT solvers can be substituted in the framework through the SMT-LIB input language. In [27] we showed how SMT-based methods can be applied to engineered biological systems, and, more specifically, in DNA computing and synthetic biology. Here we present a framework supporting this approach accompanied by an online tool, extend it to allow modeling and reasoning about biological computation within living systems via Boolean networks, and provide support for hybrid models, composed of CRNs and BNs. We outline a number of case-studies illustrating the analysis of engineered DNA circuits and genetic regulatory networks (GRNs), which we curate and make available as SMT-LIB benchmarks, with the goal of improving the evaluation of existing SMT algorithms, helping in the development of new methods, and making this auspicious application domain more accessible to the SMT community.

## 2 Chemical Reaction Networks and Boolean Networks

In the field of DNA computing, which aims at engineering and understanding forms of computation performed by biological material (*e.g.*, reacting DNA strands), chemical reaction networks (CRNs) serve as models of circuits [24, 18]. More generally, CRNs are often used to describe a number of natural and engineered biochemical mechanisms. Here, we study such systems with single-molecule resolution, abstracting from the exact reaction kinetics (rates), thereby approximating probabilistic systems by non-deterministic ones. While certain information is not captured in this representation of the behavior of a CRN, it is a useful level of detail for various studies of DNA circuits, including cases where functional correctness is under investigation. Where studies of natural biological systems are concerned, this is often also a useful abstraction, when the rates of certain reactions are unknown and a precise measurement in a wet-lab is challenging.

We treat a CRN as a pair $(\mathcal{S}, \mathcal{R})$ of species (different DNA strands) and reactions where a reaction $r \in \mathcal{R}$ is a pair of multisets $r = (R_r, P_r)$ describing the reactants (inputs) and products (outputs) of $r$ with their stoichiometries (the numbers of participating strands). We formalize the behavior of a CRN as the transition system $\mathcal{T} = (Q, T)$ where a state $q \in Q$ is a multiset of species, where $q(s)$ indicates how many strands of $s$ are available in a state $q$, and $T$ is the transition relation defined as $T(q, q') \leftrightarrow \bigvee_{r \in \mathcal{R}} [on(r, q) \wedge \bigwedge_{s \in \mathcal{S}} q'(s) = q(s) - R_r(s) + P_r(s)]$, where $on(r, q)$ is *true* if in state $q$ there are enough molecules of each reactant of $r$ for it to fire.

The complementarity of DNA sequences, dictated by the binding of Watson-Crick DNA base pairs (A-T and G-C), provides a mechanisms for engineering chemical reaction networks using DNA. In this approach, various single and double-stranded DNA molecules are designated as chemical species. The binding, unbinding and displacement reactions possible between the complementary DNA domains (subsequences) of these species form the desired CRN structure. When specific computational operations are implemented using such a strategy, the resulting system is called a *DNA circuit* (see [18] and the references therein for additional details on the formalization and design of DNA circuits).

Figure 1 (left panel) shows a simple DNA circuit implementing a logical AND gate. The system is represented as a CRN with seven different species (A, B, C, Gate, GateA, GateB, GateAB) and four reactions, two of which are reversible as indicated by the bi-directional arrows. Species A and B represent the two system inputs, species Gate is the actual AND gate, and species C

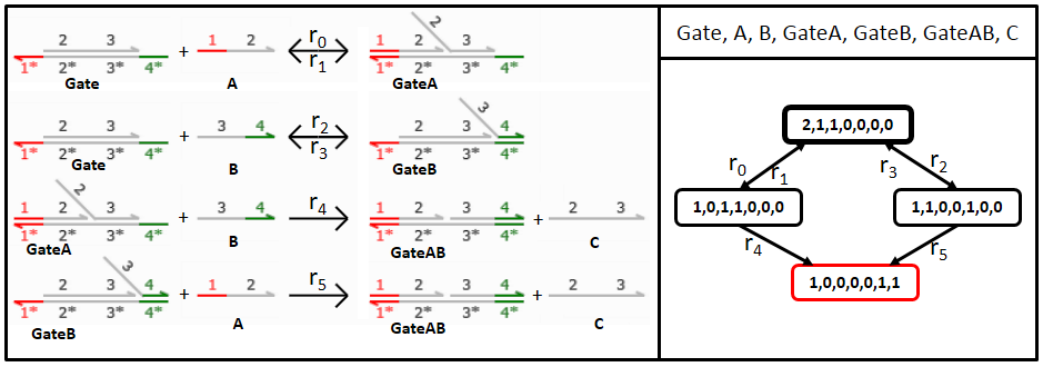


Figure 1: A simple DNA circuit implementing a logical AND gate. For each species (`Gate`,`A`,`B`,`GateA`,`GateB`,`GateAB`,`C`), domains labeled by $1, \ldots, 4$ represent different DNA sequences, while complementary sequences are denoted by $*$ (*e.g.* domains 1 and $1^*$ are complementary). The binding of complementary domains and the subsequent displacement of adjacent complementary sequences determines the possible chemical reactions $(r_0, \ldots, r_5)$ between the DNA species (left panel). The DNA circuit is represented as a transition system (right panel) where a state captures the number of molecules from each species and the initial state is highlighted using a thick black border. For this system, a state can be reached where no additional reactions are possible (shown with a red border), where computation terminates. For a single molecule of species `Gate`, the output `C` is produced at the end of the computation only if both input species `A` and `B` are present, which captures the required logical AND behavior.

is the output (all other species are intermediates). A state of the system captures the number of available molecules from each DNA species, which change as reactions take place, leading to the transition system representation in Figure 1 (right panel).

In some applications, it is sufficient to describe species more coarsely, using a small number of discrete levels of activity. This has proven to be a most useful abstraction, especially for analyzing the dynamics of species within gene regulatory networks (GRNs) [17] *e.g.* during the life-cycle of a cell or an organism. Unlike the biological engineering applications described above, the focus here is on understanding natural systems and, often, only the species' presence or absence or the activity or inactivity of *genes* is tracked. A Boolean network is a popular representation of a GRN, which is given as a pair $(\mathcal{S}, \mathcal{F})$ of species and a set of update functions. We capture the behavior over time in the transition system $\mathcal{T} = (Q, T)$ where $Q = \mathbb{B}^{|\mathcal{S}|}$ and $q(s) \in \mathbb{B}$ indicates the presence or absence of $s$. The system's dynamics are defined by $\mathcal{F}$, which is a set of functions, one for each species, i.e., $f_s \in \mathcal{F}, f_s : \mathbb{B}^{|\mathcal{S}|} \to \mathbb{B}$ where the *synchronous*[1] transition relation $T(q, q') \leftrightarrow \left(\bigwedge_{s \in \mathcal{S}} q'(s) = f_s(q)\right)$ results in a deterministic, deadlock-free system. Despite the apparent simplicity of such models, they are tremendously useful in practice, because often we do not know the quantitative interactions within the system, and a precise measurement of levels of activation in a wet-lab experiment is challenging.

`Z34Bio` supports a natural combination of CRN and BN models, allowing for "localized" abstractions, *e.g.,* to simplify the analysis of parts of a system that do not require a model on the single-molecule level. These parts may be abstracted by a BN; Figure 3 shows an example of such a combined model, using the CRN component from Figure 1 and the Boolean network

[1]This means all species update at each time step, *asynchronous* updates are also supported by our framework, but not described here.

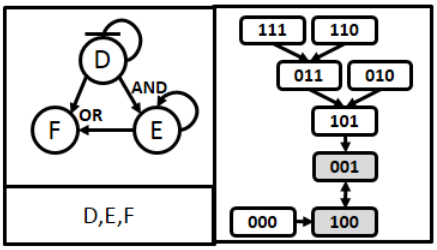


Figure 2: A Boolean network representing three species D,E, and F. The Boolean update functions are represented graphically (left panel) using pointed arrows (positive interaction), T-arrows (negative interaction), and the logical combination of inputs (*e.g.* the next state of species $F$ is given by $F' = D \wedge E$). The Boolean network is represented as a transition system (right panel) where all nodes are updated synchronously. This representation reveals that the system does not stabilize in a single state but instead reaches a cycle where the values of species D and F oscillate.

component from Figure 2 (modified to allow the interaction between the two components). In Z34Bio, we encode a BN as a single bit-vector, which leads to a compact representation, provides convenient bit-wise and arithmetic operations, while efficient decision procedures even when quantifier are used (SMT BV and UFBV), are also available [25]. We use integers to encode CRNs due to their potentially unbounded numbers of strands (or molecules) where this is required. When this is so, we first use Z34Bio in an attempt to prove the validity of mass-conservation constraints, providing us with bounds on the integer representation, thereby allowing the use of bit-vector encodings of appropriate size without sacrificing precision.

# 3 Analysis Strategies

The basic analysis strategy of Z34Bio is inspired by well-known model checking and deductive verification algorithms, most prominently Bounded Model Checking (BMC) and inductive invariants. We describe system behavior as constraints over a set of symbolic, finite paths of the transition system $\mathcal{T}$. A path is denoted as $\tau = \{q_0, \ldots, q_{K-1}\}$ where $\bigwedge_{i=0}^{K-2} T(q_i, q_{i+1})$ and $\tau[k] = q_k$ denotes the (symbolic) state at step $k$. Initial conditions of the system are described symbolically through constraints. Once all constraints describing the model as well as the property of interest are included, we encode them to a series of SMT queries, which allow us to find a model and to instantiate the abstract paths to concrete ones, or to report an unsatisfiable specification.

We use standard logical operations to construct formulas and enforce them for states. This allows us to find states with certain characteristics as (counter-) examples or prove their absence, and to test reachability and (certain) temporal properties over paths. For instance, stability and oscillations are studied in terms of paths with specific features. A *cycle* of length $K$ is a path $\tau$ where $|\tau| = K$, $T(\tau[K-1], \tau[0])$ and no other states are repeated, and a *fixed point* is a cycle of length $K = 1$. For non-deterministic systems such as CRNs, a cycle $\tau$ may be *stable* or *unstable* resulting in persistent or (possibly) transient behavior, which is tested using a path $\tau'$ of length 2, such that $\forall \tau' . \bigwedge_{i=0\ldots K} (\tau[i] = \tau'[0]) \rightarrow (\tau[i+1] = \tau'[1])$, unsatisfiability of which indicates transient behavior.

General system analysis in the case of biology also includes the analysis of existing systems that occur in nature. While for some applications this can be viewed as the estimation of black-box behavior, typically, biologists propose models of natural behavior which are meant to explain observations made in labs and allow predicting the outcome of new experiments. As the models grow, the task of refuting a model, or verifying that a model indeed explains all known

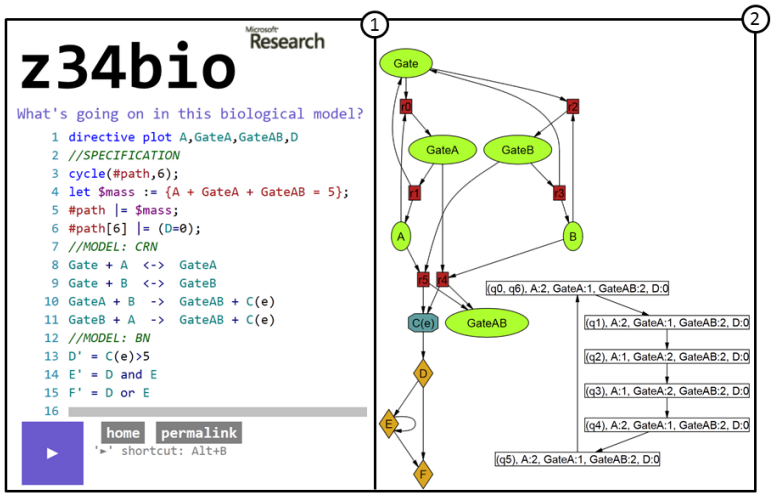


Figure 3: A screen shot illustrating the use of Z34Bio. A model (combined CRN and BN) and a specification (a mass-conservation property) are defined (1). When the "play" button is pressed, the model is visualized (Boolean, chemical and shared species are drawn as diamonds, ovals, and octagons, resp.). Interesting states and trajectories fulfilling the specification are found and displayed (2).

observations to a satisfying degree becomes harder. At the same time, the parts of the behavior of the models which are not covered by observations remain doubtful and it is imperative that new experiments for testing such models are performed to finally refine the theory. Therefore, the task of analyzing a biological model does not only include the establishment of invariants of such systems and studying their normal behavior, but also the investigation of perturbations and changes to the system behavior.

One instance of such a task is the analysis of *gene knockouts*, *i.e.,* the analysis of the system where one or more of the genes are permanently (or temporarily) disabled. Gene knockouts can occur naturally by acquiring a certain mutation in a gene, or induced by various experimental methods while studying model organisms (*e.g.*, fruit fly, worm). To automate the process of finding knockouts that effect a certain biological phenomena, and lead to interesting behavior, we augment the definition of a BN to include the bit-vector $ko \in \mathbb{B}^{|\mathcal{S}|}$, where $ko(s) \in \mathbb{B}$ indicates whether species $s$ has been knocked out, in which case it is always inactive. Additional constraints, (*e.g.*, on the cardinality of $ko$) and properties regarding the desired behavior are specified and the missing information (specific gene knockouts) is obtained from the underlying SMT solver. To close the loop back to the biological science, note that "interesting" behavior in this type of analysis means that either a new experiment is suggested (when the system behaves in an unpredicted way), or that a problem in the model is identified because the model does not explain previous observations (where some gene may have been knocked out during a wet-lab experiment).

# 4 Applications

We implemented Z34Bio as an online analysis tool [28], providing basic user-interface and visualization capabilities (Figure 3). Analysis problems can be exported as SMT-LIB benchmarks

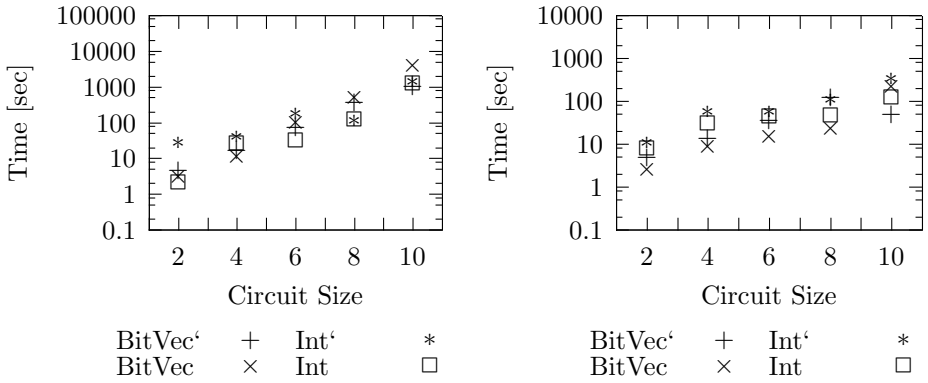


Figure 4: Computation times for the identification of traces of lengths up to $K = 100$ in the flawed transducer circuits such that a "good" state (left panel) or a "bad" state (right panel) is reached (note the difference in scales). BitVec‘ and BitVec (resp. Int‘ and Int) indicate a bit-vector (resp. integer) encoding with or without the additional mass-conservation constraints. Results from [27].

and processed offline. To illustrate some potential applications, we present a set of case-studies which are provided as benchmarks or can be explored interactively online. More details about these examples as well as additional challenging benchmarks are also available on our web-site [29]. The experimental results are obtained using the Z3 solver directly on the SMT-LIB benchmarks. All computation is performed on 2.5 Ghz Intel L5420 CPUs with a 2GB memory limit per benchmark.

The *transducer* DNA circuits described in [18, 27] are designed to convert all molecules of some chemical input to some output molecules (for an example of the structure of these models, see Figure 5). Computation terminates when a state with no possible reactions is reached but certain reactive species must also be fully consumed. First, we prove that a set of mass-conservation constraints hold for these systems, which capture the property that DNA is not created or degraded but only converted between species. Then, we use these constraints to prove that no "bad" terminal states (where reactive species remain) are possible for correct transducer circuits, but such states can be found for "faulty" designs. Finally, we show that "good" (resp. both "good" and "bad") states are reachable for correct (resp. "faulty") circuits using Bounded Model Checking [27]. Computational results from [27] shown in Figure 4, show that models take non-trivial runtimes, but are in a feasible range. The tradeoff between using bitvectors or integer representation is an interesting aspect for further exploration, as it seems each performs better for different models, sizes, and queries.

Besides the set of transducer circuits, we also applied our method to analyze a design of a DNA circuit that computes the square root of a 4-bit number [22, 4], which is one of the largest DNA computation circuits constructed experimentally. This system is designed to compute the square root of a number represented using DNA species. Our results indicate that the described methods can be applied to analyze functional correctness of systems of such complexity with large numbers of copies of the circuit operating in parallel.

Understanding the effects of gene knockouts on the dynamics of gene networks is an important biological question. Such GRN perturbations are often caused by mutations leading to a

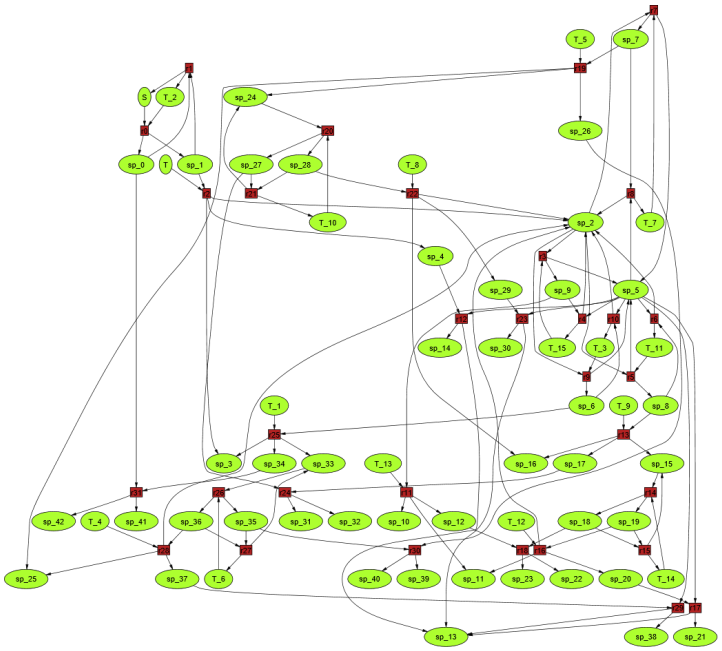


Figure 5: A chemical reaction network derived from a transducer DNA circuit.

number of diseases, while experimental techniques for introducing specific gene knockouts are also available, providing a strategy for the comparison of model predictions and experimental observations. For synchronous Boolean network models, stabilization is possible only when a fixed point exists, while oscillations are possible when a cycle of length $K > 1$ exists. We used bounded model checking to identify cycles of length up to $K = d$ (the recurrence diameter) and, while such a procedure is generally expensive, a short diameter is characteristic of some biological models. This is the case for the Boolean network models collected in [11] and the large-scale regulatory and metabolic network reconstruction studied in [23] (Table 1); for an example of the structure of such models, see Figure 6, which illustrates part of a fruit fly's GRN. A model's diameter also provides interesting information about the underlaying biological system, since it captures properties related to its response time.

To search for gene knockouts that influence stability, we first introduce the parameter $ko$ and investigate how this affects the diameters $d'$. Next we find paths that originate in the same state but change the stability/oscillatory behavior depending on $ko$, providing the target set of knockouts (genes that must be knocked out to achieve the required behavior). Results in Table 1 show that we can effectively identify single and double mutations that effect system dynamics, and the method can be effective also for larger cardinality, providing a powerful way to investigate gene knockouts which is currently very challenging utilizing simulation methods. Overall, our framework has proven to be powerful enough to tackle important biological models, suggesting that SMT-based methods have the potential to play a significant role in this emerging field. Significant advances are still needed to allow biologists to analyze some of the systems they study, and we hope this work will inspire additional progress and development of methods towards this goal.

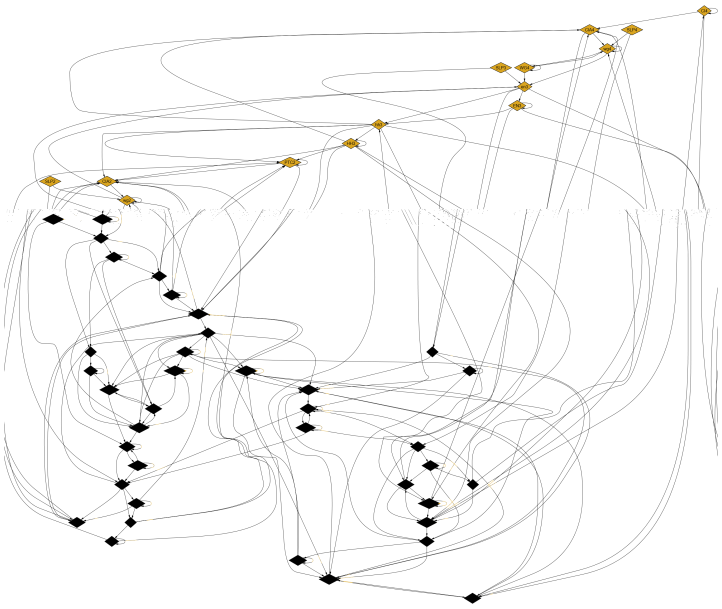

Figure 6: Boolean network model of Drosophila melanogaster's segment polarity genes. (Part of a fruit fly's gene regulatory network.)

| Model name | $\|\mathcal{S}\|$ | K | d | Behavior | Time | d' | Single knockouts | Time | Double knockouts | time |
|---|---|---|---|---|---|---|---|---|---|---|
| arabidopsis | 15 | 10 | 10 | stabil | 0.12 | 10 | LFY, SEP (2) | 0.12 | {EMF1,SEP},... (19) | 2.17 |
| budding yeast | 12 | 18 | 18 | stabil | 0.20 | 18 | none | 0.5 | none | 0.5 |
| drosophila | 52 | 34 | 34 | stabil | 2.7 | 43 | HH2 (1) | 114.96 | {SLP3,HH3},... (42) | 929.1 |
| fission yeast | 10 | 6 | 6 | stabil | 0.09 | 6 | none | 0.1 | none | 0.08 |
| mammalian | 10 | 7 | 11 | both | 0.07 | 13 | Rb,Cdc20,...(7) | 0.75 | {CycE,CycB},...(41) | 2.997 |
| tcr | 40 | 6 | 18 | both | 0.26 | 22 | CD8,CD45,...(9) | 14.6 | {CD45,IKK},... (345) | 379.1 |
| t helper | 23 | 11 | 11 | stabil | 0.12 | 17 | none | 0.34 | none | 0.31 |
| met. regulation | 693 | 7 | 7 | stabil | 35.05 | 8 | none | 1154.4 | none | 1094.8 |

Table 1: Stability analysis and gene knockout identification for the Boolean network models from [11] and [23]. $|\mathcal{S}|$ is the number of species, $d$ is the diameter ($d'$ when knockouts are allowed), $K$ is the length of the shortest cycle when one exists or $K = d$ otherwise, and all times are in sec. Only some of the identified single and double knockouts are shown (total numbers in parentheses).

# 5 Related work and Future Directions

The simulation of biological models is now supported by many specialized tools (*e.g.* SimBiology by MathWorks), and has been used as an analysis strategy, for example, in [23] but is inherently incomplete and expensive for certain problems. As an alternative, the application of formal methods in the context of biology has already attracted attention [3], providing completeness and more rigorous formalizations of properties. For example, besides simulation capabilities, `Biocham` [13] allows the analysis of rule-based models [7] using temporal logics with numerical constraints, while deriving control strategies for large Boolean networks using CTL specifications and the NuSMV model checker as described in [19].

Such expressivity is not always sufficient - to capture notions of system stability an extension is introduced in `Anelope` [1]. Stability has also been studied through dedicated BDD-based

(GINsim [20]), SAT-based (BNS [11]) and modular (BMA [2]) algorithms for Boolean networks and their generalizations. Probabilistic model checking was used in [18] to study DNA circuits with properties involving probabilities and time, while Petri-nets analysis methods [15] also serve to study chemical reaction networks and therefore DNA circuits. Synthesis methods are developed in GNBox [6] based on Constraint Logic Programming (CLP) to uncover genetic network models from incomplete information while SMT-based approaches have been applied to computer-aided chemical synthesis planning [12].

When a sufficient number of molecules is present, species concentrations can be described as continuous values ((*e.g.* using (non-linear) ODEs) [8]. Such systems, as well as other infinite-state, continuous and hybrid models used in biology, can be encoded into SMT directly but might require expensive (or incomplete) decision procedures. As an alternative, (conservative) finite transition system abstractions can be constructed (*e.g.* as in [26]), enabling the analysis and integration of infinite state systems within the framework described here. The application of formal methods to Petri Nets [5], which also describe chemical reaction networks, has been studied extensively and can provide useful analysis procedures, which can then be extended to all the formalisms we consider through their common representation. Chemical reaction networks have also been studied at steady state using flux balance analysis (FBA) [21].

The available analysis tools often focus on a specific class of models and specifications, while so far the expressivity of SMT has not been fully exploited to allow a more general framework. By doing so, we handle logical, temporal and numerical constraints and can express certain stability properties, while the model-finding capabilities of SMT solvers enable us to seamlessly synthesize parts of the model. A number of extensions can be introduced immediately in our current framework (*e.g.* to capture more general genetic network models) but novel SMT procedures are required for others *e.g.* to allow analysis for probabilistic properties when chemical kinetics are considered [14, 16, 10].

Analysis of biological models is related to hardware and software verification in general. Due to the special nature of biological models it is often the case that traditional verification tools do not perform well on models of these systems, as they are highly concurrent and non-deterministic. A more thorough investigation of the differences between models in biology, software and hardware, especially on standardized realistic models (*e.g.*, through established SMT-LIB benchmarks which we initiate here) poses an interesting challenge for future work.

# References

[1] Gustavo Arellano, Julián Argil, Eugenio Azpeitia, Mariana Benítez, Miguel Carrillo, Pedro Góngora, David A. Rosenblueth, and Elena R. Alvarez-Buyllaa. "Antelope": a hybrid-logic model checker for branching-time Boolean GRN analysis. *BMC Bioinformatics*, 12(1):490, 2011.

[2] David Benque, Sam Bourton, Caitlin Cockerton, Byron Cook, Jasmin Fisher, Samin Ishtiaq amd Nir Piterman, Alex Taylor, and Moshe Y. Vardi. BMA: Visual tool for modeling and analyzing biological networks. In *CAV*, volume 7358 of *LNCS*, pages 686–692. Springer, 2012.

[3] Miguel Carrillo, Pedro a Góngora, and David a Rosenblueth. An overview of existing modeling tools making use of model checking in the analysis of biochemical networks. *Frontiers in plant science*, 3(July):155, January 2012.

[4] Harish Chandran, Nikhil Gopalkrishnan, Andrew Phillips, and John Reif. Localized hybridization circuits. *DNA Computing and Molecular Programming (DNA17)*, pages 64–83, 2011.

[5] C. Chaouiya. Petri net modelling of biological networks. *Briefings in Bioinformatics*, 8(4):210–219, 2007.

[6] Fabien Corblin, Eric Fanchon, and Laurent Trilling. Applications of a formal approach to decipher discrete genetic networks. *BMC Bioinformatics*, 11(1):385, 2010.

[7] Vincent Danos, Jérôme Feret, Walter Fontana, and Jean Krivine. Abstract interpretation of cellular signalling networks. In *VMCAI*, pages 83–97, 2008.

[8] H. de Jong. Modeling and Simulation of Genetic Regulatory Systems: A Literature Review. *Journal of Computational Biology*, 9(1):67–103, 2002.

[9] Leonardo de Moura and Nikolaj Bjørner. Z3: An efficient SMT solver. In *TACAS*, 2008.

[10] Christian Dehnert, Joost-Pieter Katoen, and David Parker. SMT-based bisimulation minimisation of markov models. In *VMCAI*, volume 7737 of *LNCS*, pages 28–47. Springer, 2013.

[11] Elena Dubrova and Maxim Teslenko. A SAT-Based Algorithm for Finding Attractors in Synchronous Boolean Networks. *IEEE/ACM TCBB*, 8:1393–1399, 2011.

[12] Rolf Fagerberg, Christoph Flamm, Daniel Merkle, and Philipp Peters. Exploring chemistry using SMT. In *PPCP*, LNCS, pages 900–915. Springer, 2012.

[13] Franois Fages and Sylvain Soliman. Formal cell biology in BIOCHAM. In *FMCSB*, 2008.

[14] Martin Fränzle, Holger Hermanns, and Tino Teige. Stochastic satisfiability modulo theory: A novel technique for the analysis of probabilistic hybrid systems. In *HSCC*, 2008.

[15] Monika Heiner, David Gilbert, and Robin Donaldson. Petri nets for systems and synthetic biology. *FMCSB*, 5016:215–264, 2008.

[16] David Henriques, João Martins, Paolo Zuliani, André Platzer, and Edmund M. Clarke. Statistical model checking for markov decision processes. In *QEST*, pages 84–93. IEEE Computer Society, 2012.

[17] S A Kauffman. Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of Theoretical Biology*, 22(3):437–467, 1969.

[18] Matthew Lakin, David Parker, Luca Cardelli, Marta Kwiatkowska, and Andrew Phillips. Design and analysis of DNA strand displacement devices using probabilistic model checking. *J. R. Soc. Interface*, 9(72):1470–85, July 2012.

[19] Christopher James Langmead and Sumit Kumar Jha. Symbolic approaches for finding control strategies in Boolean networks. *J. BCB*, 7(2):323–338, 2009.

[20] Aurélien Naldi, Denis Thieffry, and Claudine Chaouiya. Decision diagrams for the representation and analysis of logical models of genetic networks. In *CMSB*, 2007.

[21] J.D. Orth, I. Thiele, and B.O. Palsson. What is flux balance analysis? *Nat Biotech*, 28(3), 2010.

[22] Lulu Qian and Erik Winfree. Scaling up digital circuit computation with DNA strand displacement cascades. *Science*, 332(6034):1196–201, June 2011.

[23] Areejit Samal and Sanjay Jain. The regulatory network of E. coli metabolism as a Boolean dynamical system exhibits both homeostasis and flexibility of response. *BMC Systems Biology*, 2(1):21, 2008.

[24] Georg Seelig, David Soloveichik, David Yu Zhang, and Erik Winfree. Enzyme-free nucleic acid logic circuits. *Science*, 314(5805):1585–8, 2006.

[25] Christoph M. Wintersteiger, Youssef Hamadi, and Leonardo de Moura. Efficiently solving quantified bit-vector formulas. In *FMCAD*, pages 239–246, 2010.

[26] B. Yordanov and C. Belta. Formal analysis of discrete-time piecewise affine systems. *IEEE Transactions on Automatic Control*, 55(12):2834–2840, 2010.

[27] Boyan Yordanov, Christoph M. Wintersteiger, Youssef Hamadi, and Hillel Kugler. SMT-based analysis of biological computation. In *NASA Formal Methods*, volume 7871 of *LNCS*, pages 78–92. 2013.

[28] `Z34Bio`. Online at http://rise4fun.com/Z34Biology.

[29] `Z34Biology project website`. http://research.microsoft.com/en-us/projects/z3-4biology/.