

Formal Analysis of the Biological Circuits using Higher-order-logic Theorem Proving

Sa'ed Abed

Computer Engineering Department
College of Engineering and
Petroleum, Kuwait University
Kuwait
s.abed@ku.edu.kw

Adnan Rashid

School of EE and CS (SEECs)
National University of Sciences and
Technology (NUST)
Islamabad, Pakistan
adnan.rashid@seecs.nust.edu.pk

Osman Hasan

School of EE and CS (SEECs)
National University of Sciences and
Technology (NUST)
Islamabad, Pakistan
osman.hasan@seecs.nust.edu.pk

ABSTRACT

Synthetic Biology is an interdisciplinary field that utilizes well-established engineering principles, ranging from electrical, control and computer systems, for analyzing the biological systems, such as biological circuits, enzymes, pathways and controllers. Traditionally, these biological systems, i.e., the genetic circuits are analyzed using paper-and-pencil proofs and computer-based simulations techniques. However, these methods cannot provide accurate results due to their inherent limitations such as human error-proneness, round-off errors and the unverified algorithms present in the core of the tools, providing such analyses. In this paper, we propose to use higher-order-logic theorem proving as a complementary technique for analyzing these systems and thus overcome the above-mentioned issues. In particular, we propose a higher-order-logic theorem proving based framework to formally reason about the genetic circuits used in synthetic biology. The main idea is to, first, model the continuous dynamics of the genetic circuits using differential equations. The next step is to obtain the systems' transfer function from their corresponding block diagram representations. Finally, the transfer function based analysis of these differential equation based models is performed using the Laplace transform. To illustrate the practical utilization of our proposed framework, we formally analyze the genetic circuits of activated and repressed expressions of protein.

CCS CONCEPTS

• **Bioinformatics and Computational Biology** → Computational methods for microbiology and synthetic biology; • **Computational methods for microbiology and synthetic biology** → Synthetic biology; • **Synthetic biology** → Biological circuits;

KEYWORDS

Synthetic Biology, Theorem Proving, Biological Circuits, Higher-order Logic, Laplace Transform, HOL Light

ACM Reference Format:

Sa'ed Abed, Adnan Rashid, and Osman Hasan. 2020. Formal Analysis of the Biological Circuits using Higher-order-logic Theorem Proving. In *The 35th ACM/SIGAPP Symposium on Applied Computing (SAC '20)*, March 30–April 3, 2020, Brno, Czech Republic. ACM, New York, NY, USA, 5 pages. <https://doi.org/10.1145/3341105.3373993>

1 INTRODUCTION

Nowadays, engineering principles [18] are being widely adapted in analyzing biological systems [12], signaling pathways [5] (a group of molecules working together to control various functions of a cell) and biological circuits [6] (the biological parts of a cell mimicking the logical functionality that is performed in electrical circuits) etc. For example, control systems laws [20] are used for analyzing biological systems, i.e., genetic circuits and bio-controllers. The amalgamation of such interdisciplinary fields into synthetic biology [6] allows designing and analyzing these systems in an efficient manner.

The analysis of these systems, i.e., biological circuits and bio-controllers, requires modeling their dynamics, representing the interaction of their different components, using differential equations. Next, the transfer functions of these systems, providing their dynamics in the frequency domain are extracted from their block diagram representations [19], which are commonly used in control systems. Finally, the Laplace transform is used to perform the frequency domain (transfer function based) analysis of these systems based on their differential equation models.

Traditionally, these biological systems, i.e., biological circuits, networks and pathways, are analyzed using the paper-and-pencil proof [9] and computer based numerical [11] and simulation methods [12]. However, the paper-and-pencil proof methods are prone to human error and the chances of errors increase while analyzing the larger systems. Similarly, the computer-based numerical techniques are based on the unverified numerical algorithms that are present in the core of the associated tools. Also, the simulation-based methods suffer from the limited computational resources and computer memory issues. Thus, considering the safety-critical nature of biological systems, these conventional methods cannot be relied upon for their accurate analysis.

Formal methods [15] are computer-based mathematical techniques used for the modeling, specification and verification of the systems. They have been widely adopted for the rigorous analysis of the complex real-world systems [4, 21]. They are mainly of two types, i.e., model checking [15] and theorem proving [15]. Model checking involves the development of a state-space model

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

SAC '20, March 30–April 3, 2020, Brno, Czech Republic

© 2020 Association for Computing Machinery.

ACM ISBN 978-1-4503-6866-7/20/03...\$15.00

<https://doi.org/10.1145/3341105.3373993>

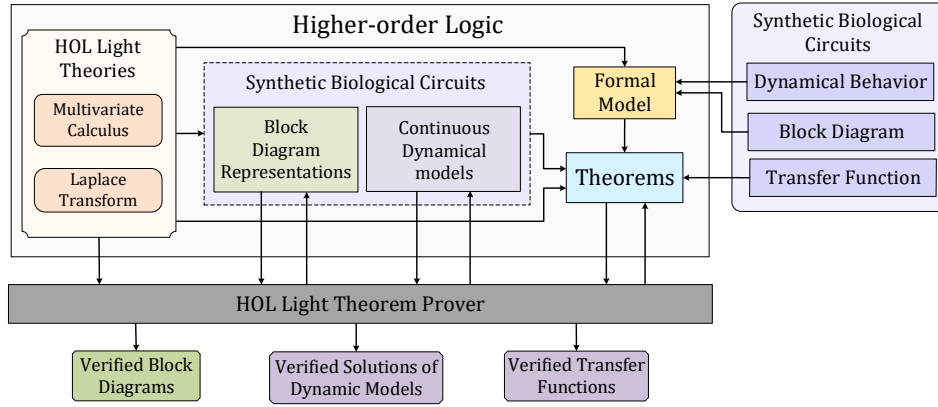


Figure 1: Proposed Framework

of the underlying system and the verification of its intended behavior by the properties specification in an appropriate logic. It has been extensively used in the area of synthetic biology for formally analyzing the biological circuits and their associated feedback controllers [7, 17, 26]. However, it suffers from the inherent state-space explosion problem [8] and thus is not suitable for analyzing larger systems. On the other hand, higher-order-logic theorem proving involves constructing a mathematical model of the system based on higher-order logic and verification of its intended properties using deductive reasoning. It has been used to formal reason about the biological systems and molecular pathways. Ahmad et al. [3] proposed a formalization of Zsyntax using HOL theorem prover. Moreover, they utilized their proposed framework for formally analyzing the TP53 degradation pathway and Glycolytic leading from D-Glucose to Fructose-1,6-bisphosphate. Recently, Rashid et al. [21] developed a framework, using the HOL Light theorem prover, which provides the formal support for the reaction kinetic based dynamical analysis of the biological systems. However, their proposed framework cannot model and analyze the genetic circuits.

In this paper, we propose to use higher-order-logic theorem proving [15] for formally analyzing the genetic circuits used in synthetic biology as shown in Figure 1. The first step is to model the continuous dynamics of the genetic circuits using differential equations. The next step requires the transfer function of these systems that can be obtained from their corresponding block diagram representations. Finally, the Laplace transform is used to perform the transfer function based analysis of the differential equation based models of these circuits. To illustrate the practical utilization of our proposed framework, we formally analyze the genetic circuits of activated and repressed expressions of protein using HOL Light.

The rest of the paper is organized as follows: We provide an introduction about the HOL Light theorem prover, multivariate calculus and the Laplace transform theories of HOL Light in Section 2. Section 3 presents the formalization of block diagram representations of the biological circuits. We provide our formal analysis of the genetic circuits of activated and repressed expressions of protein using HOL Light in Section 4. Finally, Section 5 concludes the paper.

2 PRELIMINARIES

This section presents a brief introduction to the HOL Light proof assistant and its multivariate calculus and the Laplace transform theories, which are extensively used in the rest of the paper.

2.1 Theorem Proving and HOL Light Theorem Prover

Theorem proving is a widely used formal verification method that involves developing the proofs of the mathematical theorems using a computer program (called *theorem prover*) [14]. Theorem proving systems have been commonly utilized for formally verifying the properties of both hardware and software systems [4, 24]. Based on the expressiveness requirement, these properties are modeled as theorems using propositional, first-order or higher-order logic. For example, the higher-order logic provides more expressiveness by allowing additional quantifiers. Moreover, it is best suited for conducting the mathematical analysis based on theories of multivariate calculus and the Laplace transform.

HOL Light [13] is an interactive theorem prover for developing the formal proofs of the mathematical concepts expressed in the form of theorems. It is implemented in Objective CAML (OCaml), which is a functional programming language, with an aim of automating the mathematical proofs [10]. It has a very small logical kernel of approximately 400 lines of OCaml code and has types, terms, axioms, inference rules and theorems as its main components, which are a part of its theory. Every new theorem in a theory is verified using the basic axioms and the primitive inference rules or already verified theorems, providing the soundness of this method.

2.2 Multivariable Calculus and Laplace Transform Theories

HOL Light presents an extensive support for analyzing the systems using theories of multivariable calculus and the Laplace transform. Table 1 provides some definitions from HOL Light's theory of the Laplace transform, which includes the Laplace transform, Laplace existence and the exponential-order conditions. Interested readers can refer to [21–23, 25] for more details about this theory. It is extensively utilized in our formal analysis of the biological circuits.

Table 1: Laplace Transform

Mathematical Form	Formalized Form
Laplace Transform	
$\mathcal{L}[f(t)] = F(s) = \int_0^\infty f(t)e^{-st}dt, s \in \mathbb{C}$	$\vdash_{def} \forall s f. \text{laplace_transform } f \ s = \text{integral } \{t \mid \&0 \leq \text{drop } t\} (\lambda t. \text{cexp } (-(s * Cx (\text{drop } t))) * f \ t)$
Laplace Existence	
f is piecewise smooth and is of exponential order on the positive real line	$\vdash_{def} \forall s f. \text{laplace_exists } f \ s \Leftrightarrow (\forall b. f \text{ piecewise_differentiable_on interval } [\text{lift } (\&0), \text{lift } b]) \wedge (\exists M a. \text{Re } s > \text{drop } a \wedge \text{exp_order_cond } f \ M \ a)$
Exponential-order Condition	
There exist a constant a and a positive constant M such that $ f(t) \leq Me^{at}$	$\vdash_{def} \forall f \ M \ a. \text{exp_order_cond } f \ M \ a \Leftrightarrow \&0 < M \wedge (\forall t. \&0 \leq t \Rightarrow f (\text{lift } t) \leq M * \text{exp } (\text{drop } a * t))$

3 FORMALIZATION OF BLOCK DIAGRAM REPRESENTATIONS OF BIOLOGICAL CIRCUITS

In this section, we present our formalization of block diagram representations of the biological circuits. These definitions enable us to formally model the block diagrams of a generic biological circuit in the s -domain and thus to find out the transfer function of any biological circuit. The presented formalization is basically inspired from the block diagrams of the control systems [2, 16].

Configuration 1: The transfer function of a sum of two components (subsystems) of a biological circuit, which can be any proteins or genes, in the case of a genetic circuit, is equal to the product of the transfer function of the individual components. We formalize this configuration for an arbitrary (N) number of components of a circuit as follows:

DEFINITION 3.1. Series Components

$$\vdash_{def} \forall C_1. \text{series_comp } [C_1; C_2; \dots; C_N] = \prod_{i=1}^N C_i$$

The function `series_comp` accepts the transfer functions of the individual components of the circuit as a list of complex numbers and returns the transfer function of the overall circuit as a product of all individual transfer functions.

Configuration 2: The summation junction for various components of a biological circuit is an addition module that provides the summation of the transfer functions of the individual components. We formalize this configuration for an arbitrary number (N) of components of a circuit, having transfer functions represented by a list of complex numbers as follows:

DEFINITION 3.2. Components as Summation Junction

$$\vdash_{def} \forall C_1. \text{summ_jun } [C_1; C_2; \dots; C_N] = \sum_{i=1}^N C_i$$

Configuration 3: The pickoff point configuration is the representation of a component of a biological circuit to a parallel branch of components. We model this configuration in HOL Light as follows:

DEFINITION 3.3. Components as Pickoff Point

$$\vdash_{def} \forall \alpha \ C_1. \text{pick_point } [C_1; C_2; \dots; C_N] = [\alpha * C_1; \alpha * C_2; \dots; \alpha * C_N]$$

The function `pick_point` accepts the transfer function of the first component as a complex number and the transfer functions of the components in parallel as a list of complex numbers, and returns the corresponding transfer functions corresponding to the equivalent block diagram representation as a list of complex numbers.

Configuration 4: The feedback block configuration is the fundamental representation for modeling the closed loop controllers for the biological circuits. Due to the presence of the feedback signal, it is primarily represented by an infinite summation of branches that consist of serially connected components. We formalize the transfer function of each branch as the following HOL Light function:

DEFINITION 3.4. Transfer Function of a Branch

$$\vdash_{def} \forall \alpha \ \beta \ N. \text{branch_tf } \alpha \ \beta \ N = \prod_{i=0}^N \text{series_comp } [\alpha; \beta]$$

The function `branch_tf` takes the forward path transfer function α (a protein or gene), the feedback path (feedback signal) transfer function β and the number of the branch (N), and returns a complex number representing the transfer function of the N^{th} branch.

Next, we formalize the feedback block representation using our function `branch_tf` as follows:

DEFINITION 3.5. Feedback Block Representation of Components

$$\vdash_{def} \forall \alpha \ \beta. \text{feedback_block } \alpha \ \beta = \text{series } [\alpha; \sum_{k=0}^{\infty} \text{branch } \alpha \ \beta \ k]$$

The function `feedback_block` accepts the forward path transfer function α and the feedback path transfer function β and returns the transfer function by forming the series network of the final forward path transfer function and the summation of all the possible infinite branches.

Our formalization of the foundational configurations, presented above [1], enables us to formally model the block diagram representations of the generic biological circuits as will be illustrated in the next section.

4 FORMAL ANALYSIS OF THE GENETIC CIRCUITS

Gene expression is a technique for transmitting information from the passive deoxyribonucleic acid (DNA) to the active proteins, which are widely used for performing majority of the tasks required for a human life at cellular level. The process of this transmission is performed in two steps. In the first step, a section of DNA is read out into ribonucleic acid (RNA) and is known as transcription. The second step, namely translation, involves conversions of a short strand of transcribed RNA into protein. This process is usually regulated in synthetic systems by transcription factors (TFs), which control the initiation rate of the transcription of a gene and its corresponding expression. TFs are of two types, namely activators and repressors. Activators increase the transcription rate, whereas the repressors inhibit transcription. We use our proposed formalization for formally analyzing the genetic circuits of the activated and repressed expressions of protein.

4.1 Activated Expression of Protein

The genetic circuit of the activated expression of protein is depicted in Figure 2(a). It involves the interaction of the incoming activating TF A with its promoter and the regulation of the expression of gene Y producing the protein Y. The block diagram representation of the activated expression is shown in Figure 2(c), by a gain block of $+Y_x^*$ with $x = A$.

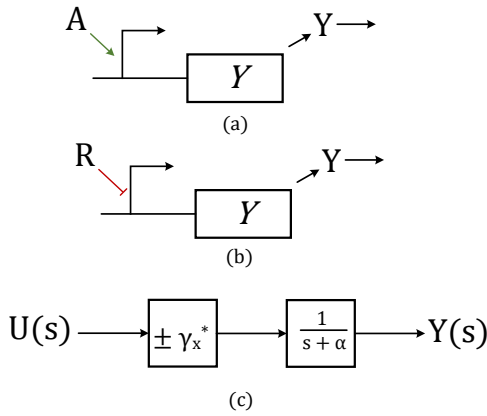


Figure 2: (a) Genetic Circuit Diagram of Activated Expression (b) Genetic Circuit Diagram of Repressed Expression (c) Block Diagram Representation for a Single Gene with Activated and Repressed Expressions

The dynamical model of the activated expression of protein is mathematically expressed as the following linear differential equation:

$$\frac{dy}{dt} + \alpha y = Y_A^* u \quad (1)$$

The corresponding transfer function is represented as:

$$\frac{Y(s)}{U(s)} = \frac{Y_A^*}{s + \alpha} \quad (2)$$

In order to model the dynamical behaviour, we first model the linear differential equation of order n as:

DEFINITION 4.1. Differential Equation of Order n

$$\vdash_{def} \forall n \ f \ t. \text{differ_equat_order_n} \ \text{lst} \ f \ t = \text{vsum} \ (\emptyset..n) \ (\lambda k. \text{EL} \ k \ [\alpha_1; \alpha_2; \dots; \alpha_k] \ * \text{higher_order_derivative} \ k \ f \ t)$$

The function `differ_equat_order_n` accepts the order of the linear differential equation n , list of coefficients `lst`, a differentiable function f and the differentiation variable t and returns the linear differential equation of order n .

Now, we model the dynamical behaviour of the activated expression as the following HOL Light function:

DEFINITION 4.2. Dynamical Model of Activated Expression

$$\vdash_{def} \forall \alpha. \text{olst_de_ae} \ \alpha = [\text{Cx} \ \alpha; \text{Cx} \ (\&1)]$$

$$\vdash_{def} \forall Y_A^*. \text{ilst_de_ae} \ Y_A^* = [\text{Cx} \ Y_A^*]$$

$$\vdash_{def} \text{differ_equat_ae} \ u \ y \ t \ \alpha \ Y_A^* \Leftrightarrow \text{differ_equat_order_n} \ 1 \ (\text{olst_de_ae} \ \alpha) \ y \ t = \text{differ_equat_order_n} \ 0 \ (\text{ilst_de_ae} \ Y_A^*) \ u \ t$$

To formally verify the transfer function of the activated expression based on its dynamical model, we first model its block diagram representation using our formalization in HOL Light as:

DEFINITION 4.3. Block Diagram Representation of Activated Expression

$$\vdash_{def} \forall \alpha \ Y_A^*. \text{bdr_ae} \ \alpha \ Y_A^* = \text{series_comp} \ \left[Y_A^*; \frac{\text{Cx}(\&1)}{s + \text{Cx} \ \alpha} \right]$$

Next, we verify the transfer function of the activated expression based on its block diagram representation as the following HOL Light theorem:

THEOREM 4.1. Transfer Function of Activated Expression

$$\vdash_{thm} \forall \alpha \ Y_A^*. \text{[A]}: (s + \text{Cx} \ \alpha) \neq \text{Cx} \ (\&0) \Rightarrow \text{bdr_ae} \ \alpha \ Y_A^* = \frac{Y_A^*}{s + \text{Cx} \ \alpha}$$

The proof of the above theorem is based on Definition 3.1 along with some arithmetic reasoning. Now, we formally verify the transfer function, obtained from Theorem 4.1 based on the dynamical model as follows:

THEOREM 4.2. Dynamical Model Implies Transfer Function

$$\vdash_{thm} \forall \alpha \ Y_A^* \ y \ u \ s. \text{[A}_1]: \&0 < Y_A^* \wedge \text{[A}_2]: \&0 < \alpha \wedge \text{[A}_3]: \forall t. \text{differentiable_higher_deriv} \ u \ y \ t \wedge \text{[A}_4]: \text{laplace_exists_higher_deriv} \ u \ y \wedge \text{[A}_5]: \text{zero_init_cond} \ u \ y \wedge \text{[A}_6]: (\forall t. \text{differ_equat_ae} \ u \ y \ t \ \alpha \ Y_A^*) \wedge \text{[A}_7]: (\text{laplace_transform} \ u \ s \neq \text{Cx} \ (\&0)) \wedge \text{[A}_8]: \left(\frac{\text{Cx}(\&1)}{s + \text{Cx} \ \alpha} \neq \text{Cx}(\&0) \right) \Rightarrow \frac{\text{laplace_transform} \ y \ s}{\text{laplace_transform} \ u \ s} = \frac{Y_A^*}{s + \text{Cx} \ \alpha}$$

Assumptions A₁-A₂ model the positivity conditions on circuit's parameters. Assumptions A₃-A₄ provide the differentiability and

condition of the existence of the Laplace transform of the higher-order derivative of y up to order 1 and the function u , respectively. Similarly, Assumption A_5 models the zero initial conditions for y and u . Assumption A_6 presents the dynamical behaviour of the activated expression. Assumptions A_7 – A_8 ensure that the denominator of the transfer function, presented in the conclusion of the above theorem, provides a valid expression. Finally, the conclusion provides the transfer function of the activated expression. The proof of the above theorem is done almost automatically using the automatic tactic `TF_TAC`, which is developed in our proposed formalization.

4.2 Repressed Expression of Protein

Figure 2(b) depicts the genetic circuit of the repressed expression of protein. It involves the interaction of the incoming repressing TF A with its promoter and the regulation of the expression of gene Y producing the protein Y . The block diagram representation of the repressed expression is shown in Figure 2(c), by a gain block of $-\gamma_R^*$ with $x = R$. The dynamical model of the repressed expression of protein is mathematically expressed as:

$$\frac{dy}{dt} + \alpha y = -\gamma_R^* u \quad (3)$$

The corresponding transfer function is represented as:

$$\frac{Y(s)}{U(s)} = \frac{-\gamma_R^*}{s + \alpha} \quad (4)$$

We formally verified the block diagram representation of repressed expression, its transfer function based on its block diagram and its dynamical model and the details about this verification can be found at [1].

Due to the undecidable nature of the higher-order logic, the verification results presented above involved manual interventions and human guidance. However, we developed the tactic `TF_TAC` [1] to automate the verification of the transfer functions of the biological circuits. The distinguishing feature of our formal analysis is that all of the verified theorems are of generic nature, i.e., all of the functions and variables are universally quantified and thus can be specialized based on the requirement of analyzing the biological circuits. Whereas, in the case of computer based simulations and numerical methods, we need to model each case individually. Moreover, the inherent soundness of the theorem proving method ensures that all the required assumptions are explicitly present along with the theorem. Similarly, due to the high expressiveness of the higher-order logic, our approach allows us to model the dynamics of the biological circuits involving differential and derivative (Equations (1), (3)) in their true form, whereas, in their model checking based analysis [26] they are discretized and modeled using a state-transition system, which may compromise the accuracy and completeness of the corresponding analysis.

5 SUMMARY

In this paper, we proposed a higher-order-logic theorem proving based framework for analyzing the dynamical behaviour of the biological circuits. We formalized the dynamical model and the block diagram representations of the biological circuits, such as the activated and repressed expressions of the protein. Finally, we formally

verified their transfer functions based on their dynamical models and their associated block diagram representations. In future, we aim to formally verify some more control systems properties of the biological circuits, such as, sensitivity and stability etc.

ACKNOWLEDGMENTS

This work was supported and funded by Kuwait University, Research Project No. (EO 01/18).

REFERENCES

- [1] Sa'ed Abed, Adnan Rashid, and Osman Hasan. 2020. Formal Analysis of the Biological Circuits using Higher-order-logic Theorem Proving. <http://save.seecs.nust.edu.pk/fabcholt/>.
- [2] Muhammad Ahmad and Osman Hasan. 2014. Formal Verification of Steady-state Errors in Unity-feedback Control Systems. In *Formal Methods for Industrial Critical Systems*. Springer, 1–15.
- [3] Sohaib Ahmad, Osman Hasan, Umair Siddique, and Sofiène Tahar. 2014. Formalization of Zsyntax to Reason About Molecular Pathways in HOL4. In *Brazilian Symposium on Formal Methods*. Springer, 32–47.
- [4] Mike Gordon, Albert Camilleri, and Tom Melham. 1986. *Hardware Verification Using Higher-Order Logic*. University of Cambridge, Computer Laboratory.
- [5] Bruce Alberts. 2017. *Molecular Biology of the Cell*. Garland Science.
- [6] Geoff Baldwin. 2012. *Synthetic Biology*. John Wiley & Sons.
- [7] Ezio Bartocci, Luca Bortolussi, and Laura Nenzi. 2013. A Temporal Logic Approach to Modular Design of Synthetic Biological Circuits. In *Computational Methods in Systems Biology*. Springer, 164–177.
- [8] Edmund M Clarke, Orna Grumberg, Somesh Jha, Yuan Lu, and Helmut Veith. 2001. Progress on the State Explosion Problem in Model Checking. In *Informatics*. Springer, 176–194.
- [9] Rokus A De Zeeuw, Glen B Baker, Ronald Thomson Coutts, and A Vercruysee. 1982. *Evaluation of Analytical Methods in Biological Systems: Analysis of Biogenic Amines*. Vol. 4. Elsevier.
- [10] Marshall Abrams et al. 2015. A History of OCaml. <http://ocaml.org/learn/history.html>.
- [11] Richard J Gaylord, Paul R Wellin, Bill Titus, Susan R McKay, and Wolfgang Christian. 1996. Computer Simulations with Mathematica: Explorations in Complex Physical and Biological Systems. *Computers in Physics* 10, 4 (1996), 349–350.
- [12] James W Haefner. 2005. *Modeling Biological Systems: Principles and Applications*. Springer Science & Business Media.
- [13] John Harrison. 1996. HOL Light: A Tutorial Introduction. In *Formal Methods in Computer-Aided Design (LNCS)*, Vol. 1166. Springer, 265–269.
- [14] John Harrison. 2009. *Handbook of Practical Logic and Automated Reasoning*. Cambridge University Press.
- [15] Osman Hasan and Sofiène Tahar. 2015. Formal Verification Methods. In *Encyclopedia of Information Science and Technology, Third Edition*. IGI Global, 7162–7170.
- [16] Constantine H Houpsis and Stuart N Sheldon. 2013. *Linear Control System Analysis and Design with MATLAB*. CRC Press.
- [17] Curtis Madsen, Chris J Myers, Nicholas Roehner, Chris Winstead, and Zhen Zhang. 2012. Utilizing Stochastic Model Checking to Analyze Genetic Circuits. In *IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*. IEEE, 379–386.
- [18] George F Muschler, Chizu Nakamoto, and Linda G Griffith. 2004. Engineering Principles of Clinical Cell-based Tissue Engineering. *JBJS* 86, 7 (2004), 1541–1558.
- [19] Norman S Nise. 2007. *Control Systems Engineering*. John Wiley & Sons.
- [20] Katsuhiko Ogata and Yanjuan Yang. 2002. *Modern Control Engineering*. Vol. 4. London.
- [21] Adnan Rashid and Osman Hasan. 2017. Formal Analysis of Linear Control Systems using Theorem Proving. In *International Conference on Formal Engineering Methods*. Springer, 345–361.
- [22] Adnan Rashid and Osman Hasan. 2017. Formalization of Transform Methods using HOL Light. In *Conference on Intelligent Computer Mathematics (LNAI)*, Vol. 10383. Springer, 319–332.
- [23] Adnan Rashid and Osman Hasan. 2018. Formalization of Lerchâ€™s Theorem using HOL Light. *Journal of Applied Logics* 5, 8 (2018), 1623–1652.
- [24] Johann M. Schumann. 2001. *Automated Theorem Proving in Software Engineering*. Springer Science & Business Media.
- [25] Syeda H. Taqdees and Osman Hasan. 2013. Formalization of Laplace Transform Using the Multivariable Calculus Theory of HOL-Light. In *Logic for Programming, Artificial Intelligence, and Reasoning (LNCS)*, Vol. 8312. Springer, 744–758.
- [26] Boyan Yordanov, Evan Appleton, Rishi Ganguly, Ebru Aydin Gol, Swati Banerjee Carr, Swapnil Bhatia, Traci Haddock, Calin Belta, and Douglas Densmore. 2012. Experimentally Driven Verification of Synthetic Biological Circuits. In *Design, Automation & Test in Europe*. IEEE, 236–241.