Techniques for automated parameter estimation in computational models of probabilistic systems

2016

Faraz Hussain
University of Central Florida

TECHNIQUES FOR AUTOMATED PARAMETER ESTIMATION IN COMPUTATIONAL MODELS OF PROBABILISTIC SYSTEMS

by

FARAZ HUSSAIN
M.S., Iowa State University, 2009
B.E., Birla Institute of Technology and Science, 2004

A dissertation submitted in partial fulfilment of the requirements
for the degree of Doctor of Philosophy
in the Department of Computer Science
in the College of Engineering and Computer Science
at the University of Central Florida
Orlando, Florida

Spring Term
2016

Major Professor: Sumit K. Jha

ABSTRACT

The main contribution of this dissertation is the design of two new algorithms for automatically synthesizing values of numerical parameters of computational models of complex stochastic systems such that the resultant model meets user-specified behavioral specifications. These algorithms are designed to operate on probabilistic systems – systems that, in general, behave differently under identical conditions. The algorithms work using an approach that combines formal verification and mathematical optimization to explore a model's parameter space.

The problem of determining whether a model instantiated with a given set of parameter values satisfies the desired specification is first defined using formal verification terminology, and then reformulated in terms of statistical hypothesis testing. Parameter space exploration involves determining the outcome of the hypothesis testing query for each parameter point and is guided using simulated annealing. The first algorithm uses the sequential probability ratio test (SPRT) to solve the hypothesis testing problems, whereas the second algorithm uses an approach based on Bayesian statistical model checking (BSMC).

The SPRT-based parameter synthesis algorithm was used to validate that a given model of glucose-insulin metabolism has the capability of representing diabetic behavior by synthesizing values of three parameters that ensure that the glucose-insulin subsystem spends at least 20 minutes in a diabetic scenario. The BSMC-based algorithm was used to discover the values of parameters in a physiological model of the acute inflammatory response that guarantee a set of desired clinical outcomes.

These two applications demonstrate how our algorithms use formal verification, statistical hypothesis testing and mathematical optimization to automatically synthesize parameters of complex probabilistic models in order to meet user-specified behavioral properties.

ACKNOWLEDGMENTS

I met my doctoral advisor Sumit Jha very serendepitously in the Fall of 2011 – I wasn't looking for an advisor at all. My then advisor, Gary Leavens, and I were hoping to prepare a proposal for a National Science Foundation (NSF) grant that would help support me. We needed an expert in model checking as co-PI[1], and Gary suggested I talk to Sumit, who had been a faculty member here at the University of Central Florida (UCF) for only about a year. After much resistance[2], I agreed to meet him. Long story short, we never submitted that proposal, but that meeting ended up changing the direction of my PhD.

Over the last four years, Sumit has been a friend, collaborator, and *mentor* who has opened my eyes to the not-so-apparent realities of the academic and research world. It may be an exaggeration to suggest that I wouldn't have graduated without his involvement – but not by much. Thanks also to Gary for continuing to serve on my PhD dissertation advisory committee. I would like to thank all committee members, Damla Turgut, Nizam Uddin, Gary and Sumit for carefully reading this dissertation and providing useful suggestions.

My doctoral work was supported by the NSF's Division of Computing and Communication Foundations (CCF) under the Software and Hardware Foundation (SHF) program (award #1422257). I also gratefully acknowledge financial support from UCF in the form of a Graduate Research Excellence Fellowship (Fall 2013 – Spring 2015) and the Graduate Dean's Dissertation Completion Fellowship (Spring 2016).

No man is an island, and I agree with Richard De Millo that much of science is a social process. I am privileged to have had many collaborators and mentors during my time at UCF.

[1]PI: Principal Investigator; term used for the person responsible for the technical direction of a project for which a grant is received from the NSF.

[2]I wanted a faculty member from my alma mater, Iowa State University, who had served on my Masters committee, as the co-PI.

The project on parameter estimation using the Sequential Probability Ratio Test (SPRT) and its application to a glucose-insulin model was done in collaboration with Christopher Langmead, Susmit Jha and Raj Dutta. A paper based on this work was presented at the *2nd International Conference on Computational Advances in Bio and Medical Sciences*[3]; an extended version was later published in the *International Journal of Bioinformatics Research and Applications*[4].

The project on parameter synthesis using Bayesian statistical model checking and its applications to a physiological model of acute inflammation was done in collaboration with Yoram Vodovotz, Joyeeta Dutta-Moscato and Qi Mi. A paper based on this work was presented at the *4th International Conference on Computational Advances in Bio and Medical Sciences*[5]; an extended version has now been published in *BMC Bioinformatics*[6].

It was a pleasure to work with Arvind Ramanathan and Laura Pullum on two projects: validation of epidemiological models, and automated verification of intelligent systems. Many thanks to Ashish Tiwari for inviting me to spend a summer at SRI International. Thanks to Narsingh Deo for his continuing support on matters ranging from parallel programming and graph theory to general career advice, and to Sumanta Pattanaik and Neslisah Torosdagli for our collaborative work on visualization of biomedical datasets.

[3]Faraz Hussain et al. "Parameter Discovery for Stochastic Biological Models against Temporal Behavioral Specifications using an SPRT based Metric for Simulated Annealing". In: *Proceedings of the 2nd IEEE International Conference on Computational Advances in Bio and Medical Sciences (ICCABS 2012)*. Las Vegas, NV. IEEE Computer Society, Feb. 2012, pp. 1–6. DOI: 10.1109/ICCABS.2012.6182640.

[4]Faraz Hussain et al. "Parameter discovery in stochastic biological models using simulated annealing and statistical model checking". In: *International Journal of Bioinformatics Research and Applications* 10.4/5 (2014), pp. 519–539. DOI: 10.1504/IJBRA.2014.062998.

[5]Faraz Hussain et al. "Parameter Discovery for Stochastic Computational Models in Systems Biology Using Bayesian Model Checking". In: *Proceedings of the 4th IEEE International Conference on Computational Advances in Bio and Medical Sciences (ICCABS 2014)*. Miami, FL. IEEE, June 2014, pp. 1–6. DOI: 10.1109/ICCABS.2014.6863925.

[6]Faraz Hussain et al. "Automated parameter estimation for biological models using Bayesian statistical model checking". In: *BMC Bioinformatics* 16(Suppl 17).S8 (2015), pp. 1–14. DOI: 10.1186/1471-2105-16-S17-S8.

Werner Deitl, Dan Zimmermann, Curt Clifton, and Vinay Deolalikar may not remember me, but they asked all the right questions about my research that woke me from up my slumber at a time when I was lost and throwing good money after bad money (in terms of time).

Surviving my very long time in graduate school would not have been possible without friendship and advice from many friends and colleagues. Many thanks to Neeraj Khanolkar, Satyadev Nandakumar, Gopalakrishnan Sivaprakasam, Steve Shaner, Mumtaz Shaikh, Nazar Khan, Nazim Ashraf, Nitish Korula, Deepak Ramachandrana, Nitish Nair, Zubair Ahmad, Rishi Arora, Benny Kenkireth, Sivaramakrishnan Sivasubramanian and Vaibhav Rajan for their help and advice. Over the last few years, I have also had numerous useful discussions with John Singleton, Rizwan Ashraf, Bernd Losert, Jonathan Jakes-Schauer and Zubir Husein – my sincere thanks to all.

My trips home to my native India have become increasingly infrequent, yet family members have been a source of strength and encouragement. Their bewilderment has long given way to (very genuine) concern over why I haven't graduated, despite my protestations that I have been making good progress. Thanks everyone. Once this is finally over, I hope to visit more often. Finally, I want to thank my parents for, well, *everything*, but most of all, their patience. This thesis is dedicated to them.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1: INTRODUCTION

This dissertation describes novel algorithmic techniques for automated parameter synthesis in computational models of probabilistic systems. A model is an abstraction of any real physical system such as a natural phenomenon or a man-made device. Its purpose is to help in the analysis of key properties of the system for a particular application. Engineers and researchers usually build mathematical models of systems to be able to use existing scientific techniques for analyzing the system. The model builder throws away details of the system that are not relevant for analyzing key properties of the application being considered. For any application, a model can be considered a reasonable representation of the actual system if it meets the following criteria:

- It can be used to explain information obtained from existing knowledge or from empirical observations, and,
- With reasonably high accuracy, it can predict the behavior of the system under a variety of circumstances (i.e. different inputs or under perturbations in the environment, etc.)

Recent advances in computational resources have led to increasing interest in algorithmic techniques for automated computational modeling and analysis of systems with complex behaviors. A well-known problem in computational modeling is that even though the fundamental structure of many models of natural, mechanical or electronic systems can be determined from the literature or via experiments, parts of the model remain unknown and almost impossible to determine experimentally [104, 97, 8]. In areas such as biochemistry and control theory, researchers spend considerable time and resources trying to find the numerical values of such *parameters* of computational models. For example, in a biological system represented as a set of ordinary differential equations (ODEs), the parameters might represent the kinetic rate constants of reactions represented by these ODEs [79] or when designing a control system, it may be required to determine those parameter values of system components that ensure safety when the system is operated [53].

Researchers often resort to brute-force methods to find appropriate parameters of their models so that the model has the desired properties. However, as the number of parameters increases, finding appropriate values becomes inefficient due to the complexity of high-dimensional state space exploration [15, §1.4]. There is a need for readily available tools that, given a model with a set of unknown parameters and a behavioral specification, can reliably discover values of these parameters, if any such valuations exist, or report infeasibility of the model ever satisfying the given specifications [64]. We address this problem by designing algorithms and tools for automatic parameter synthesis in models that are

- *parametric* (i.e. their behaviors are dependent upon a set of unknown, but constant parameters),
- *probabilistic* (i.e. in general, their behaviors vary even for a fixed valuation of all parameters), and
- *computational* (i.e. they can be executed on a computer and their behaviors observed).

We now formally define the problem of synthesizing numerical parameters in probabilistic systems.

Definition 1 (Parameter estimation problem). Given a parametric probabilistic model $\mathcal{M}(\omega)$ with unknown parameter $\omega \in \mathbb{R}^n$, a desired specification ϕ in a suitable temporal logic, and a required confidence level $\theta \in [0, 1]$, find a parameter valuation $\omega_0 \in \mathbb{R}^n$ such that $\mathcal{M}$ parameterized at ω_0 satisfies property ϕ with probability at least θ [denoted $\mathcal{M}(\omega_0) \models P_{\geq\theta}(\phi)$].

To solve this problem, we have designed two new algorithmic techniques that are discussed in Chapters 3 and 4. First, we provide some technical background to motivate the problem, and also discuss related work in the area.

CHAPTER 2: BACKGROUND

In the last few decades, massive increase in our dependence on electronic and computer-related devices has led to concern about the reliability of these systems. It is desirable that the systems in use be checked for correctness, especially in safety-critical systems like aviation-control systems. A distinction is made between *validation* (ensuring that the right system is being built) and *verification* (ensuring that the system is actually built according to the specifications). Most real-life systems have traditionally been checked using simulation-based testing [45, 68].

2.1 Formal Methods

For our work on parameter synthesis in probabilistic systems, we make extensive use of *formal methods* [23] – a set of techniques, tools and algorithms that help increase the reliability of hardware or software. Gupta provides an excellent survey on hardware verification techniques [49]. Cohn [26] discuses issues regarding the semantics of hardware verification using the example of an attempt to verify a microprocessor using the HOL system. Two important approaches for software verification [71] are *model checking* [22, 36] and *deductive verification* [56, 41, 29]. For deductive verification of a system, the properties of the system are expressed mathematically and axioms and proof rules from mathematical logic are used to verify system correctness. Much of the proof has to be done by hand, whereas part of it usually discharged to an external theorem prover [100]. Although this technique can be used to prove complex properties of infinite state systems, due to the complexity of the proof obligation, even the theorem prover may require manual intervention and guidance [50]. For algorithmically synthesizing parameters in stochastic models, we focus on techniques based on *model checking*.

2.2 Model Checking

Model checking is an important technique for the automated formal verification of finite-state models. We closely follow Clarke et al [22] when discussing the model checking problem and related ideas.

For finite state systems, or systems that can be abstracted to a finite state system, model checking is often the preferred approach for formal verification. One advantage of model checking is that it is an automatic process of verifying a finite-state model against a behavioral specification expressed in a temporal logic [91]. Clarke and Kurshan provide early examples of significant industrial applications of model checking tools [23].

A model checker needs to be provided the model in the form of a finite state transition system, and the property to be verified needs to be expressed in an appropriate temporal logic. It then exhaustively explores the entire state space of the model to verify the given property. If the property is not true, the model checker returns a counter-example i.e. a trace in of the transition system showing how the property can be violated. The counterexample serves as proof of the model not meeting the given specification. A counterexample is then usually used to modify the model being developed according to the given specification, although the fault may lie in the specification instead [22].

One drawback of model checking is that it suffers from the state-explosion problem, i.e. for models of large sizes, the state space is becomes too large to explore, and the technique does not scale. To mitigate this problem researchers have proposed a number of techniques that reduce the state space the model checking algorithm has to explore during verification. Such techniques include include partial order reduction abstraction and symmetry reduction [22].

Probabilistic model checking is a technique for automatic verification of stochastic models against specifications in temporal logic [51, 27]. We make extensive use of the *statistical* methods for probabilistic model checking [112] in developing new techniques for automatic parameter synthesis.

2.3 Model Synthesis

The problem of synthesizing programs from specification is not new. Manna and Waldinger [85] described the use of theorem provers in synthesizing programs from specifications where the main idea is to extract programs from proofs. They also discussed the *automatic debugging* problem i.e. building a tool that not only verifies, but also correct code written by programmers. They also later provided a more detailed treatment of deductive techniques for program synthesis by the same authors that emphasizes the use of mathematical induction for dealing with recursion [84]. The advances in formal verification techniques and the availability of efficient tools for constraint solving has led to renewed interest in using verification techniques for synthesis. The main idea is define the synthesis problem using formal specifications and then reduce it to a constraint solving problem that can be solved using off-the-shelf constraint solving tool [47]. Techniques for automated synthesis of code fragments and control logic for cyberphysical systems that use a combination of inductive and deductive inference have also been developed [65].

2.4 Parameter Synthesis

The problem of finding model parameters from experimental data has numerous applications in science and engineering and has been widely studied [104, 8]. For models in computational systems biology, measuring most quantitative parameters is an important challenge because usually only some parameters are experimentally measured and the rest require fitting [79]. Common methods of building models involve the identification of parameter values using estimation algorithms that allow fitting model parameters to match experimental data [7].

Given a fundamental understanding of an unknown model m's behavior G and data d, the inverse problem is to find m given d. If the number of parameters to be determined is finite (say, an n-vector) and the available data points are finite (say, an m-vector), we refer to it as the *discrete inverse problem* or a *parameter estimation problem* represented by the system of equations $G(\mathbf{m}) = \mathbf{d}$. If the model and data are functions, estimating m from d is referred to as the *continuous inverse problem* [104].

Donaldson and Gilbert use a combination of model checking and genetic algorithms [89] for parameter estimation in biochemical pathways [31]. They define a new probabilistic temporal logic (PLTLc) by incorporating quantitative and numerical aspects of $\exists$-constraint LTL [38] to define a metric for describing the distance between a model's behavior and the desired behavior that is used during the genetic algorithm driven exploration of the parameter space.

Batt et al [12] have developed a symbolic model checking-based techniques for parameter search in piecewise-affine differential equation (PADE) models of regulatory networks by which they use for finding parameters in the IRMA synthetic network [20].

In a significant control theoretic application, Henzinger and Wong-Toi [53] used the model checker HYTECH [52] to synthesize parameters for a model of a steam boiler expressed as a hybrid automaton by defining the parameter synthesis problem in terms of reachability in a linear hybrid automaton (LHA) [4]. They use a linear hybrid automaton to abstract away nonlinear behaviors of the model in order to make it more amenable to automated algorithmic analysis while ensuring that the approximation preserves key properties that are to be verified.

For linear hybrid automata, Frehse et al [42] have developed a technique for parameter synthesis based on the popular counterexample guided abstraction refinement (CEGAR) [25] model checking approach. They given an algorithm to determine the values of a set of design parameters of a LHA for which no bad locations can be reached. The key idea is to augment the traditional CEGAR counterexample feasibility check by an operation that determines constraints on the parameters that make that particular (feasible) counterexample infeasible. The feasibility check is performed using efficient techniques based on linear programming.

Calzone et al use the abstract machine BIOCHAM for model refinement and parameter discovery in a cell cycle model where the property is expressed in LTL and the numerical range of the parameters is provided [18] as input. BIOCHAM contains languages to describe both the model and the behavioral specification at different levels using different temporal logics [19].

Moles et al [90] define the parameter estimation problem as a nonlinear programming (NLP) problem with differential-algebraic constraints and survey the use of global optimization (GO) methods for parameter estimation in biochemical pathways. For parameter estimation in computational biology, Lillacci and Khammash have developed a technique that uses a Kalman filter for determining an initial estimate of the parameter value, uses a statistical test to check its reliability and then solves an optimization problem to refine the initial estimate [79]. For certain nonlinear dynamical systems, Donze et al give a parameter synthesis algorithm [32] but do not provide any formal guarantee of correctness.

Applying formal verification techniques to structural biology, Langmead and Jha [75] use symbolic model checking to predict the kinetics of protein folding using CTL model checking. They perform a search for the protein folding kinetics by using the CTL extensions made available by the probabilistic model checker PRISM [55].

An important family of approaches for global optimization are simulated annealing techniques [92] . S-systems are representations of complex biological systems modeled as a set of nonlinear ordinary differential equations. Gonzales et al successfully performed parameter estimation for a number of biochemical S-system models using simulated annealing by using a perturbation function that uses the current optimization error to determine the magnitude of parameter perturbation [46].

Boolean networks are often used to model biological systems [66]. Langmead [76] demonstrates the use of bounded model checking [14] for synthesizing control policies to drive a boolean network from an initial to a specified final state.

Continuous time Markov chains (CTMCs) are often used to mathematically model biochemical reaction networks [43]. Jha and Langmead use statistical model checking [78] and abstraction refinement [25] to develop a set of algorithms for synthesizing parameters in a CTMC model with respect to a given finitely monitorable behavioral specification [64].

In the next two chapters we discuss our novel algorithms for automated parameter estimation in probabilistic models.

CHAPTER 3: PARAMETER ESTIMATION USING THE SPRT[1]

Stochastic models are increasingly used to study the behavior of biochemical systems. While the structure of such models is often readily available from the literature, unknown quantitative features of the model are incorporated into the model as parameters. In this chapter, we make two important contributions toward addressing the parameter estimation problem in probabilistic systems (see Definition 1):

- We describe a new algorithm for automatically synthesizing parameters of stochastic computational models from experimental observations that uses a combination of simulated annealing and the sequential probability ratio test (SPRT) to reduce the number of samples required for discovering the correct parameter values. Figure 3.1 pictorially describes the parameter estimation problem in stochastic models given experimental facts and time-series data. This algorithm uses the property that during the simulated annealing-based parameter exploration process for a parametric model, *if the sequential probability ratio test (SPRT) rejects the null hypothesis, the expected number of samples required is proportional to the probability with which the model satisfies the given specification.*
- We apply this algorithm to a complex model of glucose insulin metabolism used for studying artificial pancreata. This case study shows that the model is capable of reproducing pancreatic-induced diabetes by a suitable reparameterization of three parameters related to the pancreas in the model.

[1]A preliminary version of the research reported in this chapter was presented at the *2nd International Conference on Computational Advances in Bio and Medical Sciences* [57]; an extended version was later published in the *International Journal of Bioinformatics Research and Applications* [58].

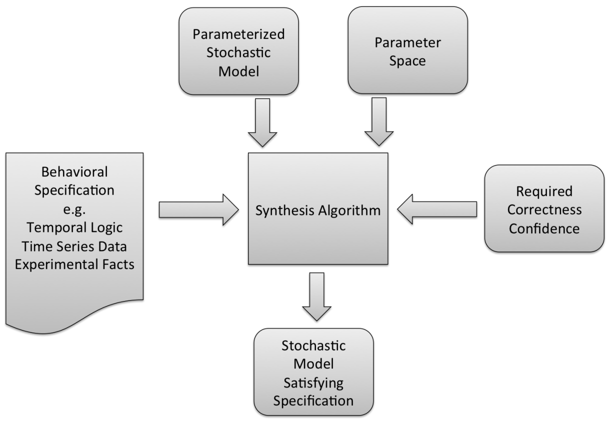

Parameterized Stochastic Model
Parameter Space
Behavioral Specification e.g. Temporal Logic Time Series Data Experimental Facts
Synthesis Algorithm
Required Correctness Confidence
Stochastic Model Satisfying Specification

Figure 3.1: Problem description: Given behavioral specifications about the computational model of a system, the parametric stochastic model itself, and a correctness confidence, find the values of the parameters that enable the model to satisfy the specification.

3.1 Background

The computational modeling of the precise dynamics of biochemical systems involves the modeling and analysis of complex continuous time Markov chain models. Such detailed biochemical models are often reduced to readily analyzable and succinct models like stochastic differential equations (SDEs) and discrete time Markov chains (DTMCs). Recent interest in the development of computer aided design (CAD) techniques for biomedical devices has led to the development of heterogeneous models that include a stochastic biochemical model coupled with an external deterministic controller. The design and formal verification of such biomedical devices requires the ability to model, analyze and verify complex stochastic models of cyberphysical systems.

The structure of such stochastic models can often be determined from first principles and a survey of existing biochemical literature. However, several quantitative features of such models cannot easily be obtained from the literature or inferred from experimental data. The discovery of such quantitative parameters of computational models from observed experimental facts remains the subject of ongoing research in computational systems biology.

The field of stochastic modeling has emerged to overcome the inherent limitations of deterministic modeling. Such a modeling approach permits randomness in the behavior of the system, and often uses sampling-based statistical inference to find the probability distribution of potential outcomes. Research in systems biology has greatly benefited from the existing literature in stochastic modeling, and has also inspired new scientific expeditions into the realm of stochastic modeling, analysis and verification.

Indeed, *in silico* modeling has been particularly useful in developing a systems view of biology. Models for whole-cell analysis, drug discovery for tuberculosis, control of Type 1 diabetes, sequence analysis, enzyme interaction with drugs, and prediction of blood-secretory proteins are some of the success stories in computational systems biology. However, several components of these models are not available from the literature or using experimental data. In such a scenario, model designers include missing information as parameters of the model. The number of parameters increases with the size and complexity of the model, and it becomes increasingly difficult to determine the value of these parameters for large and detailed models of biological systems.

The discovery of parameters for stochastic models has been carried out using various approaches [42]. Different estimation techniques have been adopted by researchers for finding parameters [93] of stochastic biochemical reactions [43]. Estimators used for deterministic models have also been extended to stochastic models [107]. Considerable research has been directed toward the use of statistical hypothesis testing for verification of stochastic models [64, 112], including those arising in systems biology.

In this section, we discuss the various classes of stochastic models that can benefit from our parameter discovery algorithm. We also describe a specification formalism for representing facts observed from experimental data that describes the properties that the stochastic model with the synthesized parameters must satisfy. Finally, we briefly survey the literature on the sequential probability ratio test (SPRT) and its relationship to statistical estimation.

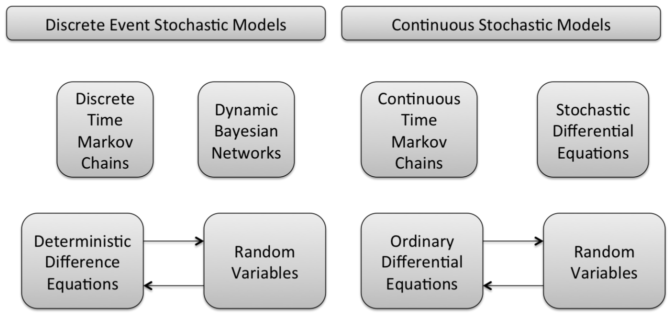

Discrete Event Stochastic Models
Continuous Stochastic Models
Discrete Time Markov Chains
Dynamic Bayesian Networks
Continuous Time Markov Chains
Stochastic Differential Equations
Deterministic Difference Equations
Random Variables
Ordinary Differential Equations
Random Variables

Figure 3.2: Stochastic models: The algorithm presented in this chapter is applicable to both discrete and continuous time stochastic models. CTMCs are particularly important for studying biochemical systems, while ODEs interacting with random variables naturally model a number of cyberphysical systems.

3.1.1 Stochastic Models

Our proposed algorithm can be applied to several classes of parametric stochastic models, including continuous-time Markov chains (CTMCs), stochastic differential equations (SDEs), jump diffusion processes, and heterogeneous models consisting of stochastic processes interacting with deterministic models like ordinary differential equations (ODEs).

Figure 3.2 illustrates the various types of stochastic models whose parameters can be discovered using our algorithm. Discrete-time Markov chains (DTMCs) are models whose state-space can be indexed by a countable set. Each transition between states of a DTMC is associated with a finite probability. Dynamic Bayesian networks (DBNs) are representations of probabilistic models amenable to analysis using statistical inference and machine learning.

All of these models are discrete event stochastic systems, where the behavior of the system over a period of time can be completely described by a finite or countable numerical sequence of values assigned to system variables. Another interesting class of systems is formed by a set of deterministic differential equations interacting with a set of random variables or stochastic processes.

Continuous-time Markov chains and stochastic differential equations represent stochastic systems that evolve continuously in time. Two kinds of stochastic models that are often used to model biochemical and cyberphysical systems are of special interest to us and merit deeper discussion:

Continuous-Time Markov Chains Biochemical systems consisting of a set of biochemical reactions in a homogeneous well-mixed volume can be precisely modeled using continuous-time Markov chains. CTMC models are often simulated using Gillespie's stochastic simulation algorithm [43]. However, the values of the rate constants or kinetic parameters that determine the transition probability in CTMC models are very difficult to obtain from first principles. In many cases, they are also difficult to measure in an *in vivo* setting. Our technique can be used to discover kinetic parameters for CTMC models of biochemical systems from data gathered by empirical observations.

ODEs and random variables A biomedical cyberphysical system is often modeled using a system of deterministic ordinary differential equations interacting with random variables to represent the biochemical system and the external controller [82]. However, several parameters and variables in such models are either unknown in biological literature or they vary substantially from one individual to another in a population. In both these cases, such parameters and variables are modeled as random variables or stochastic processes. The resulting complex cyberphysical model is a continuous time stochastic system. The technique described in this section can also be used to discover parameters of models of such complex cyberphysical biomedical systems.

3.1.2 Behavioral Specifications

The behavioral specification of stochastic biochemical and biomedical models requires complex temporal reasoning. We use probabilistic bounded linear temporal logic (PBLTL) to specify the behavior of such systems. Temporal logics can be used to formally describe the use of tense and various forms of causality in natural languages.

We first define the syntax and semantics of Bounded Linear Temporal Logic (BLTL) [39]. A Bounded Linear Temporal Logic specification is a set of predicates connected using Boolean and temporal operators. The syntax of the logic is given by the following grammar:

$$\langle \phi \rangle ::= \; x \leq v \mid x \geq v \mid \phi_1 \vee \phi_2 \mid \phi_1 \wedge \phi_2 \mid \neg \phi_1 \mid \phi_1 \mathbf{U}^t \, \phi_2$$

where $\mathcal{V}$ is set of discrete-valued variables, $x \in \mathcal{V}$, $v \in \mathbb{R}$, and $t \in \mathbb{R}_{\geq 0}$ denotes time. We can define additional temporal operators such as $\mathbf{F}^t \, \psi = \mathbf{true} \mathbf{U}^t \, \psi$, or $\mathbf{F}^t \, \psi = \neg \mathbf{F}^t \, \neg \psi$. The formula $\mathbf{F}^t \, \psi$ implies that ψ holds sometime within t time units. The formula $\mathbf{G}^t \, \psi$ implies that ψ holds at all moments for the next t time units into the future. The fact that a path $\wp$ satisfies the BLTL property ϕ is denoted by $\wp \vDash \phi$. Let $\wp = (y_0, \tau_0), (y_1, \tau_1), \ldots$ be an execution of the model along states $y_0, y_1, \ldots$ with durations $\tau_0, \tau_1, \ldots \in \mathbb{R}$. We denote the path starting at state i by $\wp^i$ (in particular, $\wp^0$ denotes the original execution $\wp$). The value of the state variable x in $\wp$ at the state i is denoted by $V(\wp, i, x)$.

Specifications about biochemical and biomedical systems are often probabilistic in nature. For example, it may be required that a respirator will not allow the oxygen level in the blood to fall below 90% of its average value for more than 4 seconds with 99.9999% probability. Such behaviors are naturally expressed as probabilistic bounded linear temporal logic (PBLTL) specifications.

If ϕ is a bounded linear temporal logic (BLTL) specification, $Pr_{\geq \rho}(\phi)$ is a probabilistic bounded linear temporal logic specification. The model $\mathcal{M}$ is said to satisfy the PBLTL specification $Pr_{\geq \rho}(\phi)$ if at least ρ fraction of independently drawn random behaviors observed from the model satisfy the BLTL specification ϕ.

3.1.3 Sequential Probability Ratio Test (SPRT)

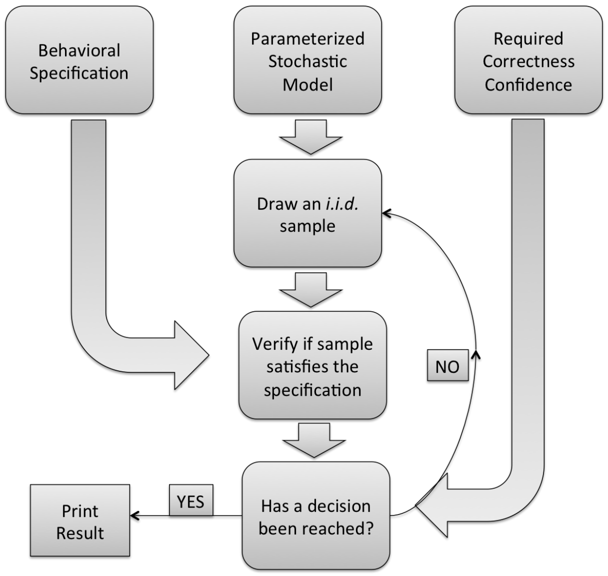

Behavioral Specification
Parameterized Stochastic Model
Required Correctness Confidence
Draw an *i.i.d.* sample
Verify if sample satisfies the specification
NO
Has a decision been reached?
YES
Print Result

Figure 3.3: Overview of the SPRT-based parameter estimation technique: Given a behavioral specification and a parametric stochastic model, our SPRT-based parameter discovery technique returns a parameter assignment at which the model satisfies the specification.

Given a stochastic model $\mathcal{M}$ and a specification ϕ, let p be the unknown probability with which the model satisfies the specification. Let ρ be the probability threshold with which the model is expected to satisfy a specification ϕ (i.e., $\mathcal{M} \models Pr_{\geq\rho}(\phi)$). Thus, our objective is to determine which of the hypotheses $H : p \geq \rho$, $K : p < \rho$ is true. The result of the SPRT procedure is correct in a probabilistic sense. There can be two kinds of errors in the answer produced by the SPRT procedure — Type I and Type II [106, §1.3.3]. A Type I error is the condition of rejecting the null hypothesis H_0 when it is actually true. Accepting the null hypothesis H_0 when the alternate hypothesis H_1 is true results in a Type II error.

In order to control the Type I and Type II error probabilities, we follow Younes [110] an introduce an *indifference region* $2 * \epsilon$, and define two thresholds $p_1 = \rho - \epsilon$ and $p_0 = \rho + \epsilon$. We then relax the test to use the following hypotheses: $H_0 : \; p \geq p_0$, and $H_1 : \; p \leq p_1$ [112]. Note that $0 < p_1 < \rho < p_0 < 1$.

The SPRT uses a sequential sampling procedure where the number of observations is determined by the outcome of the observations themselves. The goal of a sequential testing procedure is to reduce the number of samples required to decide whether to reject the null hypothesis. The sequential probability ratio test is a sequential sampling test in which each observations can lead to one of three outcomes: (i) H_0 is accepted, (i) H_1 is accepted, (i) additional observations are needed to accept either hypothesis and hence, the procedure continues. This process of generating additional samples continues until a decision is made to accept one of the hypotheses. An overview of our parameter synthesis technique that uses the SPRT is illustrated in Figure 3.3.

The SPRT bounds the probability of making Type I and Type II errors during hypothesis testing by two constants α and β that are used to parameterize the test, denoted $S(A, B)$. Note that A and B are calculated using α and β as described in [106, §3.3].

3.2 The Algorithm

We present a new parameter discovery algorithm for stochastic models that brings together the sequential probability ratio test (SPRT) and the simulated annealing procedure. The algorithm uses fewer samples than a traditional approach based on statistical estimation and simulated annealing [46]. The correctness proof of our algorithm is based on the fact that the number of samples used by this algorithm is related to the fitness value at a parameter point during the simulated annealing based exploration of the parameter space. We later formally prove the correctness proof of our algorithm (§3.4).

Figure 3.1 is a pictorial representation of the parameter discovery problem (also see Definition 1). There are three inputs to our algorithm:

- *Parametric Stochastic Model*: The algorithm discovers the parameter values for parametric stochastic models. In this setting, a parameter is a model variable whose value does not evolve during the execution of the model. As discussed earlier, our approach can be applied to a number of stochastic models, including those useful for biochemical and biomedical applications. Note that the algorithm requires that the possible space of all possible parameter values for the model be defined. A bounded (but not necessarily finite) parameter space is a requirement to ensure that the algorithm eventually terminates (§3.4.2).

- *Behavioral Specification*: The user provides a probabilistic specification about the behavior of the stochastic biological system. The specification may be provided as probabilistic bounded linear temporal logic (PBLTL) formula, or it may be available as extreme-scale time series data collected by observing a number of experiments. Various classes of experimental data and observed facts can be translated into variants of temporal logic.

- *Confidence*: The algorithm also accepts a probabilistic confidence as an input. Simulation based analysis of stochastic models requires establishing a confidence on the probability of a model satisfying a specification. This parameter (whose value should be close to 1) determines the confidence with which the algorithm must ascertain all probability estimates.

This algorithm builds on the classical simulated annealing metaheuristic (see Algorithm 3.1). Simulated annealing is a stochastic optimization method for finding the global minimum of a system that possibly has several local minima. It is a probabilistic version of the gradient descent algorithm where, instead of moving along the gradient, the algorithm decides the optimization steps stochastically [21]. A global minimum of a system is located by moving stochastically through the function space, based on the value of an objective function [92]. To escape local minima, simulated annealing uses the well-known metropolis-technique [88].

Algorithm 3.1 Simulated Annealing

```
Require: Parameter space $\Omega$, Objective function $E : \Omega \rightarrow \mathbb{R}$, Temperature Cooling Schedule
  $T : \mathbb{N} \rightarrow (0, \infty)$,
  Starting Temperature $t$, Stopping Temperature $t_0$.
  $\omega$ = Pick a random point in $\Omega$.
  $E(\omega) = \infty$
  while $t \geq t_0$ do
    Select a neighbor $\omega'$ randomly
    if $E(\omega') \leq E(\omega)$ then
      $\omega \leftarrow \omega'$
    end if
    if $E(\omega') > E(\omega)$ then
      $\omega \leftarrow \omega'$ with probability $e^{-(E(\omega') - E(\omega))/t}$
    end if
    $t = T(t)$
  end while
Ensure: Algorithm stops at $\omega^*$ that minimizes $E(\omega)$.
```

In most real optimization problems, an estimation of the objective function is made using statistical estimation and other approaches. Several heuristic estimation methods have been used for computing the probability (objective function) of a model satisfying a specification during simulated annealing. However, these estimation based methods require a large number of samples and can not be practically used for parameter discovery.

Algorithm 3.1 illustrates the classical simulated annealing algorithm. Given a parameter space Ω, the algorithm seeks to find the parameter value ω^* such that $E(\omega^*)$ represents a global minimum. The algorithm always accepts a better value for ω^* and also accepts a worse value with probability $e^{-(E(\omega') - E(\omega))/t}$. The probability of accepting a worse parameter value gets smaller with time.

The above mentioned algorithm reaches an optimal value when $E(\omega^*)$ and $E(\omega)$ are estimated correctly. The estimation procedures are tedious and challenging for stochastic systems due to the presence of randomness and the consequent necessity of observing millions of model simulations before the value of the fitness function at a given parameter point can be adequately estimated.

However, for our parameter discovery technique such a precise estimation of the probability values is not useful for most of the parameter space being explored. In reference to statistical model checking, [112] also notes that exact calculations can be replaced by a asking a weaker questions and hence making the verification procedure more efficient. We propose the use of the sequential probability ratio test (SPRT) for deciding if a parameter value is interesting *without* explicitly estimating the value of the fitness function at this parameter point. It also enables us to construct a computationally inexpensive objective function for sequential annealing. This approach helps in efficiently comparing the various states of the system towards obtaining a global minimum.

Algorithm 3.2 uses the number of simulations needed by the SPRT procedure as a metric for guiding the simulated annealing based parameter synthesis algorithm. We show that such an approach is correct. In particular, we establish a relationship between the number of samples used by the SPRT-based procedure and the probability with which a parametric model satisfies a behavioral specification.

Claim 1. Let p be the probability with which the model $\mathcal{M}$ satisfies the specification ϕ. Given the null hypothesis $H_0 : p \geq p_0$ and the alternate hypothesis $H_1 : p \leq p_1$, and error thresholds α and β, if $SPRT(A, B)$ rejects the null hypothesis, the average number of samples observed by our SPRT-based algorithm increases as p increases. (A, B are calculated from the α, β described by Wald [106].)

A proof of this claim is shown later (§3.4.1). Note that Algorithm 3.2 *does not actually use the exact value of the probability but merely needs to compare the probability for different choices of parameter values for which it uses the number of samples required by the SPRT* instead of calculating the actual probabilities. As such, any other (easily) computable metric that preserves the ordering relationship among parameter values would be sufficient. Thus, Algorithm 3.2 replaces the computation of the fitness value at a given point with the computation of the number of samples required by the hypothesis testing procedure.

3.3 Application: Validating Artificial Pancreata

The synthesis of parameters for biochemical and biomedical models is the primary focus of our research. We developed a parallel CUDA based implementation of a well-studied glucose-insulin model [82] that is used to perform in silico validation of artificial pancreata. We simulated a population of patients whose glucose intake was modeled as a normal distribution.

Our approach does not require us to fix *a priori* the size of the in-silico patient population. Instead, the size of the in-silico population depends on the region of the parameter space that the algorithm is exploring. Such an adaptive use of in silico population size has not been reported before to the best of our knowledge. Three parameters of the glucose insulin metabolism model determine the influence of the pancreas on the glucose-insulin dynamics, viz. (a) pancreatic responsivity to glucose rate of change, (b) delay between glucose signal and insulin secretion, and, (c) pancreatic response to glucose.

We synthesized parameters that ensure that the glucose-insulin subsystem model spends at least 20 minutes in a diabetic scenario, where the glucose concentration in the blood is above 140 or below 80.

The results of our synthesis algorithm are presented in Figure 3.4 (parameter: pancreatic responsivity to glucose rate of change), Figure 3.5 (parameter: delay between the glucose signal and insulin secretion) and Figure 3.6 (parameter: pancreatic responsivity to glucose).

Thus, we show that the model is capable of reproducing pancreatic-induced diabetes by a suitable parameterization of three parameters related to the pancreas in the model.

3.4 Algorithm Analysis

We present a proof of partial correctness of this algorithm (§3.2) and discuss its termination.

3.4.1 Proof of Partial Correctness

In this section, we study the relationship between the number of samples required by the SPRT algorithm and the probability of a model satisfying a given specification. Let p be the probability with which the model $\mathcal{M}$ satisfies the specification ϕ and ρ is the minimum desired probability with which the model is required to satisfy the specification.

The SPRT algorithm determines which of the following two hypotheses should be rejected:

$$\begin{aligned} \text{Null Hypothesis} \quad & H_0: \quad p \geq p_0 \\ \text{Alternate Hypothesis} \quad & H_1: \quad p \leq p_1 \end{aligned}$$

Note that $0 \leq p_1 < p_0 < 1$, $p_0 = \rho + \epsilon$, and $p_1 = \rho - \epsilon$, where 2ϵ is the indifference region [112]. Given independent and identically distributed (i.i.d.) samples x_i (ordered by index)

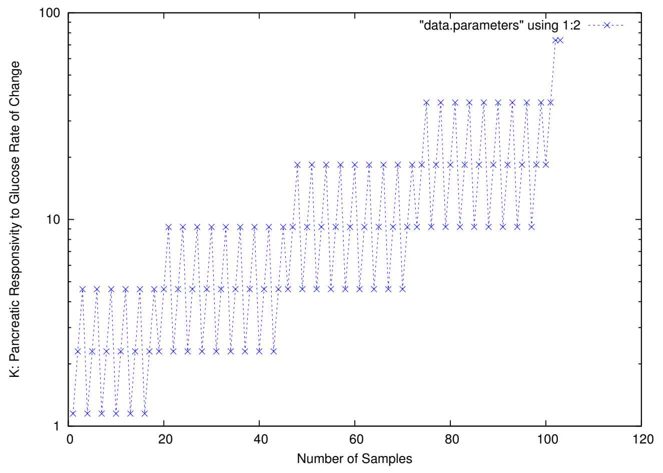


Figure 3.4: Synthesis results for the parameter 'pancreatic responsivity to glucose rate of change': The figure shows values of the first parameter of the glucose-insulin model whose value was discovered by our algorithm.

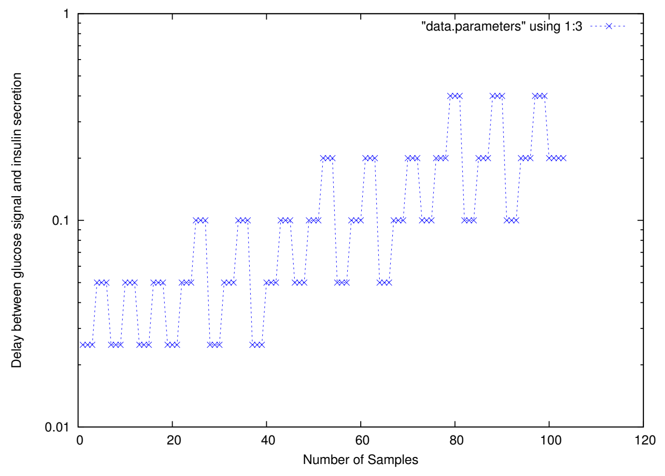


Figure 3.5: Synthesis results for the parameter 'delay between the glucose signal and insulin secretion': The figure shows values of the second parameter of the glucose-insulin model whose value was discovered by our algorithm.

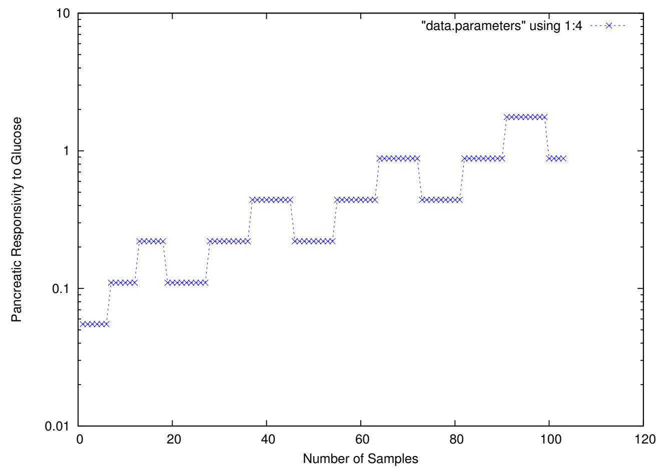


Figure 3.6: Synthesis results for the parameter 'pancreatic responsivity to glucose': The figure shows values of the third parameter of the glucose-insulin model whose value was discovered by our algorithm.

from the model $\mathcal{M}$, the SPRT procedure defines the following auxiliary quantities:

$$\overline{z_i} = \log\left(\frac{1-p_1}{1-p_0}\right) + x_i \log\left(\frac{p_1(1-p_0)}{p_0(1-p_1)}\right) \tag{3.1}$$

$$Z_n = \sum_{i=1}^{n} \overline{z_i} \tag{3.2}$$

The $SPRT(A, B)$ test accepts the null hypothesis H_0 if $Z_n \leq log(B)$, rejects the null hypothesis H_0 if $Z_n \geq log(A)$ (where $A = \frac{1-\beta}{\alpha}$ and $B = \frac{\beta}{1-\alpha}$), and continues making i.i.d. observations otherwise [112]. After n Bernoulli trials $x_1, x_2 \ldots x_n$ with the successes defined as $m = \Sigma_{i=1}^{n} x_i$, the SPRT calculates the following quantity:

$$\begin{aligned}
f_n &= \frac{\binom{n}{m} p_1^m (1-p_1)^{n-m}}{\binom{n}{m} p_0^m (1-p_0)^{n-m}} \\
&= \left(\frac{p_1}{p_0}\right)^m \left(\frac{1-p_1}{1-p_0}\right)^{n-m} \\
&= \left(\frac{p_1}{p_0}\right)^m \left(\frac{1-p_1}{1-p_0}\right)^{n} \left(\frac{1-p_1}{1-p_0}\right)^{-m} \\
\therefore f_n &= \left(\frac{p_1(1-p_0)}{p_0(1-p_1)}\right)^m \left(\frac{1-p_1}{1-p_0}\right)^{n} \qquad (3.3)
\end{aligned}$$

$$\begin{aligned}
\log(f_n) &= \log\left(\left(\frac{p_1(1-p_0)}{p_0(1-p_1)}\right)^m \left(\frac{1-p_1}{1-p_0}\right)^{n}\right) \\
&= \log\left(\left(\frac{p_1(1-p_0)}{p_0(1-p_1)}\right)^m\right) + \log\left(\left(\frac{1-p_1}{1-p_0}\right)^{n}\right) \\
&= m\log\left(\frac{p_1(1-p_0)}{p_0(1-p_1)}\right) + n\log\left(\frac{1-p_1}{1-p_0}\right) \\
&= \sum_{i=1}^{m}\left[\log\left(\frac{p_1(1-p_0)}{p_0(1-p_1)}\right)\right] + \sum_{i=1}^{n}\left[\log\left(\frac{1-p_1}{1-p_0}\right)\right] \\
&= \sum_{i=1}^{n}\left[I[x_i = 1]\log\left(\frac{p_1(1-p_0)}{p_0(1-p_1)}\right)\right] + \sum_{i=1}^{n}\left[\log\left(\frac{1-p_1}{1-p_0}\right)\right] \\
&= \sum_{i=1}^{n}\left[x_i\log\left(\frac{p_1(1-p_0)}{p_0(1-p_1)}\right)\right] + \sum_{i=1}^{n}\left[\log\left(\frac{1-p_1}{1-p_0}\right)\right] \\
&= \sum_{i=1}^{n}\left[x_i\log\left(\frac{p_1(1-p_0)}{p_0(1-p_1)}\right) + \log\left(\frac{1-p_1}{1-p_0}\right)\right] \\
&= \sum_{i=1}^{n}\overline{z_i} \\
\therefore \log(f_n) &= Z_n \qquad (3.4)
\end{aligned}$$

$I[x_i = 1]$ is the indicator random variable that indicates whether the random variable x_i has the value 1.

Next, we state the theorem relating the average number of samples with the probability of a model satisfying a specification.

Theorem 1. Let p be the probability with which the model $\mathcal{M}$ satisfies the specification ϕ. Given the null hypothesis $H_0 : p \geq p_0$ and the alternate hypothesis $H_1 : p \leq p_1$, and error thresholds α and β, if $SPRT(A, B)$ rejects the null hypothesis, the average number of samples observed by our SPRT-based algorithm increases as p increases. Note that A, B are calculated from the α, β as suggested by Wald [106].

Proof.

$$\begin{aligned}
&E[Z_n \mid p] \\
&= E[\sum_{i=1}^{n} \overline{z_i} \mid p] \\
&= \sum_{i=1}^{n} E[\overline{z_i} \mid p] \\
&= \sum_{i=1}^{n} E[\log\left(\frac{1-p_1}{1-p_0}\right) + x_i \log\left(\frac{p_1(1-p_0)}{p_0(1-p_1)}\right) \mid p] \\
&= \sum_{i=1}^{n} E[\log\left(\frac{1-p_1}{1-p_0}\right) | p] + \sum_{i=1}^{n} E[x_i \log\left(\frac{p_1(1-p_0)}{p_0(1-p_1)}\right) \mid p] \\
&= \sum_{i=1}^{n} \log\left(\frac{1-p_1}{1-p_0}\right) + \sum_{i=1}^{n} \log\left(\frac{p_1(1-p_0)}{p_0(1-p_1)}\right) E[x_i \mid p] \\
&= n \log\left(\frac{1-p_1}{1-p_0}\right) + np\left[\log\left(\frac{p_1(1-p_0)}{p_0(1-p_1)}\right)\right]
\end{aligned}$$

$$\therefore E[Z_n \mid p] = nX + nYp \tag{3.5}$$

Note that $X = \log\left(\frac{1-p_1}{1-p_0}\right)$ and $Y = \log\left(\frac{p_1(1-p_0)}{p_0(1-p_1)}\right)$ are constants. Further,

$$\begin{aligned}
&\frac{\mathrm{d}E[Z_n \mid p]}{\mathrm{d}p} \\
&= \frac{\mathrm{d}\left(n\log\left(\frac{1-p_1}{1-p_0}\right) + \log\left(\frac{p_1(1-p_0)}{p_0(1-p_1)}\right)np\right)}{\mathrm{d}p} \\
&= n\log\left(\frac{p_1(1-p_0)}{p_0(1-p_1)}\right) \\
\therefore \frac{\mathrm{d}E[Z_n \mid p]}{\mathrm{d}p} &< 0 \qquad (3.6)
\end{aligned}$$

This shows that $E[Z_n \mid p]$ is monotonically decreasing in p. Further, we reject H_0 only if $Z_n \geq log(A)$. Thus, the SPRT procedure with higher values of p takes a larger number of samples (n) for Z_n to cross the threshold A, and, hence, to reject the null hypothesis H_0. □

3.4.2 Proof Sketch of Algorithm Termination

The heart of our proposed algorithm (Algorithm 3.2) is the idea of avoiding the computationally expensive, repeated and unnecessary calculation of the precise fitness function value, i.e. the exact probability with which the stochastic model $\mathcal{M}$ satisfies the given behavioral specification ϕ. In Figure 3.7, we show two parameter points V_s and V_t in the parameter space and the corresponding probabilities p_s and p_t. Recall that the null hypothesis is $H_0 : p \geq p_0$. From Figure 3.7, it is clear that during the exploration process, we should move from V_t to V_s. Our algorithm thus ensures that we can replace the actual calculation of the exact probabilities with the expected number of samples required by the SPRT to reject the null hypothesis.

Figure 3.8 is a pictorial depiction showing that *if the null hypothesis is rejected, the number of samples required by the SPRT increases with the probability with which the model satisfies the specification.*

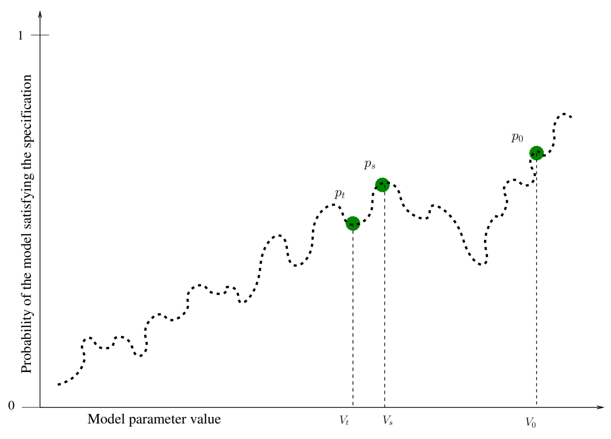


Figure 3.7: Exploring the parameter space using statistical hypothesis testing: Our aim is to find a parameterized model that satisfies the null hypothesis $H : p \geq p_0$. While exploring the parameter space, we move to the parameter that is closer to satisfying the null hypothesis.

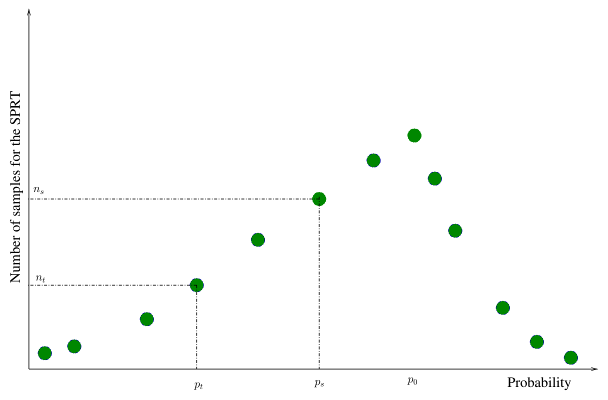


Figure 3.8: Use of the 'hardness of verification' as a metric to guide the search for parameters: While exploring the parameter space, our algorithm uses the number of samples required by the SPRT (instead of calculating the actual probability at each point of the parameterized model satisfying the specification) in order to determine which parameter is better.

The algorithm will terminate when any of the following conditions holds: *a*) the current temperature t falls below a threshold temperature t_0, *b*) the step involving the selection of a neighbor terminates, and *c*) the SPRT based statistical model checking algorithm itself terminates. Condition *a*) is true because of our monotonically decreasing temperature schedule.

Condition *b*) is true if the number of parameters is finite. Condition *c*) is known to be true almost surely for well-framed hypothesis testing queries.

3.5 Discussion

We described a new SPRT-based parameter discovery technique (Algorithm 3.2) for complex stochastic models of biochemical and biomedical systems. While the traditional approach to simulated annealing based parameter synthesis for stochastic models requires the the precise fitness value calculation by an estimation of the exact probability of a given stochastic model satisfying a given behavioral specification,we showed that such an estimate is computationally expensive to obtain. We argued that the computation of such an estimate is not needed and presented theoretical results (§3.4.1) to show that the expected number of samples required during the simulated annealing based parameter exploration procedure that uses the SPRT for verifying whether a model satisfies a given probabilistic behavioral specification is a good surrogate for the actual estimate of the probability itself.

Note, however, that a weakness of the result we established [58] is that it only gives us information about the relationship between the *expected* number of samples required by the SPRT (to reject the null hypothesis) and the probability with which model $\mathcal{M}$ satisfies specification ϕ.

We plan to investigate the theoretical developments in sequential analysis [74] to check whether we can establish a stronger result that shows the relationship between the actual number of samples needed by the SPRT and the probability that $\mathcal{M} \models \phi$.

The next chapter describes our Bayesian model checking-based technique for automated parameter estimation.

Algorithm 3.2 Parameter Estimation Using the SPRT

```
Require:
  Parameterized probabilistic model $\mathcal{M}(.)$ on parameter space $\Omega$,
  PBLTL specification $Pr_{\geq\theta}(\phi)$,
  Starting temperature $t_s$, Stopping temperature $t_f$.
  Cooling Schedule T : $\mathbb{N} \rightarrow (0, \infty)$, (T is strictly decreasing);
  Bounds on Type I/II errors: $\alpha$, $\beta$.
  {Note: Indifference region: $[\theta - \epsilon_1, \theta + \epsilon_2]$}
  {Note: $H_0 : p \geq p_0$, where $p_0 = \theta + \epsilon$ (ideal null hypotheis: $\mathcal{M}(\omega) \vDash Pr_{\geq\theta}(\phi)$)}
  {Note: $H_1 : p \leq p_1$, where $p_1 = \theta - \epsilon$ (ideal alternate hypothesis: $\mathcal{M}(\omega) \nvDash Pr_{\geq\theta}(\phi)$)}
Ensure:
  $ans = \omega$ such that $\omega \in \Omega$ and $\mathcal{M}(\omega) \vDash Pr_{\geq\theta}(\phi)$ or $ans =$ "No parameter found"

  $A = (1 - \beta)/(\alpha)$
  $B = (\beta)/(1 - \alpha)$
  $\omega \leftarrow$ an element in $\Omega$ selected randomly
  if SPRT(A,B) at $\mathcal{M}(\omega)$ leads to $H_1$ being rejected then
    $ans \leftarrow \omega$
    return
  else
    $N(\omega) \leftarrow$ number of samples needed to reject $H_0$
  end if
  $t = t_s$
  $lcount = 0$
  while $t \geq t_f$ do
    Select a neighbor $\psi$ of $\omega$ randomly
    if $SPRT(A, B)$ at $\mathcal{M}(\psi)$ leads to a rejection of $H_1$ then
      $ans \leftarrow \omega$
      return
    end if
    $N(\psi) \leftarrow$ number of samples required to reject $H_0$
    if $N(\psi) \geq N(\omega)$ then
      $\omega \leftarrow \psi; N(\omega) \leftarrow N(\psi)$
    else
      if $rand(0, 1) > exp(-(N(\psi) - N(\omega))/t)$ then
        $\omega \leftarrow \psi; N(\omega) \leftarrow N(\psi)$
      end if
    end if
    $t = T(lcount)$
  end while
  print $ans \leftarrow$ "No parameter found"
  return $ans$
```

CHAPTER 4: PARAMETER ESTIMATION USING BAYESIAN MODEL CHECKING[1]

Probabilistic models have gained widespread acceptance in the systems biology community as a useful way to represent complex biological systems. Such models are developed using existing knowledge of the structure and dynamics of the system, experimental observations, and inferences drawn from statistical analysis of empirical data. A key bottleneck in building such models is that some system variables cannot be measured experimentally. These variables are incorporated into the model as numerical *parameters*. Determining values of these parameters that justify existing experiments and provide reliable predictions when model simulations are performed is a key research problem.

Domain experts usually estimate the values of these parameters by *fitting* the model to experimental data. Model fitting is usually expressed as an optimization problem that requires minimizing a cost-function which measures some notion of distance between the model and the data. This optimization problem is often solved by combining local and global search methods that tend to perform well for the specific application domain. When some prior information about parameters is available, methods such as Bayesian inference are commonly used for parameter learning. Choosing the appropriate parameter search technique requires detailed domain knowledge and insight into the underlying system.

Using an agent-based model of the dynamics of acute inflammation, we demonstrate a novel parameter estimation algorithm by discovering the amount and schedule of doses of bacterial lipopolysaccharide that guarantee a set of observed clinical outcomes with high probability. We synthesized values of *twenty-eight unknown parameters* such that the parameterized model

[1]A preliminary version of the research reported in this chapter was presented at the *4th International Conference on Computational Advances in Bio and Medical Sciences* [60]; an extended version was later published in *BMC Bioinformatics* [59].

instantiated with these parameter values satisfies four specifications describing the dynamic behavior of the model.

We have developed a new algorithmic technique for discovering parameters in complex stochastic models of biological systems given behavioral specifications written in a formal mathematical logic. Our algorithm uses Bayesian model checking, sequential hypothesis testing, and stochastic optimization to automatically synthesize parameters of probabilistic biological models.

4.1 Introduction

Over the last few years, computational modeling has emerged as a popular tool for studying and analyzing biological systems. With rapid growth in the availability of high-performance computing (HPC) infrastructure, there is increasing interest in the construction and analysis of *in silico* models of complex biological systems [80, Chapter 5]. An essential requirement for analyzing a complex high-dimensional system is to build a sufficiently rich computational model that exhibits key properties of the real system being represented [7]. For users to have confidence in the predictions made by analyzing model simulations, it is desirable that the model be amenable to automated verification against large data-sets and expert specifications [10, 98].

This process of analysis and verification becomes complicated if the system is *not deterministic*, i.e. if repeated executions of the model, under the same inputs, may produce different results. Deterministic models, while often having a clean analytic representation, cannot capture the unpredictability of natural phenomena or multi-outcome man-made artifacts. This limitation is addressed by *stochastic models* that allow a succinct representation of variability in system behavior [108]. Such models incorporate the uncertainty inherent in the system being modeled, thus facilitating more accurate analyses and predictions [99].

Models in systems biology are usually nondeterministic, nonlinear, parameterized, and describe both functional behavior and quantitative properties [48]. We will focus on a class of stochastic models that are known in the literature as *probabilistic models* [9, Chapter 10]. The essential property of these models that is of interest to us is that it is possible to accurately assign a probability to every possible behavior that the model can exhibit [72, §3.1].

As a case study for demonstrating our parameter estimation technique, we have considered a class of probabilistic models known as *agent-based models* (ABMs). An ABM consists of a number of autonomous, independently-acting entities known as *agents*. An agent interacts with other agents in its immediate vicinity, according to fixed rules that are possibly probabilistic, enabling the system to demonstrate behavioral variability in the face of environmental uncertainty [6, 16]. Verification and validation of such agent-based models is vital for users to have confidence in the predictions generated by them [109].

An important challenge faced by designers of a biological model is to find values of unknown parameters in the model that enable it to reproduce the behavior of the relevant biological system [37, §2.2, §3.1]. Wooley and Lin describe [80, §5.3.3] the importance of parameter estimation in the computational modeling of biological systems:

> Identifying the appropriate ranges of parameters (e.g., rate constants that govern the pace of chemical reactions) remains one of the difficulties that every modeler faces sooner or later. As modelers know well, even qualitative analysis of simple models depends on knowing which "leading-order terms" are to be kept on which time scales. When the relative rates are entirely unknown—true of many biochemical steps in living cells – it is hard to know where to start and how to assemble a relevant model, a point that underscores the importance of close dialogue between the laboratory biologist and the mathematical or computational modeler.

When the state-space of the parameters is small, an exhaustive search for the correct parameter values is feasible. For high-dimensional models, brute-force methods are unlikely to terminate in sub-exponential time and hence are prohibitively expensive [11]. Again, Wooley and Lin [80, §5.2]:

> In models with many parameters, the state space to be explored may grow combinatorially fast so that no amount of data and brute force computation can yield much of value (although it may be the case that some algorithm or problem-related insight can reduce the volume of state space that must be explored to a reasonable size).

We address this problem by designing an algorithmic technique for parameter estimation in stochastic biological models that ensures that the synthesized model conforms to desired behavior as expressed in a formal temporal logic [35]. This chapter describes the following contributions:

- A *new algorithm for automatically discovering parameters of probabilistic computational models* of biological systems. Our algorithm uses simulated annealing [1] and Bayesian statistical model checking [61] to efficiently explore the system's parameter space while continually verifying that the model instantiated with the current parameters satisfies the given expert specifications.

- An application that demonstrates the effectiveness of our approach by applying this algorithm to automatically synthesize *twenty eight* parameters in an agent-based, physiological model of the acute inflammatory response to endotoxin administration [28].

4.2 Related Work

This section surveys major recent research results on parameter estimation in systems biology. We first summarize techniques that rely primarily on reformulating estimation as a non-linear optimization problem. Later, we discuss approaches based on formal verification.

4.2.1 Parameter Estimation Using Global and Local Search

Sun et al [103] survey metaheuristic techniques used in parameter estimation in systems biology, focusing on simulated annealing, evolutionary algorithms and hybrid strategies that combine multiple heuristics.

Gonzalez et al [46] use simulated annealing [21] to find parameters in S-system models of biochemical networks. At a given parameter point during the annealing process, they find a neighboring point by adding a noise term to *each component* of the parameter vector that is dependent on the current optimization error.

Lillacci and Khammash have designed a method that uses Kalman filtering, statistical testing and numerical optimization for parameter estimation and model selection [79]. Inspired by these two approaches [46, 79], we have used simulated annealing for optimization and statistical hypothesis testing-based verification in our parameter estimation algorithm for probabilistic models.

Algorithms based on maximum-likelihood estimation (MLE) and the singular value decomposition (SVD) have been proposed by Reinker et al [93], to estimate the reaction rate constants (the parameters) from discrete time series data for *molecule counts* in stochastic biochemical reactions.

Rodriquez-Fernandez et al use a hybrid approach for parameter estimation in deterministic, non-linear models of biochemical pathways [97]. They consider deterministic, non-linear models of biochemical pathways and state the parameter estimation problem as one of optimizing a scalar cost function. Their approach combining local and global optimization methods helps overcome the problem of convergence to local minima common in traditional local optimization methods (like gradient-descent) and the problem of slow convergence seen in global optimization techniques. It is interesting to note that they use user-specified switching criteria for their hybrid approach, i.e. when to switch from global to local search and when to stop the local search.

Genetic algorithm based-techniques have also been used for parameter estimation in a plucked string synthesis model [95]. Moles et al have studied the global optimization based approaches to the parameter estimation problem in nonlinear dynamic systems using a 36 parameter dynamic pathway model as a benchmark [90]. They report that only approaches based on evolutionary strategies, viz. the stochastic ranking evolutionary strategy (SRES) and unconstrained evolutionary strategy (uES), were able to successfully find parameters in the pathway.

The parameter estimation problem for biological pathway models has also been addressed by Koh et al [69]. An interesting element of their approach is to decompose the model into components whose parameters can be estimated independently. They model pathways using hybrid Petri nets (see [87]) and use evolutionary strategies for parameter estimation, but their approach can be used for any modeling framework and choice of the optimization technique.

Matthew et al have used global sensitivity analysis (GSA) to study the effect of parameter perturbations in a large, non-linear, ODE-based model of acute inflammation in mice [105], and found that Interleukin 6 (IL-6) and nitric oxide (NO) had a significant, non-linear impact on inflammatory damage [86].

In another approach that uses sensitivity analysis, Donze et al have developed an algorithm for parameter synthesis in nonlinear systems of ODEs (ordinary differential equations) [32], and applied their technique to find parameters in two models of acute inflammation [94, 70].

Next, we discuss parameter estimation approaches based on formal verification techniques.

4.2.2 Parameter Estimation Using Formal Verification

Many recent procedures for parameter estimation make use of a formal verification technique known as *model checking*.

A model checking approach to finding parameters in biological models has been used by Calzone et al [18]. They use the Biochemical Abstract Machine (BIOCHAM) modeling framework to describe the system and temporal logic model checking [24] to find parameters values in a user-specified range.

Dreossi and Dang have recently designed an algorithm that reduces the problem of parameter synthesis in polynomial (discrete-time) dynamical systems to solving a set of linear programs [33]. They use their technique to find parameters of two well-studied epidemic models.

Batt et al have used symbolic model checking to find parameters in a piecewise affine differential equation (PADE) model of the gene regulation IRMA network [12]. IRMA stands for *in vivo "benchmarking" of reverse-engineering and modeling approaches* [20]. Donaldson and Gilbert have designed a technique for parameter estimation that combines model checking with a genetic algorithm [31], and used it for estimating kinetic rate constants in a model of the mitogen-activated protein kinases (MAPK) signaling pathway.

Usually, model checking based methods of parameter estimation require that the relevant specification be expressed in a formal temporal logic, whose satisfaction against a given execution path (known as a *trace*) of the model can be determined. Rizk et al define a *continuous degree of satisfaction* of a temporal logic formula for any given trace of the model

[96] and use it as a fitness function to drive an optimization-style search for kinetic parameters in models of the cell cycle and the MAPK signal transduction.

In recent work, Mancini et al have used statistical model checking to synthesize parameters in an ODE-based biological model [83]. They have implemented a distributed, multi-core version of their algorithmic technique, and used it to estimate *patient-specific* parameters of a a human menstrual cycle model.

4.3 Background

Before describing our algorithm for synthesizing parameters in probabilistic models, we give some background on stochastic modeling in systems biology, specification of time-varying properties using temporal logic and statistical verification of probabilistic systems against behavioral specifications. We also provide definitions for formal concepts that will be used to describe our algorithm.

Remark 1. One of our main objectives in this section is to develop a formal definition of computational probabilistic models: i.e. those stochastic models for which *a probability can be assigned to any observed model behavior* when the model is executed. The reader familiar with this concept can quickly skim through the next two subsections.

4.3.1 Stochastic Biological Models

Many biological systems have traditionally been described using *deterministic, mathematical* models, often in terms of ordinary differential equations [37, §2.1]. However, in recent years, the need for incorporating the uncertainty inherent in biological systems has led to the development of *stochastic models*: these are harder to analyze but more accurately reflect the behavior of the underlying system [108]. Common stochastic models in biology include

Markov jump processes [108], stochastic differential equations [30], discrete-time Markov chains [3, Chapter 3] and continuous-time Markov chains [73].

Also, researchers are increasingly developing *computational* models that naturally capture the bottom-up nature of biological phenomena and hence are more amenable to *in-silico* implementation [40]. Such models are constructed by observing large sets of time-series data, combined with their expert insight into the system being modeled [67]. Some of these models are based on experimental data of varying veracity and error propagation into the designed model is an ever-present challenge.

We will focus on *discrete-time Markov chains*, a class of stochastic models that are widely used in the sciences, engineering, economics and other areas. We closely follow Baier and Katoen [9] in formally defining them.

Definition 2 (Discrete-time Markov chain). A discrete time Markov chain (DTMC) is given by $\mathcal{M} = (S, P, init, AP, L)$ where:

- S is a countable, nonempty, set of states,
- $init : S \to [0, 1]$ is the initial distribution such that $\Sigma_{s \in S}\ init(s) = 1$,
- $P : S \times S \to [0, 1]$ is the transition probability function,
- AP is a set of boolean valued atomic propositions,
- $L : S \to 2^{AP}$ is a labeling function for states. ■

We next define (a) a parameterized family of DTMCs, given a set of parameters and (b) execution paths over them.

Definition 3 (Parameterized DTMC). A parameterized discrete time Markov chain (ParDTMC) is given by $\mathcal{M} = (S, \Theta_m, P, init, AP, L)$ where

- S is a countable, nonempty, set of states,
- $\Theta_m = \mathbb{R}^m, m \geq 1$ is the parameter space,
- $init : S \to [0, 1]$ is the initial distribution such that $\Sigma_{s \in S}\ init(s) = 1$,
- $P : S \times S \times \Theta_m \to [0, 1]$ is the transition probability function over an m-dimensional parameter space,
- AP is a set of boolean valued atomic propositions, and
- $L : S \to 2^{AP}$ is a labeling function for states. ■

Definition 4 (ParDTMC execution paths). An execution path of a ParDTMC $\mathcal{M} = (S, \Theta_m, P, init, AP, L)$ is given by $\sigma = s_0, s_1, \ldots$. The suffix of a path σ that starts at state s_i (i.e. $s_i, s_{i+1}, \ldots$) is denoted σ^i.

We have applied our automated parameter estimation technique to an agent-based model of the acute inflammatory response that we discuss later (§4.5). We now briefly discuss major characteristics of ABMs.

4.3.2 Agent-Based Modeling in Systems Biology

Agent-based models (ABMs) form an interesting, well-studied subset of complex probabilistic models. In recent years, agent-based modeling has emerged as a popular method for the representation, analysis and simulation of biological models [6].

ABMs allow the specification of high-level model properties as well as fine-grained component behavior. An ABM is composed of autonomous elements whose individual properties can be specified. At a macro-level, model-wide agent-interaction rules can be defined and enforced. Each agent has a physical location and its state evolves with time based on messages exchanged with other agents, allowing rich spatio-temporal properties to emerge [6].

ABMs can incorporate parallelism, object-oriented behavior and stochasticity, making them conducive to the development of computational models in systems biology. Another advantage of using ABMs is that they are *bottom-up* models that mirror the natural behavior of biological systems [77]. Agents interact with one another based on fixed rules and also have a spatial location [6].

Most agent-based models do not have compact analytic descriptions, and simulations across the input space must be performed to in order to infer general model properties [81]. Also, agent-based models tend to have a large number of parameters, some of them highly-sensitive in the sense that a small change can radically alter model behavior [17]. To the best of our knowledge, the parameter estimation problem in ABMs has only been addressed for very simple models [2] and most approaches use standard optimization techniques [18, 44].

As a case study for our parameter estimation technique, we have used an agent-based model of the acute inflammatory response due to endotoxin administration [5] written using the SPARK tool [102]. SPARK is an ABM framework designed for multi-scale modeling of biomedical models that is implemented in Java and allows users to develop models using its own programming language (SPARK-PL) [101, 34].

One we have a basic model design available, we need to find the *exact model instantiation* in terms of concrete parameter values that meets desired behavior (usually experimental data). Therefore, we need a way to check if the models all requirements. We now discuss how to *formally specify* and *automatically verify* probabilistic computational models.

An ABM $\mathcal{A}$ consists of a fixed number of agents $A_1, \ldots A_n$, where the state of agent A_i is determined by the values of variables $V_1^i \ldots V_m^i$. Assuming that for any agent $A_i, i \in \{1 \ldots n\}$, variable $V_j^i, j \in \{1 \ldots m\}$ can take values in the set Var_j, $\mathcal{A}$ can be represented as a DTMC.

Definition 5 (DTMC corresponding to an ABM). Consider ABM $\mathcal{A}(n, m)$ with n agents $A_1, \ldots A_n$ and m variables $V_1 \ldots V_m$ that take values over countably finite sets $Var_1, \ldots, Var_m$

respectively. The DTMC $\mathcal{M}_{\mathcal{A}} = (S, P, init, AP, L)$ corresponding to $\mathcal{A}(n, m)$ is:

- $S = (Var_1 \times \ldots \times Var_m)^n$,
- $init : S \to [0, 1]$, the initial distribution is imposed by $\mathcal{A}$,
- $P : S \times S \to [0, 1]$ is determined by the transition rules over variables of $\mathcal{A}$,
- AP consists of (boolean-valued) atomic propositions over agent variables $V_1^1 \ldots V_m^1, \ldots, V_1^n \ldots V_m^n$ and
- $L : S \to 2^{AP}$ is the usual function marking each state with propositions that hold there. ■

We generalize agent-variable ABMs, by adding the notion of parameters to obtain a *family* of ABMs. More formally, we talk of a parametric ABM $\mathcal{A}(n, m, k)$ over n agents, m variables and k parameters whose dynamic behavior (i.e. its transition function P) now also depends on the parameter values. For a given parametric ABM, we can defined an equivalent ParDTMC.

Definition 6 (ParDTMC corresponding to a parametric ABM). Consider ABM $\mathcal{A}(n, m, k)$ with n agents $A_1, \ldots A_n$ and m variables $V_1 \ldots V_m$ that take values over countably finite sets $Var_1 \ldots Var_m$ and k parameters $\theta_1 \ldots \theta_k$. The ParDTMC $\mathcal{M}_{\mathcal{A}} = (S, \Theta_k, P, init, AP, L)$ corresponding to $\mathcal{A}(n, m, k)$ is:

- $S = (Var_1 \times \ldots \times Var_m)^n$,
- $init : S \to [0, 1]$ the initial distribution is imposed by $\mathcal{A}$,
- $P : S \times S \times \Theta_k \to [0, 1]$, where $\Theta_k = \theta_1 \times \ldots \times \theta_k$ is determined by the transition rules over variables and parameters of $\mathcal{A}$,
- AP consists of boolean propositions over variable values of all agents $V_1^1 \ldots V_m^1, \ldots, V_1^n \ldots V_m^n$,

- $L : S \to 2^{AP}$ is the usual function marking each state with propositions that hold there. ■

Remark 2. In order to use statistical model checking to solve the probabilistic model checking problem, we need to associate probabilities with executions paths of the model. For DTMCs, this has been shown by Kwiatkowska et al [72] and Baier and Katoen [9, Chapter 10]. Although we do not prove this, this is also true for ParDTMCs. From now on, we assume that our models can be represented as ParDTMCs and thus have a unique underlying probability measure.

4.3.3 Specifying and Checking Biological Properties

Our algorithm discovers parameters of stochastic biological from experimental data and expert behavioral specifications. While exploring the parameter space, we continually verify whether we have a parameter assignment at which the model meets the given specifications.

Typically, properties of biological systems that need to be formally specified are *qualitative*, *quantitative* and *time-varying*. Since the models are usually stochastic, specifications too should allow reasonable deviations from desired behavior. To capture such rich behavioral properties, we use a *probabilistic temporal logic* (see Definition 9) to specify expected model behavior.

In order to develop automated techniques for analyzing biological models, we also need the properties to be specified in a way that can be *monitored* as we perform ABM simulations. Monitoring is the process of determining if an execution trace of the model satisfies given specifications [39] Therefore, we write specifications using a logic belonging to a class of languages for which monitors can be derived algorithmically [63].

We translate natural language expert insights representing desired model behavior into a logic whose sentences are *adapted finitely monitorable* (AFM) formulas. The truth value of an AFM formula with respect to a trace of a model execution can be determined by observing a finite prefix of the trace [78, §2],[63, §2.2]. Next, we formally define our specification language:

Definition 7 (BLTL grammar). Given a ParDTMC $\mathcal{M} = (S, \Theta_k, P, init, AP, L)$, the grammar of bounded linear temporal logic (BLTL) formulas ϕ in Backus-Naur Form is as follows:

$$\langle\phi\rangle ::= a \mid \phi \wedge \phi \mid \phi \vee \phi \mid \neg\phi \mid \phi \mathbf{U}^d \phi$$

where $a \in AP$, $d \in \mathbb{N}$, and $\wedge, \vee$ and $\neg$ are the usual proposition logic operators. We call $\mathbf{U}^d$ the bounded until operator. ■

The semantics of a BLTL formula ϕ is defined over an execution path of a ParDTMC $\mathcal{M}$ (see Definition 4).

Definition 8 (BLTL semantics). Given a path σ of a ParDTMC $\mathcal{M} = (S, P, init, AP, L)$, the satisfaction of a BLTL formula ϕ on a path suffix $\sigma^i = (s_1, s_2, \ldots)$ is denoted $\sigma^i \models \phi$, and is determined by the following rules:

- $\sigma^i \models a$, where $a \in AP$ iff $a \in L(s_i)$,
- $\sigma^i \models \phi_1 \wedge \phi_2$ iff $\sigma^i \models \phi_1$ and $\sigma^i \models \phi_2$,
- $\sigma^i \models \phi_1 \vee \phi_2$ iff $\sigma^i \models \phi_1$ or $\sigma^i \models \phi_2$,
- $\sigma^i \models \neg\phi$ iff $\sigma^i \not\models \phi$,
- $\sigma^i \models \phi_1 \mathbf{U}^d \phi_2$ iff $\exists l \in \mathbb{N}$ such that $l <= d$, $\sigma^{i+l} \models \phi_2$ and $\forall 0 \leq j < l, \sigma^{i+j} \models \phi_1$. ■

We now state, without proof, the fact that it is possible to check if any path of a ParDTMC satisfies a given BLTL formula, by observing a finite prefix of the path. For a proof of a similar lemma for continuous-time Markov chains, see Legay et al [113].

Lemma 1 (Monitorability of BLTL formulas over DTMCs)**.** *It is always possible to algorithmically decide if any simulation path $\sigma = (s_1, s_2, \ldots)$ of a ParDTMC $\mathcal{M}$ satisfies given BLTL formula ϕ by observing a finite prefix of σ.*

Note 1 (Additional bounded operators)*.* We can define bounded versions of the usual linear temporal logic operators **G** (*always*) and **F** (*eventually*) as follows: $\mathbf{F}^d\psi = \mathbf{true}\ \mathbf{U}^d\psi$, $\mathbf{G}^d\psi = \neg\mathbf{F}^d\neg\psi$. $\mathbf{F}^d\psi$ means that ψ holds at some state within the next d state-transitions, and $\mathbf{G}^d\psi$ means that ψ continually holds for the next d state-transitions.

The expressive power of our agent-based models emanates from their ability to capture uncertainty. For such stochastic models, a single execution trace that violates a given property cannot serve as a counterexample. We need a more flexible specification language whose formulas allow reasonable deviations from expected model behavior. For this, we define a probabilistic version of BLTL.

Definition 9 (Probabilistic Bounded Linear Temporal Logic)*.* A specification of the form $P_{\geq\theta}(\phi)$ is a probabilistic bounded linear temporal logic (PBLTL) formula if ϕ is a bounded linear temporal logic formula and θ is a probability threshold such that $\theta \in \mathbb{R}, 0 \leq \theta \leq 1$. If a θ fraction of the traces of a model satisfy ϕ, we deem the model to satisfy the (probabilistic) AFM specification $P_{\geq\theta}(\phi)$. ■

4.3.4 Automated Verification Using Model Checking

Model checking is an automated technique for verifying finite and infinite state transition systems that is widely used for formal assurance of safety-critical systems [9]. Techniques based on model checking have been used for verification in a number of areas such as hybrid dynamical systems, computer software, and systems biology.

In order to use model checking to verify a time-varying system, the model is described using a Kripke structure and the property to be checked is written in a formal temporal logic. We will use PBLTL (Definition 9) to define the probabilistic model checking problem.

Definition 10 (Probabilistic model checking (PMC)). Given a probabilistic model $\mathcal{M}$ and a PBLTL specification $P_{\geq\theta}(\phi)$, determine if $\mathcal{M}$ satisfies ϕ with probability at least θ. ■

There are two approaches for solving the PMC problem:

- Symbolic and numerical techniques that estimate the exact value of the probability with which $\mathcal{M}$ satisfies ϕ (by exhaustively exploring all possible model behaviors) and then compare it to the specification threshold probability θ.
- Statistical techniques [54, 112] that use a set of sample simulations to determine if $\mathcal{M} \models P_{\geq\theta}(\phi)$.

Statistical approaches for probabilistic model checking are more scalable since they avoid the expensive calculation associated with accurately estimating exact probabilities. However, a limitation of statistical model checking algorithms is that the reported result is not guaranteed to be correct, i.e., there may be false positives or false negatives [111].

Since computational models in systems biology often have large parameter spaces, we use statistical model checking as part of our parameter estimation algorithm. Next, we describe a reformulation of the probabilistic model checking problem in terms of statistical hypothesis testing – an approach first used by Younes [111].

4.3.5 Hypothesis Testing Based Statistical Model Checking

We are interested in determining an answer to the probabilistic model checking problem, i.e. "Does $\mathcal{M} \models P_{\geq\theta}(\phi)$?" (see Definition 10).

Assuming that the model's actual probability of satisfying the specification is $u \in [0, 1]$, i.e. $\mathcal{M} \models P_{=u}(\phi)$, we test the (null) hypothesis $H : u \geq \theta$ against the (alternative) hypothesis

$K : u < \theta$. If K is rejected we conclude that $\mathcal{M} \models P_{\geq\theta}(\phi)$. If H is rejected we conclude that $\mathcal{M} \not\models P_{\geq\theta}(\phi)$.

Note 2 (Errors in hypothesis testing). For a hypothesis test procedure, the critical region is the part of the sample space where the null hypothesis is rejected. If the test rejects H when its true, it is considered a Type I error. On the other hand, if the test accepts H when its false, it is a Type II error. Once the critical region for a test procedure has been decided, it uniquely determines the probabilities of Type I and Type II errors. For a given critical region, we denote by α (resp. β) the probability of a Type I (resp. Type II) error.

Naturally, for any statistical hypothesis testing procedure we want a critical region that minimizes the probabilities α and β of Type I and Type II errors (see Note 2). However, this implies either using a large value for either α or β, or drawing a large number of samples to ensure test accuracy.

Younes [112] suggests that the solution is to use the more relaxed test of $H_0 : u \geq u_r$ against $H_1 : u \leq u_l$, where $0 \leq u_l < \theta < u_r \leq 1$). If H_0 (H_1) is accepted, we consider H (resp. K) to be true.

Remark 3 (Indifference regions). $[u_l, u_r]$ is known as the indifference region. If $u \in [u_l, u_r]$ (i.e. when both H_0 and H_1 are false), we do not care about the result of the test; thus the test procedure is allowed to accept either hypothesis. In practice, we often choose the indifference region $[u_l, u_r]$ to be of width $2 * \epsilon$ (where $0 < \epsilon \ll 1$), i.e. $[u - \epsilon, u + \epsilon]$. ■

Our parameter estimation technique uses the *Bayesian statistical model checking* (BSMC) algorithm developed by Jha at al [61], to algorithmically check if a probabilistic model satisfies given behavioral specifications.

Next, we briefly describe the main idea behind BSMC and refer to the reader to the literature for details [63, 78, 61].

4.3.6 Bayesian Statistical Model Checking

Recall that we are interested in testing whether a probabilistic model meets a PBLTL specification with a minimum threshold probability, i.e. "Does $\mathcal{M} \models P_{\geq\theta}(\phi)$"? (Note that the BLTL specification ϕ does not contain a probability operator.)

Assuming that $\mathcal{M} \models P_{=u}(\phi)$, we have posed this problem as hypothesis testing query: test $H : u \geq \theta$ against $K : u < \theta$, and then relaxed it to use indifference regions: $H_0 : u \geq u_r$ against $H_1 : u \leq u_l$ (where $0 \leq u_l < \theta < u_r \leq 1$).

Algorithm 4.3 Bayesian Statistical Model Checking

```
Require:
    Probabilistic model $\mathcal{M}$,
    PBLTL specification $P_{\geq\theta}(\phi)$,
    Threshold $L > 1$,
    Prior density function $g(.)$ of the unknown parameter $u$ where $\mathcal{M} \models P_{=u}(\phi)$,
    Indifference region bounds: $(\epsilon_1, \epsilon_2)$, where $\epsilon_1 > 0, \epsilon_2 > 0$.
    {Note: Indifference region is $[\theta - \epsilon_1, \theta + \epsilon_2]$}
    {Note: $H_0 : u \geq \theta + \epsilon_2$; $H_1 : u \leq \theta - \epsilon_1$}
Ensure:
    $ans$ = false if $H_0$ rejected,
    $ans$ = true if $H_0$ is accepted,
    $n$ = Total number of traces sampled from $\mathcal{M}$.
 1: $n \leftarrow 0$ {Number of traces drawn from $\mathcal{M}$.}
 2: $z \leftarrow 0$ {Number of traces satisfying $\phi$.}
 3: repeat
 4:   Draw sample $\sigma$ from $\mathcal{M}$
 5:   $n \leftarrow n + 1$
 6:   if $\sigma \models \phi$ then
 7:     $z \leftarrow z + 1$
 8:   end if
 9:   $B \leftarrow \frac{\int_{\theta+\epsilon_2}^{1} u^z (1-u)^{n-z} g(u) du}{\int_{0}^{\theta-\epsilon_1} u^z (1-u)^{n-z} g(u) du}$ {From Eqn. 4.2.}
10: until $(B > L) \vee (B < \frac{1}{L})$
11: if $B > L$ then
12:   $ans \leftarrow true$
13: else
14:   $ans \leftarrow false$
15: end if
16: return $(ans; n)$
```

If H_1 is rejected, we consider that $H : u \geq \theta$ holds, and hence conclude that $\mathcal{M} \models P_{\geq\theta}(\phi)$. If H_0 is rejected, we conclude that $H_1 : \mathcal{M} \models P_{<\theta}(\phi)$, i.e. $\mathcal{M} \not\models P_{\geq\theta}(\phi)$.

Recall that we do not know the value of the actual probability u with which the model satisfies the specification i.e. $M \models P_{=u}(\phi)$. For Bayesian testing, we model this unknown probability as a random variable U and assume the availability of a *prior density* function $g(.)$ that incorporates our existing knowledge of U.

The Bayesian statistical model checking procedure (see Algorithm 4.3) sequentially draws traces from $\mathcal{M}$ in an i.i.d. fashion until it rejects either H_0 or H_1. For each sample trace $\sigma_i, (i \in \{0, 1, \ldots\})$ of $\mathcal{M}$, it uses a monitoring algorithm to determine if the path satisfies the BLTL specification ϕ.

The outcome of each such test of path satisfiability can be represented by a Bernoulli random variable X_i i.e. $\forall i \in \{1, \ldots n\}$, if $\sigma_i \models \phi, X_i = 1$, otherwise (i.e. when $\sigma_i \not\models \phi$) $X_i = 0$. At each iteration, the algorithm calculates a quantity known as the Bayes Factor (BF) that, given the observation of a sample $(x_1, x_2, \ldots x_n), x_i \in \{0, 1\}$, reflects confidence in H_0 holding versus H_1:

$$BayesFactor : \frac{P(x_1, \ldots, x_n | H_0)}{P(x_1, \ldots, x_n | H_1)} \tag{4.1}$$

We need to calculate the probability $P(d|H_i)$ of observing sample $d = (x_1, x_2, \ldots x_n)$ given hypothesis $H_i, i \in 0, 1$. Therefore, we will need to consider all cases where H_0 (resp. H_1) holds. Assuming the indifference region $[u_l, u_r]$ to be $[\theta - \epsilon_1, \theta + \epsilon_2]$, whether $H_0 : u \geq u_r$ or $H_1 : u \leq u_l$ holds depends on the actual probability u of $\mathcal{M}$ satisfying ϕ. For both cases, we integrate over all possible values of u according to our prior g:

$$P(d|H_0) = \int_{\theta+\epsilon_2}^{1} f(x_1|u) \ldots f(x_n|u) g(u) du$$

$$P(d|H_1) = \int_0^{\theta-\epsilon_1} f(x_1|u)\dots f(x_n|u)g(u)du$$

Here, $f(x_i|u) = u^{x_i}(1-u)^{1-x_i}$ is the conditional density of Bernoulli random variable X_i given the actual probability u. Now, the probability of observing a sample d given hypothesis $H_i (i = 0, 1)$, depends on its joint density h and the prior $g(.)$:

$$BayesFactor = \frac{\int_{\theta+\epsilon_2}^{1} u^z(1-u)^{n-z}g(u)du}{\int_0^{\theta-\epsilon_1} u^z(1-u)^{n-z}g(u)du} \tag{4.2}$$

In summary, calculating the Bayes factor requires knowing the number of total traces n drawn from $\mathcal{M}$, the number z of traces that satisfied specification ϕ, the indifference region $[u_l, u_r]$ and the prior density $g(.)$ of the unknown probability u. For more details about the Bayesian model checking algorithm, we refer the reader to the work of Jha [63, Chapter 4] .

4.4 Algorithm for Discovering Parameters

We formally state the problem of estimating parameters in computational models of probabilistic systems:

Definition 11 (Parameter estimation problem). Given a parameterized probabilistic model $\mathcal{M}(\omega)$, with parameter set $\Omega \subseteq \mathbb{R}^n$, a desired specification ϕ in a bounded (finitely monitorable) temporal logic, and a probabilistic threshold $\theta \in [0, 1]$, find a parameter value $\omega \in \Omega$ such that $\mathcal{M}(\omega)$ satisfies property ϕ with probability at least θ, i.e. $\mathcal{M}(\omega) \models P_{\geq\theta}(\phi)$. ■

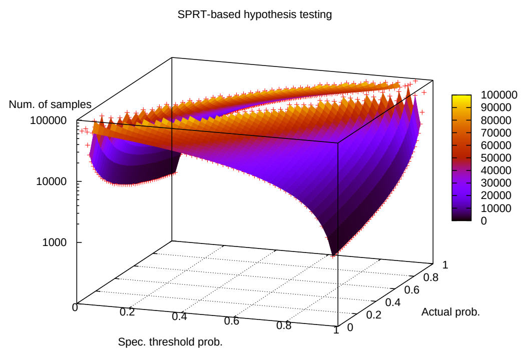
Our algorithmic procedure for synthesizing parameters is shown in Algorithm 4.4. We use simulated annealing [92] for exploring the parameter space of the stochastic model. Simulated annealing is a stochastic optimization technique that avoids local minima in two ways (lines 17–27): (a) by sometimes accepting points with lower fitness and (b) via the temperature schedule that causes fewer bad choices to be accepted as we move closer to one of the global optima.

When considering any candidate parameter $\omega \in \Omega$, we invoke the Bayesian model checking routine (lines 2 and 12) to check if the *parameterized model* matches expected behavior (i.e. if $\mathcal{M}(\omega) \models P_{\geq\theta}(\phi)$).

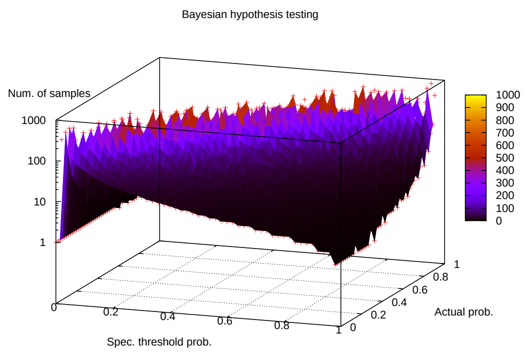
Note that since the BSMC algorithm expects as input a prior density g for the unknown probability u where $\mathcal{M} \models P_{=u}(\phi)$, our synthesis algorithm needs a parameterized prior $h(.)$ that represents the prior for each model instantiation $\mathcal{M}(\omega)$.

To guide the annealing process, we use the number of samples returned by the Bayesian model checking procedure, moving to the parameter point that needed *more samples to reject the null hypothesis during Bayesian statistical model checking* (lines 14 and 15 in Algorithm 4.4). For verifying a model $\mathcal{M}$ against a PBLTL formula $P_{\geq\theta}(\phi)$, given that $\mathcal{M}$ actually satisfies ϕ with probability u (i.e. $\mathcal{M} \models P_{=u}(\phi)$), the Bayesian statistical model checking algorithm takes increasingly larger number of samples to reject the null hypothesis as the specification threshold probability (θ) approaches the actual probability (u) with which $\mathcal{M}$ satisfies ϕ, as shown in Figure 4.1. The figure shows how, for a fixed threshold θ, the Bayesian hypothesis testing algorithm takes more samples for verification when we consider parameter points ω at which the model's probability $p(\omega)$ of satisfying ϕ is close to θ.

In earlier work, we had demonstrated the use of statistical hypothesis testing for parameter search in stochastic models by using a metric based on the Sequential Probability Ratio Test (SPRT) hypothesis testing technique [106] as a fitness function to drive the global optimization procedure used for searching the state space [57, 58]. Figure 4.1 shows results from two sets of experiments that demonstrate how Bayesian hypothesis testing uses significantly fewer samples than a verification procedure based on the SPRT when the model is far away from the desired behavior.



(a) Hypothesis test using the SPRT with Type I error bound $\alpha = 0.001$, Type II error bound $\beta = 0.001$ and indifference region $2 * 0.0001$.



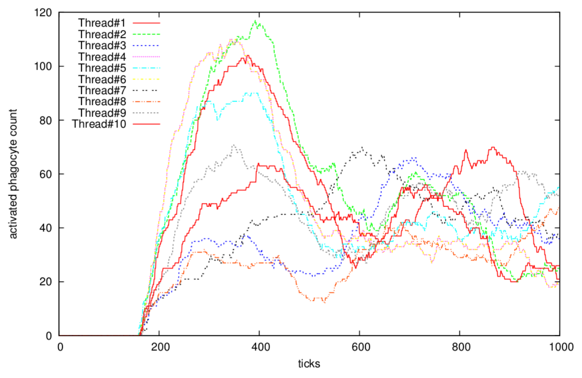
(b) Bayesian hypothesis test with Bayes factor $T = 1000$, and using Beta prior with parameters $\alpha = 1$, $\beta = 1$.

Figure 4.1: Comparison of the efficiency of SPRT-based (a) and Bayesian (b) hypothesis testing. In both cases, the number of samples required for hypothesis testing increases as the specification threshold probability approaches the actual probability with which the model satisfies the specification. Bayesian hypothesis testing required fewer samples than the SPRT when the model is obviously flawed with respect to the desired behavior. The number of samples for the Bayesian hypothesis testing vary from 1 to 1000 while those for the SPRT become as large as 100000.

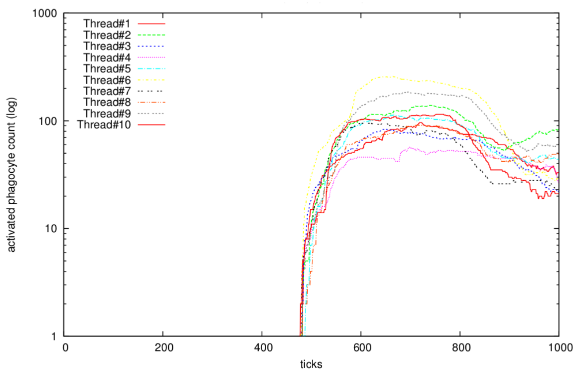
Unlike the SPRT-based technique for parameter discovery (Chapter 3), Algorithm 4.4 does not need *the Type I, Type II error bounds* (α, β resp.) because it uses Bayesian statistical model checking for verification, thereby reducing the number of samples required for discovering a model's parameters.

4.5 Application: Verifying Properties of a Physiological Model of Acute Inflammation

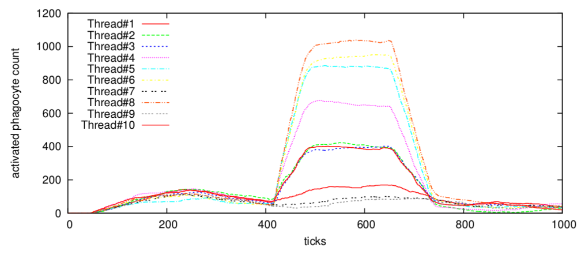
We demonstrate our algorithm for discovering parameters of probabilistic systems on a physiological model that describes the acute inflammatory response (AIR) to the administration of the Gram-negative Bacterial endotoxin lipopolysaccharide (LPS) [28].



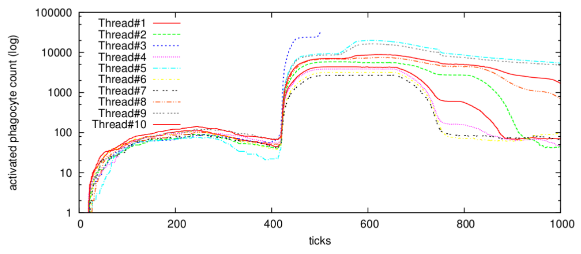
(a) Simulations validating specification ($spec_1$).



(b) Simulations validating specification ($spec_2$).



(c) Simulations validating specification ($spec_3$).



(d) Simulations validating specification ($spec_4$).

Figure 4.2: Simulation results showing SPARK output that demonstrate that the model instantiated with the synthesized parameters meets the desired specifications. Parallel simulations show SPARK output for 10 threads. Figures (a), (b), (c) and (d) show traces for the activated phagocyte count over time on invocation of the ABM simulator. Satisfaction of the four specifications was determined by a monitoring script that checks traces for each of the desired behaviors. One can visually verify that the ABM parameterized with the values in Table 4.1 satisfies all the four expert-provided specifications.

With the aim of discovering the schedule and doses of LPS that make the model exhibit desired properties, we synthesized *twenty eight model parameters* for a set of *four* specifications given to us by experts with extensive experience with the model. Simulations were performed using the SPARK (Simple Platform for Agent-based Representation of Knowledge) agent-based modeling and simulation framework [101, 102]. We describe both the (natural language) expert specifications and their translations into PBLTL (see Definition 9):

($spec_1$) There exists a low dose of the lipopolysaccharide (LPS) that stimulates an episode of inflammation which eventually resolves – the system returns to baseline.
Formal specification: $D_L \to \mathbf{F}^{\delta_1}(I \wedge \mathbf{F}^{\delta_2}(N))$.

($spec_2$) There exists a high dose of LPS that stimulates an episode of inflammation which does not resolve, i.e. the system reaches levels of inflammation from which there is no recovery.
Formal specification: $D_H \to \mathbf{F}^{\delta_3}(\mathbf{G}^{\delta_4} I_H)$.

($spec_3$) *Desensitization*: For a certain time interval, when one administration of LPS is followed by a second administration of the same dose, the inflammatory response resulting from the second administration is lesser than that from the first.
Formal specification: $D \to \mathbf{F}^{\delta_5}(I_L \wedge \mathbf{F}^{\delta_6}(D \to \mathbf{F}^{\delta_7} I_H))$.

($spec_4$) *Priming*: For a certain time interval, when one administration of LPS is followed by a second administration of the same dose, the inflammatory response resulting from the second administration is greater than that from the first.
Formal specification: $D \to \mathbf{F}^{\delta_8}(I_H \wedge \mathbf{F}^{\delta_9}(D \to \mathbf{F}^{\delta_{10}} I_L))$.

D_L (D_H) represents a low (high) dose of LPS, D is a dose of unknown magnitude, but likely to be neither too low nor very high, I indicates that an inflammatory event occurred, N is the event of entering a non-inflammatory state, and I_L (I_H) stands for lower (higher) level of inflammation. Also, $\forall i \in \{1 \dots 10\}$, $\delta_i \in \mathbb{N}$ represents the (discrete) simulation time steps between the relevant events. For initial values of each of the parameters, we used a randomly chosen value within bounds provided as part of the specification.

We *successfully synthesized 28 parameters* (shown in Table 4.1) for the acute inflammatory response model against four behavioral specifications. Our algorithm took *less than 24 hours to synthesize this parameter set* on a 1400 MHz, 64-core machine running the Linux operating system. Figure 4.2 shows the satisfaction of all four specifications by depicting model simulations when parameterized at the synthesized parameter set from Table 4.1.

The first 14 parameters are fundamental to the model and denote various attributes that cannot be measured experimentally. The remaining 14 parameters describe the endotoxin administration schedule. Of these, the first 3 parameters indicate the case where inflammatory event occurs but is later resolved; the next 3 parameters show the scenario where an inflammatory event occurs that is never resolved. The last 4 parameters denote the *priming* scenario ($spec_3$), and the second-last set of 4 parameters denotes the *desensitization* scenario ($spec_4$). Both priming and desensitization are phenomena in which repeated administration of endotoxin leads to either an augmented (priming) or reduced (desensitization) level of inflammation as compared to a single administration of endotoxin [28].

We conclude that a low dose for a high duration causes priming behavior whereas an even lower dose administered for a short period of time results in desensitization. Thus, our algorithm synthesizes a model that demonstrates all four behavioral properties, i.e. ($spec_1$), ($spec_2$), ($spec_3$) and ($spec_4$).

Table 4.1: Parameters of the acute inflammatory response model synthesized by our algorithm.

(param. 1) LPS-evap	0.932575	(param. 15) exp1-dose-time	149.574
(param. 2) mac-act-LPS	0.661416	(param. 16) exp1-dose-duration	4.80575
(param. 3) mac-act-pro	0.326682	(param. 17) exp1-dose-amount	3352.54
(param. 4) mac-regen	12.655	(param. 18) exp2-dose-time	467.262
(param. 5) mac-age	60.5967	(param. 19) exp2-dose-duration	458.451
(param. 6) mac-act-dam	0.3916	(param. 20) exp2-dose-amount	896067
(param. 7) max-pro-dam	18.5986	(param. 21) exp3-1st-dose-time	33.3838
(param. 8) pro-dam-thresh	0.51023	(param. 22) exp3-2nd-dose-time	407.352
(param. 9) damage-evap	0.276594	(param. 23) exp3-doses-duration	41.6759
(param. 10) anti-heal-thresh	7.92487	(param. 24) exp3-doses-amount	2628.97
(param. 11) mac-anti	0.442621	(param. 25) exp4-1st-dose-time	8.24293
(param. 12) anti-evap	0.623503	(param. 26) exp4-2nd-dose-time	411.959
(param. 13) pro-evap	0.142298	(param. 27) exp4-doses-duration	4.40842
(param. 14) mac-prop	8.39519	(param. 28) exp4-doses-amount	4494.65

Algorithm 4.4 Parameter Estimation Using Bayesian Statistical Model Checking

```
Require:
    Parameterized probabilistic model $\mathcal{M}(.)$ on parameter space $\Omega$,
    PBLTL specification $P_{\geq\theta}(\phi)$,
    Starting temperature $t_s$,
    Stopping temperature $t_f$,
    Cooling schedule $T : \mathbb{N} \mapsto [0, \infty)$ (where $T$ is strictly decreasing),
    Parameterized prior density $h(.)$ on parameter space $\Omega$,
    Threshold $L$,
    Indifference region bounds $(\epsilon_1, \epsilon_2)$ where $\epsilon_1 > 0, \epsilon_2 > 0$.
Ensure:
    ans = $\omega$ such that $\omega \in \Omega$ and $\mathcal{M}(\omega) \models P_{\geq\theta}(\phi)$ or
    ans = "No parameter found."
 1: $\omega \leftarrow$ an element in $\Omega$ selected randomly
 2: $(f, n) \leftarrow BSMC(\mathcal{M}(\omega), P_{\geq\theta}(\phi), L, h(\omega), (\epsilon_1, \epsilon_2))$
 3: if $f = true$ then
 4:     $ans \leftarrow \omega$
 5:     return
 6: end if
 7: $t = t_s$
 8: $lcount = 0$
 9: while $t \geq t_f$ do
10:     $lcount \leftarrow lcount + 1$
11:     Select a neighbor $\omega'$ of $\omega$ randomly.
12:     $(f', n') \leftarrow BSMC(\mathcal{M}(\omega'), P_{\geq\theta}(\phi), L, h(\omega), (\epsilon_1, \epsilon_2)]$
13:     if $f' = true$ then
14:         $ans \leftarrow \omega'$
15:         return
16:     end if
17:     if $n' > n$ then
18:         $\omega \leftarrow \omega'$
19:         $f \leftarrow f'$
20:         $n \leftarrow n'$
21:     else
22:         if $rand(0, 1) < \exp(-(n' - n)/t)$ then
23:             $\omega \leftarrow \omega'$
24:             $f \leftarrow f'$
25:             $n \leftarrow n'$
26:         end if
27:     end if
28:     $t \leftarrow T(lcount)$
29: end while
30: $ans \leftarrow$ "No parameter found"
31: return $ans$
```

CHAPTER 5: CONCLUSION[1]

Probabilistic computational models have been to used to analyze complex phenomena in areas that include the study of complex economic phenomena, global ecology, forced migration, the spread of infectious diseases and threats to international security. In recent years, there have been attempts to automate the discovery of model parameters using modern high-performance computing techniques. The ongoing exponential increase in computational power provides an opportunity for building software that can automatically find parameter values of complex stochastic models, given specifications describing the relevant properties the completed model should ideally have.

This dissertation discussed new algorithmic techniques for parameter synthesis that use the Sequential Probability Ratio Test (SPRT), Bayesian statistical model checking and simulated annealing to find a model instantiation that meets expert-provided behavioral specifications. We applied our SPRT-based parameter estimation technique to discover three parameters in a complex glucose-insulin model that helped validate key properties of artificial pancreata. We applied our Bayesian model checking-based algorithm to discover twenty-eight parameters in a model of the acute inflammatory response in humans to satisfy four behavioral specification describing a variety of responses due to varying schedules and doses of administering endotoxin.

We have thus demonstrated how, given user-provided behavioral properties, our new algorithms based on formal verification, statistical hypothesis testing and mathematical optimization automatically synthesize numerical parameters in computational models of probabilistic systems. Next, we briefly discuss possibilities for future work in the area.

[1]This chapter is based on concluding remarks from our earlier papers on algorithmic techniques for parameter estimation [57, 58, 60, 59].

5.1 Future Work

We plan to pursue several directions for future work. First, the current sampling strategy of drawing a fresh set of samples for every parameter perturbation is wasteful and may be avoided by employing change of measures arguments and reusing earlier samples [62].

We also plan to automate the process of *learning specifications* from time-series data, which will allow researchers to build better models using specifications generated automatically from experimental data that they can then curate. Another interesting area of research is the use of unbounded temporal specifications. This would permit the specification of interesting properties including cyclic behavior and periodic oscillations [35].

After comparing different parameter estimation techniques for stochastic models in systems biology, we are convinced of the need to develop open benchmarks of computational models and experimental datasets (like time-series data) that would help evaluate existing and proposed solutions to the parameter estimation problem.

We are also keenly interested in studying the problem of parameter sensitivity, i.e. ensuring that parameter values discovered should be robust enough so that a slight change in them does not cause drastically different model behavior. This problem is especially important in biology because experimental data are often not only sparse but also contains measurement errors [67].

Finally, users of search algorithms are always concerned about the issue of scalability, i.e. whether or not the technique would work efficiently when the problem size is large, resulting in a high-dimensional parameter space. To address this issue, we plan to investigate various model reduction techniques [13].

REFERENCES

[1] Emile Aarts, Jan Korst, and Wil Michiels. “Simulated annealing”. In: *Search methodologies*. Springer, 2005, pp. 187–210. [see p. 32]

[2] Simone Alfarano, Thomas Lux, and Friedrich Wagner. “Estimation of agent-based models: the case of an asymmetric herding model”. In: *Computational Economics* 26.1 (2005), pp. 19–49. [see p. 39]

[3] Linda JS Allen. *An introduction to stochastic processes with applications to biology*. CRC Press, 2010. [see p. 37]

[4] Rajeev Alur, Costas Courcoubetis, Nicolas Halbwachs, Thomas A Henzinger, P-H Ho, Xavier Nicollin, Alfredo Olivero, Joseph Sifakis, and Sergio Yovine. “The algorithmic analysis of hybrid systems”. In: *Theoretical computer science* 138.1 (1995), pp. 3–34. [see p. 6]

[5] Gary An. “Concepts for developing a collaborative in silico model of the acute inflammatory response using agent-based modeling”. In: *Journal of critical care* 21.1 (2006), pp. 105–110. [see p. 39]

[6] Gary An, Qi Mi, Joyeeta Dutta-Moscato, and Yoram Vodovotz. “Agent-based models in translational systems biology”. In: *Wiley Interdisciplinary Reviews: Systems Biology and Medicine* 1.2 (2009), pp. 159–171. [see pp. 31, 38, 39]

[7] Marco Antoniotti, Alberto Policriti, Nadia Ugel, and Bud Mishra. “Model building and model checking for biochemical processes”. In: *Cell Biochemistry and Biophysics* 38.3 (2003), pp. 271–286. [see pp. 5, 30]

[8] Richard C Aster, Brian Borchers, and Clifford H Thurber. *Parameter estimation and inverse problems*. Academic Press, 2013. [see pp. 1, 5]

[9] Christel Baier, Joost-Pieter Katoen, et al. *Principles of model checking*. Vol. 26202649. MIT press Cambridge, 2008. [see pp. 31, 37, 41, 43]

[10] Osman Balci. “Verification, validation, and testing”. In: *Handbook of simulation* (1998), pp. 335–393. [see p. 30]

[11] Julio R Banga. “Optimization in computational systems biology”. In: *BMC systems biology* 2.1 (2008), p. 1. [see p. 32]

[12] Gregory Batt, Michel Page, Irene Cantone, Gregor Goessler, Pedro Monteiro, and Hidde De Jong. “Efficient parameter search for qualitative models of regulatory networks using symbolic model checking”. In: *Bioinformatics* 26.18 (2010), pp. i603–i610. [see pp. 6, 35]

[13] Peter Benner, Serkan Gugercin, and Karen Willcox. “A survey of model reduction methods for parametric systems”. In: (2013). [see p. 56]

[14] Armin Biere, Marijn Heule, and Hans van Maaren. *Handbook of satisfiability*. Vol. 185. IOS Press, 2009. [see p. 8]

[15] Christopher M. Bishop. *Pattern recognition and Machine Learning*. Vol. 1. Springer, 2006. ISBN: 978-81-322-0906-5. [see p. 2]

[16] Eric Bonabeau. “Agent-based modeling: Methods and techniques for simulating human systems”. In: *Proceedings of the National Academy of Sciences* 99.suppl 3 (2002), pp. 7280–7287. [see p. 31]

[17] Benoit Calvez and Guillaume Hutzler. “Parameter space exploration of agent-based models”. In: *Knowledge-Based Intelligent Information and Engineering Systems*. Springer. 2005, pp. 633–639. [see p. 39]

[18] Laurence Calzone, Nathalie Chabrier-Rivier, François Fages, and Sylvain Soliman. “Machine learning biochemical networks from temporal logic properties”. In: *Transactions on Computational Systems Biology VI*. Springer, 2006, pp. 68–94. [see pp. 7, 35, 39]

[19] Laurence Calzone, François Fages, and Sylvain Soliman. “BIOCHAM: an environment for modeling biological systems and formalizing experimental knowledge”. In: *Bioinformatics* 22.14 (2006), pp. 1805–1807. [see p. 7]

[20] Irene Cantone, Lucia Marucci, Francesco Iorio, Maria Aurelia Ricci, Vincenzo Belcastro, Mukesh Bansal, Stefania Santini, Mario Di Bernardo, Diego Di Bernardo, and Maria Pia Cosma. “A yeast synthetic network for in vivo assessment of reverse-engineering and modeling approaches”. In: *Cell* 137.1 (2009), pp. 172–181. [see pp. 6, 35]

[21] Roger Carr. *Simulated Annealing*. `http://mathworld.wolfram.com/SimulatedAnnealing.html`. [From MathWorld–A Wolfram Web Resource, created by Eric W. Weisstein]. [see pp. 17, 33]

[22] Edmund M. Clarke, Orna Grumberg, and Doron Peled. *Model checking*. MIT Press, 2001. [see pp. 3, 4]

[23] Edmund M Clarke and Robert P Kurshan. “Computer-aided verification”. In: *Spectrum, IEEE* 33.6 (1996), pp. 61–67. [see pp. 3, 4]

[24] Edmund Clarke, Ansgar Fehnker, Sumit Kumar Jha, and Helmut Veith. “Temporal logic model checking”. In: *Handbook of Networked and Embedded Control Systems*. Springer, 2005, pp. 539–558. [see p. 35]

[25] Edmund Clarke, Orna Grumberg, Somesh Jha, Yuan Lu, and Helmut Veith. “Counterexample-guided abstraction refinement”. In: *Computer aided verification*. Springer. 2000, pp. 154–169. [see pp. 6, 8]

[26] Avra Cohn. “The notion of proof in hardware verification”. In: *Journal of Automated Reasoning* 5.2 (1989), pp. 127–139. [see p. 3]

[27] Costas Courcoubetis and Mihalis Yannakakis. “The complexity of probabilistic verification”. In: *Journal of the ACM (JACM)* 42.4 (1995), pp. 857–907. [see p. 4]

[28] Judy Day, Jonathan Rubin, Yoram Vodovotz, Carson C Chow, Angela Reynolds, and Gilles Clermont. “A reduced mathematical model of the acute inflammatory response II. Capturing scenarios of repeated endotoxin administration”. In: *Journal of theoretical biology* 242.1 (2006), pp. 237–256. [see pp. 32, 50, 53]

[29] Edsger Wybe Dijkstra, Edsger Wybe Dijkstra, Edsger Wybe Dijkstra, and Edsger Wybe Dijkstra. *A discipline of programming*. Vol. 1. prentice-hall Englewood Cliffs, 1976. [see p. 3]

[30] Susanne Ditlevsen and Adeline Samson. “Introduction to stochastic models in biology”. In: *Stochastic biomathematical models*. Springer, 2013, pp. 3–35. [see p. 37]

[31] Robin Donaldson and David Gilbert. “A model checking approach to the parameter estimation of biochemical pathways”. In: *Computational Methods in Systems Biology*. Springer. 2008, pp. 269–287. [see pp. 6, 35]

[32] Alexandre Donzé, Gilles Clermont, and Christopher J Langmead. “Parameter synthesis in nonlinear dynamical systems: Application to systems biology”. In: *Journal of Computational Biology* 17.3 (2010), pp. 325–336. [see pp. 7, 35]

[33] Tommaso Dreossi and Thao Dang. “Parameter synthesis for polynomial biological models”. In: *Proceedings of the 17th international conference on Hybrid systems: computation and control*. ACM. 2014, pp. 233–242. [see p. 35]

[34] Joyeeta Dutta-Moscato, Alexey Solovyev, Qi Mi, Taichiro Nishikawa, Alejandro Soto-Gutierrez, Ira J Fox, and Yoram Vodovotz. “A multiscale agent-based in silico model of liver fibrosis progression”. In: *Frontiers in bioengineering and biotechnology* 2 (2014), p. 18. [see p. 39]

[35] E Allen Emerson. “Temporal and modal logic.” In: *Handbook of Theoretical Computer Science, Volume B: Formal Models and Sematics (B)* 995.1072 (1990), p. 5. [see pp. 32, 56]

[36] E Allen Emerson. “The beginning of model checking: A personal perspective”. In: *25 Years of Model Checking*. Springer, 2008, pp. 27–45. [see p. 3]

[37] Heinz W Engl, Christoph Flamm, Philipp Kügler, James Lu, Stefan Müller, and Peter Schuster. “Inverse problems in systems biology”. In: *Inverse Problems* 25.12 (2009), p. 123014. [see pp. 31, 36]

[38] François Fages and Aurélien Rizk. “On the analysis of numerical data time series in temporal logic”. In: *Computational Methods in Systems Biology*. Springer. 2007, pp. 48–63. [see p. 6]

[39] Bernd Finkbeiner and Henny Sipma. “Checking finite traces using alternating automata”. In: *Formal Methods in System Design* 24.2 (2004), pp. 101–127. [see pp. 14, 41]

[40] Jasmin Fisher and Thomas A Henzinger. “Executable cell biology”. In: *Nature biotechnology* 25.11 (2007), pp. 1239–1249. [see p. 37]

[41] Robert W Floyd. “Assigning meanings to programs”. In: *Mathematical aspects of computer science* 19.19-32 (1967), p. 1. [see p. 3]

[42] Goran Frehse, Sumit Kumar Jha, and Bruce H Krogh. “A counterexample-guided approach to parameter synthesis for linear hybrid automata”. In: *Hybrid Systems: Computation and Control*. Springer, 2008, pp. 187–200. [see pp. 6, 11]

[43] Daniel T Gillespie. “Exact stochastic simulation of coupled chemical reactions”. In: *The journal of physical chemistry* 81.25 (1977), pp. 2340–2361. [see pp. 8, 11, 13]

[44] Manfred Gilli and Peter Winker. “A global optimization heuristic for estimating agent based models”. In: *Computational Statistics & Data Analysis* 42.3 (2003), pp. 299–312. [see p. 39]

[45] Antoine Girard and George J Pappas. “Verification using simulation”. In: *Hybrid Systems: Computation and Control*. Springer, 2006, pp. 272–286. [see p. 3]

[46] Orland R Gonzalez, Christoph Küper, Kirsten Jung, Prospero C Naval, and Eduardo Mendoza. “Parameter estimation using Simulated Annealing for S-system models of biochemical networks”. In: *Bioinformatics* 23.4 (2007), pp. 480–486. [see pp. 8, 16, 33]

[47] Sumit Gulwani, Saurabh Srivastava, and Ramarathnam Venkatesan. “Program analysis as constraint solving”. In: *ACM SIGPLAN Notices* 43.6 (2008), pp. 281–292. [see p. 5]

[48] Jeremy Gunawardena. “Models in systems biology: the parameter problem and the meanings of robustness”. In: *Elements of computational systems biology* 1 (2010). [see p. 31]

[49] Aarti Gupta. “Formal hardware verification methods: A survey”. In: *Computer-Aided Verification*. Springer. 1993, pp. 5–92. [see p. 3]

[50] John Harrison. “Formal proof–theory and practice”. In: *Notices of the AMS* 55.11 (2008), pp. 1395–1406. [see p. 3]

[51] John Heath, Marta Kwiatkowska, Gethin Norman, David Parker, and Oksana Tymchyshyn. “Probabilistic model checking of complex biological pathways”. In: *Theoretical Computer Science* 391.3 (2008), pp. 239–257. [see p. 4]

[52] Thomas A Henzinger, Pei-Hsin Ho, and Howard Wong-Toi. “HyTech: A model checker for hybrid systems”. In: *Computer aided verification*. Springer. 1997, pp. 460–463. [see p. 6]

[53] Thomas A Henzinger and Howard Wong-Toi. *Using HyTech to synthesize control parameters for a steam boiler*. Springer, 1996. [see pp. 2, 6]

[54] Thomas Hérault, Richard Lassaigne, Frédéric Magniette, and Sylvain Peyronnet. “Approximate probabilistic model checking”. In: *Verification, Model Checking, and Abstract Interpretation*. Springer. 2004, pp. 73–84. [see p. 44]

[55] Andrew Hinton, Marta Kwiatkowska, Gethin Norman, and David Parker. "PRISM: A tool for automatic verification of probabilistic systems". In: *Tools and Algorithms for the Construction and Analysis of Systems*. Springer, 2006, pp. 441–444. [see p. 7]

[56] Charles Antony Richard Hoare. "An axiomatic basis for computer programming". In: *Communications of the ACM* 12.10 (1969), pp. 576–580. [see p. 3]

[57] Faraz Hussain, Raj Gautam Dutta, Sumit Kumar Jha, Christopher James Langmead, and Susmit Jha. "Parameter Discovery for Stochastic Biological Models against Temporal Behavioral Specifications using an SPRT based Metric for Simulated Annealing". In: *Proceedings of the 2nd IEEE International Conference on Computational Advances in Bio and Medical Sciences (ICCABS 2012)*. Las Vegas, NV. IEEE Computer Society, Feb. 2012, pp. 1–6. DOI: `10.1109/ICCABS.2012.6182640`. [see pp. v, 9, 49, 55]

[58] Faraz Hussain, Sumit Kumar Jha, Susmit Jha, and Christopher James Langmead. "Parameter discovery in stochastic biological models using simulated annealing and statistical model checking". In: *International Journal of Bioinformatics Research and Applications* 10.4/5 (2014), pp. 519–539. DOI: `10.1504/IJBRA.2014.062998`. [see pp. v, 9, 27, 49, 55]

[59] Faraz Hussain, Christopher J. Langmead, Qi Mi, Joyeeta Dutta-Moscato, Yoram Vodovotz, and Sumit K. Jha. "Automated parameter estimation for biological models using Bayesian statistical model checking". In: *BMC Bioinformatics* 16(Suppl 17).S8 (2015), pp. 1–14. DOI: `10.1186/1471-2105-16-S17-S8`. [see pp. v, 29, 55]

[60] Faraz Hussain, Christopher James Langmead, Qi Mi, Joyeeta Dutta-Moscato, Yoram Vodovotz, and Sumit Kumar Jha. "Parameter Discovery for Stochastic Computational Models in Systems Biology Using Bayesian Model Checking". In: *Proceedings of the 4th IEEE International Conference on Computational Advances in Bio and Medical*

Sciences (ICCABS 2014). Miami, FL. IEEE, June 2014, pp. 1–6. DOI: 10.1109/ICCABS.2014.6863925. [see pp. v, 29, 55]

[61] Sumit K Jha, Edmund M Clarke, Christopher J Langmead, Axel Legay, André Platzer, and Paolo Zuliani. “A bayesian approach to model checking biological systems”. In: *Computational Methods in Systems Biology*. Springer. 2009, pp. 218–234. [see pp. 32, 45]

[62] Sumit K Jha and Christopher J Langmead. “Exploring behaviors of stochastic differential equation models of biological systems using change of measures”. In: *BMC bioinformatics* 13.Suppl 5 (2012), S8. [see p. 56]

[63] Sumit Kumar Jha. “Model validation and discovery for complex stochastic systems”. PhD thesis. Carnegie Mellon University, 2010. [see pp. 41, 42, 45, 48]

[64] Sumit Kumar Jha and Christopher James Langmead. “Synthesis and infeasibility analysis for stochastic models of biochemical systems using statistical model checking and abstraction refinement”. In: *Theoretical Computer Science* 412.21 (2011), pp. 2162–2187. [see pp. 2, 8, 11]

[65] Susmit Jha. “Towards Automated System Synthesis Using SCIDUCTION”. PhD thesis. PhD thesis, University of California at Berkeley, 2011. [see p. 5]

[66] Stuart A. Kauffman. *The origins of order: Self organization and selection in evolution*. Oxford University Press, USA, 1993. [see p. 8]

[67] Hiroaki Kitano. “Systems biology: a brief overview”. In: *Science* 295.5560 (2002), pp. 1662–1664. [see pp. 37, 56]

[68] Jack PC Kleijnen. “Verification and validation of simulation models”. In: *European Journal of Operational Research* 82.1 (1995), pp. 145–162. [see p. 3]

[69] Geoffrey Koh, Huey Fern Carol Teong, Marie-Véronique Clément, David Hsu, and PS Thiagarajan. “A decompositional approach to parameter estimation in pathway modeling: a case study of the Akt and MAPK pathways and their crosstalk”. In: *Bioinformatics* 22.14 (2006), e271–e280. [see p. 34]

[70] Rukmini Kumar, Gilles Clermont, Yoram Vodovotz, and Carson C Chow. “The dynamics of acute inflammation”. In: *Journal of theoretical biology* 230.2 (2004), pp. 145–155. [see p. 35]

[71] Robert P Kurshan. “Program verification”. In: *Notices of the AMS* 47.5 (2000), pp. 534–545. [see p. 3]

[72] Marta Kwiatkowska, Gethin Norman, and David Parker. “Stochastic model checking”. In: *Formal methods for performance evaluation*. Germany. Springer, 2007, pp. 220–270. [see pp. 31, 41]

[73] Marta Kwiatkowska, Gethin Norman, and David Parker. “Using probabilistic model checking in systems biology”. In: *ACM SIGMETRICS Performance Evaluation Review* 35.4 (2008), pp. 14–21. [see p. 37]

[74] Tze Leung Lai. *Sequential analysis*. Wiley Online Library, 2001. [see p. 27]

[75] Christopher James Langmead and Sumit Kumar Jha. “Predicting protein folding kinetics via temporal logic model checking”. In: *Algorithms in Bioinformatics*. Springer, 2007, pp. 252–264. [see p. 7]

[76] Christopher James Langmead and Sumit Kumar Jha. “Symbolic approaches for finding control strategies in Boolean networks”. In: *Journal of Bioinformatics and Computational Biology* 7.02 (2009), pp. 323–338. [see p. 8]

[77] Reinhard Laubenbacher, Abdul S Jarrah, Henning S Mortveit, and SS Ravi. “Agent based modeling, mathematical formalism for”. In: *Encyclopedia of complexity and systems science* (2009), pp. 160–176. [see p. 39]

[78] Axel Legay, Benoit Delahaye, and Saddek Bensalem. “Statistical model checking: An overview”. In: *Runtime Verification*. Springer. 2010, pp. 122–135. [see pp. 8, 42, 45]

[79] Gabriele Lillacci and Mustafa Khammash. “Parameter estimation and model selection in computational biology”. In: *PLoS computational biology* 6.3 (2010), e1000696. [see pp. 1, 5, 7, 33]

[80] Herbert S Lin, John C Wooley, et al. *Catalyzing inquiry at the interface of computing and biology*. Washington DC, USA. National Academies Press, 2005. [see pp. 30–32]

[81] Charles M Macal and Michael J North. “Tutorial on agent-based modelling and simulation”. In: *Journal of simulation* 4.3 (2010), pp. 151–162. [see p. 39]

[82] C Dalla Man, Robert A Rizza, and Claudio Cobelli. “Meal simulation model of the glucose-insulin system”. In: *Biomedical Engineering, IEEE Transactions on* 54.10 (2007), pp. 1740–1749. [see pp. 13, 20]

[83] Toni Mancini, Enrico Tronci, Ivano Salvo, Federico Mari, Annalisa Massini, and Igor Melatti. “Computing Biological Model Parameters by Parallel Statistical Model Checking”. In: *Bioinformatics and Biomedical Engineering*. Switzerland. Springer, 2015, pp. 542–554. [see p. 36]

[84] Zohar Manna and Richard Waldinger. “A deductive approach to program synthesis”. In: *ACM Transactions on Programming Languages and Systems (TOPLAS)* 2.1 (1980), pp. 90–121. [see p. 5]

[85] Zohar Manna and Richard J Waldinger. “Toward automatic program synthesis”. In: *Communications of the ACM* 14.3 (1971), pp. 151–165. [see p. 5]

[86] Shibin Mathew, John Bartels, Ipsita Banerjee, and Yoram Vodovotz. “Global sensitivity analysis of a mathematical model of acute inflammation identifies nonlinear dependence of cumulative tissue damage on host interleukin-6 responses”. In: *Journal of theoretical biology* 358 (2014), pp. 132–148. [see p. 34]

[87] Hiroshi Matsuno, Yukiko Tanaka, Hitoshi Aoshima, Mika Matsui, Satoru Miyano, et al. "Biopathways representation and simulation on hybrid functional Petri net". In: *In silico biology* 3.3 (2003), pp. 389–404. [see p. 34]

[88] Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. "Equation of state calculations by fast computing machines". In: *The journal of chemical physics* 21.6 (1953), pp. 1087–1092. [see p. 17]

[89] Melanie Mitchell. *An introduction to genetic algorithms.* MIT press, 1998. [see p. 6]

[90] Carmen G Moles, Pedro Mendes, and Julio R Banga. "Parameter estimation in biochemical pathways: a comparison of global optimization methods". In: *Genome research* 13.11 (2003), pp. 2467–2474. [see pp. 7, 34]

[91] Arthur N Prior. *Past, present and future.* Vol. 154. Clarendon Press Oxford, 1967. [see p. 4]

[92] Sanguthevar Rajasekaran. "On simulated annealing and nested annealing". In: *Journal of Global Optimization* 16.1 (2000), pp. 43–56. [see pp. 7, 17, 48]

[93] S Reinker, RM Altman, and J Timmer. "Parameter estimation in stochastic biochemical reactions". In: *IEE Proceedings-Systems Biology* 153.4 (2006), pp. 168–178. [see pp. 11, 33]

[94] Angela Reynolds, Jonathan Rubin, Gilles Clermont, Judy Day, Yoram Vodovotz, and G Bard Ermentrout. "A reduced mathematical model of the acute inflammatory response: I. Derivation of model and analysis of anti-inflammation". In: *Journal of theoretical biology* 242.1 (2006), pp. 220–236. [see p. 35]

[95] Janne Riionheimo and Vesa Valimaki. "Parameter estimation of a plucked string synthesis model using a genetic algorithm with perceptual fitness calculation". In: *EURASIP Journal on Advances in Signal Processing* 2003.8 (2003), pp. 1–15. [see p. 34]

[96] Aurélien Rizk, Grégory Batt, François Fages, and Sylvain Soliman. “On a continuous degree of satisfaction of temporal logic formulae with applications to systems biology”. In: *Computational Methods in Systems Biology*. Springer. 2008, pp. 251–268. [see p. 36]

[97] Maria Rodriguez-Fernandez, Pedro Mendes, and Julio R Banga. “A hybrid approach for efficient and robust parameter estimation in biochemical pathways”. In: *Biosystems* 83.2 (2006), pp. 248–265. [see pp. 1, 34]

[98] Robert G Sargent. “Verification and validation of simulation models”. In: *Journal of simulation* 7.1 (2013), pp. 12–24. [see p. 30]

[99] Russell Schwartz. *Biological modeling and simulation: a survey of practical models, algorithms, and numerical methods*. Cambridge, Massachusetts, USA. MIT Press, 2008. [see p. 30]

[100] Natarajan Shankar. *Metamathematics, machines and Gödel’s proof*. 38. Cambridge University Press, 1997. [see p. 3]

[101] Alexey Solovyev, Maxim Mikheev, Leming Zhou, Joyeeta Dutta-Moscato, Cordelia Ziraldo, Gary An, Yoram Vodovotz, and Qi Mi. “SPARK: A Framework for Multi-Scale Agent-Based Biomedical Modeling”. In: *International Journal of Agent Technologies and Systems* 2.3 (2010), pp. 18–30. [see pp. 39, 51]

[102] *Simple Platform for Agent-based Representation of Knowledge (SPARK)*. `http://www.pitt.edu/~cirm/spark/`. Accessed: 2015-06-23. [see pp. 39, 51]

[103] Jianyong Sun, Jonathan M Garibaldi, and Charlie Hodgman. “Parameter estimation using metaheuristics in systems biology: a comprehensive review”. In: *Computational Biology and Bioinformatics, IEEE/ACM Transactions on* 9.1 (2012), pp. 185–202. [see p. 33]

[104] Albert Tarantola. *Inverse problem theory: Methods for data fitting and model parameter estimation*. Elsevier Science, 2002. [see pp. 1, 5, 6]

[105] Andres Torres, Timothy Bentley, John Bartels, Joydeep Sarkar, Derek Barclay, Rajaie Namas, Gregory Constantine, Ruben Zamora, Juan Carlos Puyana, and Yoram Vodovotz. “Mathematical modeling of posthemorrhage inflammation in mice: studies using a novel, computer-controlled, closed-loop hemorrhage apparatus”. In: *Shock* 32.2 (2009), pp. 172–178. [see p. 34]

[106] Abraham Wald. *Sequential analysis.* Dover Phoenix, 2014. ISBN: 978-0-486-61579-0. [see pp. 15, 16, 19, 24, 49]

[107] Darren J Wilkinson. “Parameter inference for stochastic kinetic models of bacterial gene regulation: a Bayesian approach to systems biology”. In: *Proceedings of 9th Valencia International Meeting on Bayesian Statistics.* 2010, pp. 679–705. [see p. 11]

[108] Darren J Wilkinson. “Stochastic modelling for quantitative description of heterogeneous biological systems”. In: *Nature Reviews Genetics* 10.2 (2009), pp. 122–133. [see pp. 30, 36, 37]

[109] Xiaorong Xiang, Ryan Kennedy, Gregory Madey, and Steve Cabaniss. “Verification and validation of agent-based scientific simulation models”. In: *Agent-Directed Simulation Conference.* 2005, pp. 47–55. [see p. 31]

[110] Håkan LS Younes, Marta Kwiatkowska, Gethin Norman, and David Parker. “Numerical vs. statistical probabilistic model checking”. In: *International Journal on Software Tools for Technology Transfer* 8.3 (2006), pp. 216–228. [see p. 15]

[111] Håkan LS Younes and Reid G Simmons. “Probabilistic verification of discrete event systems using acceptance sampling”. In: *Computer Aided Verification.* Springer. 2002, pp. 223–235. [see p. 44]

[112] Håkan LS Younes and Reid G Simmons. “Statistical probabilistic model checking with a focus on time-bounded properties”. In: *Information and Computation* 204.9 (2006), pp. 1368–1409. [see pp. 5, 11, 15, 19, 21, 23, 44, 45]

[113] Paolo Zuliani, André Platzer, and Edmund M Clarke. "Bayesian statistical model checking with application to Stateflow/Simulink verification". In: *Formal Methods in System Design* 43.2 (2013), pp. 338–367. [see p. 42]