Numerical Analysis of Long-Run Properties for Markov Population Models

Dissertation

zur Erlangung des Grades des
Doktors der Naturwissenschaften
an der Naturwissenschaftlich-Technischen Fakultät I
der Universität des Saarlandes

von
David Spieler

Saarbrücken
2014

Tag des Kolloquiums:	28.11.2014
Dekan :	Prof. Dr. Markus Bläser
Berichterstatter:	Prof. Dr. Paolo Ballarini Prof. Dr. Boudewijn Haverkort Prof. Dr. Verena Wolf
Vorsitzender:	Prof. Bernd Finkbeiner, Ph.D.
Akademischer Mitarbeiter:	Dr. Jan Krčál

Abstract

One of the most versatile modeling formalism is the one given by Markov chains as used for the performance analysis of queuing systems or for cost benefit ratio optimizations in the financial sector

In systems biology, chemical reaction networks have originally been studied using deterministic models. However, when it recently became apparent that only stochastic effects can explain certain phenomenons, Markov chains again turned out to be a suitable modeling formalism in the form of Markov population models. Those Markov chains possess a structured but potentially infinite state space where each state encodes the current counts of a fixed number of population types.

Due to the infinite state space, classical steady state analysis methods can not be used directly. Hence, this doctoral thesis presents a highly efficient method to compute a finite state space truncation entailing most of the steady state probability mass. Further, stochastic complementation is lifted to the infinite state space setting and is combined with truncation based reachability analysis and aggregation to derive state wise steady state bounds. This method achieves high performance even for stiff systems. Particular attention is paid on a system's ability to maintain stable oscillations and thus optimized analysis methods are developed alongside. In order to prove their applicability, all techniques are evaluated on a large variety of biological models.

Zusammenfassung

Ursprünglich wurden chemische Reaktionsnetzwerke in der Systembiologie mit Hilfe von deterministischen Modellen analysiert. Als jedoch klar wurde, dass bestimmte Phänomene nur durch stochastische Effekte erklärt werden können, erwiesen sich Markovsche Populationsmodelle als geeigneter Formalismus. Diese Markovketten besitzen einen strukturierten Zustandsraum, wobei ein Zustand die aktuelle Anzahl einer oder mehrerer Populationstypen kodiert.

Oft ist dieser Zustandsraums jedoch unendlich groß und klassische Methoden für die Analyse des Equilibriums können nicht benutzt werden. Diese Doktorarbeit präsentiert daher eine effiziente Methode, ein endlich großes Fenster im Zustandsraum zu berechnen, welches den Großteil der Equilibriumswahrscheinlichkeitsmasse umschließt. Zudem wird das Verfahren der stochastischen Komplementierung für das vorliegende Szenario erweitert und mit Aggregationsmethoden und der Erreichbarkeitsanalyse basierend auf dem Abschneiden von nicht signifikanten Zuständen kombiniert. Diese Methode erlaubt die Berechnung von Schranken für die Equilibriumswahrscheinlichkeit aller Zustände innerhalb des Fensters und ist performant – sogar für steife Systeme. Der Fähigkeit eines Systems, stabile Oszillationen aufrecht zu erhalten, wird spezielle Aufmerksamkeit geschenkt und die entsprechende Analyse optimiert. Um ihre Einsetzbarkeit zu zeigen werden alle Methoden anhand einer Vielzahl biologischer Modelle evaluiert.

Acknowledgements

When I started studying computer science at Saarland university, I immediately felt that I am in good hands. In my oppinion it is due to the dedication of the lecturers as well as the strong emphasis on the theoretical foundations of computer science that I was finally able to take a doctoral degree. Consequently, it was already in the first semester, when Holger Hermanns with great enthusiasm introduced me to the world of formal methods by showing what to do against evil vending machines modelled as CCS processes. Thus, it was hardly surprising that I decided to do my Bachelor's degree at the chair of *Dependable Software and Systems*. This way, I got in contact with business processes and a language called WS-BPEL 2.0 that was in desperate need of a formal sementics, I was determined to give.

Although, I liked that topic a lot, when it was time to find a suitable topic for my Master's thesis, I consulted Holger Hermanns to give me a new and trendy topic. It was at that time when also Verena Wolf joined the university as a junior research group leader and brought the facinating application area of systems biology along. We quickly decided that I should investigate how one could capture the essence of oscillatory behavior using formal methods like model checking. Consequently, it is Holger and Verena, my scientific mentors, I want to thank a lot for preparing me so well for the big task of doing a doctorate and finally giving me a position at the respective chairs such that I could

do the research in the best scientific environment one could think of. Alongside, I would like to thank all the people that have worked at the chair of *Modeling and Simulation* as well as at the chair of Dependable Software and Systems I have worked with: Alexander Andreychenko, Tuğrul Dayar, Christian Eisentraut, Luis Maria Ferrer Fioriti, Alexander Graf-Brill, Ernst Moritz Hahn, Arnd Hartmanns, Vahid Hashemi, Hassan Hatefi, Thilo Krüger, Linar Mikeev, Martin Neuhäußer, Werner Sandmann, Lei Song, Andrea Turrini, Lijun Zhang, and Sascha Zickenrott. Many thanks also go to all the people I have met in the academic world at conferences and workshops, especially in the ROCKS project and at the various AVACS meetings.

Last but not least, I would like to thank my family, especially my parents Michaela and Dieter as well as my sister Johanna, who always cheered me up when I was stuck. Special thanks goes to all my friends, especially to those in Bavaria who reminded me of my roots and never to forget that *dahoam* is *dahoam*. And finally, above all, I want to thank my wife Natalia who entered my life exactly at the right time.

Contents

CHAPTER 1

Introduction

The task of the highly inter-disciplinary field of *systems biology* is to capture the important characteristics of biological systems in mathematical models and to study those models in order to discover emergent properties. The traditional approach [FTY11, Kri06, SMH02] is to formalize chemical reaction networks using *ordinary differential equations* (ODEs) which have a *deterministic* semantics. This means, that given an initial condition, the future behavior is fully determined. Recent insights however suggest that a completely deterministic approach might not be appropriate in all of the cases since *noise* resulting from low population counts of certain chemical species plays an important role. For example, *circadian clocks*, the basic mechanism behind the 24 hour day-night rhythm affecting the sleeping behavior of mammals, rely on stochastic effects to maintain an oscillatory pattern and to escape equilibrium states [BL00]. This and other results [ARM98, MA97] clearly motivate the use of stochastic modeling. The common stochastic framework justified [Gil92] for those systems are *continuous-time Markov chains* (CTMCs) which are also widely used in the world of formal methods [BHHK03]. Since in those cases, a state encodes the number of molecules of each chemical species or *population type*, the respective CTMC is called a *Markov population model* (MPM).

Most quantities of MPMs that are of interest can be determined either via *transient*, *reachability*, or *steady state analysis* or combinations thereof developed for CTMCs [Ste94, BHHK03]. While efficient numerical techniques for CTMCs with finite state space exist [Ste94], biologically motivated models often posses a countably infinite state space. The reason is that usually there are no bounds on the respective molecule counts given a priori. Consequently, those techniques – based on vector matrix multiplications and solutions of linear equation systems – can not be used without adaption.

This doctoral thesis specializes on the development of effective and efficient techniques to study the *long-term behavior* of MPMs mainly associated with the steady state, applying those results mainly to models from the area of systems biology. The focus lies on a broad applicability of those methods and thus, occurring problems, like different time scales in the behavior of certain sub-systems which would hinder or slow down the analysis, are taken care of. Finally, this thesis studies *oscillatory* MPMs, where the quantity of certain populations fluctuates regularly over time while considering noise introduced by the stochastic nature of the model class.

1.1 Contribution

The contribution of this doctoral thesis consists of:

(i) a *semi-automatic* method involving user-defined *Lyapunov functions* to compute a *finite truncation* of the potentially infinite state space of an MPM that contains a major part of the total steady state probability mass,

(ii) provable *lower* and *upper bounds* for the steady state probabilities of all states inside that truncation,

(iii) an efficient approach to model check the *steady state operator* of CSL for MPM,

(iv) a fast approximation algorithm for the *multimodal* steady state distribution of MPMs possessing two or more *attracting regions*,

(v) a truncation-based approximation algorithm for the probability measure of all MPM paths that are *accepted* by a *(single-clock) DTA*, and

(vi) an efficient method to compute the *oscillation characteristics* of an MPM with respect to a user-defined observation measure on states, resembling the probability distribution functions of the measure's period length for a chosen minimal amplitude level.

Several of these topics have been developed in chronological succession since they depend on previous results. These dependencies are reflected in the order of the chapters of this thesis as explained in Section 1.2. All these contributions have been published and thus have been peer-reviewed. The corresponding list of publications is:

[DHSW11] T. Dayar, H. Hermanns, D. Spieler, and V. Wolf. Bounding the equilibrium distribution of Markov population models. *Numerical Linear Algebra with Applications*, 18(6):931–946, 2011.

[SW13] D. Spieler and V. Wolf. Efficient steady state analysis of multimodal Markov chains. In *Analytical and Stochastic Modeling Techniques and Applications*, volume 7984 of *Lecture Notes in Computer Science*, pages 380–395. Springer, Berlin Heidelberg, 2013.

[MNSW13] L. Mikeev, M. R. Neuhäußer, D. Spieler, and V. Wolf. On-the-fly verification and optimization of DTA-properties for large Markov chains. *Formal Methods in System Design*, 43(2):313–337, 2013.

[Spi13a] D. Spieler. Characterizing oscillatory and noisy periodic behavior in Markov population models. In *Proceedings of QEST*, pages 106–122, 2013.

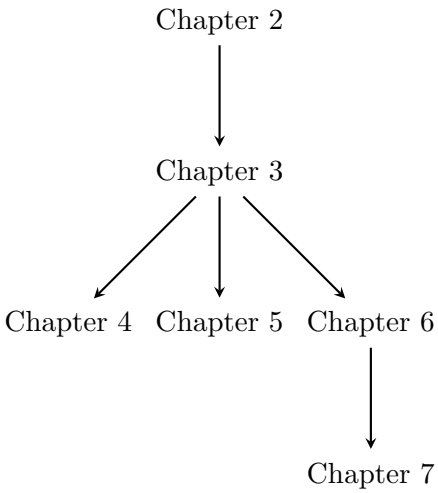


Figure 1.1: Chapter dependencies.

[SHZ14] D. Spieler, E. M. Hahn, and L. Zhang. Model checking CSL for Markov population models. In *Proceedings of QAPL*, Electronic Proceedings in Theoretical Computer Science, 2014.

[AKS14] A. Andreychenko, T. Kruüger, and D. Spieler. Analyzing oscillatory behavior with formal methods. In *Proceedings of the ROCKS autumn school 2013*, Lecture Notes in Computer Science, 2014.

1.2 Structure

Before we sketch the contents of the respective chapters of this thesis, we would like to point at the dependencies of the chapters on each other in Figure 1.1.

Chapter 2: In the second chapter we clarify the notation and define the basic mathematical concepts. Further, we recall those topics of probability theory that are needed later in this thesis and formally introduce *Markov population models* as well as a related symbolic description mechanisms called *transition classes*. Moreover, we review different transient and long-run analysis methods in the literature and argue why they fail in many situations when dealing with biologically motivated models.

Chapter 3: This chapter is the core of this thesis and provides a method to determine a *finite subset* of the state space of a model that comprises most of the total steady state probability mass circumventing the above mentioned problems. In addition we show how to bound the steady state probability mass *per state* by only taking into account those states inside that finite subset.

Chapter 4: A first application of this method is given in Chapter 4, where the *model checking* problem is tackled for the *continuous stochastic logic* and Markov population models with particular attention being paid on the steady state operator.

Chapter 5: When the model of consideration is *multimodal*, that is, its equilibrium distribution has two or more peaks, computation of the equilibrium is often complicated and time consuming. The reason usually is that the transitions work on different time scales, resulting in a stiff system of differential equations for the individual state probabilities. Consequently, Chapter 5 shows a way to combine techniques like stochastic complementation and aggregation to provide an efficient algorithm specially suited to those problematic models.

Chapter 6: The next chapter paves the way for analyzing the oscillatory character of a model. More precisely, we show how to approximate the probability of paths of a (possibly infinite state) continuous-time Markov chain that are accepted by a (single-clock) *deterministic timed automaton* on-the-fly using truncation based transient analysis. Further, we show how to approximate the derivatives of the acceptance probability with respect to model parameters in order to provide a way

to maximize that probability by exploiting local and global optimization strategies.

Chapter 7: Finally, we formally define what it means for a Markov population model to *oscillate* in Chapter 7 and give a specialized algorithm to approximate the distribution of period lengths of those oscillations depending of the minimal amplitude of interest.

CHAPTER 2

Preliminaries

In the further course of this thesis, many fundamental definitions and results mainly from the areas of linear algebra and stochastic processes will be used. In order to make the presentation consistent, we will also introduce several naming conventions.

2.1 Notation

Sets: We will sometimes use the term *infinite* set which shall always mean *countably infinite* set as we demand the existence of a bijection of the set to the natural numbers, if not stated otherwise. Given a family of sets $\mathcal{A}_i$ with $i \in \mathcal{I}$ for some index set $\mathcal{I}$, we denote the *disjoint union* by $\dot{\bigcup}_{i\in\mathcal{I}}\mathcal{A}_i$. Disjoint union of two sets $\mathcal{A}_1$ and $\mathcal{A}_2$ is denoted by $\mathcal{A}_1 \,\dot\cup\, \mathcal{A}_2$. Given an equivalence relation $\mathcal{R} \subseteq \mathcal{S} \times \mathcal{S}$, we denote the *set of equivalence classes* by $\mathcal{S}/\mathcal{R}$. When we write $\exists! a \in \mathcal{A}$ we mean the *unique existence* of an element that satisfies a certain property.

Numbers: The set of *natural numbers* $\mathbb{N} = \{0, 1, 2, \dots\}$ is extended to the set of *integers* $\mathbb{Z} = \{\dots, -2, -1, 0, 1, 2, \dots\}$. When we restrict to positive integers, we write $\mathbb{N}^+$ with $\mathbb{N}^+ = \mathbb{N} \setminus \{0\}$. We associate

the truth value *true* with 1 and *false* with 0. Consequently, the set of *booleans* is $\mathbb{B} = \{0, 1\} \subset \mathbb{N}$. The character $\mathbb{R}$ shall denote the set of *real numbers* and $\mathbb{C} = \{a + b \cdot \mathrm{i} \mid a, b \in \mathbb{R}\}$ the set of *complex numbers* with the *imaginary unit* i defined as $\mathrm{i} = (-1)^{-1}$. The restriction of $\mathbb{R}$ to the non-negative reals is denoted by $\mathbb{R}_0^+ = \{x \mid x \in \mathbb{R}, x \geq 0\}$ and to the positive reals by $\mathbb{R}^+ = \{x \mid x \in \mathbb{R}, x > 0\}$.

Functions: Functions like $f : \mathcal{A}_1 \to \mathcal{A}_2$ with *domain* $\mathrm{dom}(f) = \mathcal{A}_1$ and *image* $\mathrm{img}(f) = \mathcal{A}_2$ are considered to be *total*, that is, for all $a \in \mathcal{A}_1$ the function is defined and has a *unique* result $f(a) \in \mathcal{A}_2$.

Hash Maps: In various algorithms described later in this thesis, *hash maps* are used. We will treat hash maps as total functions $f : \mathcal{A}_1 \to \mathcal{A}_2$, where the domain $\mathcal{A}_1$ will be adjusted dynamically after key insertion and deletion, especially in order to guarantee totality. Given a hash map $f : \mathcal{A}_1 \to \mathcal{A}_2$, the statement $f(a) \leftarrow b$ will extend the domain of f to $\mathcal{A}_1 \cup \{a\}$ and the mapping is changed or extended, respectively, such that $f(a) = b$. The statement remove(f, a) will reduce the domain of f to $\mathcal{A}_1 \setminus \{a\}$. When we use hash maps with $0 \in \mathcal{A}_2 \subseteq \mathbb{R}$, we will sometimes evaluate them for keys $a \notin \mathcal{A}_1$ and implicitly assume they will return the value zero in these cases. Note that by using this convention, memory space can be saved when storing sparse probability distributions. Furthermore, given two hash maps f and g with $\mathrm{img}(f), \mathrm{img}(g) \subseteq \mathbb{R}$, we can define the hash map $h = c_1 \cdot f + c_2 \cdot g$ for $c_1, c_2 \in \mathbb{R}$ as the hash map $h : \mathrm{dom}(f) \cup \mathrm{dom}(g) \to \mathbb{R}$ with $h(a) = c_1 \cdot f(a) + c_2 \cdot g(a)$ for any $a \in \mathrm{dom}(f) \cup \mathrm{dom}(g)$. This way, also more complex arithmetic operations can be defined recursively. Note that we implicitly assume that the domains of f and g have been expanded to $\mathrm{dom}(f) \cup \mathrm{dom}(g)$ as well as that mappings to zero have been added where the functions were not defined originally, that is, on $\mathrm{dom}(f) \setminus \mathrm{dom}(g)$, respectively on $\mathrm{dom}(g) \setminus \mathrm{dom}(f)$. In order to delete all entries of a hash map $f : \mathcal{A}_1 \to \mathcal{A}_2$ we use the statement reset(f) which effectively sets both, the domain $\mathcal{A}_1$ and the image $\mathcal{A}_2$ to the empty set $\emptyset$.

Vectors: *Vectors* are represented by boldface lower case letters like $\mathbf{x}$ and are indexed by $\mathbf{x}_i$. Discrete *distributions* like $\pi(t)$ are row-vectors but are not represented by boldface letters. Scalars can be treated as

one-dimensional vectors. Vector $\mathbf{0}$ is the zero vector, $\mathbf{e}$ represents a vector of ones, and $\mathbf{e}^{(i)}$ is the vector of zeros with a one at position i. Whether these vectors denote a row- or column-vector is dependent on the context. Given a finite vector $\mathbf{x} = \begin{bmatrix}\mathbf{x}_1 & \dots & \mathbf{x}_n\end{bmatrix}$ and (possibly infinite) vector $\mathbf{y} = \begin{bmatrix}\mathbf{y}_1 & \mathbf{y}_2 & \dots\end{bmatrix}$, we define their concatenation as

$$\begin{bmatrix}\mathbf{x} & \mathbf{y}\end{bmatrix} = \begin{bmatrix}\mathbf{x}_1 & \dots & \mathbf{x}_n & \mathbf{y}_1 & \mathbf{y}_2 & \dots\end{bmatrix}.$$

Sometimes, we index a vector $\mathbf{x}$ by a set $\mathcal{A}$ as in $\mathbf{x}_\mathcal{A}$ and use it as a shorthand notation for the (scalar) sum $\sum_{a\in\mathcal{A}} \mathbf{x}_a$, where we rely on an index function defined later on. By $\mathbb{P}[\mathbf{x}]$ we denote the set of *multivariate polynomial functions* over $\mathbf{x} \in \mathbb{R}^N$, that is,

$$\mathbb{P}[\mathbf{x}] = \{p : \mathbb{R}^N \to \mathbb{R} \mid \exists c_0, c_i, c_{ij}, \dots \in \mathbb{R}. \forall \mathbf{x}. \\ p(\mathbf{x}) = c_0 + \sum_{i=1}^{N} c_i \cdot \mathbf{x}_i + \sum_{i=1}^{N}\sum_{j=1}^{i} c_{ij} \cdot \mathbf{x}_i \cdot \mathbf{x}_j + \dots\}.$$

To each subset relationship $\mathcal{A} \subseteq \mathcal{B}$, we associate a *characteristic function* $\phi^\mathcal{B}_\mathcal{A} : \mathcal{B} \to \mathbb{B}$ with $\phi^\mathcal{B}_\mathcal{A}(b) = 1$ if $b \in \mathcal{A}$ and 0 otherwise.

Matrices: *Matrices* like $\mathbf{Q}$ are symbolized by capital boldface letters with their components indexed by $\mathbf{Q}_{ij}$. Vectors can be treated as matrices where one of the two dimensions is one depending on whether it is a row- or column-vector. A matrix indexed by a row index and a set of column indices is defined as the sum $\mathbf{M}_{i\mathcal{A}} = \sum_{j\in\mathcal{A}} \mathbf{M}_{ij}$. A matrix $\mathbf{M}_\mathcal{A}$ indexed by a set of indices $\mathcal{A}$ is defined as the block $\mathbf{M}_\mathcal{A} = \begin{bmatrix}\mathbf{M}_a\end{bmatrix}_{a\in\mathcal{A}}$. The *identity matrix* $\mathbf{I}$ is a matrix with ones on the diagonal and zero for all off-diagonal entries. Transposition of a vector or a matrix is indicated by ^{T} as in $\mathbf{Q}^T$. A matrix $\mathbf{M} = \begin{bmatrix}f(i,j)\end{bmatrix}_{ij}$ denotes the matrix where each entry $\mathbf{M}_{ij}$ is defined by $\mathbf{M}_{ij} = f(x, y)$. We overload symbol $\mathbf{0}$ to denote the matrix of zeros depending on the context. The matrix exponential $e^\mathbf{M}$ of matrix $\mathbf{M}$ is defined as

$$e^\mathbf{M} = \sum_{i=0}^{\infty} \frac{\mathbf{M}^i}{i!},$$

where in general by $\mathbf{M}^i$ we mean the i-th power of a matrix $\mathbf{M}$. A matrix is *reducible* iff it can be placed into block upper-triangular form

by simultaneous row/column permutations, otherwise it is *irreducible*. Note that our notational formalism allows matrices to be of (countably) infinite size. Whenever this leads to possible problems or pitfalls like in Chapter 3, we will take special care. In general, we restrict to *row-/column-finite* (infinite) matrices [Hal82], that is, matrices where each row and each column has a finite number of non-zero entries.

Index Functions and Notational Simplifications: Index functions as defined in Definition 1 allow us to access the entries of vectors and matrices by vectors, which comes in handy when dealing with stochastic processes with multi-dimensional state spaces.

▷**Definition 1**

Definition 1: Index Function

An *index function* for a finite set $\mathcal{A}$ is a bijective function $f : \mathcal{A} \to \{1, \ldots, |\mathcal{A}|\}$. If $\mathcal{A}$ is a countably infinite set, an index function is a bijective function $f : \mathcal{A} \to \mathbb{N}$.

For example, given matrix $\mathbf{Q}$ with entries $\mathbf{Q}_{ij}$, those entries might be accessed via $\mathbf{Q}_{f(\mathbf{x})f(\mathbf{y})}$ with $\mathbf{x}, \mathbf{y} \in \mathbb{N}^N$ via an index function for the set $\mathbb{N}^N$. We will assume that suitable index functions $f(\mathbf{x})$ are implicitly given and simply write $\mathbf{Q}_{\mathbf{xy}}$ instead of $\mathbf{Q}_{f(\mathbf{x})f(\mathbf{y})}$. We note that index functions exist for all vector spaces $\mathbb{N}^N$ regardless of the dimension $N \in \mathbb{N}$ [Pol02].

Further, we will sometimes treat vectors and matrices like functions and vice versa. For example for matrix $\mathbf{Q}$ with $\mathbf{Q}_{\mathbf{xy}} \in \mathbb{R}$ and function $g(\mathbf{x}) : \mathbb{N}^N \to \mathbb{R}$, the product $d(\mathbf{x}) = (\mathbf{Q} \cdot g)(\mathbf{x})$ is a function $d(\mathbf{x}) : \mathbb{N}^N \to \mathbb{R}$ with $d(\mathbf{x}) = \sum_{\mathbf{y} \in \mathbb{N}^N} \mathbf{Q}_{\mathbf{xy}} \cdot g(\mathbf{y})$.

Transformations: For several proofs, *hyper-spherical coordinate transformations* as defined in the following definition come in handy.

Definition 2: Hyper-Spherical Coordinates ▷Definition 2

For each $N \in \mathbb{N}, N > 1$, we define the bijective function

$$\psi^N : (\mathbb{R}^+ \times [0, \pi/2]^{N-1}) \cup \{\mathbf{0}\} \to {\mathbb{R}_0^+}^N$$

which maps the origin $\mathbf{0}$ to itself ($\psi^N(\mathbf{0}) = \mathbf{0}$) and assigns to a *hyper-spherical coordinate* consisting of the distance r from the origin and $N-1$ (radiant) angles $\boldsymbol{\alpha} = [\alpha_1 \ \ldots \ \alpha_{N-1}] \in [0, \pi/2]^{N-1}$ the point

$$\psi^N(r, \alpha_1, \ldots, \alpha_{N-1}) = r \cdot \begin{bmatrix} \cos(\alpha_1) \\ \sin(\alpha_1) \cdot \cos(\alpha_2) \\ \sin(\alpha_1) \cdot \sin(\alpha_2) \cdot \cos(\alpha_3) \\ \ldots \\ \sin(\alpha_1) \ldots \cos(\alpha_{N-1}) \\ \sin(\alpha_1) \ldots \sin(\alpha_{N-1}) \end{bmatrix}$$

of the non-negative quadrant of $\mathbb{R}^N$. Further, we define $\psi^1 : \mathbb{R}_0^+ \to \mathbb{R}_0^+$ with $\psi^1(r) = r$. We write ψ if N is clear from the context.

Note that using function ψ^N, each point $\mathbf{x} \in {\mathbb{R}_0^+}^N \setminus \{\mathbf{0}\}$ can be represented by exactly one combination of a distance r and angles $\boldsymbol{\alpha}$ such that $\psi^N(r, \boldsymbol{\alpha}) = \mathbf{x}$ and the origin is mapped to itself. This way, the hyper-spherical coordinate transformation can be used to re-parameterize any function on ${\mathbb{R}_0^+}^N$ for $N > 1$.

Probability Theory: For the formal treatment of the topics introduced in this thesis, a basic probability theoretic tool set is required. This tool set for example includes definitions of probability spaces, probability functions, sigma-algebrae, random variables and so on. Since there are far better text books covering these topics like for example [JP04] than what could be explained in this thesis, we assume those definitions to be given. Thus, in this paragraph, we mainly deal with the description of the related notation.

By $\mathbf{Pr}[A]$, we denote the *probability* of an event A and $\mathbf{Pr}[A \mid B]$ is

the *conditional probability* of event A conditioned on event B. A *random variable* like $\mathcal{X}$ is indicated by a capital calligraphic letter, $\mathbf{Exp}[\mathcal{X}]$ denotes its *expectation*, $\mathbf{Var}[\mathcal{X}]$ its *variance*, and $\mathbf{Std}[\mathcal{X}]$ its *standard deviation*. We write $\mathcal{X} \sim \mathrm{Exp}(\lambda)$ to state that $\mathcal{X}$ is distributed according to the (continuous negative) *exponential* distribution with rate $\lambda \in \mathbb{R}^+$. Accordingly, we will use the symbol $\mathrm{Uni}(a, b)$ with $a < b$ and $a, b \in \mathbb{R}$ to denote the (continuous) *uniform* distribution in the interval $[a, b]$. A *discrete* distribution can be expressed as a vector $\mathbf{d}$ and we write $\mathcal{X} \sim \mathbf{d}$ to describe that a random variable $\mathcal{X}$ is distributed according to distribution $\mathbf{d}$, that is, $\mathbf{Pr}[\mathcal{X} = i] = \mathbf{d}_i$. If $\mathbf{d}$ is a probability distribution over $\mathbb{N}^N$, we define the *marginal distribution* $\mathbf{d}_{j|}$ as the distribution defined by

$$\mathbf{d}_{j|i} = \sum_{\mathbf{x}_1=1}^{\infty} \cdots \sum_{\mathbf{x}_{j-1}=1}^{\infty} \sum_{\mathbf{x}_{j+1}=1}^{\infty} \cdots \sum_{\mathbf{x}_N=1}^{\infty} \mathbf{d}_{\left[\mathbf{x}_1 \ \ldots \mathbf{x}_{j-1} \ i \ \mathbf{x}_{j+1} \ \ldots \mathbf{x}_N\right]}.$$

2.2 Stochastic Processes

This thesis mainly concentrates on systems modeled as stochastic processes. Consequently, we will introduce some basic definitions which we will rely on in order to develop the further theories and approaches. For an introduction into probability theory defining concepts like random variables and expectations, we refer to [JP04].

▷Definition 3

Definition 3: Stochastic Process

A *stochastic process* is a family of random variables $\{\mathcal{X}(t)\}_{t \in T}$, defined on a common probability space $(\Omega, \mathcal{F}, \mathbf{Pr})$. We demand the existence of a set $\mathcal{S}$, such that for each $t \in T$, we have that $\mathcal{X}(t) : \Omega \to \mathcal{S}$. We restrict set $\mathcal{S}$ to be finite or countably infinite.

Index t usually denotes *time* and set $\mathcal{S}$ is called the *state space*. If $T = \mathbb{N}$, the stochastic process is referred to as a *discrete-time* stochastic process. On the other hand if $T = \mathbb{R}_0^+$, then $\mathcal{X}(t)$ is a *continuous-time* stochastic process. We will restrict to stochastic processes satisfying

the *Markov property* as stated in Equation (2.1) for all $s_0, \ldots, s_n \in \mathcal{S}$ and $t_0, \ldots, t_n \in T$ with $t_0 < \cdots < t_n$.

$$\begin{aligned} & \mathbf{Pr}\left[\mathcal{X}(t_n) = s_n \mid \mathcal{X}(t_{n-1}) = s_{n-1}, \ldots, \mathcal{X}(t_0) = s_0\right] \\ = \; & \mathbf{Pr}\left[\mathcal{X}(t_n) = s_n \mid \mathcal{X}(t_{n-1}) = s_{n-1}\right] \end{aligned} \tag{2.1}$$

Furthermore, we only consider *time-homogeneous* stochastic processes as formalized in Equation (2.2) for all $t, t' \in T$ with $t < t'$ and all $s, s' \in \mathcal{S}$.

$$\begin{aligned} & \mathbf{Pr}\left[\mathcal{X}(t') = s' \mid \mathcal{X}(t) = s\right] \\ = \; & \mathbf{Pr}\left[\mathcal{X}(t' - t) = s' \mid \mathcal{X}(0) = s\right] \end{aligned} \tag{2.2}$$

Intuitively, the possible future behavior of a process only depends on the current state and does not change over time. If these two requirements are met, we call such processes *(time-homogeneous) Markov chains*. If in addition $T = \mathbb{N}$, we abbreviate such a process by the term *discrete-time Markov chain* (DTMC), respectively if $T = \mathbb{R}_0^+$, we call the process a *continuous-time Markov chain* (CTMC). In the case of CTMCs, we further demand that they are *standard*, that is, state changes can only occur when time evolves as formally stated in Equation (2.3).

$$\left[\lim_{t \to 0} \mathbf{Pr}\left[\mathcal{X}(t) = j \mid \mathcal{X}(0) = i\right]\right]_{ij} = \mathbf{I} \tag{2.3}$$

Further, we only consider *non-exploding* CTMCs [And91, vD88]. Intuitively, if a CTMC explodes, then there is a non-zero probability of infinitely many transitions being taken in finite time. A CTMC that is standard and non-exploding is called *regular*.

The state space $\mathcal{S}$ is not necessarily $\mathbb{N}$ or a subset thereof. In fact, $\mathcal{S}$ will become multi-dimensional in most cases, that is, $\mathcal{S} \in \mathbb{N}^N$ for some dimension $N \in \mathbb{N}$. But still, we want to base our formulations on matrices and vectors for simplicity. This is where bijective *index functions* $f : \mathcal{S} \to \mathbb{N}$ as defined in Definition 1 will be used, mapping each state $s \in \mathcal{S}$ to an index $f(s)$. As mentioned, we will abuse notation and directly use s as the index and mean $f(s)$. For example, we define

the *transient (probability) distribution* of a Markov chain as the row-vector defined in Equation (2.4).

$$\pi_s(t) = \pi_{f(s)}(t) = \mathbf{Pr}\left[\mathcal{X}(t) = s\right] \tag{2.4}$$

If the limit of the transient distribution converges for $t \to \infty$ as in Equation (2.5), the resulting distribution is called *limiting distribution.* If the Markov process is ergodic [Cin75], this distribution is *unique* and *independent* of the initial distribution and is called the *steady state (probability) distribution* synonymous with the *equilibrium distribution.*

$$\pi_s = \lim_{t\to\infty} \mathbf{Pr}\left[\mathcal{X}(t) = s\right] \tag{2.5}$$

Precise conditions for ergodicity will be presented later on. In order to be able to describe those conditions, we need to define that two states i and j *communicate* written $i \leftrightarrow j$ iff

$$\exists t, t' \in T. \mathbf{Pr}\left[\mathcal{X}(t) = j \mid \mathcal{X}(0) = i\right] > 0 \wedge \mathbf{Pr}\left[\mathcal{X}(t') = i \mid \mathcal{X}(0) = j\right] > 0. \tag{2.6}$$

Further, a set $\mathcal{A} \subseteq \mathcal{S}$ is a *communicating class* if

$$\forall i, j \in \mathcal{A}. i \leftrightarrow j \wedge \nexists k \notin \mathcal{A}. i \leftrightarrow k. \tag{2.7}$$

Finally, a Markov chain $\mathcal{X}$ with state space $\mathcal{S}$ is *irreducible* if $\mathcal{S}$ is a single communication class. We want to emphasize that we enforce a strict definition of the state space $\mathcal{S}$. More precisely, $\mathcal{S}$ is defined such that all states $s \in \mathcal{S}$ are *reachable* from the initial distribution as formalized in Equation (2.8).

$$\forall s \in \mathcal{S}.\ \exists t \in T.\ \pi_s(t) > 0 \tag{2.8}$$

Further, given a set $\mathcal{A} \subseteq \mathcal{S}$, we define the probability $\mathbf{reach}_{\mathcal{A}}$ to *reach* some state in $\mathcal{A}$ starting from the initial distribution as

$$\mathbf{reach}_{\mathcal{A}} = \mathbf{Pr}\left[\mathcal{X}(t) \in \mathcal{A} \text{ for some } t \geq 0\right]. \tag{2.9}$$

2.2.1 Discrete-Time Markov Chains

To sum up, we can fully describe a discrete-time Markov chain as in Definition 4.

Definition 4: DTMC (Tuple Representation) ▷Definition 4

A *discrete-time Markov chain* (DTMC) is fully determined by a tuple

$$(\mathcal{S}, \mathbf{P}, \pi(0))$$

where the finite or countably infinite set $\mathcal{S}$ is the *state space*, $\mathbf{P}$ is the *transition matrix*, and $\pi(0)$ is the *initial distribution*.

The transition matrix $\mathbf{P}$ captures all one step transition probabilities as described in Equation (2.10).

$$\mathbf{P}_{ij} = \mathbf{Pr}\left[\mathcal{X}(1) = j \mid \mathcal{X}(0) = i\right] \tag{2.10}$$

Due to the *law of total probability*, the transient distribution $\pi(t)$ at time point t of a DTMC $(\mathcal{S}, \mathbf{P}, \pi(0))$ can therefore be formulated as in Equation (2.11).

$$\pi(t) = \pi(0) \cdot \mathbf{P}^t \tag{2.11}$$

Equivalently, the transient distribution can be stated using a recursive definition starting with a given $\pi(0)$ as the base case and the recursive step in Equation (2.12).

$$\pi(t+1) = \pi(t) \cdot \mathbf{P} \tag{2.12}$$

For an ergodic DTMC, the *equilibrium* or *steady state* distribution π is characterized by the equation system stated in Equation (2.13).

$$\pi = \pi \cdot \mathbf{P}, \pi \cdot \mathbf{e} = 1 \tag{2.13}$$

An *absorbing state* of a DTMC $(\mathcal{S}, \mathbf{P}, \pi(0))$ is a state $s \in \mathcal{S}$ that can not be left, that is, $\mathbf{P}_{ss} = 1$.

2.2.2 Continuous-Time Markov Chains

An analogous tuple-based definition can be derived for continuous-time Markov chains as well. Note that since we restrict to regular CTMCs, the behavior of a CTMC is uniquely determined by its infinitesimal generator matrix.

▷Definition 5

Definition 5: CTMC

A *continuous-time Markov chain* (CTMC) is fully determined by a tuple

$$(\mathcal{S}, \mathbf{Q}, \pi(0))$$

where the finite or countable set $\mathcal{S}$ is the *state space*, matrix $\mathbf{Q}$ is the *infinitesimal generator*, and $\pi(0)$ is the *initial distribution*.

The infinitesimal generator $\mathbf{Q}$ defines the time derivatives of the transition probabilities as defined in Equation (2.14).

$$\mathbf{Q}_{ij} = \begin{cases} \lim_{h\to 0} \frac{\mathbf{Pr}[\mathcal{X}(h)=j \mid \mathcal{X}(0)=i] - \mathbf{I}_{ij}}{h} & \text{if } i \neq j, \text{and} \\ -\sum_{k\neq i} \mathbf{Q}_{ik} & \text{otherwise.} \end{cases} \tag{2.14}$$

We call the negated diagonal rates $-\mathbf{Q}_{ii}$ *exit rates* since they amount for the total rate to take any transition. The Chapman-Kolmogorov Equation (2.15) [Kol31] relates the development of the transient distribution in time and the infinitesimal generator via a set of ordinary differential equations

$$\frac{d}{dt}\pi(t) = \pi(t) \cdot \mathbf{Q} \tag{2.15}$$

which has the general solution

$$\pi(t) = \pi(0) \cdot e^{\mathbf{Q}\cdot t}. \tag{2.16}$$

if the state space is finite. For an ergodic CTMC, the equilibrium distribution exists and is characterized by the equation system stated in Equation (2.17).

$$\pi \cdot \mathbf{Q} = 0, \pi \cdot \mathbf{e} = 1 \tag{2.17}$$

Given a CTMC $(\mathcal{S}, \mathbf{Q}, \pi(0))$, an *absorbing state* $s \in \mathcal{S}$ is a state that can not be left, that is, $\mathbf{Q}_{ss} = 0$.

2.2.3 Embedding of Continuous-Time Markov Chains

In certain cases, we are not interested in the time spent in each state, although we analyze a continuous-time stochastic process $\mathcal{X}(t)$. More precisely, we observe the system at the *jump time points*, $t_0, t_1, \cdots \in \mathbb{R}_0^+$ with $t_0 < t_1 < \ldots$ where the current state is left for a successor state. We define the resulting process as $Y(n) = X(t_n)$ for all $n \in \mathbb{N}$ as long as state $X(t_n)$ can be left and $Y(n') = X(t_n)$ for all $n' > n$ if $X(t_n)$ can not be left. The resulting process $\{Y(n)\}_{n\in\mathbb{N}}$ is a *discrete-time* stochastic process. In the case of CTMCs, the resulting process is a DTMC as defined by the following matrix.

Definition 6: Embedded Matrix ▷**Definition 6**

Let $\mathbf{Q}$ be the infinitesimal generator of a CTMC. We define the corresponding *embedded matrix* $\mathbf{E}$ via

$$\mathbf{E}_{ij} = \begin{cases} -\mathbf{Q}_{ij} \cdot \mathbf{Q}_{ii}^{-1} & \text{if } i \neq j \wedge \mathbf{Q}_{ii} \neq 0 \\ 0 & \text{if } i = j \wedge \mathbf{Q}_{ii} \neq 0 \\ 0 & \text{if } i \neq j \wedge \mathbf{Q}_{ii} = 0 \\ 1 & \text{if } i = j \wedge \mathbf{Q}_{ii} = 0. \end{cases}$$

▷Definition 7

Definition 7: Embedded DTMC

Given a CTMC $(\mathcal{S}, \mathbf{Q}, \pi(0))$ we define the corresponding *embedded DTMC* as the DTMC $(\mathcal{S}, \mathbf{E}, \pi(0))$ where $\mathbf{E}$ is the embedded matrix of $\mathbf{Q}$.

2.2.4 Paths of Continuous-Time Markov Chains

When we deal with formal methods like model checking described in Chapters 4 and 6, we will need the notion of *paths* in continuous-time Markov chains.

▷Definition 8

Definition 8: CTMC Paths

Given a CTMC $(\mathcal{S}, \mathbf{Q}, \pi(0))$, we define the set of *finite paths* as $Path^* \subseteq \bigcup_{n\in\mathbb{N}} \mathcal{S} \times (\mathbb{R}_0^+ \times \mathcal{S})^n$, where

$$(s_0, t_0, s_1, \ldots, t_{n-1}, s_n) \in Path^*$$

implies that $\pi_{s_0}(0) > 0$, $\mathbf{Q}_{s_n s_n} = 0$, and for all $0 \leq i < n$ we have $\mathbf{Q}_{s_i s_{i+1}} > 0$. We extend this definition to the set of *infinite paths* $Path^\omega \subseteq (\mathcal{S} \times \mathbb{R}_0^+)^\omega$, where we require $\pi_{s_0}(0) > 0$ and $\mathbf{Q}_{s_i s_{i+1}} > 0$ for all $i \geq 0$. Let $Path = Path^* \cup Path^\omega$ be the set of all paths and $Path(s)$ the set of those paths that start in state s.

In order to access specific states and times of a path, we define the following path accessors.

▷Definition 9

Definition 9: CTMC Path Accessors

For the path $\sigma = (s_0, t_0, s_1, \ldots) \in Path$ and $i \in \mathbb{N}$, let $\sigma[i] = s_i$ denote the i-th state, and let $\delta(\sigma, i) = t_i$ denote the time spent in state s_i. For $t \in \mathbb{R}_0^+$, let $\sigma@t$ denote $\sigma[i]$ such that i is the smallest index with $t \leq \sum_{j=0}^{i} t_j$.

We note that a probability measure $\mathbf{Pr}_s$ on measurable subsets $\mathcal{A} \subseteq Path$ is uniquely defined [BHHK03] with $\mathbf{Pr}_s[\mathcal{A}] = \mathbf{Pr}[\mathcal{A} \mid \mathcal{X}(0) = s]$.

2.3 Markov Population Models

Subject to examination in this thesis are mainly systems which describe the behavior of several different *population types*. But instead of modeling each individual of each population type on its own, we make use of a higher level view where we aggregate all individuals of a type to a counter expressing their number. Formally, we distinguish between N population types and assume that the number of individuals can be represented by non-negative integers from $\mathbb{N}$. The corresponding state space therefore resembles a subset of the N-dimensional vector space of the non-negative integers, that is, $\mathcal{S} \subseteq \mathbb{N}^N$. We call such a systems a *Markov population model* (MPM) if its underlying behavior is *Markov* and *homogeneous* in *continuous time*.

Definition 10: MPM ▷**Definition 10**

A *Markov population model* (MPM) for $N \in \mathbb{N}$ population types is a CTMC $(\mathcal{S}, \mathbf{Q}, \pi(0))$ with $\mathcal{S} \subseteq \mathbb{N}^N$.

2.3.1 Transition Classes

Since the number of states $|\mathcal{S}|$ might be countably infinite, computing all entries of $\mathbf{Q}$ is infeasible. What we propose instead is to induce the infinitesimal generator by a compact symbolic representation in the form of a finite set of change vectors with state-dependent functions that determine the transition rate [BKPR06, Eng08, Gra91]. In this thesis, we call those pairs *transition classes* [HS07, HJW09].

▷Definition 11

Definition 11: Transition Class

A *transition class* (TC) for $N \in \mathbb{N}$ population types is a tuple $(\alpha, \mathbf{v})$, where $\alpha : \mathbb{N}^N \to \mathbb{R}_0^+$ is the *propensity function* and $\mathbf{v} \in \mathbb{Z}^N \setminus \{\mathbf{0}\}$ is the *change vector*.

Intuitively, a transition class $(\alpha, \mathbf{v})$ encodes a possible transition type from any state $\mathbf{x} \in \mathcal{S}$ to state $\mathbf{x} + \mathbf{v}$ with rate $\alpha(\mathbf{x})$ if $\alpha(\mathbf{x}) \neq 0$. This way, given a set of $R \in \mathbb{N}$ transition classes, the corresponding generator matrix can have a maximum of R outgoing transitions per state.

▷Definition 12

Definition 12: TC Induced MPM

A set $\{(\alpha^{(r)}, \mathbf{v}^{(r)}\}_{1 \leq r \leq R}$ of $R \in \mathbb{N}$ transition classes for N population types determines the entries of the infinitesimal generator of an MPM $(\mathcal{S}, \mathbf{Q}, \pi(0))$ with $\mathcal{S} \subseteq \mathbb{N}^N$ via

$$\mathbf{Q}_{\mathbf{xy}} = \begin{cases} \sum_{\{r \mid \mathbf{x}+\mathbf{v}^{(r)}=\mathbf{y}\}} \alpha^{(r)}(\mathbf{x}) & \text{if } \mathbf{x} \neq \mathbf{y} \text{ and} \\ -\sum_{\mathbf{z} \neq \mathbf{x}} \mathbf{Q}_{\mathbf{xz}} & \text{otherwise,} \end{cases}$$

where we demand that the resulting $\mathbf{Q}$ uniquely defines a regular CTMC.

Let the infinitesimal generator $\mathbf{Q}$ of an MPM be induced by a finite number of transition classes. It might be the case that $\sup_i |\mathbf{Q}_{ii}| = \infty$ and the number of reachable states from a given initial state is infinite. But still, $\mathbf{Q}$ has only a finite number of nonzero entries in each row and in each column and is therefore a *row-/column-finite (infinite) matrix*.

Theorem 1: Row-/Column-Finiteness ▷Theorem 1

The generator matrix of an MPM induced by a finite set of transition classes is row-/column-finite.

Proof: Let an MPM with state space $\mathcal{S}$ be described by R transition classes with change vectors $\mathbf{v}^{(1)}, \ldots, \mathbf{v}^{(R)}$. Then, each state $\mathbf{x} \in \mathcal{S}$ has at most R transitions to states $\mathbf{x} + \mathbf{v}^{(1)}, \ldots, \mathbf{x} + \mathbf{v}^{(R)}$, which limits the number of non-zero entries of $\mathbf{Q}$ per row by R. But also state $\mathbf{x}$ can only be reached by a maximum of R states $\mathbf{x} - \mathbf{v}^{(1)}, \ldots, \mathbf{x} - \mathbf{v}^{(R)}$ which limits the number of non-zero entries per column also by R. □

Obviously, if $\mathbf{Q}$ is row-/column-finite, also its embedded matrix $\mathbf{E}$ as defined in Definition 6 is *well-defined* and row-/column-finite. A small model from the theory of queuing networks shall provide an intuitive example for the usage of transition classes.

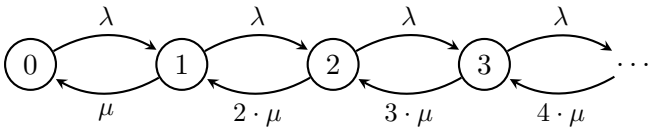


Figure 2.1: Graphical representation of an M/M/∞ queue.

Example 1: TC Description of an M/M/∞ Queue ▷Example 1

An *M/M/∞* queue [HP92] describes a queuing network which consists of an unbounded number of servers where incoming packets are immediately processed. The arrival and service processes are Markovian with exponential rates λ (arrival) and μ (service). The system is modeled by a CTMC with state space $\mathcal{S} = \mathbb{N}$, where each state $\mathbf{x} \in \mathcal{S}$ encodes the number $\mathbf{x}$ of packets in the buffer. Note that since we only model a single queue, the state space is one-dimensional. A graphical representation of the CTMC is de-

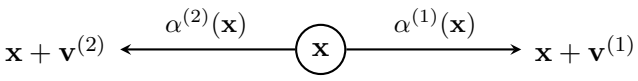


Figure 2.2: Graphical representation of the transition classes of an M/M/∞ queue.

picted in Figure 2.1. The corresponding generator matrix $\mathbf{Q}$ has the diagonal band structure

$$\mathbf{Q} = \begin{bmatrix} -\lambda & \lambda & & & & \mathbf{0} \\ \mu & -(\lambda+\mu) & \lambda & & & \\ & 2\cdot\mu & -(\lambda+2\cdot\mu) & \lambda & & \\ & & 3\cdot\mu & -(\lambda+3\cdot\mu) & \lambda & \\ \mathbf{0} & & & \ddots & \ddots & \ddots \end{bmatrix}.$$

The same generator matrix can be induced by using only two transition classes. The first transition class $(\alpha^{(1)}, \mathbf{v}^{(1)})$ with $\alpha^{(1)}(\mathbf{x}) = \lambda$ and $\mathbf{v}^{(1)} = 1$ describes the arrival process and the second transition class $(\alpha^{(2)}, \mathbf{v}^{(2)})$ with $\alpha^{(2)}(\mathbf{x}) = \mathbf{x} \cdot \mu$ and $\mathbf{v}^{(2)} = -1$ encodes the service process. A symbolic illustration of the two transition classes is given in Figure 2.2.

Note that the state space $\mathcal{S}$ of an MPM induced by a set of transition classes is not necessarily unique and depends on the initial distribution as illustrated by the following example.

▷Example 2

Example 2: State Space of TC Induced MPMs

A single transition class $(\alpha, \mathbf{v})$ with $\alpha(\mathbf{x}) = 1$ for all $\mathbf{x} \in \mathbb{N}$ and $\mathbf{v} = 2$ and initial distribution $\pi_0(0) = 1$ induces an MPM with state space $\mathcal{S} = \{2 \cdot n \mid n \in \mathbb{N}\}$ where as for $\pi_1(0) = 1$ we would have state space $\mathcal{S} = \{2 \cdot n + 1 \mid n \in \mathbb{N}\}$.

Consequently, if the state space of an MPM is not trivially $\mathcal{S} = \mathbb{N}^N$ for some $N \in \mathbb{N}$, we will either specify the initial distribution or directly constrain the state space to make the model unique.

2.3.2 Chemical Reaction Networks

The majority of models in this thesis are motivated by the field of systems biology, where biological processes are analyzed on the level of *chemical reaction networks*. More precisely, molecules of different *chemical species* react with each other and are finally transformed into other chemical species. Each possible transformation is described by a chemical *reaction* happening at a certain *reaction rate*, which determines the likelihood and speed of the respective reaction.

Definition 13: Chemical Reaction Network ▷Definition 13

A *chemical reaction network (CRN)* involving N different chemical species $C_1, \dots, C_N$ is a system of R different *reactions*. Each reaction is of the form

$$\mathbf{u}_1^{(r)} \cdot C_1 + \cdots + \mathbf{u}_N^{(r)} \cdot C_N \xrightarrow{c_r} \mathbf{w}_1^{(r)} \cdot C_1 + \cdots + \mathbf{w}_N^{(r)} \cdot C_N,$$

for $1 \leq r \leq R$ and where

$$\mathbf{u}^{(r)} = \left[\mathbf{u}_1^{(r)}, \dots, \mathbf{u}_N^{(r)}\right] \in \mathbb{N}^N$$

and

$$\mathbf{w}^{(r)} = \left[\mathbf{w}_1^{(r)}, \dots, \mathbf{w}_N^{(r)}\right] \in \mathbb{N}^N$$

are the *stoichiometric coefficients* and $c_r \in \mathbb{R}_0^+$ is the *reaction constant*.

We use the symbol $\varnothing$ to explicitly state that there are no reactants ($\mathbf{u}^{(r)} = \mathbf{0}$) or products ($\mathbf{w}^{(r)} = \mathbf{0}$), respectively. Also, we do not number the chemical reactions of a CRN explicitly but rather assume an implicit numbering of the reactions in the order of appearance. Further, in most cases, we will use more intuitive descriptors than just $C_1, \dots, C_N$ to

denote the respective chemical species. Those descriptors will serve as synonyms for the species and their index. In the model description, we will indicate the species index in parentheses. For example, we might write $A(1)$ and $B(2)$ to introduce two chemical species A and B, where A is the species with index 1 and B is the species with index 2. This will allow us later to associate the components of a state vector in an MPM with the respective chemical species and vice versa.

▷Model 1

Model 1: Gene Expression

In every living organism, the complete building plan is stored in the form of DNA. The information on the DNA is divided into *genes* which mainly encode the proteins that are (potentially) used during the lifetime of an organism. Genes are *transcribed* into *messenger RNA* (mRNA), an intermediate encoding, before they are finally *translated* into the respective proteins. We can model the transcription/translation process of a single protein via a small CRN consisting of two species, mRNA $M(1)$ and the protein species $P(2)$. The corresponding chemical reactions are

$$\begin{aligned} \varnothing &\xrightarrow{\rho} M \\ M &\xrightarrow{\tau} M + P \\ M &\xrightarrow{\delta_M} \varnothing \\ P &\xrightarrow{\delta_P} \varnothing \end{aligned}$$

where the first reaction describes the transcription of a gene into mRNA at a constant rate ρ. The second reaction describes the translation process of mRNA into proteins with reaction constant τ. The remaining reactions are related to the degradation of mRNA and the proteins at reaction constants δ_M and δ_P, respectively. Figure 2.3 illustrates that model.

An example of a chemical reaction network is given in Model 1. Approximations of the behavior of chemical reaction networks by interpreting

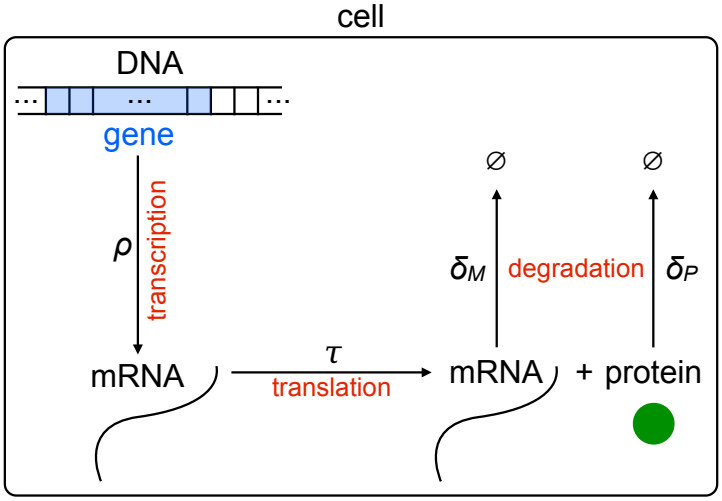


Figure 2.3: Illustration of the gene expression model.

the originally discrete state space as a continuum and approximating population counts solely by their respective expectation are widely used. The resulting system of ordinary differential equations resemble the *law of mass action* as known in chemistry. Recent discoveries however suggest that for certain models, a stochastic treatment on the molecule level is more appropriate. We refer to Chapter 7 for a detailed discussion. Then, as it has been shown in the seminal work of Gillespie et al. [Gil92], the underlying process semantics is best characterized by a continuous-time Markov chain. We refer to [Kur72] for an investigation of the relationship between the two views. The quintessence is that if at least one population count may become small, the effects of intrinsic noise can not be neglected, hence continuous approximations become inaccurate [HMMW10], and a stochastic treatment is more appropriate. This phenomenon has been observed very often in the case of chemical reaction networks [TvO01, SES02, Pau04].

The use of Markov population models in the stochastic setting allows for an intuitive way of modeling those systems. More precisely, there is a one-to-one relationship between chemical species and MPM population types as well as reactions and MPM transitions. Formally, this

relationship is defined as in the following.

▷Definition 14

Definition 14: CRN Induced TCs

Let a CRN with N chemical species and R reactions be given. Every reaction r induces a transition class $(\alpha^{(r)}, \mathbf{v}^{(r)})$ for the MPM $(S, \mathbf{Q}, \pi(0))$ with $S \subseteq \mathbb{N}^N$ as defined in Equations (2.18) and (2.19).

$$\alpha^{(r)}(\mathbf{x}_1, \ldots, \mathbf{x}_N) = c_r \cdot \prod_{i=1}^{N} \frac{\mathbf{x}_i!}{\mathbf{u}_i^{(r)}! \cdot (\mathbf{x}_i - \mathbf{u}_i^{(r)})!} \tag{2.18}$$

$$\mathbf{v}^{(r)} = \mathbf{w}^{(r)} - \mathbf{u}^{(r)} \tag{2.19}$$

Note that the propensity functions are polynomial functions, that is, $\alpha^{(r)} \in \mathbb{P}[\mathbf{x}]$ since the factorials reduce to polynomials in the end.

Also note that models with more general propensity functions exist. However, most of the time, we restrict to the form in Equation (2.18). The following example shall provide an intuitive understanding about the relationship between chemical reaction networks and Markov population models.

▷Example 3

Example 3: TCs for the Gene Expression Model

The state space of the gene expression model as defined in Model 1 is $\mathcal{S} = \mathbb{N}^2$ and the four transition classes are

$$\alpha^{(1)}(\mathbf{x}) = \rho, \qquad \mathbf{v}^{(1)} = \begin{bmatrix} 1 & 0 \end{bmatrix},$$

$$\alpha^{(2)}(\mathbf{x}) = \tau \cdot \mathbf{x}_M, \qquad \mathbf{v}^{(2)} = \begin{bmatrix} 0 & 1 \end{bmatrix},$$

$$\alpha^{(3)}(\mathbf{x}) = \delta_M \cdot \mathbf{x}_M, \qquad \mathbf{v}^{(3)} = \begin{bmatrix} -1 & 0 \end{bmatrix},$$

$$\alpha^{(4)}(\mathbf{x}) = \delta_P \cdot \mathbf{x}_P, \qquad \mathbf{v}^{(4)} = \begin{bmatrix} 0 & -1 \end{bmatrix}.$$

We would like to exemplify the different forms of propensity functions that usually appear when dealing with CRN in the following.

Example 4: Types of Reactions ▷**Example 4**

Any reaction

$$\varnothing \xrightarrow{c_1} \dots$$

with no reactant molecules has a constant propensity function $\alpha(\mathbf{x}) = c_1$. The propensity function corresponding to a mono-molecular reaction of the form

$$A \xrightarrow{c_2} \dots$$

has the form $\alpha(\mathbf{x}) = c_2 \cdot \mathbf{x}_A$ since each of the $\mathbf{x}_A$ molecules of type A can react at rate c_2. Bi-molecular reactions

$$A + B \xrightarrow{c_3} \dots$$

have propensity functions $\alpha(\mathbf{x}) = c_3 \cdot \mathbf{x}_A \cdot \mathbf{x}_B$ accounting for each of the $\mathbf{x}_A \cdot \mathbf{x}_B$ combinations of A and B molecules to react at rate c_3. But also two molecules of the same type can react. Such a process is called *dimerization*, where two reactants P form a dimer D as in

$$2 \cdot P \xrightarrow{c_4} D$$

whose propensity function has the form $\alpha(\mathbf{x}) = 0.5 \cdot c_3 \cdot \mathbf{x}_P \cdot (\mathbf{x}_P - 1)$, where each of the $\frac{\mathbf{x}_P \cdot (\mathbf{x}_P - 1)}{2}$ combinations of two different P molecules can react at rate c_4. More than two reactant molecules

are rare and usually, those reactions can be split into a combination of these elementary reaction types.

2.4 Transient Analysis for Markov Population Models

Although this thesis focuses primarily on equilibrium analysis for Markov population models, various approaches will rely on an approximation of the transient distribution of an MPM as well. Since MPMs are a sub-class of CTMCs, this computation boils down to efficiently solving Equation (2.15). In the following, we would like to give a small overview of the respective standard methods in the literature.

2.4.1 Stochastic Simulation

The *stochastic simulation algorithm* (SSA) [Gil76] also widely known as *Gillespie simulation* (direct method) is stated in Algorithm 1 and uses pseudo-random numbers to compute a sample trajectory according to the model's underlying stochastic process semantics. Keeping track of the current time t and state $\mathbf{x}$ over multiple simulation runs allows for the generation of statistics about the events of interest. For example, if interested in the transient probability $\pi_{\mathbf{x}'}(t')$ to be in state $\mathbf{x}'$ at time t' and a total of n out of N simulation runs until time t' would end in state $\mathbf{x}'$, one would estimate $\pi_{\mathbf{x}'}(t') \approx \frac{n}{N}$. One could also estimate the corresponding variance in order to compute confidence intervals for those estimates and further statistics [LK00].

Algorithm 1 SSA($\{(\alpha^{(r)}, \mathbf{v}^{(r)})\}_{1 \leq r \leq R}, \pi(0), T$)

```
1: t ← 0
2: sample x ~ π(0)
3: while t < T do
4:     α_0 ← Σ_{r=1}^{R} α^(r)(x)
5:     if α_0 = 0 then
6:         t ← T
7:     else
8:         sample τ ~ Exp(−α_0)
9:         sample j ~ [α^(r)(x) · α_0^{−1}]_r
10:        x ← x + v^(j)
11:        t ← t + τ
12:    end if
13: end while
```

In general, stochastic simulation based on Monte-Carlo simulation is often used in order to derive quantities based on the transient or steady state probability distribution [ROB05].

Several techniques like *explicit* [Gil01] and *implicit* [RPCG03] τ-*leaping* have been developed to speed up the simulation time by not advancing the simulation step by step but by larger time intervals τ in which several transitions might occur. But despite those improvements, the use of these simulation methods is still limited in the case when the full transient or steady state probability distribution shall be approximated. The reason is that a large number of simulation runs is needed to estimate the individual probabilities at a sufficient precision [LK00]. This problem becomes worse when interested in the estimation of small probabilities since the corresponding events are rarely encountered and thus increase the number of needed simulation runs even further. Although, techniques like *rare-event* simulation try to combat that problem, they are usually not directly usable without model-specific adaptations. For example, when using an approach based on *importance sampling* [Jun06], an appropriate *change of measure* is needed, that is, the transition rates of the system have to be altered in order to make the rare event more likely. This is complicated without prior knowledge about the system. One of the biggest challenges when using standard

simulation for approximating the steady state, is to decide when the stationary distribution is reached, that is, when the bias introduced by the choice of the initial state vanishes. Remedy comes in the form of methods like *perfect simulation* based on a technique called *coupling from the past* [PW96], where the goal is to directly generate samples of the steady state distribution, provided that structural properties like *monotonicity* [PW96] hold.

2.4.2 Uniformization

Assuming an appropriate precision, numerical solution techniques circumvent the problem of rare events by computing the full transient probability distribution directly exploiting the Chapman-Kolmogorov Equation (2.15). Intuitively, this can be seen as considering all possible simulation runs at once. While there exists a wide variety of methods like computing [ML78] the matrix exponential of Equation (2.16) in the case of a finite state space, a prominent and very efficient technique called *uniformization* [Gra77a, Gra77b] usually gives better results [Ste94]. The main idea is that the underlying CTMC is decomposed into a Poisson process governing the progress of time and a DTMC governing the distribution of the probability mass among the states.

▷**Definition 15**

Definition 15: Uniformized CTMC

Given a CTMC $(\mathcal{S}, \mathbf{Q}, \pi(0))$ and a *uniformization rate*

$$u \geq \sup_{\mathbf{x} \in \mathcal{S}} -\mathbf{Q}_{\mathbf{xx}} \neq \infty,$$

the *uniformized CTMC* (UCTMC) is defined as the DTMC

$$(\mathcal{S}, \mathbf{P}, \pi(0))$$

with

$$\mathbf{P} = \mathbf{I} + u^{-1} \cdot \mathbf{Q}.$$

Using the identity $\mathbf{Q} = u \cdot (\mathbf{P} - \mathbf{I})$ and the general solution of the Chapman-Kolmogorov Equation (2.16), one can derive

$$\pi(t) = \sum_{i=0}^{\infty} \pi(0) \cdot \mathbf{P}^i \cdot e^{-u \cdot t} \cdot \frac{(u \cdot t)^i}{i!}. \tag{2.20}$$

The first part of the sum's inner term can be computed using the recursion

$$\pi(0) \cdot \mathbf{P}^i = \mathbf{p}^{(i)} \text{ with } \mathbf{p}^{(0)} = \pi(0), \mathbf{p}^{(k)} = \mathbf{p}^{(k-1)} \cdot \mathbf{P}$$

resembling the transient solution of the uniformized CTMC as in Equation (2.12). Application of the *Fox-Glynn* algorithm [FG88] finally gives lower and upper bounds for the infinite sum's index i such that $\pi(t)$ can be approximated up to an a priori chosen error ϵ. Further, the algorithm also efficiently computes the Poisson probabilities in the second part of the sum's inner term.

In fact, an infinite state space poses a serious problem for most standard numerical solution algorithms, since the full probability distribution can not be stored explicitly in memory. Note that techniques based on simulation do not suffer from this problem since only the current state and the relevant statistics need to kept in memory. We would like to illustrate an example where indeed an infinite amount of memory would be needed.

Example 5: Poisson Process ▷**Example 5**

Assume we are given a CRN with a single chemical species A and the only reaction

$$\varnothing \xrightarrow{\lambda} A$$

for $\lambda > 0$. The induced TC is given by $\alpha(\mathbf{x}) = \lambda$ and $\mathbf{v} = 1$ on state space $\mathcal{S} = \mathbb{N}$. The corresponding generator matrix $\mathbf{Q}$ has a simple diagonal band structure $\mathbf{Q}_{\mathbf{xy}} = \lambda$ if $\mathbf{y} = \mathbf{x}+1$, $\mathbf{Q}_{\mathbf{xy}} = -\lambda$ if $\mathbf{x} = \mathbf{y}$ and 0 otherwise. The resulting process is a *Poisson* process with rate λ and assuming $\pi_0(0) = 1$, the corresponding analytical

transient solution is

$$\pi_{\mathbf{x}}(t) = \frac{(\lambda \cdot t)^{\mathbf{x}}}{\mathbf{x}!} \cdot e^{-(\lambda \cdot t)}.$$

Obviously, for all $\mathbf{x} \in \mathbb{N}$ we have $\pi_t(\mathbf{x}) > 0$ for any $t > 0$, that is, intuitively the probability mass after a non-zero amount of time is spread over a (countably) infinite amount of states.

As we have seen in Example 5, approximating and storing the transient probabilities for each state is not possible. One could argue that a Poisson process has this problem since it is not ergodic, but also an ergodic M/M/1 queue with a service rate greater than its arrival rate, shows the same difficulties when examining its analytical solution as for example exercised in [AW87]. Further, only for very simple models derived from CRNs [JH07], that is, models where only reactions are involved with $\sum_{i=1}^{d} \mathbf{u}_i^{(r)} \leq 1$, an analytical solution can be derived easily. Consequently, the transient probability distribution needs to be approximated.

2.4.3 Numerical Methods Based on Dynamic State Space Truncation

A remedy for the problem of infinite storage needs is to *truncate* the state space dynamically. The seminal idea as introduced by Munsky and Mhammash in 2006 [MK06] and in Munsky's PhD thesis [Mun08] is to project the infinite state space onto a finite subset and an additional single absorbing state corresponding to the rest. The probability mass absorbed in that state can be used to compute bounds on statistical quantities of the full system by only taking the behavior inside the projected state space into account.

Follow-up approaches artificially truncate the support of the transient probability distribution $\pi(t)$ dynamically after each numerical transient analysis iteration step to keep only *significant* states, that is, states s with transient probability mass $\pi_s(t) > \delta$ for a chosen *truncation threshold* $\delta > 0$. The basic idea of keeping track of the major part of the transient probability mass using a finite *sliding window* possessing specific

geometric shapes (like rectangles, cubes, and hypercubes) in a multi-dimensional state space was developed in [HMW09]. Further improvements like an *adaptive uniformization* scheme taking into account that the uniformization rate might change over time depending on the current set of significant states, were introduced in [DHMW09]. A tool that includes sophisticated implementations of truncation based approaches is SHAVE [LMW11], which is under active development at the chair of modeling and simulation at Saarland University. Still, a problem of basic truncation-based approaches is that despite the infinite state space has been reduced to a finite subset via truncation, for many MPMs, the significant part of the state space grows exponentially with the number of population types. In addition, if the expected population counts grow large, the state space significally grows as well. In order to overcome that issue, the work [HMMW10] proposes to approximate populations by their expectations and covariances if their population counts grow above a given threshold. The resulting system becomes a *hybrid system* where the state space contains a discrete stochastic part (population types with a population count below the threshold) and a continuous deterministic part (population types with a population count above the threshold). In each time step, the populations that are treated discrete stochastically evolve according to the MPM semantics given in this chapter. The deterministic species however are each aggregated by their mean, a single scalar value and optionally higher moments, and thus evolve deterministically according to a reduced number of ordinary differential equations. The SHAVE tool features this technique as well. Note that in this thesis, whenever we need to solve a system transiently, we will assume that the significant part of the state space, that is, the set of states with a transient probability mass greater than a given threshold, can be kept in memory and that thus we do not need to use hybrid systems as an approximation.

In later chapters, we will rely on truncation based transient analysis as stated in Algorithm 2 inspired by the mentioned work. The algorithm takes

- the initial distribution $\pi(0)$,
- a description of the infinitesimal generator $\mathbf{Q}$,
- the time point of interest t,

- and a function *advance* that implements a single integration step

as its input and returns p, an approximation of $\pi(t)$, and the total approximation error e containing the probability mass of all states that have been deleted along the way (cf. line 8). Note that the creation of the hash map p via argument $\mathcal{S}$, the state space, is only meant to allow the hash map initialization routine to infer the type of the keys. Consequently, a reachability analysis to determine the full set $\mathcal{S}$ is not needed.

Algorithm 2 transient($\pi(0), \mathbf{Q}$,$\mathcal{S}$, t,δ,advance)

```
1: t' ← 0; e ← 0
2: p ← new hash map S → [0, 1]
3: ∀s with π_s(0) > 0: p(s) ← π_s(0)
4: while t' < t do
5:     choose appropriate h
6:     Δt ← min(t − t', h)
7:     p ← advance(p, Q, Δt)
8:     ∀s with p(s) < δ: e ← e + p(s) and remove(p, s)
9:     t' ← t' + min(t − t', h)
10: end while
11: return [p, e]
```

Algorithm 2 propagates the probability mass currently stored in hash map p for h time units as long as the desired time point t has not been reached. After each propagation, all insignificant states s with $p(s) < \delta$ are deleted and the total error, that is, the truncated probability mass is accumulated in e. Note that for a reasonable choice of δ =1e-20, the total error e remains negligible small (in the order of 1e-10 or less) for many systems. We refer to [DHMW09, MNSW13] for more detailed discussions.

In the following Algorithm 3, the flow $\pi(t) \cdot \mathbf{Q}$ as defined in Equation (2.15) is approximated for all states in dom(p).

Algorithm 3 flow$(p, \mathbf{Q}, \mathcal{S})$

1: $p' \leftarrow$ new hash map $\mathcal{S} \to \mathbb{R}$
2: **for** $s \in \mathrm{dom}(p)$ **do**
3: **for** s' with $\mathbf{Q}_{ss'} \neq 0$ **do**
4: $p'(s') \leftarrow p'(s') + p(s) \cdot \mathbf{Q}_{ss'}$
5: **end for**
6: **end for**
7: **return** p'

The *Euler method* as stated in Algorithm 4, following the flow evaluated at the current approximation of the transient probability mass for h time units can be used as a simple integration method supplied to Algorithm 2.

Algorithm 4 euler$(p, \mathbf{Q}, h)$

1: **return** $p + h \cdot \mathrm{flow}(p, \mathbf{Q})$

A more advanced integration scheme is the *Fourth-Order Runge-Kutta* (RK4) method which is a standard method used to numerically integrate systems of ordinary differential equations and provides a superior numerical precision compared to many approaches including the Euler method. For a general overview of integration methods, we refer to [Atk89]. The usage of RK4 as the numerical integration scheme of choice is mainly motivated pragmatically. It is usually considered as very stable for small time steps and easy to implement in the proposed truncation based framework. As mentioned previously, numerical integration based on uniformization would also have been a good choice. However, sophisticated heuristics for the uniformization rate would have been needed in advance which becomes non-trivial for infinite state-spaces. Moreover, since the set of significant states and thus the maximum exit rate might change over time, an adaptive uniformization scheme varying the uniformization rate over time would be needed to efficiently cope with that dynamics. Intuitively, the chosen scheme using RK4 does not concentrate on how to approximate the transient distribution for a single time point in the most efficient way but allows the computation of its evolution over time. This is needed for example

when dealing with oscillatory systems in Chapter 7. Another advantage is that it is easy to adapt the only parameter of RK4, the time step h, at each time point since the current maximum exit rate of all significant states is known in each time point and does not have to be guessed in advance.

Algorithm 5 RK4$(p, \mathbf{Q}, h)$

1: $k_1 = \text{flow}(p, \mathbf{Q})$
2: $k_2 = \text{flow}(p + 0.5 \cdot h \cdot k_1, \mathbf{Q})$
3: $k_3 = \text{flow}(p + 0.5 \cdot h \cdot k_2, \mathbf{Q})$
4: $k_4 = \text{flow}(p + h \cdot k_3, \mathbf{Q})$
5: **return** $p + \frac{h}{6} \cdot (k_1 + 2 \cdot k_2 + 2 \cdot k_3 + k_4)$

Note that in order to provide an efficient computation, Algorithm 2 relies on an appropriate implementation of hash maps from states to their current transient probabilities. Also, all outgoing transitions of a state must be computable in an efficient way. One such way is provided by the use of MPMs induced by sets of transition classes as defined in Definition 12 allowing for a symbolic representation of $\mathbf{Q}$. The implementation of Algorithm 2 with a RK4 integration scheme that is used in this thesis contains further tweaks. On the one hand, the integration function is defined as a C++ template which has direct access to all data structures of the transient analysis algorithm eliminating the need of several hash maps and function calls. On the other hand, for each significant state, all outgoing transitions with rate and successor state are cached in a graph like structure. Note that this is possible since we only consider *time-homogeneous* Markov chains. From a theoretical perspective, the time complexity of Algorithm 2 using RK4 for time step integration is $\mathcal{O}(u \cdot t \cdot n)$ where u is the maximum exit rate encountered until time t and n is the maximum number of significant states until time t. Note that we assume a sparse model which is given in the case of an MPM induced by a constant number of R transition classes. Hence, the space complexity is $\mathcal{O}(n)$.

2.5 Steady State Analysis for Markov Population Models

The main topic of this thesis is to analyze the steady state distribution of MPMs. For that, we will review general requirements for the existence of the equilibrium distribution of CTMCs as well as approaches to compute it. Afterwards, we will clarify the problems of those standard techniques when trying to apply them to MPMs with an infinite state space.

2.5.1 Finite Irreducible Continuous-Time Markov Chains

At first, we would like to restrict to the case of CTMCs with a finite state space and review the conditions for the unique existence of the steady state distribution. Recall, if the Markov process is ergodic [Cin75], the limiting distribution coincides with the steady state distribution as stated in Equation (2.5).

Theorem 2: Ergodicity for Finite CTMCs ▷**Theorem 2**

A CTMC with *finite* state space is *ergodic* iff it is *irreducible*.

Proof: We refer to the respective proof in [Tri02]. □

Checking Irreducibility: Consequently, when analyzing a CTMC with finite state space, all we need to check in order to ensure the unique existence of the equilibrium distribution is whether the state space is a single communicating class as defined in Equation (2.7). We can do that by analyzing the CTMC's graph structure.

▷Definition 16

Definition 16: Directed Graph

A *directed graph* is a tuple (V, E) with *vertex* set V and *edge* relation $E \subseteq V \times V$.

▷Definition 17

Definition 17: Graph Structure of a CTMC

The *graph structure* of a CTMC $(\mathcal{S}, \mathbf{Q}, \pi(0))$ is defined as the graph $(\mathcal{S}, E)$ with $E = \{(i, j) \in \mathcal{S} \times \mathcal{S} \mid \mathbf{Q}_{ij} > 0\}$.

A communicating class of a CTMC corresponds to a *bottom strongly connected component* in its graph structure.

▷Definition 18

Definition 18: Strongly Connected Component (SCC)

Given a graph (V, E), a *strongly connected component* (SCC) is a subset $\mathcal{A} \subseteq \mathcal{S}$ such that

$$\forall v_s, v_e \in \mathcal{A}, v_s \neq v_e . \exists v_1, \ldots, v_n \in \mathcal{A}. \\ (v_s, v_1), (v_1, v_2), \ldots, (v_{n-1}, v_n), (v_n, v_e) \in E. \tag{2.21}$$

▷Definition 19

Definition 19: BSCC

A SCC $\mathcal{A}$ for a graph (V, E) is a *bottom strongly connected component* (BSCC) if

$$\forall v \in \mathcal{A}.\ \nexists (v, v') \in E . v' \notin \mathcal{A} \tag{2.22}$$

Theorem 3: Communicating Classes and BSCCs ▷Theorem 3

Given a CTMC $(\mathcal{S}, \mathbf{Q}, \pi(0))$ and its graph structure $(\mathcal{S}, E)$, a subset $\mathcal{A} \subseteq \mathcal{S}$ is a communicating class of the CTMC iff it is a BSCC in the graph.

Proof: We refer to the respective parts of the proof of Theorem 3.2.1 in [Nor98]. □

An efficient algorithm to compute all SCCs of a graph is *Tarjan's strongly connected components algorithm* [Tar72]. In order to compute all BSCCs of a graph, one can use Tarjan's algorithm to first compute all SCCs and filter out all those SCCs that have a transition leaving the SCC. If there is a single (B)SCC $\mathcal{A} = \mathcal{S}$, the CTMC is irreducible and thus has a unique steady state distribution. We note that in this thesis, we will not explicitly prove irreducibility in cases where it is obvious.

Relationship Between the Equilibrium Distribution of a CTMC and its UCTMC: Before we state possible ways of computing the equilibrium distribution of CTMC, we would like to clarify the relationship between the equilibrium distribution of a CTMC and its UCTMC.

Theorem 4: Equilibrium Distribution of UCTMCs ▷Theorem 4

The equilibrium distribution of the uniformized CTMC with an uniformization rate $u > \sup_{\mathbf{x} \in \mathcal{S}} -\mathbf{Q}_{\mathbf{xx}} \neq \infty$ coincides with the equilibrium distribution of the original Markov chain.

Proof: Let the DTMC $(\mathcal{S}, \mathbf{P}, \pi(0))$ be the uniformized CTMC of the continuous-time Markov chain $(\mathcal{S}, \mathbf{Q}, \pi(0))$ using uniformization rate u. Further let π be the steady state distribution of the uniformized CTMC. Consequently, we have that $\pi \cdot \mathbf{P} = \pi \Leftrightarrow \pi \cdot (\mathbf{P} - \mathbf{I}) = \mathbf{0}$. Inserting the definition of $\mathbf{P}$, we get $\pi \cdot (\mathbf{I} + u^{-1} \cdot \mathbf{Q} - \mathbf{I}) = 0$. Elimination of $\mathbf{I}$ and multiplication of both sides with u finally yields $\pi \cdot \mathbf{Q} = \mathbf{0}$. We need to choose a slightly higher uniformization rate than just $\sup_{\mathbf{x} \in \mathcal{S}} -\mathbf{Q}_{\mathbf{xx}}$

such that the resulting uniformized CTMC is *aperiodic* [Ste94] and possesses a unique steady state distribution. □

Note that uniformization is only applicable, if the exit rates are bounded. In the case of MPMs with infinite state spaces, this is not necessarily the case since the rates $\alpha(\mathbf{x})$ might grow without bounds with increasing population counts $\mathbf{x}_i$. Nevertheless, there is work on numerical analysis methods that *approximate* the transient probability distribution of an MPM [HBS+07, SBM07]. For example, techniques like *Fast Adaptive Uniformization* [DHMW09] based on adaptive uniformization [vMS94]. They work on a finite subset of the state space similar to the technique explained in Chapter 2.4.3. Thus, they enable the application of uniformization even for infinite MPMs without any a priori bound on the uniformization rate.

Computing the Steady State Distribution: In the case of an ergodic system with finite state space, Equation (2.17) can be solved using a huge variety of methods depending on the structure of the infinitesimal generator.

- Transient Analysis: According to its definition, the transient probability distribution $\pi(t)$ converges to the steady state distribution π with increasing time horizon t. Consequently, numerical methods for transient analysis as discussed in Chapter 2.4.2 can be used to approximate the equilibrium distribution assuming suitable convergence criteria and tests [Ste94]. Depending on the transition rates and chosen time step size, the convergence speed might not be satisfactory.

- Simulation: A long simulation run using the Stochastic Simulation Algorithm 1 can also be used to approximate the equilibrium distribution. For that, for each state, the algorithms needs to keep track of the time needed to return to the state after it has been left [Nas04]. A problem of simulative approaches are stiff systems, where transitions from one region of the state space to another are very unlikely. In order to faithfully capture the equilibrium distribution in all regions, the switching between regions must be observed very often, although it is a *rare event*. Consequently,

the needed simulation time is large. Note that τ-leaping methods as described in Chapter 2.4.1 can not be used since they skip intermediate states in-between two time steps.

- Iterative Methods: According to Theorem 4, the original problem can be reduced to the computation of the equilibrium distribution of a discrete-time Markov chain. Using Equation 2.12 to approximate π by recursively computing $\pi(t)$ for growing t is called *power method* [MPG29]. Depending on the uniformization rate, the convergence speed might still be slow. Nevertheless, even for huge state spaces like in the case of the Google page rank computation [BL06], this method is widely used. Other iterative methods like *Jacobi*, *Gauss-Seidel*, and *successive over-relaxation* (SOR) can directly be applied to Equation (2.17). For an overview of those iterative methods we refer to [Ste94]. In case of large linear equation systems, iterative *Krylov subspace* methods like *(bi-)conjugate gradient* methods can be applied as well. For an overview we refer to [VdV03] and [Ste94].

- Direct Solutions: If the state space is small, the equation system can also be solved directly, for example using *Gaussian elimination* [Atk89] or matrix decomposition methods like *LU factorization* [BH72].

- Product Form Solutions: Depending on the structure of the infinitesimal generator, the equilibrium distribution of an MPM with N population types can be represented by the product of marginal distributions

$$\pi_{\left[\mathbf{x}_1 \ \ldots \mathbf{x}_N\right]} = c \cdot \prod_{i=1}^{N} \pi_{i|\mathbf{x}_i}$$

with a scaling factor $c \in \mathbb{R}^+$. Product form solutions provide an efficient way to evaluate the steady state distribution even for larger dimensions N since it suffices to compute and store the marginal distribution for each dimension. The state wise probabilities then can be computed by a simple product. Population models induced by chemical reactions only posses a product form

solution if the chemical reaction network is *weakly reversible* and has a *deficiency of zero* [ACK10].

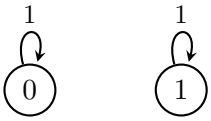
2.5.2 Finite Reducible Continuous-Time Markov Chains

If a CTMC is not irreducible, the limiting distribution is not unique since it depends on the initial distribution. We want to illustrate that situation using the following example.

▷Example 6

Example 6: Reducible CTMC

Consider the following CTMC which is not irreducible since it has the two BSCCs $\mathcal{B}_1 = \{0\}$ and $\mathcal{B}_2 = \{1\}$.



Here, the limiting distribution coincides with the initial distribution $\pi(0)$ since none of the two states can be left.

In those cases, the *limiting distribution* might be of interest instead which can be computed efficiently as formalized in Theorem 5.

▷Theorem 5

Theorem 5: Limiting Distribution

Given a *finite* and *reducible* CTMC $(\mathcal{S}, \mathbf{Q}, \pi(0))$ with $n \in \mathbb{N}$ BSCCs $\mathcal{B}_1, \ldots, \mathcal{B}_n$, the corresponding *limiting distribution* is given by

$$\lim_{t\to\infty} \mathbf{Pr}\left[\mathcal{X}(t) = s\right] = \begin{cases} 0 & \text{if } s \notin \mathcal{B}_i \quad \forall 1 \leq i \leq n \\ \mathbf{reach}_{\mathcal{B}_i} \cdot \pi_s^{\mathcal{B}_i} & \text{if } s \in \mathcal{B}_i, \end{cases}$$

where $\pi^{\mathcal{B}_i}$ is the equilibrium distribution of the (irreducible) CTMC with state space $\mathcal{B}_i$ and infinitesimal generator $\mathbf{Q}_{\mathcal{B}_i}$ (and arbitrary

initial distribution) and $\mathbf{reach}_{\mathcal{B}_i}$ denotes the probability to reach BSCC $\mathcal{B}_i$ starting with the initial distribution.

Proof: We refer to the proof of Theorem 6.16 in [Kul95]. □

Consequently, when interested in the limit distribution of a finite reducible CTMC, Theorem 5 also gives a suitable computation scheme. The first task is to compute all BSCCs $\mathcal{B}_1, \ldots, \mathcal{B}_n$ for example using Tarjan's algorithm [Tar72] and filtering out all those SCCs that can be left by a transition. Next, the probability $\mathbf{reach}_{\mathcal{B}_i}$ to reach each BSCC $\mathcal{B}_i$ needs to be computed. Also, the local steady states $\pi^{\mathcal{B}_i}$ of BSCCs $\mathcal{B}_i$ need to be computed as described in Chapter 2.5.1. Finally, the limiting distribution can be assembled by weighting the local steady state distributions $\pi^{\mathcal{B}_i}$ with the respective reachability probabilities $\mathbf{reach}_{\mathcal{B}_i}$ and setting the long run probability of all *transient states*, that is, states $s \in \mathcal{S} \setminus (\mathcal{B}_1 \cup \cdots \cup \mathcal{B}_n)$, to zero.

2.5.3 Infinite Continuous-Time Markov Chains

Most stochastic models based on chemical reaction networks as introduced in Chapter 2.3.2 do not have an upper limit on the molecule counts per chemical species. Consequently, the resulting MPM has an infinite state space. Unfortunately, if the state space of a CTMC is infinite, neither the results nor the methods for finite CTMCs in Chapters 2.5.1 and 2.5.2 are directly applicable.

The Problem of Deciding Ergodicity: More precisely, for an infinite state CTMC, a further condition for ergodicity called *positive recurrence* alongside irreducibility must be satisfied which is hard to check directly.

▷Theorem 6

Theorem 6: Ergodicity for Infinite CTMCs

A CTMC $\mathcal{X}(t)$ with *infinite* state space $\mathcal{S}$ is *ergodic* iff it is *irreducible* and at least one state $s \in \mathcal{S}$ is *positive recurrent*, that is, the expected return time

$$\mathbf{Exp}\left[\inf\{t \geq t^* : \mathcal{X}(t) = s\} \mid \mathcal{X}(0) = s\right]$$

is finite, where $t^* = \inf\{t \geq 0 : \mathcal{X}(t) \neq \mathcal{X}(0)\}$ is the time of the first jump of $\mathcal{X}$.

Proof: We refer to the proof of Theorem 3.5.3 in [Nor98]. □

Note that also checking *irreducibility* is not straightforward since a graph analysis traversing all states as described in Chapter 2.5.1 is not possible due to the infinite amount of states.

The Problem of Infinite Support: Even if ergodicity has been shown, the computation of the equilibrium distribution can not be done as in the finite case. Standard direct methods like Gauss-elimination operate on the complete infinitesimal generator matrix which is infinite in size and therefore can not be kept in memory. But also standard iterative methods can not be used as they require the current approximation of the complete equilibrium distribution to be kept in memory although it might have infinite support as illustrated in Example 7.

▷Example 7

Example 7: The M/M/∞ Queue in Equilibrium

The M/M/∞ queue as shown in Example 1 is *ergodic* and has

$$\pi_i = \frac{(\frac{\lambda}{\mu})^i}{i!} \cdot e^{-\frac{\lambda}{\mu}} > 0 \qquad \forall i \in \mathbb{N}$$

as its equilibrium distribution, that is, the *Poisson* distribution with parameter $\rho = \frac{\lambda}{\mu}$.

Consequently, the state space needs to be truncated to a finite set that captures the true nature of the steady state distribution. We want to remark that such an *analytical solution* only exists when there is a *single population type* and the number of individuals in each state may only change by one [GW76]. When such a change exceeds one, *approximation* techniques for example exploiting that the fraction π_{i+1}/π_i is a *slowly varying quantity* [Lan62] can be used [GW76] to reason about the mean and variance of π.

One could argue that approaches based on inexact transient analysis as introduced in Chapter 2.4.2 could be used to approximate the equilibrium distribution since at each time point the algorithm concentrates on a finite set of significant states. But unfortunately, exactly that way of truncating the state space in each iteration poses serious problems for certain systems. We want to illustrate those problems with the following simple multimodal system. Note that for simplicity and in order to be able to compare with standard methods, a finite state system was chosen but the inherent problems are present in infinite systems as well.

Model 2: Simple Multi-Stable System ▷**Model 2**

The *simple multi-stable system* consists of chemical species $A(1)$ and $B(2)$ and the reactions

$$A + B \xrightarrow{\tau_A} 2 \cdot B$$

$$B + A \xrightarrow{\tau_B} 2 \cdot A$$

$$A \xrightarrow{\nu_A} B$$

$$B \xrightarrow{\nu_B} A$$

These four reactions can be described by the transition classes

$$\alpha^{(1)}(\mathbf{x}) = \nu_A \cdot \mathbf{x}_A + \tau_A \cdot \mathbf{x}_A \cdot \mathbf{x}_B, \quad \mathbf{v}^{(1)} = \begin{bmatrix} -1 & 1 \end{bmatrix}$$

$$\alpha^{(2)}(\mathbf{x}) = \nu_B \cdot \mathbf{x}_B + \tau_B \cdot \mathbf{x}_A \cdot \mathbf{x}_B, \quad \mathbf{v}^{(2)} = \begin{bmatrix} 1 & -1 \end{bmatrix}$$

since reactions 1 and 3 as well as reactions 2 and 4 share the same change vector.

Depending on the choice of the initial distribution, the extent of the state space varies. More precisely, if for example we choose initial distribution

$$\pi_{\begin{bmatrix} a & b \end{bmatrix}}(0) = 1 \text{ with } \begin{bmatrix} a & b \end{bmatrix} \in \mathbb{N}^2,$$

the resulting state space is

$$\mathcal{S} = \{ \begin{bmatrix} \mathbf{x}_A & \mathbf{x}_B \end{bmatrix} \in \mathbb{N}^2 \mid \mathbf{x}_A + \mathbf{x}_B = z \}$$

with $z = a + b$. Since the total number of molecules present initially is preserved by both change vectors, we have $\mathbf{x}_B = z - \mathbf{x}_A$ for all states $\begin{bmatrix} \mathbf{x}_A & \mathbf{x}_B \end{bmatrix} \in \mathcal{S}$. Consequently, we can express the exit rate $-\mathbf{Q}_{\mathbf{xx}} = \alpha^{(1)}(\mathbf{x}) + \alpha^{(2)}(\mathbf{x})$ of a state $\mathbf{x} = \begin{bmatrix} \mathbf{x}_A & z - \mathbf{x}_A \end{bmatrix} \in \mathcal{S}$ as

$$f(\mathbf{x}_A) = -\mathbf{Q}_{\mathbf{xx}} = -(\tau_A + \tau_B) \cdot \mathbf{x}_A^2 + (\nu_A - \nu_B + [\tau_A + \tau_B] \cdot z) \cdot \mathbf{x}_A + \nu_B \cdot z.$$

If we interpret $f(\mathbf{x}_A)$ as a function $f : \mathbb{R} \to \mathbb{R}$, we can bound the exit rate from above in order to find a suitable uniformization rate. Since $f(\mathbf{x}_A)$ is a quadratic concave function, it reaches its maximum where $\mathbf{x}_A$ satisfies $\frac{d}{d\mathbf{x}_A} f(\mathbf{x}_A) = 0$ with

$$\frac{d}{d\mathbf{x}_A} f(\mathbf{x}_A) = -2 \cdot (\tau_A + \tau_B) \cdot \mathbf{x}_A + \nu_A - \nu_B + [\tau_A + \tau_B] \cdot z$$

which is the case at

$$\mathbf{x}_A = \frac{\nu_A - \nu_B + (\tau_A + \tau_B) \cdot z}{2 \cdot (\tau_A + \tau_B)}.$$

Now, we fix the parameters to

$$\tau_A = \tau_B = 10.0 \text{ and } \nu_A = \nu_B = 0.00001$$

and start with probability one in state $\begin{bmatrix}50 & 0\end{bmatrix}$ which gives $z = 50$. The maximal exit rate therefore is 1250.00025 which is reached in state [25 25] and also serves as the uniformization rate u.

First, we have a look at the steady state distribution of this system illustrated in Figure 2.4 which was computed by the PRISM tool [KNP11] using the power method on the UTCMC with a maximum of one million iterations and relative termination epsilon of $1e-6$. Note that this was only possible since the state space is finite. Clearly, the system is *bi-*

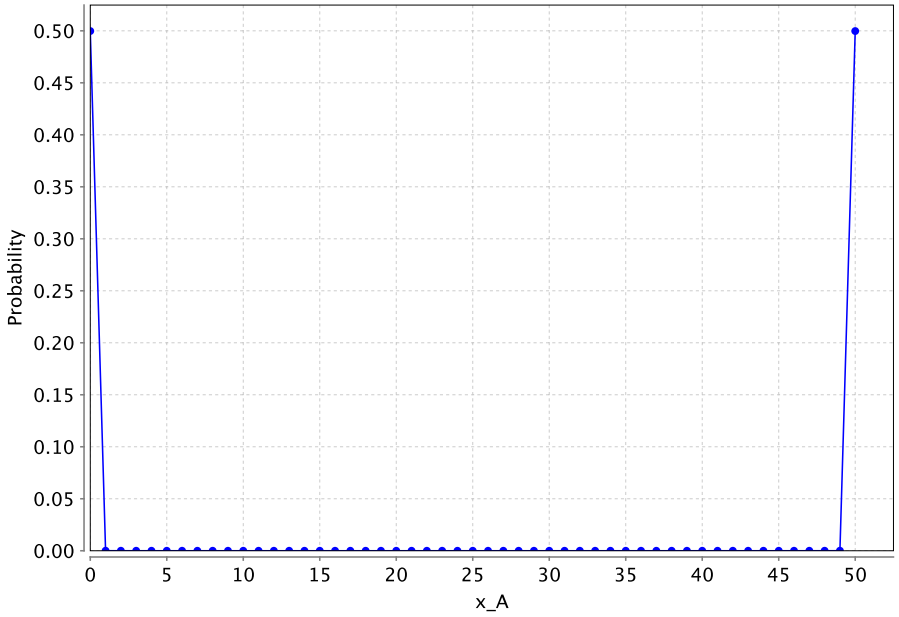


Figure 2.4: Steady state distribution of Model 2 with initial state [50 0] and parameter set $\tau_A = \tau_B = 10.0, \nu_A = \nu_B = 0.00001$.

stable in the sense that the steady state distribution has *two peaks*, one in state $\begin{bmatrix}0 & 50\end{bmatrix}$ and another one in state $\begin{bmatrix}50 & 0\end{bmatrix}$, each with approximately 50% of the total steady state probability mass.

Now, we perform an approximate transient analysis using Algorithm 2 with truncation threshold $\delta = 1e-6$, RK4 as the integration method and a time step size of $h = 0.5 \cdot u^{-1}$. Figure 2.5 illustrates that scenario where states with high transient probability are blue, states with

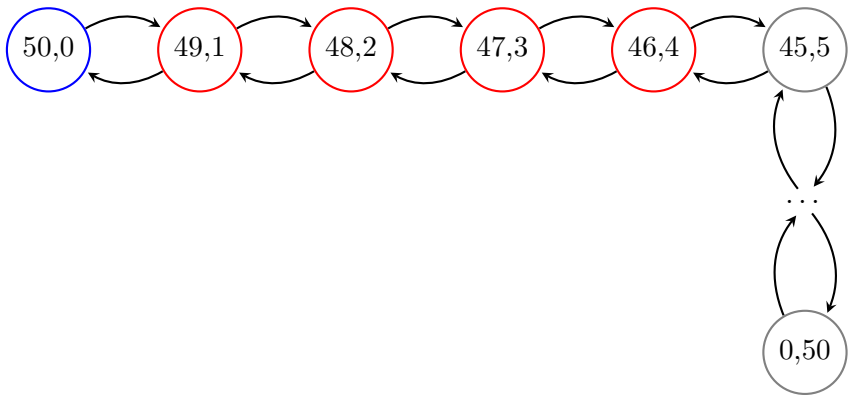


Figure 2.5: State space of Model 2 with initial state [50 0].

low transient probability mass that are truncated by Algorithm 2 are red and unreachable states are gray. The actual transition rates are omitted for simplicity. During the first iteration, state $\begin{bmatrix}50 & 0\end{bmatrix}$ looses around $1e-6$ of its probability mass to states $\begin{bmatrix}49 & 1\end{bmatrix}$,$\begin{bmatrix}48 & 2\end{bmatrix}$, $\begin{bmatrix}47 & 3\end{bmatrix}$, and $\begin{bmatrix}46 & 4\end{bmatrix}$. Since none of these successor states has a probability mass *greater* than $\delta = 1e-6$ those states are deleted together with their probability mass. This process can be witnessed in all further iterations while state $\begin{bmatrix}50 & 0\end{bmatrix}$ gradually looses all of its probability mass and the other states (gray states) including $\begin{bmatrix}0 & 50\end{bmatrix}$ are never reached. Consequently, an approximate transient analysis method based on state space truncation like Algorithm 2 might not capture the true shape of the steady state distribution. The reason is that regions with low steady state probability mass separating regions with high steady state probability mass will not be crossed since they will be truncated due to their low transient probabilities. Note that even when assuming that all relevant states will be kept with their probability mass, it is not obvious when to stop the approximate transient analysis, that is, when a sufficiently good approximation of the steady state distribution has been reached that also incorporates the algorithm's truncation error e.

To sum up, the task of correctly approximating the steady state distribution of an MPM with *infinite* state space can be split into three sub-tasks:

1. *Ergodicity* needs to be proven as it is the condition for the unique existence of the steady state distribution.

2. A *finite subset* of the potentially infinite state space needs to be computed that faithfully captures the shape of the equilibrium distribution like a subset that contains the major part of the steady state probability mass.

3. The *individual* steady state probabilities for all states in that finite subset need to be approximated.

These tasks will be treated in Chapters 3 and 5, where in general, we will concentrate on *ergodic* infinite state MPMs and leave reducible infinite state MPMs with ergodic BSCCs as future work.

CHAPTER 3

Geometric Bounds for the Equilibrium Distribution of Markov Population Models

As discussed in the previous chapter, the first two tasks in order to approximate the steady state distribution of an MPM with *infinite* state space is to prove ergodicity and to find a suitable truncation of the state space which contains most of the steady state probability mass.

3.1 Ergodicity Proofs and Geometric Bounds

We will tackle both problem at once by using *Lyapunov functions* following the course published in [DHSW11].

Definition 20: Lyapunov Function ▷**Definition 20**

A *Lyapunov function* for set $\mathcal{A} \subseteq \mathbb{N}^N$ with $N \in \mathbb{N}$ is a function $g : \mathcal{A} \to \mathbb{R}_0^+$ such that the set $\{s \in \mathcal{A} \mid g(s) \leq l\}$ is finite $\forall l \in \mathbb{R}_0^+$.

The following theorem from the literature states necessary and sufficient conditions for the ergodicity of infinite state CTMCs using the notion of Lyapunov functions.

▷Theorem 7

Theorem 7: Ergodicity via Lyapunov Functions

An irreducible CTMC $\mathcal{X}$ with (possibly infinite) state space $\mathcal{S}$ is *ergodic* iff there exists a Lyapunov function g for set $\mathcal{S}$, a finite subset $\mathcal{C} \subseteq \mathcal{S}$, and a constant $\gamma > 0$ such that

1. $d(s) \leq -\gamma$ for all $s \in \mathcal{S} \setminus \mathcal{C}$ and
2. $d(s) < \infty$ for all $s \in \mathcal{C}$

where

$$d(s) = \frac{d}{dt}\mathbf{Exp}\left[g(\mathcal{X}(t)) \mid \mathcal{X}(t) = s\right]$$

is the *drift* in state $s \in \mathcal{S}$.

Proof: For the proof, we refer to [Twe75]. □

For a CTMC with infinitesimal generator $\mathbf{Q}$ and a Lyapunov function g, the drift can be expressed by the product

$$d(s) = (\mathbf{Q} \cdot g)(s). \tag{3.1}$$

In the case of an MPM $(\mathcal{S}, \mathbf{Q}, \pi(0))$ induced by a set of transition classes $\{(\alpha^{(r)}, \mathbf{v}^{(r)}\}_{1 \leq r \leq R}$, the drift for state $\mathbf{x} \in \mathcal{S}$ as formulated in Equation (3.1) becomes

$$d(\mathbf{x}) = \sum_{r=1}^{R} \alpha^{(r)}(\mathbf{x}) \cdot \left[g(\mathbf{x} + \mathbf{v}^{(r)}) - g(\mathbf{x})\right]. \tag{3.2}$$

In the following, we assume that both conditions of Theorem 7 are satisfied, that is, we are given a Lyapunov function g which serves as aon *witness* for ergodicity. Thus, we can conclude that the MPM is ergodic and therefore has a unique equilibrium probability distribution π which satisfies $\pi \cdot \mathbf{Q} = 0$ subject to $\pi \cdot \mathbf{e} = 1$, where $\mathbf{Q}$ is defined as stated in Definition 12. Moreover, due to the conditions of Theorem 7, the drift is bounded from above, that is, there exists a number $c \in \mathbb{R}_0^+$ with

$$\infty > c \geq \sup_{\mathbf{x} \in \mathcal{S}} d(\mathbf{x}). \tag{3.3}$$

We use that upper bound c to define the *scaled Lyapunov function* g^* as

$$g^*(\mathbf{x}) = \frac{g(\mathbf{x})}{c+\gamma}. \tag{3.4}$$

Inserting Equation (3.4) in Equation (3.2) gives the *scaled drift* d^* which simplifies to

$$\begin{aligned} d^*(\mathbf{x}) &= \sum_{r=1}^{R} \alpha^{(r)}(\mathbf{x}) \cdot \left[g^*(\mathbf{x}+\mathbf{v}^{(r)}) - g^*(\mathbf{x}) \right] \\ &= \sum_{r=1}^{R} \alpha^{(r)}(\mathbf{x}) \cdot \left[\frac{g(\mathbf{x}+\mathbf{v}^{(r)})}{c+\gamma} - \frac{g(\mathbf{x})}{c+\gamma} \right] \\ &= \frac{1}{c+\gamma} \cdot \sum_{r=1}^{R} \alpha^{(r)}(\mathbf{x}) \cdot \left[g(\mathbf{x}+\mathbf{v}^{(r)}) - g(\mathbf{x}) \right] \\ &= \frac{d(\mathbf{x})}{c+\gamma}. \end{aligned} \tag{3.5}$$

Intuitively, the drift scales as the Lyapunov function scales. Combining (3.3) and the conditions of Theorem 7 gives

$$d^*(\mathbf{x}) \leq \frac{c}{c+\gamma} - \phi^{\mathcal{S}}_{\mathcal{S}\setminus\mathcal{C}}(\mathbf{x}), \tag{3.6}$$

where $\phi^{\mathcal{S}}_{\mathcal{S}\setminus\mathcal{C}}(\mathbf{x})$ denotes the characteristic function of set $\mathcal{S}\setminus\mathcal{C}$, since for states $\mathbf{x} \in \mathcal{C}$ we have

$$d^*(\mathbf{x}) \leq \frac{c}{c+\gamma} < \infty \text{ iff } d(\mathbf{x}) \leq c < \infty \tag{3.7}$$

due to Equation (3.5) and for states $\mathbf{x} \in \mathcal{S}\setminus\mathcal{C}$ we have

$$d^*(\mathbf{x}) \leq \frac{c}{c+\gamma} - 1 = \frac{-\gamma}{c+\gamma} \text{ iff } d(\mathbf{x}) \leq -\gamma \tag{3.8}$$

also due to Equation (3.5). Next, we multiply both sides of Equation (3.6) with $\pi_{\mathbf{x}}$, sum over all $\mathbf{x} \in \mathcal{S}$, and get

$$\sum_{\mathbf{x}\in\mathcal{S}} \pi_{\mathbf{x}} \cdot d^*(\mathbf{x}) \leq \sum_{\mathbf{x}\in\mathcal{S}} \pi_{\mathbf{x}} \cdot \left(\frac{c}{c+\gamma} - \phi^{\mathcal{S}}_{\mathcal{S}\setminus\mathcal{C}}(\mathbf{x}) \right).$$

We insert the definition $d^*(\mathbf{x}) = (\mathbf{Q} \cdot g^*)(\mathbf{x})$ and arrive at

$$\sum_{\mathbf{x}\in\mathcal{S}} \pi_{\mathbf{x}} \cdot \sum_{\mathbf{y}\in\mathcal{S}} \mathbf{Q}_{\mathbf{xy}} \cdot g^*(\mathbf{y}) \leq \sum_{\mathbf{x}\in\mathcal{S}} \pi_{\mathbf{x}} \cdot \left(\frac{c}{c+\gamma}\right) - \sum_{\mathbf{x}\in\mathcal{S}} \pi_{\mathbf{x}} \cdot \phi^{\mathcal{S}}_{\mathcal{S}\setminus\mathcal{C}}(\mathbf{x}).$$

By exploiting distributivity and associativity, the sum on the left-hand side becomes

$$\begin{aligned}&\sum_{\mathbf{x}\in\mathcal{S}} \pi_{\mathbf{x}} \cdot \sum_{\mathbf{y}\in\mathcal{S}} \mathbf{Q}_{\mathbf{xy}} \cdot g^*(\mathbf{y}) = \sum_{\mathbf{y}\in\mathcal{S}} \sum_{\mathbf{x}\in\mathcal{S}} \pi_{\mathbf{x}} \cdot \mathbf{Q}_{\mathbf{xy}} \cdot g^*(\mathbf{y}) \\ =&\sum_{\mathbf{y}\in\mathcal{S}} g^*(\mathbf{y}) \cdot \sum_{\mathbf{x}\in\mathcal{S}} \pi_{\mathbf{x}} \cdot \mathbf{Q}_{\mathbf{xy}} = \sum_{\mathbf{y}\in\mathcal{S}} g^*(\mathbf{y}) \cdot (\pi \cdot \mathbf{Q}) = 0,\end{aligned}$$

since $\pi \cdot \mathbf{Q} = 0$. Inserting the definition of $\phi^{\mathcal{S}}_{\mathcal{S}\setminus\mathcal{C}}(\mathbf{x})$ and bringing the negative term from the right-hand side to the left-hand side finally gives

$$\sum_{\mathbf{x}\in\mathcal{S}\setminus\mathcal{C}} \pi_{\mathbf{x}} \leq \frac{c}{c+\gamma}. \tag{3.9}$$

Consequently, we can use Equation (3.9) as an upper bound on the steady state probability mass outside $\mathcal{C}$ as well as a lower bound on the probability mass inside $\mathcal{C}$. More precisely, we choose some $\epsilon \in (0,1)$ and an appropriate Lyapunov function $g(\mathbf{x})$. Then, we compute the corresponding drift $d(\mathbf{x})$ as well as its supremum c, and let $\gamma = \frac{c\cdot(1-\epsilon)}{\epsilon}$ which gives $\frac{c}{c+\gamma} = \epsilon$. Thus, the scaled drift becomes

$$d^*(\mathbf{x}) = \frac{d(\mathbf{x})}{c+\gamma} = \frac{\epsilon}{c} \cdot d(\mathbf{x}) \tag{3.10}$$

with Equation (3.5) and

$$\frac{\epsilon}{c} = \frac{\frac{c}{c+\gamma}}{c} = \frac{1}{c+\gamma}.$$

In order to satisfy the fist condition of Theorem 7, we determine set $\mathcal{C}$ such that for all $\mathbf{x} \in \mathcal{S} \setminus \mathcal{C}$ we have that $d(\mathbf{x}) \leq -\gamma$, that is in terms of the scaled drift that $d^*(\mathbf{x}) \leq \frac{c}{c+\gamma} - 1 = \epsilon - 1$. Thus, we choose set $\mathcal{C}$ as

$$\mathcal{C} = \{\mathbf{x} \in \mathcal{S} \mid d^*(\mathbf{x}) > \epsilon - 1\}.$$

Also the second condition of Theorem 7 holds with Equation (3.7), that is, $d^*(\mathbf{x}) \leq \frac{c}{c+\gamma} = \epsilon < \infty$ since $d^*(\mathbf{x}) \leq \epsilon$ (because $d(\mathbf{x}) \leq c$) and Equation (3.10) for all $\mathbf{x} \in \mathcal{S}$ and especially for all $\mathbf{x} \in \mathcal{C}$. Consequently, since both conditions hold, also Equation (3.9) holds which gives $\sum_{\mathbf{x}\in\mathcal{S}\setminus\mathcal{C}} \pi_{\mathbf{x}} \leq \frac{c}{c+\gamma} = \epsilon$ and finally we get

$$\sum_{\mathbf{x}\in\mathcal{C}} \pi_{\mathbf{x}} > 1 - \epsilon.$$

We summarize that result in the following definition and theorem.

Definition 21: Drift ▷Definition 21

Given an MPM induced by the transition classes $\{(\alpha^{(r)}, \mathbf{v}^{(r)})\}_{1\leq r\leq R}$ with state space $\mathcal{S}$ and a Lyapunov function g on $\mathcal{S}$, we define the function

$$d(\mathbf{x}) = \sum_{r=1}^{R} \alpha^{(r)}(\mathbf{x}) \cdot \left[g(\mathbf{x} + \mathbf{v}^{(r)}) - g(\mathbf{x})\right]$$

as the corresponding *drift* of the MPM.

Intuitively, the drift of a state $\mathbf{x}$ is equivalent to the expected change of the Lyapunov function at this state as stated in Theorem 7.

Theorem 8: Geometric Bounds ▷Theorem 8

Given an irreducible MPM with state space $\mathcal{S}$ and drift d for some Lyapunov function, if there exists an upper bound on the drift

$$\infty > c \geq \sup_{\mathbf{x}\in\mathcal{S}} d(\mathbf{x})$$

and if set

$$\mathcal{C}_\epsilon = \{\mathbf{x} \in \mathcal{S} \mid \frac{\epsilon}{c} \cdot d(\mathbf{x}) > \epsilon - 1\}$$

for a chosen $\epsilon \in (0, 1)$ is *finite*, the MPM is *ergodic* and in addition

we have that

$$\sum_{\mathbf{x}\in\mathcal{C}_\epsilon} \pi_{\mathbf{x}} > 1 - \epsilon.$$

Proof: We refer to the preceding argumentation. □

Note that we demand an irreducible MPM in order to be able to use Theorem 8. As we will see in Section 3.7.2, generally proving irreducibility for MPMs is *undecidable*. However, the transition structure of MPMs is usually relatively regular such that it becomes trivial to show that each state is reachable by every other state as in the following example.

▷**Example 8**

Example 8: Irreducibility of the Gene Expression Model

In Model 1, the production of mRNA and its degradation have opposing change vectors just like the translation and protein degradation reactions. The resulting transition structure is shown in Figure 3.1, where an arrow denotes a transition with a non-zero rate (assuming the reactions rate constants are positive). Note that there are no arrows correspoding to the translation reaction for states with $M = 0$ since mRNA is needed for protein production. However, those states can reach any other state via the successor state of the mRNA producing reaction and each state with $M = 0$ can be reached via a sequence of mRNA degradation reactions.

Indeed, a similar transition pattern is present in all of our models we will use in this thesis and we will only show proofs of irreducibility in the non-trivial case.

3.2 Polynomial Drift

In general, it is not obvious how to check whether the set $\mathcal{C}_\epsilon$ is finite, but fortunately, our main application area is systems biology, where we

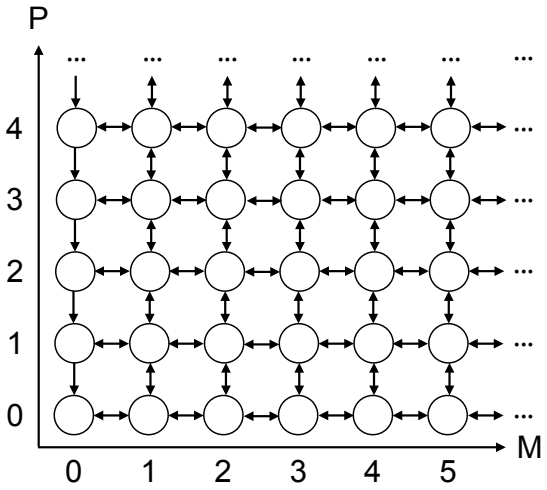


Figure 3.1: Transition structure of the gene expression model.

study MPMs induced by chemical reaction networks. For those systems, the transition rates induced by the respective propensity functions as specified in Definition 14 are *polynomials*. If we also restrict to *polynomial Lyapunov functions*, the drift as defined in Definition 21 becomes a polynomial function as well and we can make use of Theorem 9. Note that we generalize the use of Lyapunov function to domain $\mathbb{R}^N$ in order to simplify the analysis of the occurring polynomials. As a consequence, the drift is not only defined for states $s \in \mathcal{S}$ but for all $\mathbf{x} \in \mathbb{R}^N$.

Theorem 9: Polynomial Geometric Bounds ▷**Theorem 9**

Given an MPM with $\mathcal{S} \subseteq \mathbb{N}^N$ and polynomial drift d. If

$$\lim_{t\to\infty} d(\psi^N(t, \alpha_1, \ldots, \alpha_{N-1})) = -\infty$$

for all $\boldsymbol{\alpha} \in [0, \pi/2]^{N-1}$, then there exists an upper bound $\infty > c \geq \sup_{\mathbf{x}\in\mathcal{S}} d(\mathbf{x})$ and set $\mathcal{C}_\epsilon$ as defined in Theorem 8 is *finite* for all $\epsilon \in (0, 1)$.

Proof: For $N > 1$, we have that

$$\lim_{t\to\infty} d(\psi^N(t, \alpha_1, \ldots, \alpha_{N-1})) = -\infty$$

holds for all $\boldsymbol{\alpha} \in [0, \pi/2]^{N-1}$ and thus for each such $\boldsymbol{\alpha}$, there exists a $t_{\boldsymbol{\alpha}} \in \mathbb{R}_0^+$ such that

$$d(\psi^N(t, \alpha_1, \ldots, \alpha_{N-1})) \leq \epsilon - 1 \tag{3.11}$$

for all $t > t_\alpha$. Now, let $t_{\text{sup}} = \sup_{\boldsymbol{\alpha}\in[0,\pi/2]^{N-1}} t_{\boldsymbol{\alpha}} < \infty$. If $N = 1$, there is a $t_{\text{sup}} \in \mathbb{R}_0^+$ for which

$$d(t) \leq \epsilon - 1$$

for all $t > t_{\text{sup}}$. In both cases we have that $\mathcal{C}_\epsilon \subseteq [0, \lceil t_{\text{sup}} \rceil]^N \subset \mathbb{N}^N$, that is, $\mathcal{C}_\epsilon$ is a finite set. Further, in the case of $N > 1$, each function $d(\psi^N(t, \alpha_1, \ldots, \alpha_{N-1}))$ for any $\boldsymbol{\alpha} \in [0, \pi/2]^{N-1}$ is an univariate polynomial in $t \in \mathbb{R}_0^+$ which has no singularity and diverges to $-\infty$ and thus must have an upper bound $c_{\boldsymbol{\alpha}}$. Taking $c = \sup_{\boldsymbol{\alpha}\in[0,\pi/2]^{N-1}} c_{\boldsymbol{\alpha}} < \infty$ gives the desired global upper bound. The same holds for $N = 1$, that is, $d(t)$ is a univariate polynomial with no singularities which diverges to $-\infty$ and thus must have an upper bound c. □

When analyzing polynomial drift functions $d(\mathbf{x})$ it also becomes clear, why the bounds are called *geometric* bounds. The reason is that the set of solutions to the equation

$$d(\mathbf{x}) = \epsilon - 1$$

marking the boundaries of set $\mathcal{C}$ are geometric objects. For example, if the drift is a quadratic polynomial, the above equation describes a *quadric surface* like a circle, sphere or ellipsoid.

What is left is to prove the criteria of Lyapunov functions for a collection of polynomial functions.

▷**Theorem 10**

Theorem 10: Polynomial Lyapunov Functions

Every N-dimensional *multivariate polynomial function* $g \in \mathbb{P}[\mathbf{x}]$ with $N \in \mathbb{N}^+$, only non-negative coefficients, and at least one

monomial $c \cdot \mathbf{x}_1^{d_1} \cdot \ldots \cdot \mathbf{x}_N^{d_N}$ with $c > 0$ and $d_i > 0$ for some $1 \leq i \leq N$ is a Lyapunov function for $\mathcal{S} \subseteq \mathbb{N}^N$.

Proof: For $N = 1$, we have that $g(t)$ is *strictly monotonic increasing*. Note that constant functions are excluded since there needs to be at least one monomial. Consequently, for any $l \in \mathbb{R}_0^+$ there exists a $r \in \mathbb{R}_0^+$ such that $g(t) > l$ for all $t > r$ which ensures that the set $\{s \in \mathcal{S} \mid g(t) \leq l\} \subseteq [0, \lceil r \rceil] \subset \mathbb{N}$ is finite.

In the case of $N > 1$, we study the hyper-spherical re-parameterization $g(\psi(t, \alpha_1, \ldots, \alpha_{N-1}))$ of g. Like in the one-dimensional case, each function $g(\psi(t, \alpha_1, \ldots, \alpha_{N-1}))$ with fixed angles $\boldsymbol{\alpha} \in [0, \pi/2]^{N-1}$ is strictly monotonic increasing in t. Consequently, for any $l \in \mathbb{R}_0^+$ and any choice of angles $\boldsymbol{\alpha} \in [0, \pi/2]^{N-1}$ there exists a $r_{\boldsymbol{\alpha}}$ such that

$$g(\psi(t, \alpha_1, \ldots, \alpha_{N-1})) > l$$

for all $t > r_{\boldsymbol{\alpha}}$. Taking the supremum $r_{\sup} = \sup_{\boldsymbol{\alpha} \in ([0,\pi/2]^{N-1})} r_{\boldsymbol{\alpha}}$ gives a distance from the origin after which regardless of the angles $\boldsymbol{\alpha}$, the function $g(\psi(t, \alpha_1, \ldots, \alpha_{N-1}))$ will evaluate to a value greater than l. Consequently, we can use the over-approximation $\{s \in \mathcal{S} \mid g(s) \leq l\} \subseteq [0, \lceil r_{\sup} \rceil]^N \subset \mathbb{N}^N$ to show the required finiteness. □

Thus, according to Theorem 10, *linear* polynomials of the form $p(\mathbf{x}) = \mathbf{c}_1^T \cdot \mathbf{x}$ with $\mathbf{c}_1 \in {\mathbb{R}_0^+}^N \setminus \{\mathbf{0}\}$ like the *one-norm* on the positive quadrant

$$||\mathbf{x}||_1 = \mathbf{e}^T \cdot \mathbf{x}$$

and *quadratic* polynomials of the form $p(\mathbf{x}) = \mathbf{x}^T \cdot \mathbf{C}_2 \cdot \mathbf{x} + \mathbf{c}_2^T \cdot \mathbf{x}$ with $\mathbf{c}_2 \in {\mathbb{R}_0^+}^N$ and $\mathbf{C}_2 \in {\mathbb{R}_0^+}^{N \times N} \setminus \{\mathbf{0}\}$ like the *squared Euclidean norm*

$$||\mathbf{x}||_2^2 = \mathbf{x}^T \cdot \mathbf{x}$$

are Lyapunov functions for MPMs.

We will also make use of the *weighted distance* to a point as a Lyapunov function as stated in the following theorem.

▷Theorem 11

Theorem 11: Weighted Distance as a Lyapunov Function

Functions $g : \mathbb{R}_0^{+N} \to \mathbb{R}_0^+$ with $N \in \mathbb{N}$ of the form

$$g(\mathbf{x}) = ||\mathbf{x} - \widetilde{\mathbf{x}}||_2^2 \circ \mathbf{c}$$

with $\widetilde{\mathbf{x}} \in \mathbb{N}^N$ and $\mathbf{c} \in \mathbb{R}_{>0}^N$ are Lyapunov functions for state spaces $\mathcal{S} \subseteq \mathbb{N}^N$.

Proof: The proof for $N = 1$ is trivial. Here, we will only give a proof for $N = 2$ (since such a Lyapunov function will be used later on) and note that proofs for $N > 2$ can be derived analogously. We observe that the area

$$\{\mathbf{x} \in \mathbb{R}^2 \mid a \cdot (\mathbf{x}_1 - \widetilde{\mathbf{x}}_1)^2 + b \cdot (\mathbf{x}_2 - \widetilde{\mathbf{x}}_2)^2 \leq l\}$$

is finite since it belongs to the interior (including the border) of an *ellipse* with major and axes $\sqrt{(\frac{l}{a})}$ and $\sqrt{(\frac{l}{b})}$ that coincide with the Cartesian axes. Consequently, the set

$$\{\mathbf{x} \in \mathbb{N}^2 \mid a \cdot (\mathbf{x}_1 - \widetilde{\mathbf{x}}_1)^2 + b \cdot (\mathbf{x}_2 - \widetilde{\mathbf{x}}_2)^2 \leq l\}$$

is finite for all l. □

Now, we have discussed all relevant techniques to practically apply Theorem 7 in order to retrieve a *finite* set $\mathcal{C} \subseteq \mathcal{S}$ that provably contains at least $1 - \epsilon$ of the total steady state probability mass (for a chosen $\epsilon \in (0, 1)$). But what is still missing is a way to approximate the steady state probabilities of *individual* states $\mathbf{x} \in \mathcal{C}$.

3.3 Stochastic Complementation for Infinite State CTMCs

For that, we first need to extend the theory of *stochastic complementation*, originally developed for finite DTMCs [Mey89], to infinite CTMCs. We assume that the CTMC $\mathcal{X}$ with state space $\mathcal{S}$ is *ergodic* and has an *irreducible* row-/column-finite infinitesimal generator matrix $\mathbf{Q}$. First,

we partition $\mathbf{Q}$ with respect to a *finite* set $\mathcal{C}$ as

$$\mathbf{Q} = \begin{bmatrix} \mathbf{Q}_{\mathcal{C}\mathcal{C}} & \mathbf{Q}_{\mathcal{C}\overline{\mathcal{C}}} \\ \\ \mathbf{Q}_{\overline{\mathcal{C}}\mathcal{C}} & \mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}} \end{bmatrix},$$

where the *finite* square submatrix $\mathbf{Q}_{\mathcal{C}\mathcal{C}}$ contains all transitions within set $\mathcal{C}$, and the possibly *infinite* matrices $\mathbf{Q}_{\mathcal{C}\overline{\mathcal{C}}}$, $\mathbf{Q}_{\overline{\mathcal{C}}\mathcal{C}}$, and $\mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}$ contain all transitions from $\mathcal{C}$ to $\overline{\mathcal{C}} = \mathcal{S} \setminus \mathcal{C}$, from $\overline{\mathcal{C}}$ to $\mathcal{C}$, and within $\overline{\mathcal{C}}$, respectively. Further, we partition the probability transition matrix of the *embedded CTMC* likewise as

$$\mathbf{E} = \begin{bmatrix} \mathbf{E}_{\mathcal{C}\mathcal{C}} & \mathbf{E}_{\mathcal{C}\overline{\mathcal{C}}} \\ \\ \mathbf{E}_{\overline{\mathcal{C}}\mathcal{C}} & \mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}} \end{bmatrix}.$$

Note that matrix $\mathbf{E}$ is well-defined and row-/column-finite since $\mathbf{Q}$ is a row-/column-finite matrix. We make use of those partitions in the following definition which provides a finite abstraction of the CTMC where the stochastic process is only observed within set $\mathcal{C}$.

Definition 22: Stochastic Complement ▷Definition 22

For a CTMC with infinitesimal generator $\mathbf{Q}$ and state space $\mathcal{S}$, the *stochastic complement* (SC) $\overline{\mathbf{Q}}_{\mathcal{C}}$ of (finite) set $\mathcal{C} \subseteq \mathcal{S}$ is defined as

$$\overline{\mathbf{Q}}_{\mathcal{C}} = \mathbf{Q}_{\mathcal{C}\mathcal{C}} + \mathbf{Q}_{\mathcal{C}\overline{\mathcal{C}}} \cdot \sum_{i=0}^{\infty} \mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^{i} \cdot \mathbf{E}_{\overline{\mathcal{C}}\mathcal{C}},$$

where $\overline{\mathcal{C}} = \mathcal{S} \setminus \mathcal{C}$ and the infinite series $\sum_{i=0}^{\infty} \mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^{i}$ is meant to be element-wise convergent.

By element-wise convergence of a matrix $\mathbf{M}$ we mean that each element $\mathbf{M}_{ij}$ converges.

3.3.1 Properties of the Stochastic Complement

The stochastic complement for a finite block of an infinite generator matrix has already been defined in [Ris02] (cf. page 22). However, in

this definition, the matrix inverse of the infinite block $-\mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}$ is used, but no proof of well-definedness is given. However, our definition of the stochastic complement circumvents this problem by using embedding and allows us to prove well-definedness of $\overline{\mathbf{Q}}_{\mathcal{C}}$.

▷Theorem 12

Theorem 12: Properties of the Stochastic Complement

The stochastic complement $\overline{\mathbf{Q}}_{\mathcal{C}}$ of $\mathcal{C}$ for an ergodic CTMC is a *well-defined infinitesimal generator* matrix and it is *irreducible*.

Proof: For $\overline{\mathbf{Q}}_{\mathcal{C}}$ to be well-defined, we need to ensure that $\sum_{i=0}^{\infty} \mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}$ converges element-wise. Now, since $\mathbf{Q}$ is ergodic and therefore irreducible, its embedded matrix $\mathbf{E}$ is irreducible as well. Consequently, we can follow [RT96] to reason that $\mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}$ is strictly substochastic, and with Corollary 2 of Lemma 5.4^D of [Sen73] we have that $\sum_{i=0}^{\infty} \mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}$ is element-wise finite.

In order to prove that $\overline{\mathbf{Q}}_{\mathcal{C}}$ is an infinitesimal generator matrix, we will show that

(i) $\overline{\mathbf{Q}}_{\mathcal{C}} \cdot \mathbf{e} = \mathbf{0}$ and

(ii) $\left[\overline{\mathbf{Q}}_{\mathcal{C}}\right]_{\mathbf{xy}} \geq 0$ for $\mathbf{x} \neq \mathbf{y}$.

For this, we first observe that

$$\mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}} \cdot \mathbf{e} + \mathbf{E}_{\overline{\mathcal{C}}\mathcal{C}} \cdot \mathbf{e} = \mathbf{e}$$

can be rewritten as

$$\mathbf{E}_{\overline{\mathcal{C}}\mathcal{C}} \cdot \mathbf{e} = (\mathbf{I} - \mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}) \cdot \mathbf{e}.$$

We use this equality to show that $\sum_{i=0}^{n} \mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^{i} \cdot \mathbf{E}_{\overline{\mathcal{C}}\mathcal{C}}$ is a stochastic matrix. Obviously, it is nonnegative and we are left to show that the row sums converge element-wise to one. We begin with considering the finite

sequence

$$\begin{aligned}\sum_{i=0}^{n} \mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^{i} \cdot \mathbf{E}_{\overline{\mathcal{C}}\mathcal{C}} \cdot \mathbf{e} &= (\mathbf{I} + \mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}} + \mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^{2} + \cdots + \mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^{n}) \cdot (\mathbf{I} - \mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^{n}) \\ &= (\mathbf{I} - \mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^{n+1}) \cdot \mathbf{e} \\ &= \mathbf{e} - \mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^{n+1} \cdot \mathbf{e},\end{aligned}$$

and have to show $\lim_{n\to\infty} \mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^{n+1} \cdot \mathbf{e} = \mathbf{0}$, that is, that the sequence $\mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^{n+1} \cdot \mathbf{e}$ converges element-wise to the zero vector. For that, we will reason about the k-th row sum of $\mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^{n}$, that is, we have

$$\mathbf{e}_k^T \cdot \mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^{n} \cdot \mathbf{e} = (\mathbf{e}_k^T \cdot \mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^{n} \cdot \mathbf{e})^T = \mathbf{e} \cdot (\mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^{T})^n \cdot \mathbf{e}_k = ||(\mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^{T})^n \cdot \mathbf{e}_k||_1 \quad (3.12)$$

which holds since $\mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}$ and therefore also $\mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^{n}$ is row/column-finite for all n. Now, we observe that $\lim_{n\to\infty}(\mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^{T})^n \cdot \mathbf{e}_k = \mathbf{0}$ element-wise since all entries of $\mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^{n}$ tend to zero. By Schur's Theorem [Sch21] (cf. also [Ban55], page 137) according to which in the sequence space

$$l^1 := \{\mathbf{x} = (\xi)_i^\infty \mid ||\mathbf{x}||_1 := \sum_{i=1}^{\infty} |\xi_i| < \infty\}$$

convergence in the 1-norm coincides with element-wise convergence, we can conclude that $\lim_{n\to\infty} ||(\mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^{T})^n \cdot \mathbf{e}_k||_1 = 0$ element-wise for all k. Therefore, in combination with Equation (3.12) we also get that $\lim_{n\to\infty} \mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^{n+1} \cdot \mathbf{e}_k = \mathbf{0}$ which finally results in $\sum_{i=0}^{n} \mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^{i} \cdot \mathbf{E}_{\overline{\mathcal{C}}\mathcal{C}} \cdot \mathbf{e} = \mathbf{e}$. Consequently, for the row sums (i) of $\overline{\mathbf{Q}}_{\mathcal{C}}$ we get

$$\overline{\mathbf{Q}}_{\mathcal{C}} \cdot \mathbf{e} = \mathbf{Q}_{\mathcal{C}\mathcal{C}} \cdot \mathbf{e} + \mathbf{Q}_{\mathcal{C}\overline{\mathcal{C}}} \cdot \underbrace{\sum_{i=0}^{n} \mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^{i} \cdot \mathbf{E}_{\overline{\mathcal{C}}\mathcal{C}} \cdot \mathbf{e}}_{\mathbf{e}} = (\mathbf{Q}_{\mathcal{C}\mathcal{C}} + \mathbf{Q}_{\mathcal{C}\overline{\mathcal{C}}}) \cdot \mathbf{e} = \mathbf{0}$$

since $\mathbf{Q}$ is an infinitesimal generator with $\mathbf{Q} \cdot \mathbf{e} = \mathbf{0}$. The nonnegativity of the off-diagonal entries of $\overline{\mathbf{Q}}_{\mathcal{C}}$ (ii) follows from $\mathbf{Q}_{\mathcal{C}\overline{\mathcal{C}}} \geq \mathbf{0}$, $\sum_{i=0}^{n} \mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^{i} \cdot \mathbf{E}_{\overline{\mathcal{C}}\mathcal{C}} \geq \mathbf{0}$, and $\mathbf{Q}_{\mathcal{C}\mathcal{C}\mathbf{xy}} \geq 0$ for $\mathbf{x} \neq \mathbf{y}$.

In order to show the irreducibility of $\overline{\mathbf{Q}}_{\mathcal{C}}$, we notice that if $\mathbf{Q}_{\mathcal{C}\mathcal{C}}$ is irreducible, $\overline{\mathbf{Q}}_{\mathcal{C}}$ is trivially irreducible as well. Otherwise, $\mathbf{Q}_{\mathcal{C}\mathcal{C}}$ can be

symmetrically permuted into a block lower-triangular form in which the diagonal blocks are irreducible [DR78]. Without loss of generality, consider partitionings of $\mathbf{Q}$ and $\overline{\mathbf{Q}}_{\mathcal{C}}$ with two diagonal blocks as in

$$\mathbf{Q} = \left[\begin{array}{cc|c} \mathbf{Q}_{\mathcal{C}\mathcal{C}}{}^{(11)} & \mathbf{0} & \mathbf{Q}_{\mathcal{C}\overline{\mathcal{C}}}{}^{(1)} \\ \mathbf{Q}_{\mathcal{C}\mathcal{C}}{}^{(21)} & \mathbf{Q}_{\mathcal{C}\mathcal{C}}{}^{(22)} & \mathbf{Q}_{\mathcal{C}\overline{\mathcal{C}}}{}^{(2)} \\ \hline \mathbf{Q}_{\overline{\mathcal{C}}\mathcal{C}}{}^{(1)} & \mathbf{Q}_{\overline{\mathcal{C}}\mathcal{C}}{}^{(2)} & \mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}} \end{array}\right] \qquad \overline{\mathbf{Q}}_{\mathcal{C}} = \begin{bmatrix} \overline{\mathbf{Q}}_{\mathcal{C}}{}^{(11)} & \overline{\mathbf{Q}}_{\mathcal{C}}{}^{(12)} \\ \overline{\mathbf{Q}}_{\mathcal{C}}{}^{(21)} & \overline{\mathbf{Q}}_{\mathcal{C}}{}^{(22)} \end{bmatrix}.$$

In this partitioning, $\mathbf{Q}_{\mathcal{C}\mathcal{C}}{}^{(11)}$ and $\mathbf{Q}_{\mathcal{C}\mathcal{C}}{}^{(22)}$ are irreducible, and therefore, $\overline{\mathbf{Q}}_{\mathcal{C}}{}^{(11)}$ and $\overline{\mathbf{Q}}_{\mathcal{C}}{}^{(22)}$ are irreducible as well. Let $\mathbf{Pr}\left[x \rightsquigarrow y\right]$ denote the probability of reaching state y from state x independent of time. Obviously, there exist states $x, w \in \mathcal{C}$ and $y, z \in \overline{\mathcal{C}}$ such that

$$\mathbf{Q}_{\mathcal{C}\overline{\mathcal{C}}}{}^{(1)}{}_{xy} > 0, \mathbf{E}_{\overline{\mathcal{C}}\mathcal{C}}{}^{(2)}{}_{zw} > 0, \text{ and } \mathbf{Pr}\left[y \rightsquigarrow z\right] > 0,$$

otherwise $\mathbf{Q}$ must have been reducible. Here, $\mathbf{E}_{\overline{\mathcal{C}}\mathcal{C}}{}^{(2)}$ corresponds to the block $\mathbf{Q}_{\overline{\mathcal{C}}\mathcal{C}}{}^{(2)}$ of $\mathbf{Q}$ in $\mathbf{E}$ assuming the same partitioning. Moreover, there exists an $i \geq 0$ such that

$$(\mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^{i})_{yz} > 0$$

since $\mathbf{Pr}\left[y \rightsquigarrow z\right] > 0$. Consequently, we get

$$\overline{\mathbf{Q}}_{\mathcal{C}}{}^{(12)}_{xw} = \left(\mathbf{Q}_{\mathcal{C}\overline{\mathcal{C}}} \cdot \sum_{i=0}^{\infty} \mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^{i} \cdot \mathbf{E}_{\overline{\mathcal{C}}\mathcal{C}}\right)_{xw} > 0.$$

If $\mathbf{Q}_{\mathcal{C}\mathcal{C}}{}^{(21)} = \mathbf{0}$ or there are more than two irreducible diagonal blocks, similar arguments can be used to prove that appropriate off-diagonal blocks in the upper-triangular part of $\mathbf{Q}$ are nonzero. □

3.3.2 Equilibrium Distribution of the Stochastic Complement

The following theorem relates the equilibrium distribution of the original CTMC with the equilibrium distribution of the stochastic complement.

Theorem 13: Equilibrium Distribution of the SC

▷Theorem 13

Let $\overline{\pi}$ be the unique equilibrium distribution that corresponds to the generator matrix $\overline{\mathbf{Q}}_{\mathcal{C}}$ of the stochastic complement of an ergodic CTMC for set $\mathcal{C}$. Then,

$$\overline{\pi}_x = \frac{\pi_x}{\sum_{y \in \mathcal{C}} \pi_y}$$

holds for all $x \in \mathcal{C}$, where π denotes the (unique) equilibrium distribution of the original CTMC.

Proof: First, we proof the identity

$$\begin{bmatrix} \mathbf{Q}_{\mathcal{C}\mathcal{C}} & \mathbf{Q}_{\mathcal{C}\overline{\mathcal{C}}} \\ \mathbf{Q}_{\overline{\mathcal{C}}\mathcal{C}} & \mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \sum_{i=0}^{\infty} \mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^{i} \cdot \mathbf{E}_{\overline{\mathcal{C}}\mathcal{C}} & \mathbf{I} \end{bmatrix}$$

$$= \begin{bmatrix} \mathbf{Q}_{\mathcal{C}\mathcal{C}} + \mathbf{Q}_{\mathcal{C}\overline{\mathcal{C}}} \cdot \sum_{i=0}^{\infty} \mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^{i} \cdot \mathbf{E}_{\overline{\mathcal{C}}\mathcal{C}} & \mathbf{Q}_{\mathcal{C}\overline{\mathcal{C}}} \\ \mathbf{0} & \mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}} \end{bmatrix} \tag{3.13}$$

and express the probability transition matrix of the embedded matrix of $\mathbf{Q}$ as

$$\mathbf{E} = \mathbf{I} + \mathit{dinv}(\mathbf{Q}) \cdot \mathbf{Q}$$

where $\mathit{dinv}(\mathbf{Q})$ is a diagonal matrix with

$$\mathit{dinv}(\mathbf{Q})_{xy} = |\mathbf{Q}_{xx}|^{-1}$$

if $x = y$ and 0 otherwise. Note that the product $\mathit{dinv}(\mathbf{Q}) \cdot \mathbf{Q}$ of the two infinite matrices is well-defined since both factors are row-/column-finite infinite matrices. By $\mathit{dinv}(\mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}})$ we denote the block of $\mathit{dinv}(\mathbf{Q})$ that corresponds to block $\mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}$ in $\mathbf{Q}$. Further, we observe that

$$(\mathbf{I} - \mathbf{T}) \cdot \sum_{i=0}^{\infty} \mathbf{T}^i = \mathbf{I}$$

for any strictly substochastic row-/column-finite infinite matrix $\mathbf{T}$ where the infinite series $\sum_{i=0}^{\infty} \mathbf{T}^i$ is meant to converge element-wise. Now we

choose $\mathbf{T} = \mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}} = \mathbf{I} + dinv(\mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}) \cdot \mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}$ for which substochasticity has been shown in the proof of Theorem 12 and get

$$(\mathbf{I} - (\mathbf{I} + dinv(\mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}) \cdot \mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}})) \cdot \sum_{i=0}^{\infty} (\mathbf{I} + dinv(\mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}) \cdot \mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}})^i = \mathbf{I},$$

which reduces to

$$dinv(\mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}) \cdot \mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}} \cdot \sum_{i=0}^{\infty} (\mathbf{I} + dinv(\mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}) \cdot \mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}})^i = -\mathbf{I}.$$

Now, let the infinite matrix $diag(\mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}})$ be defined by

$$diag(\mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}})_{xy} = |\mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}xy}|$$

if $x = y$ and 0 otherwise. Then, multiplying the previous equality on both sides by $diag(\mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}})$ from the left gives

$$\mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}} \cdot \sum_{i=0}^{\infty} (\mathbf{I} + dinv(\mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}) \cdot \mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}})^i = -diag(\mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}})$$

since $diag(\mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}) \cdot dinv(\mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}) = \mathbf{I}$. Exploiting this equality finally yields

$$\begin{aligned}
& \mathbf{Q}_{\overline{\mathcal{C}}\mathcal{C}} + \mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}} \cdot \sum_{i=0}^{\infty} \mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}{}^i \cdot \mathbf{E}_{\overline{\mathcal{C}}\mathcal{C}} \\
= \; & \mathbf{Q}_{\overline{\mathcal{C}}\mathcal{C}} + \mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}} \cdot \sum_{i=0}^{\infty} (\mathbf{I} + dinv(\mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}) \cdot \mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}})^i \cdot (dinv(\mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}) \cdot \mathbf{Q}_{\overline{\mathcal{C}}\mathcal{C}}) \\
= \; & \mathbf{Q}_{\overline{\mathcal{C}}\mathcal{C}} + \underbrace{(-diag(\mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}})) \cdot dinv(\mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}})}_{-\mathbf{I}} \cdot \mathbf{Q}_{\overline{\mathcal{C}}\mathcal{C}} = \mathbf{0}.
\end{aligned}$$

Note that the multiplications involved in taking the powers of $\mathbf{E}_{\overline{\mathcal{C}}\overline{\mathcal{C}}}$ and in the product of $dinv(\mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}})$ and $diag(\mathbf{Q}_{\overline{\mathcal{C}}\overline{\mathcal{C}}})$ are well-defined since these matrices are row-/column-finite. Now, let

$$\pi = \begin{bmatrix} \pi^{\mathcal{C}} & \pi^{\overline{\mathcal{C}}} \end{bmatrix}$$

be the unique equilibrium distribution of the original CTMC partitioned according to the sets $\mathcal{C}$ and $\overline{\mathcal{C}}$ satisfying $\pi \cdot \mathbf{Q} = \mathbf{0}$ subject to $\pi \cdot \mathbf{e} = 1$. If we multiply Equation (3.13) by π from the left, we get

$$\pi_{\mathcal{C}} \cdot \overline{\mathbf{Q}} + \pi_{\overline{\mathcal{C}}} \cdot \mathbf{0} = \mathbf{0}.$$

With Theorem 12, $\overline{\mathbf{Q}}$ is irreducible and therefore $\pi_{\mathcal{C}}$ is the unique solution of $\pi_{\mathcal{C}} \cdot \overline{\mathbf{Q}} = \mathbf{0}$ up to the unit 1-norm and thus, $\overline{\pi} = \pi^{\mathcal{C}} \cdot (\sum_{x \in \mathcal{C}} \pi_x)^{-1}$. □

3.4 State-Wise Bounds

Our goal is to derive lower and upper bounds for the equilibrium probability of each state in the finite subset $\mathcal{C}$. For that, we will adapt the results in [CS84] and [Cou85]. As in the previous section, we assume that $\mathbf{Q}$ is the infinitesimal generator matrix of an ergodic CTMC. Let the substochastic matrix $\mathbf{C}$ be defined as

$$\mathbf{C} = \mathbf{I} + u^{-1} \cdot \mathbf{Q}_{\mathcal{C}\mathcal{C}} \tag{3.14}$$

where $u > \max_x |\mathbf{Q}_{\mathcal{C}\mathcal{C}xx}|$. We define the transition probability matrices $\mathbf{C}^{(x)}$ for $x \in \mathcal{C}$ as

$$\mathbf{C}^{(x)} = \mathbf{C} + (\mathbf{e} - \mathbf{C} \cdot \mathbf{e}) \cdot \mathbf{e}_x^T, \tag{3.15}$$

that is, we increase the elements of the column corresponding to state x in $\mathbf{C}$ to obtain a stochastic matrix. Thus, $\mathbf{C}^{(x)}$ corresponds to a DTMC for which we want to compute the steady state distribution. We remark that $\mathbf{Q}_{\mathcal{C}\mathcal{C}}$, and hence, $\mathbf{C}$ may be reducible. In that case, the quality of the bounds computed using $\mathbf{C}^{(x)}$ deteriorate [Cou85, Section 6]. However, in all problems we consider, $\mathbf{Q}_{\mathcal{C}\mathcal{C}}$ turns out to be irreducible, implying the irreducibility of $\mathbf{C}^{(x)}$. Therefore, in the following derivations, we assume $\mathbf{Q}_{\mathcal{C}\mathcal{C}}$ to be irreducible. The following theorem gives the desired bounds for the equilibrium distribution conditioned on set $\mathcal{C}$.

Theorem 14: State-Wise Bounds ▷Theorem 14

Let π be the unique equilibrium probability distribution of a CTMC $(\mathcal{S}, \mathbf{Q}, \pi(0))$ and let $\mathcal{C} \subseteq \mathcal{S}$ be a finite set. If the matrices $\mathbf{C}^{(y)}$ as defined in Equations (3.14) and (3.15) are irreducible for all

$y \in \mathcal{B} = \{z \in \mathcal{C} \mid \exists w \in \mathcal{S} \setminus \mathcal{C}.\mathbf{Q}_{wz} > 0\}$, then, for all $x \in \mathcal{C}$ holds

$$\min_{y \in \mathcal{B}} \pi_x^{(y)} \leq \frac{\pi_x}{\sum_{z \in \mathcal{C}} \pi_z} \leq \max_{y \in \mathcal{B}} \pi_x^{(y)}, \tag{3.16}$$

where $\pi^{(y)}$ is the unique equilibrium probability distribution of the DTMC associated with $\mathbf{C}^{(y)}$.

Proof: From Theorem 12 we know that $\overline{\mathbf{Q}}_{\mathcal{C}}$ is well-defined and irreducible. Furthermore, Theorem 13 ensures that π conditioned on $\mathcal{C}$ is identical to the equilibrium distribution of $\overline{\mathbf{Q}}_{\mathcal{C}}$ which also coincides with the equilibrium distribution of

$$\overline{\mathbf{C}} = \mathbf{I} + u^{-1} \cdot \overline{\mathbf{Q}}_{\mathcal{C}}.$$

Note that

$$u > \max_{x \in \mathcal{C}} |\mathbf{Q}_{\mathcal{C}\mathcal{C}xx}| \geq \max_{x \in \mathcal{C}} |\overline{\mathbf{Q}}_{\mathcal{C}xx}|$$

implies that $\overline{\mathbf{C}}$ is a stochastic matrix. The same arguments as in [CS84] can be used to conclude that the equilibrium distribution of $\overline{\mathbf{C}}$ is contained in the convex hull of the unit 1-norm Perron vectors of the irreducible matrices $\mathbf{C}^{(x)}$ for $x \in \mathcal{B}$ (see also [Cou85] and [MdSeSG89]). □

Now, if we choose set $\mathcal{C}$ for $\epsilon \in (0, 1)$ according to Theorem 8, that is, we have

$$1 - \epsilon < \sum_{x \in \mathcal{C}} \pi_x \leq 1,$$

Theorem 14 allows us to bound the (unconditional) individual equilibrium probabilities for states $x \in \mathcal{C}$ by

$$(1 - \epsilon) \cdot \min_{y \in \mathcal{B}} \pi_x^{(y)} < \pi_x \leq \max_{y \in \mathcal{B}} \pi_x^{(y)}. \tag{3.17}$$

This equation reveals an algorithmic procedure for the computation of lower and upper bounds on the equilibrium probabilities. More precisely, we determine the set $\mathcal{C}$ as suggested in Chapter 3.1. Then, for each $y \in \mathcal{B}$, we construct $\mathbf{C}^{(y)}$ and, provided it is irreducible, compute the distribution $\pi^{(y)}$. Note that if the CTMC is an MPM induced by a

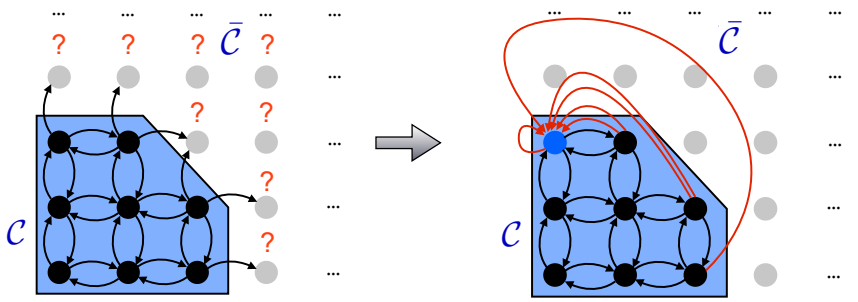


Figure 3.2: Redirection of transitions leaving set $\mathcal{C}$.

set of transition classes $\{(\alpha^{(r)}, \mathbf{v}^{(r)}\}_{1 \leq r \leq R}$, it is easy to compute the set $\mathcal{B} \subseteq \mathcal{C}$ of states that have incoming transitions from $\overline{\mathcal{C}}$, since

$$\mathcal{B} = \{\mathbf{x} \in \mathcal{C} \mid \exists r \in \{1, \ldots, R\}.\alpha^{(r)}(\mathbf{x} - \mathbf{v}^{(r)}) > 0 \wedge (\mathbf{x} - \mathbf{v}^{(r)}) \in \overline{\mathcal{C}}\},$$

that is, for each state in the finite set $\mathcal{C}$, a finite number of R possible predecessor states have to be checked. Finally, for all $x \in \mathcal{C}$, we derive bounds according to Equation (3.17) where for the states $x \in \overline{\mathcal{C}}$, we have the trivial bound $0 < \pi_x \leq \epsilon$. Figure 3.2 illustrates the redirection scheme.

3.5 Automated Ergodicity Proofs and Efficient Computation of Geometric and State-Wise Bounds

We developed a prototypical tool called *Geobound* [Spi13b] written in Java 6 which implements both, the computation of geometric bounds via Lyapunov functions and the computation of the conditional probabilities, using the approach presented in the previous section.

3.5.1 Extrema of Polynomial Functions

Several steps in Geobound rely on the computation of *all* global and local extrema of polynomial functions $f(\mathbf{x}) \in \mathbb{P}[\mathbf{x}]$ in $\mathbb{R}_0^{+N}$. For that,

we equate the gradient

$$\nabla f(\mathbf{x}) = \left[\frac{\partial}{\partial \mathbf{x}_1} f(\mathbf{x}), \ldots, \frac{\partial}{\partial \mathbf{x}_N} f(\mathbf{x})\right]$$

with the zero vector $\mathbf{0}$. Note that this alone would not yield all extrema since we neglect the borders of $\mathbb{R}_0^{+N}$. Therefore, we additionally solve $\nabla f(\mathbf{x}) = \mathbf{0}$ for every projection of $f(\mathbf{x})$ onto each subspace of $\mathbb{R}^N$ by setting all combinations of variables $\mathbf{x}_i$ with $i \in \{1, \ldots, N\}$ to zero. Finally, we filter out all extrema outside of $\mathbb{R}_0^{+N}$. The resulting nonlinear equation systems are solved using the *HOM4PS-2.0* package [LLT08], an implementation of the *polyhedral homotopy continuation method*. Note that traditional global optimization techniques like gradient descent or simulated annealing cannot be used since we need to guarantee that the found maximum is indeed the *global* maximum. The time and space complexities are those of solving 2^N multi-variate polynomial equation systems whose actual complexity highly depends on the solution method of choice. For a discussion of the costs of the polyhedral continuation method for polynomial systems we refer to [Ver96].

3.5.2 Ergodicity Proof

Geobound takes as input a description based on a set of transition classes $\{(\alpha^{(r)}, \mathbf{v}^{(r)})\}_{1 \leq r \leq R}$ of an MPM with state space $\mathcal{S} \subseteq \mathbb{N}^N$ and tries to prove ergodicity using a user specified Lyapunov function. If the drift $d(\mathbf{x})$ tends to $-\infty$ with increasing distance to the origin, then only finitely many states have a nonnegative drift and the first two conditions of Theorem 7 hold. We can check this condition automatically by reparameterizing the drift $d(\mathbf{x})$ via $\mathbf{x} = t \cdot \overline{\mathbf{x}}$ with $t \in \mathbb{R}_0^+$ and $\overline{\mathbf{x}} \in [0,1]^N$. Now, if m is the maximum degree of all monomials in $f(\mathbf{x})$, the reparameterized $f(\mathbf{x})$ has the form

$$f(t, \overline{\mathbf{x}}) = t^m \cdot p_m(\overline{\mathbf{x}}) + t^{m-1} \cdot p_{m-1}(\overline{\mathbf{x}}) + \cdots + t \cdot p_1(\overline{x}) + p_0$$

with $p_m(\overline{\mathbf{x}}), \ldots, p_1(\overline{\mathbf{x}}) \in \mathbb{P}[\overline{\mathbf{x}}]$ and $p_0 \in \mathbb{R}$. We can interpret t as the distance from the origin with respect to the direction vector $\overline{\mathbf{x}}$. Since we are only interested in the behavior inside $\mathbb{R}_0^{+N}$, we restrict $\overline{\mathbf{x}}$ to $[0,1]^N$. We could restrict $\overline{\mathbf{x}}$ even further like we did in the proof of Theorem 10 but from a computational perspective, the region $[0,1]^N$ is

easier to manage. Consequently, we observe that for a fixed $\overline{\mathbf{x}} \in [0,1]^N$, the limit of $f(t, \overline{\mathbf{x}})$ for $t \to \infty$ is determined by the monomial with maximum degree, that is, $t^m \cdot p_m(\overline{\mathbf{x}})$. More precisely, if $p_m(\overline{\mathbf{x}}) < 0$, we have that $\lim_{t\to\infty} f(t, p_m(\overline{\mathbf{x}})) = -\infty$. Thus, all we need no check is whether $p_m(\overline{\mathbf{x}}) < 0$ for all $\overline{\mathbf{x}} \in [0,1]^N$. For that, we compute the global maximum $\overline{c}$ of $p_m(\overline{\mathbf{x}})$ inside $[0,1]^N$ using the technique described in the previous paragraph where we make sure that we also project on the borders with $\overline{\mathbf{x}}_i = 0$ and $\overline{\mathbf{x}}_i = 1$ for the respective indices $1 \leq i \leq N$ and ignore extrema in $\mathbb{R}_0^{+N} \setminus [0,1]^N$. Finally, if $\overline{c} < 0$, we can conclude that the drift tends to $-\infty$ in all directions. Again, the time and space complexities are those of solving 2^N multi-variate polynomial equation systems as mentioned in the previous subsection.

3.5.3 Geometric Bounds

In order to determine the maximum drift c, we compute all extrema of $d(\mathbf{x})$ as described above and assemble the scaled drift polynomial $d^*(x)$. Next, we start with the local maxima and recursively add all neighboring points of the integer grid that satisfy the condition in Theorem 8 for the scaled drift. In order to compute $\mathcal{B}$, we calculate for each $\mathbf{x} \in \mathcal{C}$ and each transition class $(\alpha^{(r)}, \mathbf{v}^{(r)})$ the predecessor $(\mathbf{x} - \mathbf{v}^{(r)})$. If $(\mathbf{x} - \mathbf{v}^{(r)}) \notin \mathcal{C}$ and $\alpha^{(r)}(\mathbf{x} - \mathbf{v}^{(r)}) > 0$, then we add $\mathbf{x}$ to $\mathcal{B}$. Since we are not aware of any border for the number of states $|\mathcal{C}|$ inside $\mathcal{C}$, we can not express the time and space complexities of the computation of $\mathcal{C}$ a priori. However, if $|\mathcal{C}|$ is known, the time and space complexity is $\mathcal{O}(|\mathcal{C}|)$. Time and space complexity of computing $\mathcal{B}$ then lies in $\mathcal{O}(R \cdot |\mathcal{C}|)$

3.5.4 State-Wise Bounds

Next, we compute the nonzero entries of the finite matrix $\mathbf{Q}_{\mathcal{C}\mathcal{C}}$ and the conditional bounds of $\pi_{\mathbf{x}}$ for $\mathbf{x} \in \mathcal{C}$ as explained in the previous section. For this, the tool employs the *SuperLU* [DEG+99] package for LU factorization. More precisely, in order to compute the equilibrium probability distributions $\pi^{(y)}$ of $\mathbf{C}^{(y)}$ for $y \in \mathcal{B}$ efficiently, we consider $\pi^{(y)} \cdot \mathbf{C}^{(y)} = \pi^{(y)}$ subject to $\pi^{(y)} \cdot \mathbf{e} = 1$. Since this equation needs to be solved $|\mathcal{B}|$ times for matrices $\mathbf{C}^{(y)}$ that differ from each other only in two columns, we uncouple the slack probability mass added to column y

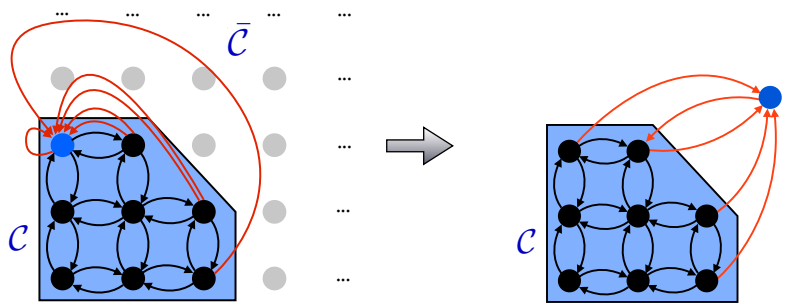


Figure 3.3: Modified redirection scheme of transitions leaving set $\mathcal{C}$ in order to allow LU factorization.

and, assuming that $\mathbf{Q}_{\mathcal{C}\mathcal{C}}$ is irreducible, obtain the irreducible stochastic matrix

$$\mathbf{P}^{(y)} = \begin{bmatrix} \mathbf{C} & \mathbf{e} - \mathbf{C} \cdot \mathbf{e} \\ \mathbf{e}_y^T & \mathbf{0} \end{bmatrix}$$

of order $|\mathcal{C}| + 1$. Intuitively, an additional state collects all transitions leaving set $\mathcal{C}$ and redirects them to a single state in $\mathcal{B}$. Figure 3.3 illustrates this new redirection scheme. Now, let $\mathbf{p}^{(y)} = \begin{bmatrix} \mathbf{p}^{(y)'} & p^{(y)'} \end{bmatrix}$ denote the equilibrium probability distribution of $\mathbf{P}^{(y)}$, that is, $\mathbf{p}^{(y)} \cdot \mathbf{P}^{(y)} = \mathbf{p}^{(y)}$ subject to $\mathbf{p}^{(y)} \cdot \mathbf{e} = 1$. Multiplying $(\mathbf{P}^{(y)} - \mathbf{I})$ on the left-hand side with $\mathbf{p}^{(y)}$ and equating to $\mathbf{0}$, we obtain $\mathbf{p}^{(y)'} \cdot (\mathbf{C} - \mathbf{I}) = -p^{(y)'} \cdot \mathbf{e}_y^T$, which can be solved subject to $\mathbf{p}^{(y)'} \cdot \mathbf{e} = 1$ for $y \in \mathcal{B}$. We remark that the coefficient matrix $\mathbf{C} - \mathbf{I}$ does not depend on y and therefore can be LU factorized once. Consequently, the time complexity of the whole procedure of computing conditional bounds reduces to that of a sparse LU factorization of a matrix of order $|\mathcal{C}|$ plus $|\mathcal{B}|$ forward and backward substitutions giving $\mathcal{O}(|\mathcal{C}|^3)$. However, we trade time for space, that is, the resulting LU decomposition matrix might not be sparse any more resulting in a $\mathcal{O}(|\mathcal{C}|^2)$ space complexity.

3.6 Case Studies

In the following we will evaluate the approach to bound the equilibrium distribution of infinite state CTMCs using a series of case studies. All computations were performed on a dual-core 2.66 GHz machine with 3 gigabytes of memory under Ubuntu 10.04.

3.6.1 Exclusive Switch

First, we consider a case study from biology, called the *exclusive switch*. In previous work, the equilibrium distribution of this gene regulatory network has been approximated by Monte-Carlo simulations [LLBB07], but no execution times are reported. Here, we present a direct method to truncate the state space using the approach based on Lyapunov functions presented in Chapter 3.1 and we approximate the equilibrium by computing state-wise bounds as explained in Chapter 3.4.

Model 3: Exclusive Switch ▷**Model 3**

The *exclusive switch* is a gene regulatory network consisting of two genes that share a promotor region (cf. Figure 3.4). The state of the promotor region determines whether the genes are used to produce proteins or not. Let $P_1(1)$ be the product of the first gene and let $P_2(2)$ be the product of the second gene. Each of the proteins can bind to the unbound promotor region $G(3)$ at rates β_1 and β_2, respectively, and thereby inhibit the production of the other protein. More precisely, if the promotor region is free, both proteins, P_1 and P_2, are produced at rates ρ_1 and ρ_1, respectively. Otherwise, if P_1 (P_2) is bound to the promotor region represented by chemical species $G.P_1(4)$ ($G.P_1(5)$), only the first (second) gene is active and thus only P_1 (P_2) is produced. The proteins can also unbind from the promotor region at rates ν_1 and ν_2 and degrade at rates δ_1 and δ_2, respectively.

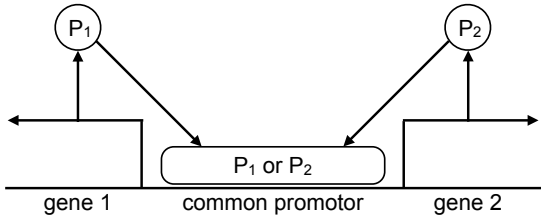


Figure 3.4: Illustration of the exclusive switch model. The picture has been adapted from [LLBB07].

$$
\begin{array}{llll}
G & \xrightarrow{\rho_1} & G + P_1 & \qquad G \xrightarrow{\rho_2} G + P_2 \\
P_1 & \xrightarrow{\delta_1} & \emptyset & \qquad P_2 \xrightarrow{\delta_2} \emptyset \\
G + P_1 & \xrightarrow{\beta_1} & G.P_1 & \qquad G + P_2 \xrightarrow{\beta_2} G.P_2 \\
G.P_1 & \xrightarrow{\nu_1} & G + P_1 & \qquad G.P_2 \xrightarrow{\nu_2} G + P_2 \\
G.P_1 & \xrightarrow{\rho_1} & G.P_1 + P_1 & \qquad G.P_2 \xrightarrow{\rho_2} G.P_2 + P_2
\end{array}
$$

Geometric Bounds: For sake of simplicity we choose the squared Euclidean norm as the Lyapunov function for our case study. Note that this choice influences the accuracy of the probability bounds. Obviously, it is possible to take as Lyapunov function a multivariate polynomial and optimize over the coefficients. Our experimental results, however, indicate that for the models that we consider, the squared Euclidean norm performs well. Thus, the Lyapunov function g is defined as

$$g(\mathbf{x}) = \mathbf{x}^T \cdot \mathbf{x}.$$

We choose symmetric model parameters $\rho = \rho_1 = \rho_2 = 0.05, \delta = \delta_1 = \delta_2 = 0.005, \beta = \beta_1 = \beta_2 = 0.01, \nu = \nu_1 = \nu_2 = 0.008$ and the drift

function $d(\mathbf{x})$ becomes

$$\begin{aligned} d(\mathbf{x}) &= \sum_{j=1}^{10} \alpha^{(j)}(\mathbf{x})(g(\mathbf{x}+\mathbf{v}^{(j)}) - g(\mathbf{x})) \\ &= \rho \cdot \mathbf{x}_3 \cdot (2 \cdot \mathbf{x}_1 + 1) + \rho \cdot \mathbf{x}_3 \cdot (2 \cdot \mathbf{x}_2 + 1) \\ &+ \delta \cdot \mathbf{x}_1 \cdot (-2 \cdot \mathbf{x}_1 + 1) + \delta \cdot \mathbf{x}_2 \cdot (-2 \cdot \mathbf{x}_2 + 1) \\ &+ \beta \cdot \mathbf{x}_1 \cdot \mathbf{x}_3 \cdot (-2 \cdot \mathbf{x}_1 - 2 \cdot \mathbf{x}_3 + 2 \cdot \mathbf{x}_4 + 3) \\ &+ \beta \cdot \mathbf{x}_2 \cdot \mathbf{x}_3 \cdot (-2 \cdot \mathbf{x}_2 - 2 \cdot \mathbf{x}_3 + 2 \cdot \mathbf{x}_5 + 3) \\ &+ \nu \cdot \mathbf{x}_4 \cdot (-2 \cdot \mathbf{x}_4 + 2 \cdot \mathbf{x}_1 + 2 \cdot \mathbf{x}_3 + 3) \\ &+ \nu \cdot \mathbf{x}_5 \cdot (-2 \cdot \mathbf{x}_5 + 2 \cdot \mathbf{x}_2 + 2 \cdot \mathbf{x}_3 + 3) \\ &+ \rho \cdot \mathbf{x}_4 \cdot (2 \cdot \mathbf{x}_1 + 1) + \rho \cdot \mathbf{x}_5 \cdot (2 \cdot \mathbf{x}_2 + 1). \end{aligned}$$

With the initial condition $\mathbf{x} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \end{bmatrix}$, it is easy to see that $\mathbf{x}_k \in \{0, 1\}$ for $k \in \{3, 4, 5\}$ and $\mathbf{x}_3 + \mathbf{x}_4 + \mathbf{x}_5 = 1$, that is, at any time the promotor region is either free or a molecule of type P_1 or P_2 is bound to it. We consider the drift functions

$$\begin{aligned} d_3(\mathbf{x}_1, \mathbf{x}_2) &= d(\mathbf{x}_1, \mathbf{x}_2, 1, 0, 0) \\ &= -2{\cdot}(\delta + \beta){\cdot}(\mathbf{x}_1^2 + \mathbf{x}_2^2) + (2{\cdot}\rho + \delta + \beta){\cdot}(\mathbf{x}_1 + \mathbf{x}_2) + 2{\cdot}\rho \\ d_4(\mathbf{x}_1, \mathbf{x}_2) &= d(\mathbf{x}_1, \mathbf{x}_2, 0, 1, 0) \\ &= -2{\cdot}\delta{\cdot}(\mathbf{x}_1^2 + \mathbf{x}_2^2) + \delta{\cdot}(\mathbf{x}_1 + \mathbf{x}_2) + 2{\cdot}(\nu + \rho){\cdot}\mathbf{x}_1 + \rho + \nu \\ d_5(\mathbf{x}_1, \mathbf{x}_2) &= d(\mathbf{x}_1, \mathbf{x}_2, 0, 0, 1) \\ &= -2{\cdot}\delta{\cdot}(\mathbf{x}_1^2 + \mathbf{x}_2^2) + \delta{\cdot}(\mathbf{x}_1 + \mathbf{x}_2) + 2{\cdot}(\nu + \rho){\cdot}\mathbf{x}_2 + \rho + \nu \end{aligned}$$

that is, one for each state of the promotor. It turns out that the global maximum drift $c = \max_{\mathbf{x} \in \mathcal{S}} d(\mathbf{x}) = 0.42465$ is reached in

$$\mathbf{x}^{(1)} = \begin{bmatrix} 6.05 & 0.25 & 0 & 1 & 0 \end{bmatrix} \text{ and } \mathbf{x}^{(2)} = \begin{bmatrix} 0.25 & 6.05 & 0 & 0 & 1 \end{bmatrix}.$$

Note that the coefficients of the highest order terms are negative, which implies that $\mathcal{C} = \{\mathbf{x} \in \mathcal{S} \mid d(\mathbf{x}) \geq -\gamma\}$ is finite. We choose $\epsilon = 0.1$ which yields a set of states that contains at least 90% of the equilibrium probability mass. The value of γ in this case is $\gamma = \frac{c \cdot (1-\epsilon)}{\epsilon} = 3.82185$. We derive the scaled drift functions

$$d_k^*(\mathbf{x}) = \frac{\epsilon}{c} \cdot d_k(\mathbf{x}) \text{ with } k \in \{3, 4, 5\}$$

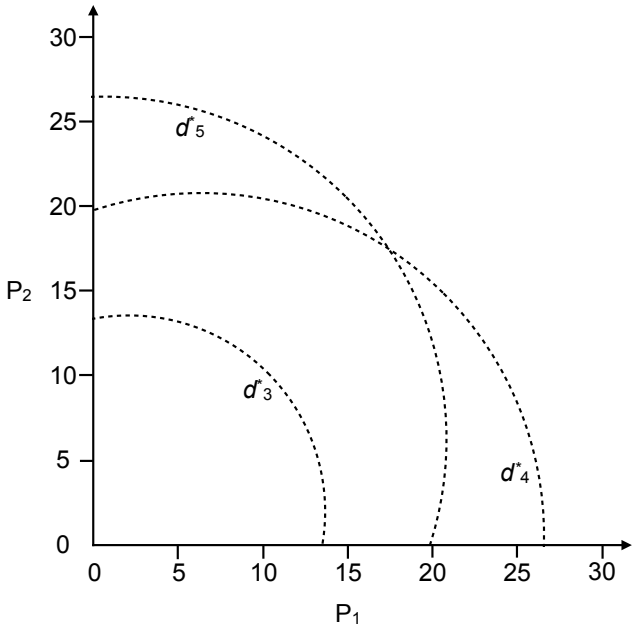


Figure 3.5: A set $\mathcal{C}$ that contains at least 90% of the steady state probability mass for the exclusive switch model. The dashed lines confine the three sets whose union is $\mathcal{C}$.

and finally compute

$$\begin{aligned}\mathcal{C} &= \{\mathbf{x} \in \mathcal{S} \mid d^*(\mathbf{x}) > -0.9\} \\ &= \{(\mathbf{x}_1, \mathbf{x}_2, 1, 0, 0) \in \mathcal{S} \mid d_3^*(\mathbf{x}_1, \mathbf{x}_2) > -0.9\} \\ &\cup \{(\mathbf{x}_1, \mathbf{x}_2, 0, 1, 0) \in \mathcal{S} \mid d_4^*(\mathbf{x}_1, \mathbf{x}_2) > -0.9\} \\ &\cup \{(\mathbf{x}_1, \mathbf{x}_2, 0, 0, 1) \in \mathcal{S} \mid d_5^*(\mathbf{x}_1, \mathbf{x}_2) > -0.9\}.\end{aligned}$$

We illustrate the set $\mathcal{C}$ in Figure 3.5, with respect to the state space projected on P_1 and P_2. The set $\mathcal{C}$ contains 1,680 states altogether.

ϵ	$\lvert\mathcal{C}\rvert$	$\lvert\mathcal{B}\rvert$	Time (s)		Memory (MB)	Δ
			$\mathcal{C}$,$\mathcal{B}$	Bounds		
1.0e-1	1,680	108	1.0	5.0	43.5	2.9e-3
5.0e-2	2,934	144	1.4	5.7	64.2	1.4e-3
1.0e-2	11,994	297	1.4	83.2	304.6	2.8e-4

Table 3.1: Numerical results (geometric bounds) for the exclusive switch.

State-wise Bounds: We apply Theorem 14 to bound the individual equilibrium probabilities conditioned on being in $\mathcal{C}$. In order to compute unconditional bounds, the lower bounds must be multiplied by $(1-\epsilon) = 0.9$. We list the numerical results for the exclusive switch in Table 3.1 where we vary ϵ (first column). The columns "$|\mathcal{C}|$" and "$|\mathcal{B}|$" list the size of the sets $\mathcal{C}$ and $\mathcal{B}$. In the two columns with heading "Time (s)" we give the execution time in seconds that was needed to determine the sets $\mathcal{C}$ and $\mathcal{B}$ as well as the execution time for the LU factorization that was used to compute the bounds on the conditional probabilities. Moreover, we list our memory requirements (in MegaBytes) in column "Memory (MB)". Finally, in the last column we give the maximum absolute difference of the bounds of the unconditional equilibrium probabilities, Δ, that is,

$$\Delta = \max_{\mathbf{x}\in\mathcal{C}}(\max_{\mathbf{y}\in\mathcal{B}} \pi_{\mathbf{x}}^{(\mathbf{y})} - (1-\epsilon)\cdot \min_{\mathbf{y}\in\mathcal{B}} \pi_{\mathbf{x}}^{(\mathbf{y})}).$$

In Table 3.2 we illustrate the lower conditional bounds (left) and the difference between upper and lower bounds (right) for various choices of ϵ. Note that the maximum absolute difference of the conditional probabilities is smaller than Δ. In order to compute bounds on the unconditional probabilities, each lower bound of the conditional probability is multiplied by $(1-\epsilon)$ while the upper bound remains the same.

3.6.2 Toggle Switch

The next case study is the *toggle switch* [GCC00, LLBB06], a gene regulatory system that has even been implemented *in-vivo*.

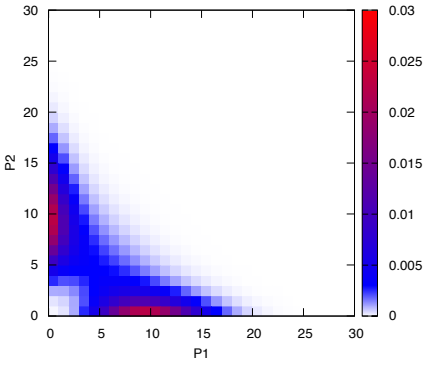
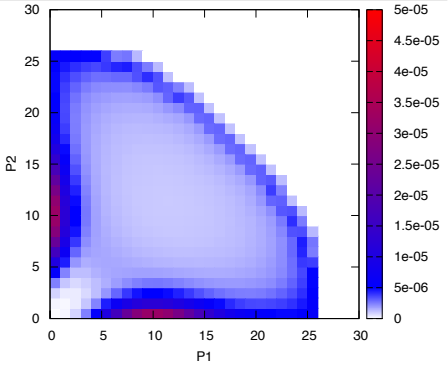
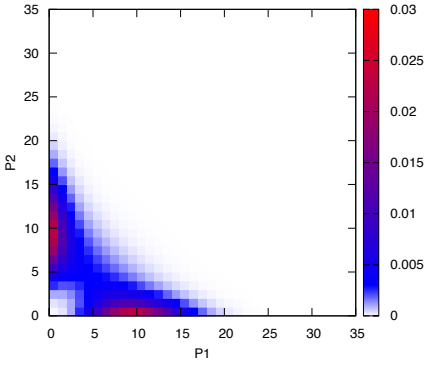
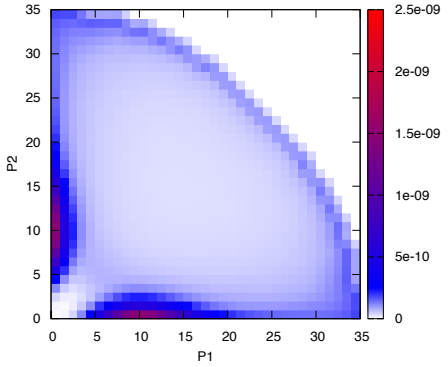
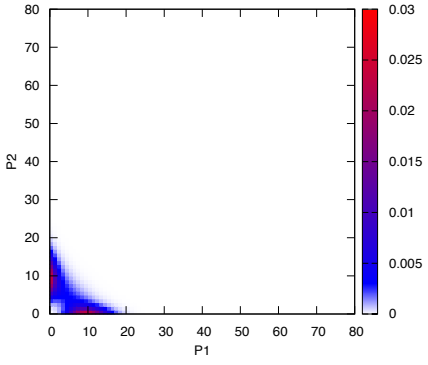
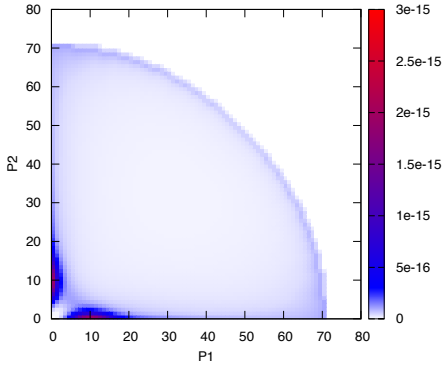
ϵ	$\min_{\mathbf{y}\in\mathcal{B}} \pi_{\mathbf{x}}^{(\mathbf{y})}$	$\max_{\mathbf{y}\in\mathcal{B}} \pi_{\mathbf{x}}^{(\mathbf{y})} - \min_{\mathbf{y}\in\mathcal{B}} \pi_{\mathbf{x}}^{(\mathbf{y})}$
0.1		
0.05		
0.01		

Table 3.2: State-wise lower bounds of the conditional equilibrium distribution of the exclusive switch (left) and the difference between state-wise lower and upper bounds (right) for $\epsilon \in \{0.01, 0.05, 0.1\}$.

ϵ	$\lvert\mathcal{C}\rvert$	$\lvert\mathcal{B}\rvert$	Time (s) $\mathcal{C}$,$\mathcal{B}$	Time (s) Bounds	Memory (MB)	Δ
1.0e-1	7,964	292	1.5	29.0	227.3	1.9e-3
5.0e-2	14,016	384	1.9	65.8	433.2	9.7e-4

Table 3.3: Numerical results (geometric bounds) for the toggle switch model.

▷Model 4

Model 4: Toggle Switch

The toggle switch is a gene regulatory network that is similar to the exclusive switch but instead of a single promotor it has two promotor regions. If the first promotor $\overline{R_B}(1)$ is free, the production of protein A (3) at rate ρ_A is enabled. The other promotor $\overline{R_A}(2)$ controls the production of protein $B(4)$ at rate ρ_B. Both proteins can repress the expression of the other protein by binding to the promotor of the corresponding gene at rates β_A and β_B, respectively. In the repressed state the promotors are represented by the species R_A and R_B. Proteins bound to the promotor can unbind at rates ν_A and ν_B, respectively, and protein molecules degrade over time at rates δ_A and δ_B, respectively.

$$
\begin{aligned}
\overline{R_B} &\xrightarrow{\rho_A} \overline{R_B} + A, & A &\xrightarrow{\delta_A} \emptyset,\\
\overline{R_A} + A &\xrightarrow{\beta_A} R_A, & R_A &\xrightarrow{\nu_A} \overline{R_A} + A,\\
\overline{R_A} &\xrightarrow{\rho_B} \overline{R_A} + B, & B &\xrightarrow{\delta_B} \emptyset,\\
\overline{R_B} + B &\xrightarrow{\beta_B} R_B, & R_B &\xrightarrow{\nu_B} \overline{R_B} + B
\end{aligned}
$$

We choose parameters $\rho_A = \rho_B = 6.0, \delta_A = \delta_B = 0.4, \beta_A = \beta_B = 1.0$, and $\nu_A = \nu_B = 0.5$. The reachable state space is $\mathcal{S} = \mathbb{B}^2 \times \mathbb{N}^2$ for any initial condition with $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{B}$. We state the numerical results in Table 3.3 and the bounds are plotted in Table 3.4. The global maximum drift $c = 124.225$ was found at $\mathbf{x}_3 = \mathbf{x}_4 = 8.375$ where both promotor regions are free.

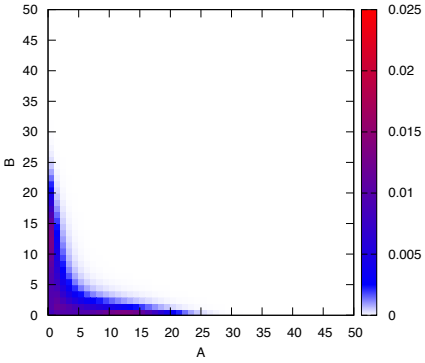
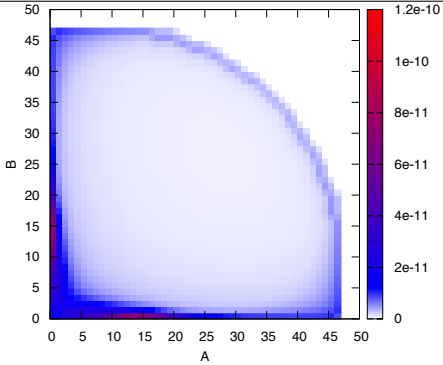
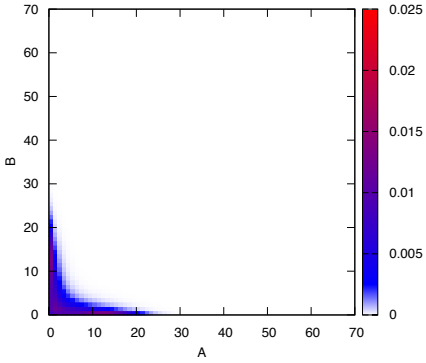
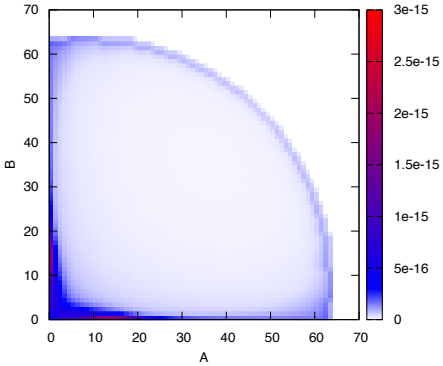
ϵ	$\min_{\mathbf{y}\in\mathcal{B}} \pi_{\mathbf{x}}^{(\mathbf{y})}$	$\max_{\mathbf{y}\in\mathcal{B}} \pi_{\mathbf{x}}^{(\mathbf{y})} - \min_{\mathbf{y}\in\mathcal{B}} \pi_{\mathbf{x}}^{(\mathbf{y})}$
0.1		
0.05		

Table 3.4: State-wise lower bounds of the conditional equilibrium distribution of the toggle switch (left) and the difference between state-wise lower and upper bounds (right) for $\epsilon \in \{0.05, 0.1\}$.

For this system we could verify tristability, that is, in addition to the two regions with a high number of protein molecules of one of the two species, we could identify another peak in the equilibrium distribution in the region with a low number of protein molecules. In [LLBB06], the authors explain that peak by a deadlock situation caused by both promotor regions being bound simultaneously.

3.6.3 Protein Synthesis

The following model is a simple protein synthesis model as described in [GP98] as a stochastic Petri net (SPN). Often, MPMs and SPNs (without zero-arcs) can trivially be encoded within each other.

Model 5: Protein Synthesis ▷Model 5

The *protein synthesis* model describes the transcription of a single gene $G(1)$ that switches between being active (represented by G) and inactive (represented by $\overline{G}$). If the gene is active, protein molecules $P(2)$ are produced which degrade over time.

$$\overline{G} \xrightarrow{\lambda} G \qquad G \xrightarrow{\mu} \overline{G} \qquad G \xrightarrow{\nu} G + P \qquad P \xrightarrow{\delta} \emptyset$$

Note that in the transition class description, we encode both, the activity and the inactivity of the gene, by the variable $\mathbf{x}_1$ since $\mathbf{x}_1 = 0$ if the gene is inactive.

$$\alpha^{(1)} = \lambda \cdot (1 - \mathbf{x}_1) \quad \mathbf{v}^{(1)} = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

$$\alpha^{(2)} = \mu \cdot \mathbf{x}_1 \quad \mathbf{v}^{(2)} = \begin{bmatrix} -1 & 0 \end{bmatrix}$$

$$\alpha^{(3)} = \nu \cdot \mathbf{x}_1 \quad \mathbf{v}^{(3)} = \begin{bmatrix} 0 & 1 \end{bmatrix}$$

$$\alpha^{(4)} = \delta \cdot \mathbf{x}_2 \quad \mathbf{v}^{(4)} = \begin{bmatrix} 0 & -1 \end{bmatrix}$$

We use the same parameters as in [GP98], that is, $\lambda = 1.0, \mu = 5.0, \nu = 1.0, \delta = 0.02$. The reachable state space is $\mathcal{S} = \mathbb{B} \times \mathbb{N}$ for any initial condition with $\mathbf{x}_1 \in \mathbb{B}$ and we summarize our results in Table 3.5. The global maximum drift $c = 21.5025$ was found at $\mathbf{x}_2 = 25.25$ when the gene is active. We plot the lower bounds on the conditional equilibrium for $\epsilon = 0.1$ in Figure 3.6. Note that we do not plot the difference between lower and upper bounds since Δ is lower than the double floating point precision of 1.0e–15. Due to the same reason, we do not plot the bounds for smaller values of ϵ.

ϵ	$\lvert\mathcal{C}\rvert$	$\lvert\mathcal{B}\rvert$	Time (s)		Memory (MB)	Δ
			$\mathcal{C}$,$\mathcal{B}$	Bounds		
1.0e–1	198	2	0.1	0.7	21.5	$<$ 1.0e–15
5.0e–2	258	2	0.1	0.7	24.6	$<$ 1.0e–15
1.0e–2	516	2	0.1	0.7	27.7	$<$ 1.0e–15
1.0e–3	1,518	2	0.4	1.7	36.9	$<$ 1.0e–15
1.0e–4	4,688	2	0.9	3.5	58.4	$<$ 1.0e–15
1.0e–5	14,716	2	1.0	5.4	61.4	$<$ 1.0e–15
1.0e–6	46,422	2	1.0	7.8	150.5	$<$ 1.0e–15

Table 3.5: Numerical results (geometric bounds) for the protein synthesis model.

3.6.4 Gene Expression

Our next case study is the gene expression model described in Model 1 with the parameter set $\rho = 100.0$, $\tau = 0.01$, $\delta_M = 0.2$ and $\delta_P = 0.02$ and reachable state space $\mathbb{N}^2$ for any initial condition. The parameters are chosen such that the *attracting regions* are located far from the origin. This implies that the state space must be truncated from "above" and additionally from "below" because otherwise the number of states in the set $\mathcal{C}$ becomes intractably large. In the examples presented so far, the set $\mathcal{C}$ always included the origin. The reason is that all states "below" an attractor have positive drift if the squared Euclidean norm is used. If we take as Lyapunov function the distance to attracting regions, it is possible to derive a set $\mathcal{C}$ that does not include the origin. The reason is

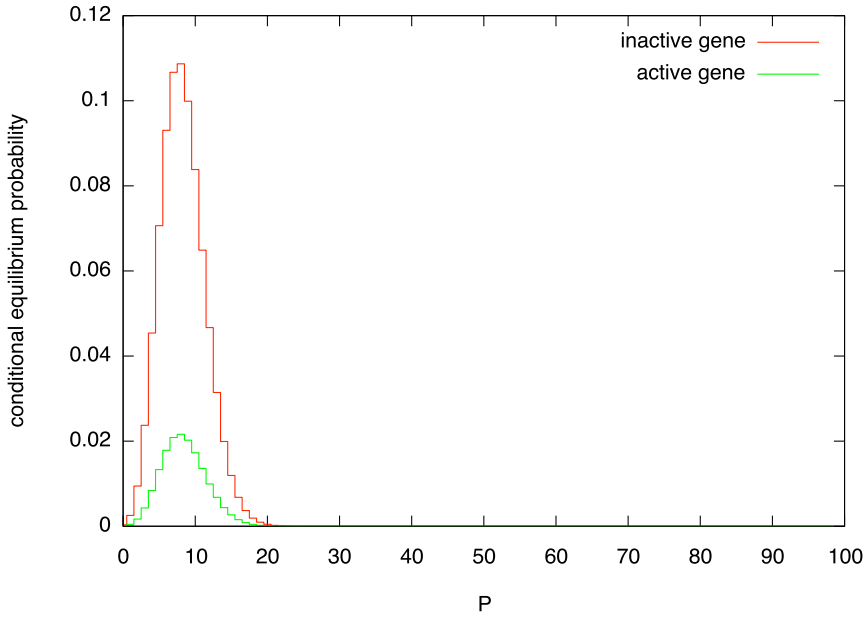


Figure 3.6: State-wise lower bounds of the conditional equilibrium distribution of the protein synthesis for $\epsilon = 0.1$.

that a state that is located far "below" an attractor may have a negative drift. In general, it is difficult to guess attracting regions. We refer to Chapter 5 for a more detailed discussion of attracting regions. One possibility to locate them is to find points where the multi-dimensional drift function

$$\widetilde{d}(\mathbf{x}) = \frac{d}{dt}\mathbf{Exp}\left[\mathcal{X}(t) \mid \mathcal{X}(t) = \mathbf{x}\right]$$

becomes zero. Then, another Lyapunov function is used to bound the equilibrium probability, namely, the function that measures the distance to these points. Thus, we set $\widetilde{d}(\mathbf{x})$ to zero and get

$$\rho - \delta_M \cdot \mathbf{x}_1 = 0 \qquad \tau \cdot \mathbf{x}_1 - \delta_P \cdot \mathbf{x}_2 = 0,$$

which has the unique solution $(\widetilde{\mathbf{x}}_1, \widetilde{\mathbf{x}}_2) = (\frac{\rho}{\delta_M}, \frac{\rho \cdot \tau}{\delta_M \cdot \delta_P})$. Next, we consider the Lyapunov function

$$g(\mathbf{x}) = a \cdot (\mathbf{x}_1 - \widetilde{\mathbf{x}}_1)^2 + b \cdot (\mathbf{x}_2 - \widetilde{\mathbf{x}}_2)^2,$$

that is, the weighted squared distance to the possible attractor $(\widetilde{\mathbf{x}}_1, \widetilde{\mathbf{x}}_2)$ computed before. We tested different weights and found that the choice $a = 1.0, b = 20.0$ provides good numerical results which we list in Table 3.6. Figure 3.7 shows the corresponding geometric bounds for $\epsilon = 0.1$ and Table 3.7 illustrates the state-wise bounds for $\epsilon = 0.1$ and $\epsilon = 0.05$. Note that for $a = b = 1.0$ and $\epsilon = 0.1$ the set $\mathcal{C}$ contains 52,316 states (vs. 23,770 for $a = 1.0, b = 20.0$) and the results (not shown) are slightly more accurate. This indicates that the choice of the Lyapunov function is particularly important if the attracting regions must be bounded from below as well.

ϵ	$\lvert\mathcal{C}\rvert$	$\lvert\mathcal{B}\rvert$	Time (s)		Memory (MB)	Δ
			$\mathcal{C}$,$\mathcal{B}$	Bounds		
1.0e-1	23,770	515	0.7	19.0	605.2	1.4e-4
5.0e-2	47,517	728	1.3	90.1	866.3	2.2e-5

Table 3.6: Numerical results (geometric bounds) for the gene expression model.

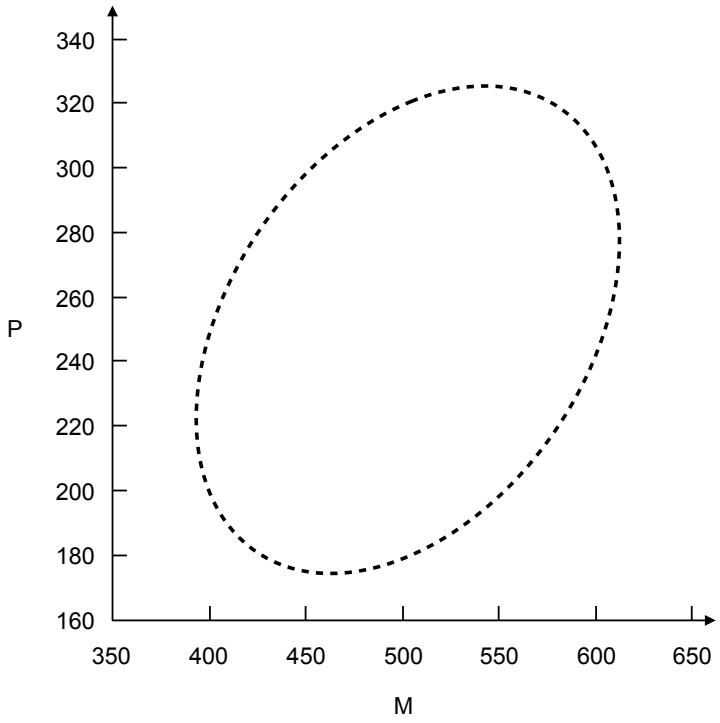


Figure 3.7: A set $\mathcal{C}$ (dashed line) that contains at least 90% of the steady state probability mass for the gene expression model.

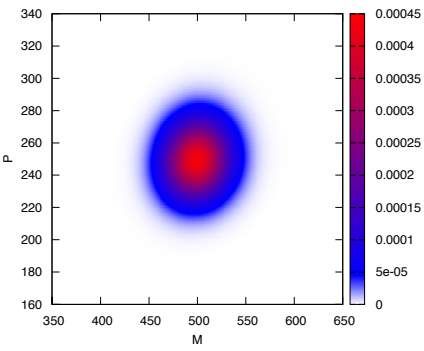
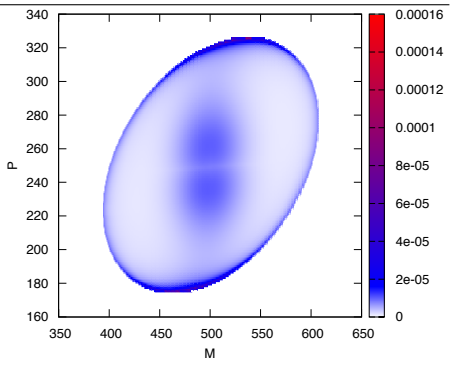
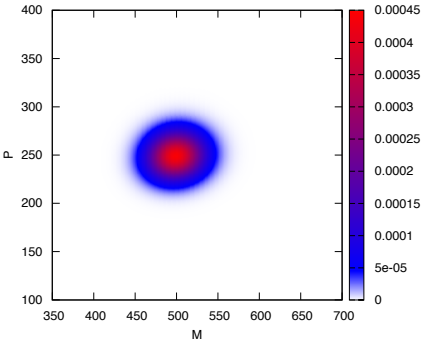
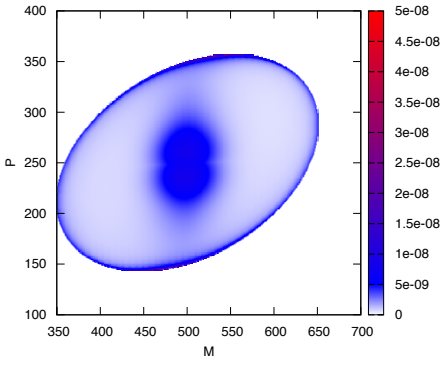
ϵ	$\min_{\mathbf{y}\in\mathcal{B}} \pi_{\mathbf{x}}^{(\mathbf{y})}$	$\max_{\mathbf{y}\in\mathcal{B}} \pi_{\mathbf{x}}^{(\mathbf{y})} - \min_{\mathbf{y}\in\mathcal{B}} \pi_{\mathbf{x}}^{(\mathbf{y})}$
0.1		
0.05		

Table 3.7: State-wise lower bounds of the conditional equilibrium distribution of the gene expression model (left) and the difference between state-wise lower and upper bounds (right) for $\epsilon \in \{0.05, 0.1\}$.

3.7 Extensions

Several models need extensions to the theory and methods presented so far, be it more expressible propensity functions or more flexible structural possibilities. In the following section, we will show two of those examples.

3.7.1 Rational Propensity Functions

As already described in Chapter 2.3.2, the propensity functions of ordinary chemical reactions are polynomial functions. However, when for example analyzing systems where reactions exhibit different time-scales, that is, propensities of different magnitudes, it might be beneficial to apply the *quasi-steady-state* assumption originating from [BH25], where one chemical species is assumed to stagnate on the time-scale of the other. In the end, it turns out that the amount of one species can be expressed by a *rational function* involving the amount of one or multiple other species. This greatly helps to reduce the complexity of the analysis since the dimensionality is decreased.

We want to illustrate this at the example of the toggle switch from Chapter 3.6.2. Usually, the binding-unbinding processes are magnitudes faster than the DNA transcription/translation and protein degradation processes [LLBB06]. Thus, when assuming the binding-unbinding to be in equilibrium with respect to the other processes, the toggle switch can be formulated in Michaelis-Menten form as summarized in Model 6.

Model 6: Toggle Switch – Michaelis-Menten Form ▷Model 6

The Michaelis-Menten form of the toggle switch consists of the two protein species $A(1)$ and $B(2)$ as well as the four chemical reactions

$$\varnothing \xrightarrow{\frac{\rho_A}{1+B^{n_A}}} A, \qquad A \xrightarrow{\delta_A} \varnothing,$$

$$\varnothing \xrightarrow{\frac{\rho_B}{1+A^{n_B}}} B, \qquad B \xrightarrow{\delta_B} \varnothing$$

with protein production rates ρ_A and ρ_B, protein degradation rates δ_A and δ_B and repression coefficients n_A and n_B.

In this model, the mutual repression of the two protein species becomes apparent when looking at reactions 1 and 3, where the total production rate of one protein species is inversely proportional to a power (the repression coefficient) of the other species. Further, the propensity functions

$$\alpha^{(1)}(\mathbf{x}) = \frac{\rho_A}{1+\mathbf{x}_2^{n_A}} \text{ and } \alpha^{(3)}(\mathbf{x}) = \frac{\rho_B}{1+\mathbf{x}_1^{n_B}}$$

of these two reactions are not polynomials as in the previous sections but *rational functions*. Now, when we choose the Lyapunov function

$$g(\mathbf{x}) = \mathbf{x}_1^2 + \mathbf{x}_2^2$$

and the parameter set $\rho_A = 60.0$, $\rho_B = 30.0$, $n_A = 3$, $n_B = 2$, and $\delta_A = \delta_B = 1.0$, we get the drift function

$$\begin{aligned} d(\mathbf{x}) &= -2 \cdot \left(\mathbf{x}_1^2 + \mathbf{x}_2^2\right) + \left(\frac{120}{1+\mathbf{x}_2^3} + 1\right) \cdot \mathbf{x}_1 + \left(\frac{60}{1+\mathbf{x}_1^2} + 1\right) \cdot \mathbf{x}_2 \\ &+ \frac{120}{1+\mathbf{x}_2^3} + \frac{60}{1+\mathbf{x}_1^2}. \end{aligned}$$

We observe that also the drift $d(\mathbf{x})$ became a rational function. Nevertheless, the denominators do not become zero for any $\mathbf{x} \in {\mathbb{R}_0^+}^2$ and thus (i) $d(\mathbf{x}) < \infty$ for all $\mathbf{x} \in \mathbb{N}^2$. In order to analyze the limiting behavior of the drift function, we change the coordinate system to polar coordinates, that is, we define the line equation $l_\beta : \mathbb{R}_0^+ \to {\mathbb{R}_0^+}^2$ in the positive quadrant by

$$l_\beta(t) = \begin{bmatrix} t \cdot \sin(\beta) & t \cdot \cos(\beta) \end{bmatrix}$$

for $\beta \in [0, \pi]$ and get

$$d_\beta(t) = d(l_\beta(t)) = -2 \cdot t^2 + U_\beta \cdot t + V_\beta$$

with

$$U_\beta = \left[\frac{120}{1+t^3 \cdot \cos^3(\beta)} + 1\right] \cdot \sin(\beta) + \left[\frac{60}{1+t^2 \cdot \sin^2(\beta)} + 1\right] \cdot \cos(\beta)$$

and

$$V_\beta = \frac{120}{1+t^3 \cdot \cos^3(\beta)} + \frac{60}{1+t^2 \cdot \sin^2(\beta)}.$$

We can derive the limit

$$\lim_{t\to\infty} d_\beta(t) = -\infty$$

for every direction $\beta \in [0, \pi]$ since $d_\beta(t)$ is clearly dominated by the $-2 \cdot t^2$ term. Therefore, there is a $\gamma \in \mathbb{R}^+$ such that for each $\beta \in [0, \pi]$ there is a $t_\beta \in \mathbb{R}_0^+$ such that $d_\beta(t) < -\gamma$ for all $t > t_\beta$. Thus, (ii) the set of all $\mathbf{x} \in \mathbb{N}^2$ with $d(\mathbf{x}) \geq -\gamma$ is finite. Finally, since the MPM is irreducible, that is, the reachable state space is $\mathcal{S} = \mathbb{N}^2$ (independent of the initial distribution) and every state is reachable from any other state as well as sub-results (i) and (ii), the MPM of the toggle switch is *ergodic* according to Theorem 7.

Next, we are interested in an upper bound $\infty > c \geq \sup_{\mathbf{x}\in\mathbb{N}^2} d(\mathbf{x})$ of the drift function. Consequently, we determine the partial derivatives of the drift as

$$\frac{\partial}{\partial \mathbf{x}_1} d(\mathbf{x}) = -4 \cdot \mathbf{x}_1 + \frac{120}{1 + \mathbf{x}_2^3} - \frac{60 \cdot \mathbf{x}_1 \cdot (2 \cdot \mathbf{x}_2 + 1)}{(1 + \mathbf{x}_1^2)^2} + 1 \text{ and}$$

$$\frac{\partial}{\partial \mathbf{x}_2} d(\mathbf{x}) = -4 \cdot \mathbf{x}_2 + \frac{60}{1 + \mathbf{x}_1^2} - \frac{180 \cdot \mathbf{x}_2^2 \cdot (2 \cdot \mathbf{x}_1 + 1)}{(1 + \mathbf{x}_2^3)^2} + 1.$$

Setting the gradient to zero in order to find local extrema, that is, setting

$$\nabla d(\mathbf{x}) = \left[\frac{\partial}{\partial \mathbf{x}_1} d(\mathbf{x}) \quad \frac{\partial}{\partial \mathbf{x}_2} d(\mathbf{x})\right] = \mathbf{0}$$

boils down to solving

$$\frac{f_1(\mathbf{x})}{(1 + \mathbf{x}_2^3) \cdot (1 + \mathbf{x}_1^2)^2} = 0 \quad \text{and} \quad \frac{f_2(\mathbf{x})}{(1 + \mathbf{x}_1^2) \cdot (1 + \mathbf{x}_2^3)^2} = 0$$

with

$$\begin{aligned} f_1(\mathbf{x}) &= (-4 \cdot \mathbf{x}_1 + 1) \cdot (1 + \mathbf{x}_2^3) \cdot (1 + \mathbf{x}_1^2)^2 \\ &+ 120 \cdot (1 + \mathbf{x}_1^2)^2 - 100 \cdot \mathbf{x}_1 \cdot (2 \cdot \mathbf{x}_2 + 1) \cdot (1 - \mathbf{x}_2^3) \end{aligned}$$

and

$$\begin{aligned} f_2(\mathbf{x}) &= (-4 \cdot \mathbf{x}_2 + 1) \cdot (1 + \mathbf{x}_1^2) \cdot (1 + \mathbf{x}_2^3)^2 \\ &+ 100 \cdot (1 + \mathbf{x}_2^3)^2 - 180 \cdot \mathbf{x}_2^2 \cdot (2 \cdot \mathbf{x}_1 + 1) \cdot (1 - \mathbf{x}_1^2) \end{aligned}$$

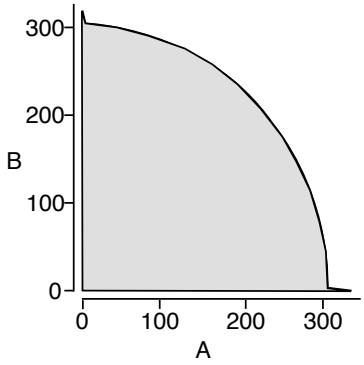


Figure 3.8: Geometric bounds for the Michaelis-Menten form of the toggle switch model for $\epsilon = 0.01$. The set $\mathcal{C}$ contains all integer grid points in the grey region.

where each fraction has been multiplied by the denominators of the other fractions in order to establish a common denominator. These equations are satisfied if the numerators become zero, that is, if

$$f_1(\mathbf{x}) = 0 \text{ and } f_2(\mathbf{x}) = 0.$$

From here, we again use the *homotopy continuation method* as implemented in the HOM4PS-2.0 tool [LLT08] to compute all solutions. In addition, we analyze the drift at the borders where $\mathbf{x}_1$ respectively $\mathbf{x}_2$ is zero and at the origin $\mathbf{0}$. The maximal drift is $c \approx 1.890165e+3$ and we over-approximate the global maximum by setting $c = 1891$. If we choose a moderate ϵ of 0.01, we get a set $\mathcal{C}$ as illustrated in Figure 3.8. Consequently, inside that region $\mathcal{C}$, most, that is at least 99%, of the steady state probability mass must be located.

We conclude that the geometric bounding technique presented in this chapter is also applicable when the propensity functions are rational functions as they appear in Michaelis-Menten approximations or in models that exhibit Hill-functions [Hil10] in general. More precisely, if the drift $d(\mathbf{x})$ can be written as

$$d(\mathbf{x}) = \sum_{i=1}^{n} \frac{p_i(\mathbf{x})}{q_i(\mathbf{x})}$$

with $p_i(\mathbf{x}), q_i(\mathbf{x}) \in \mathbb{P}[\mathbf{x}]$ and $q_i(\mathbf{x}) > 0$ for all $\mathbf{x} \in \mathbb{R}_0^{+N}$, re-parameterization of $d(\mathbf{x})$ via $\mathbf{x} = t \cdot \overline{\mathbf{x}}$ with direction vector $\overline{\mathbf{x}} \in [0,1]^N$ and distance value $t \in \mathbb{R}_0^+$ can be used to reason about the limit behavior. However, note that parameter t might also appear in denominators which makes the argumentation harder. Further, equating each partial derivative $\frac{\partial}{\partial j} d(\mathbf{x})$ with zero, that is

$$\frac{\partial}{\partial \mathbf{x}_j} d(\mathbf{x}) = \sum_{i=1}^{n} \frac{\frac{\partial}{\partial \mathbf{x}_j} p_i(\mathbf{x}) \cdot q_i(\mathbf{x}) - p_i(\mathbf{x}) \cdot \frac{\partial}{\partial \mathbf{x}_j} q_i(\mathbf{x})}{(q_i(\mathbf{x}))^2} = 0$$

becomes

$$\sum_{i=1}^{n} \prod_{k \neq i} (q_k(\mathbf{x}))^2 \cdot \left(\frac{\partial}{\partial \mathbf{x}_j} p_i(\mathbf{x}) \cdot q_i(\mathbf{x}) - p_i(\mathbf{x}) \cdot \frac{\partial}{\partial \mathbf{x}_j} q_i(\mathbf{x}) \right) = 0$$

which is a polynomial. Consequently, we can apply the methods presented in Chapter 3.5. Again, note that the problem becomes harder from a computational perspective when compared to the polynomial drift case since in the worst case, the degree of the polynomial equation system grows with the doubled degrees of the denominator polynomials.

3.7.2 Guards

A further extension to the standard modeling formalism are *guards*, that is, predicates over states. Guards annotate transition classes such that a transition is only enabled if the corresponding guard evaluates to true.

Definition 23: Guard

A *guard* $\beta(\mathbf{x})$ with $\mathbf{x} = \begin{bmatrix} \mathbf{x}_1 & \dots & \mathbf{x}_N \end{bmatrix}$ is a boolean expression over $\mathbf{x}_1, \dots, \mathbf{x}_N$.

▷Definition 23

▷Definition 24

Definition 24: Guarded Transition Class

A *guarded transition class* (GTC) for N population types is a tuple $(\beta, \alpha, \mathbf{v})$, where $\beta(\mathbf{x})$ is a *guard*, $\alpha : \mathbb{N}^N \to \mathbb{R}_0^+$ is the *propensity function*, and $\mathbf{v} \in \mathbb{Z}^N \setminus \{\mathbf{0}\}$ is the *change vector*.

▷Definition 25

Definition 25: GTC Induced MPM

A set $\{(\beta^{(r)}, \alpha^{(r)}, \mathbf{v}^{(r)}\}_{1 \le r \le R}$ of R guarded transition classes for N population types determines the entries of the infinitesimal generator of an MPM $(\mathcal{S}, \mathbf{Q}, \pi(0))$ with $\mathcal{S} \subseteq \mathbb{N}^N$ via

$$\mathbf{Q}_{\mathbf{xy}} = \begin{cases} \sum_{\{r \mid \beta^{(r)}(\mathbf{x}) \wedge \mathbf{x}+\mathbf{v}^{(r)}=\mathbf{y}\}} \alpha^{(r)}(\mathbf{x}) & \text{if } \mathbf{x} \neq \mathbf{y} \text{ and} \\ -\sum_{\mathbf{z} \neq \mathbf{x}} \mathbf{Q}_{\mathbf{xz}} & \text{otherwise.} \end{cases}$$

We want to exemplify the use of guards using the following performance model of a Bit-Torrent-like peer-to-peer network.

▷Model 7

Model 7: Bit-Torrent-Like Peer-To-Peer Network

We study a peer-to-peer network primarily used for file sharing which consists of two population types, that is, downloaders $x(1)$ and uploaders $y(2)$. The guarded transition classes are

$$\begin{aligned} \alpha^{(1)}(\mathbf{x}) &= \lambda, & \mathbf{v}^{(1)} &= \begin{bmatrix} 1 & 0 \end{bmatrix}, \\ \alpha^{(2)}(\mathbf{x}) &= \theta \cdot \mathbf{x}_1, & \mathbf{v}^{(2)} &= \begin{bmatrix} -1 & 0 \end{bmatrix}, \\ \alpha^{(3)}(\mathbf{x}) &= c \cdot \mathbf{x}_1, & \mathbf{v}^{(3)} &= \begin{bmatrix} -1 & 1 \end{bmatrix}, \\ \alpha^{(4)}(\mathbf{x}) &= \mu \cdot (\eta \cdot \mathbf{x}_1 + \mathbf{x}_2), & \mathbf{v}^{(4)} &= \begin{bmatrix} -1 & 1 \end{bmatrix}, \\ \alpha^{(5)}(\mathbf{x}) &= \gamma \cdot \mathbf{x}_2, & \mathbf{v}^{(5)} &= \begin{bmatrix} 0 & -1 \end{bmatrix}, \end{aligned}$$

with the guards

$$\begin{aligned}
\beta^{(1)}(\mathbf{x}) &= \text{true} \\
\beta^{(2)}(\mathbf{x}) &= \text{true} \\
\beta^{(3)}(\mathbf{x}) &= c \cdot \mathbf{x}_1 \leq \mu \cdot (\eta \cdot \mathbf{x}_1 + \mathbf{x}_2) \\
\beta^{(4)}(\mathbf{x}) &= c \cdot \mathbf{x}_1 > \mu \cdot (\eta \cdot \mathbf{x}_1 + \mathbf{x}_2) \\
\beta^{(5)}(\mathbf{x}) &= \text{true}
\end{aligned}$$

where the first GTC models new requests at a rate of λ, the second GTC represents the impatience of peers that abort at rate θ, and the fifth GTC describes the process of seeds y leaving the system at a rate of γ each. The interesting ones are the third and fourth GTCs, where peers x that are serviced are transformed into seeds y. The rate of that process is $\mu \cdot (\eta \cdot \mathbf{x}_1 + \mathbf{x}_2)$ constrained by the total downloading bandwidth $\mathbf{c} \cdot \mathbf{x}_1$, where c (μ) is the downloading (uploading) bandwidth of a peer and η is the effectiveness of the file sharing. The reachable state space of this model (for positive rates) is $\mathcal{S} = \mathbb{N}^2$. We refer to [QS04] for details about the model.

When using guarded transition classes, the drift function becomes a *piecewise-defined* function with the partition of the domain being induced by the validity of the guard expressions. For example, if we choose the Lyapunov function

$$g(\mathbf{x}) = \mathbf{x}_1^2 + \mathbf{x}_2^2 + \mathbf{x}_1 \cdot \mathbf{x}_2,$$

we retrieve the drift function $d(\mathbf{x})$ defined as

$$d(x) = \begin{cases} d_1(\mathbf{x}) & \text{if } c \cdot \mathbf{x}_1 \leq \mu \cdot (\eta \cdot \mathbf{x}_1 + \mathbf{x}_2) \text{ and} \\ d_2(\mathbf{x}) & \text{otherwise.} \end{cases}$$

with

$$\begin{aligned} d_1(\mathbf{x}) = &-(2\cdot\theta + c)\cdot\mathbf{x}_1^2 - 2\cdot\gamma\cdot\mathbf{x}_2^2 - (\theta+\gamma-c)\cdot\mathbf{x}_1\cdot\mathbf{x}_2 \\ &+ (2\cdot\lambda + c + \theta)\cdot\mathbf{x}_1 + (\lambda+\gamma)\cdot\mathbf{x}_2 + \lambda, \text{ and} \end{aligned}$$

$$\begin{aligned} d_2(\mathbf{x}) = &-(2\cdot\theta + \mu\cdot\eta)\cdot\mathbf{x}_1^2 - (2\cdot\gamma-\mu)\cdot\mathbf{x}_2^2 \\ &- (\theta+\gamma+[1-\eta]\cdot\mu)\cdot\mathbf{x}_1\cdot\mathbf{x}_2 \\ &+ (2\cdot\lambda+\theta+\mu\cdot\eta)\cdot\mathbf{x}_1 + (\lambda+\gamma+\mu)\cdot\mathbf{x}_2 + \lambda. \end{aligned}$$

Choosing the parameter set

$$\eta = 1, \mu = 0.00125, c = 0.002, \theta = 0.001, \gamma = 0.001, \lambda = 0.04$$

as in [QS04] yields

$$d_1(\mathbf{x}) = -0.004\cdot\mathbf{x}_1^2 - 0.002\cdot\mathbf{x}_2^2 + 0.083\cdot\mathbf{x}_1 + 0.041\cdot\mathbf{x}_2 + 0.04, \text{ and}$$

$$\begin{aligned} d_2(\mathbf{x}) = &-0.00325\cdot\mathbf{x}_1^2 - 0.00075\cdot\mathbf{x}_2^2 - 0.002\cdot\mathbf{x}_1\cdot\mathbf{x}_2 \\ &+ 0.08225\cdot\mathbf{x}_1 + 0.04225\cdot\mathbf{x}_2 + 0.04. \end{aligned}$$

The global maximum of $d_1(\mathbf{x})$ and $d_2(\mathbf{x})$ is approximatively 0.6806875 respectively 0.7226304. Consequently, we scale the drift $d(\mathbf{x})$ by $\epsilon \cdot 0.7226304^{-1}$ by effectively scaling $d_1(\mathbf{x})$ and $d_2(\mathbf{x})$ yielding $d_1^*(\mathbf{x})$ and $d_2^*(\mathbf{x})$. Finally, we retrieve the geometric bounds as illustrated in Figure 3.9 (left) for $\epsilon = 0.01$. Note the bend at around $\begin{bmatrix}30 & 17.5\end{bmatrix}$ which is the discontinuity introduced by switching between $d_1(\mathbf{x})$ and $d_2(\mathbf{x})$. Figure 3.9 (right) illustrates the conditional geometric bounds induced by $d_1^*(\mathbf{x})$ and $d_2^*(\mathbf{x})$ as well as the line $c\cdot\mathbf{x}_1 = \mu\cdot(\eta\cdot\mathbf{x}_1 + \mathbf{x}_2)$ that separates the two regions of different guard validity.

Consequently, given a set $\{(\beta^{(r)}, \alpha^{(r)}, \mathbf{v}^{(r)}\}_{1\le r\le R}$ of R guarded transition classes in general, the first task is to determine the partitioning of the state space

$$\mathcal{S} = \dot{\bigcup}_{\mathbf{b}\in\mathbb{B}^N} \mathcal{P}_{\mathbf{b}}$$

with

$$\mathcal{P}_{\mathbf{b}} = \{\mathbf{x}\in\mathcal{S} \mid \beta^{(i)}(\mathbf{x}) \text{ evaluates to } \mathbf{b}_i\}$$

which is induced by the validity of the guards. Each region $\mathcal{P}_{\mathbf{b}}$ has a specific drift function $d_{\mathbf{b}}(\mathbf{x})$ that incorporates exactly those GTCs

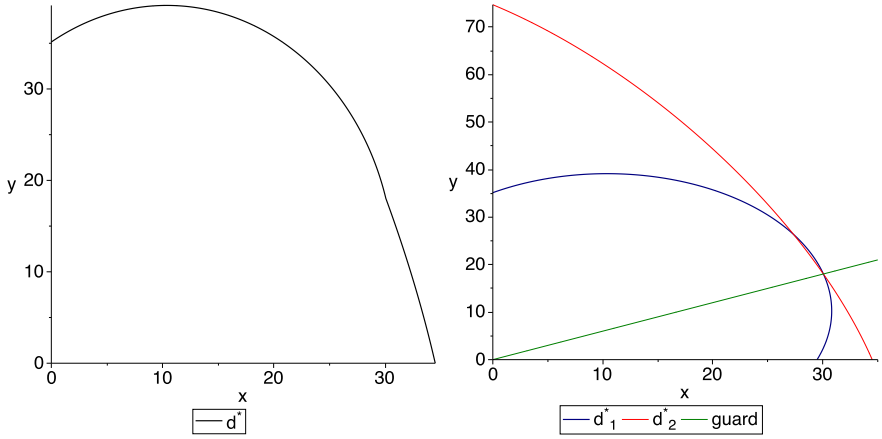


Figure 3.9: Geometric bounds (left) and conditional geometric bounds (right) of the peer-to-peer model for $\epsilon = 0.01$ as well as the line $c \cdot \mathbf{x}_1 = \mu \cdot (\eta \cdot \mathbf{x}_1 + \mathbf{x}_2)$.

whose guard evaluates to true inside the respective region. The global maximum drift can be found by examining all drift functions $d_{\mathbf{b}}(\mathbf{x})$ inside their respective regions as well as on their borders. Note that in the worst case, 2^R regions have to be examined since in principle every combination where each of the R guards might evaluate to true or false can manifest as a region. On the other hand, regions might also collapse due to overlapping guard validities as it is the case in the peer-to-peer example. Moreover, the expressivity added by allowing general boolean expressions for guards makes the problem of (dis-)proving ergodicity undecidable as stated in the following theorem.

▷Theorem 15

Theorem 15: Undecidability of Ergodicity of GTCMPMs

The problem of determining whether an MPM induced by a set of guarded transition classes $\{(\beta^{(r)}, \alpha^{(r)}, \mathbf{v}^{(r)}\}_{1 \leq r \leq R}$ is ergodic is *undecidable*.

Proof: We will show that sets of guarded transition classes allow us to simulate any *Random Access Machine* (RAM) [SS63] which is a Turing powerful formalism. A RAM consists of a finite set of registers $r_1, \ldots, r_n$ which can be assigned any natural number as well as of a finite set of instructions $(1 : I_1), \ldots, (m : I_m)$. The state of a RAM is defined by a tuple $(i, \mathbf{c})$ where i is the index of the next instruction to be executed and $\mathbf{c} \in \mathbb{N}^n$ contains the current values of the registers $r_1, \ldots, r_n$. In fact, only two instruction types are needed to be able to simulate recursive functions [Min67]. Assuming the current state is $(i, \mathbf{c})$ and the next instruction is

- $(i : \mathrm{INC}(r_j))$, the next state will be $(i + 1, \mathbf{c} + \mathbf{e}_j)$ and if it is
- $(i : \mathrm{DECJ}(r_j, s))$, the successor state is $(i+1, \mathbf{c}-\mathbf{e}_j)$ if $\mathbf{c}_j > 0$ and $(s, \mathbf{c})$ otherwise,

for $j \in \{1, \ldots, n\}$. If a state $(i, \mathbf{c})$ with $i > m$ is reached, the machine halts in that state. We model a given RAM by an MPM with state space $\mathcal{S} \subseteq \mathbb{N}^{n+1}$ and add for each instruction

- $(i : \mathrm{INC}(r_j))$, the guarded transition classes $(\mathbf{x}_{n+1} = i, \alpha, \mathbf{e}_j + \mathbf{e}_{n+1})$ and $(\mathbf{x}_{n+1} = i + 1, \alpha, -\mathbf{e}_j - \mathbf{e}_{n+1})$
- $(i : \mathrm{DECJ}(r_j, s))$, the guarded transition classes $(\mathbf{x}_{n+1} = i \wedge \mathbf{x}_j > 0, \alpha, -\mathbf{e}_j + \mathbf{e}_{n+1})$, $(\mathbf{x}_{n+1} = i \wedge \mathbf{x}_j = 0, \alpha, (s - i) \cdot \mathbf{e}_{n+1})$, $(\mathbf{x}_{n+1} = i + 1, \alpha, \mathbf{e}_j - \mathbf{e}_{n+1})$, and $(\mathbf{x}_{n+1} = i \wedge \mathbf{x}_j = 0, \alpha, (i - s) \cdot \mathbf{e}_{n+1})$.

where $\alpha(\mathbf{x}) = 1$. We also add the guarded transition class $(\mathbf{x}_{n+1} > m, \alpha, \mathbf{e}_{n+1})$. First of all, we observe that already the problem of computing the set of reachable states assuming initial condition $\pi_{\mathbf{e}_{n+1}}(0) = 1$ is not decidable, since states $\mathbf{x} \in \mathbb{N}^{n+1}$ with $\mathbf{x}_{n+1} > m$ will be reached if and only if the RAM halts. Further, note that for each instruction

$i \in \{1, \ldots, m\}$, we added guarded transition classes in both directions, that is, from one state to its successor state and back. But as soon as any halting state $\mathbf{x}$ with $\mathbf{x}_{n+1} = i > m$ is reached, only states $\mathbf{x}'$ with $\mathbf{x}'_{n+1} > i$ are reachable. This scheme ensures that the MPM is irreducible if and only if the RAM does not halt. Consequently, the problem of determining irreducibility and thus ergodicity is undecidable. □

We want to remark that for simplicity, we decided to proof the undecidability result via irreducibility. In fact, we could also have proven it via positive-recurrence, by ensuring the behavior corresponding to instructions $I_1, \ldots, I_m$ to be ergodic and connecting a null-recurrent process like an M/M/1 queue with a larger arrival than service rate to the halting states.

Theorem 16: Undecidability of Ergodicity for MPMs ▷**Theorem 16**

Theorem 15 also implies the undecidability of ergodicity for MPMs that are induced by ordinary transition classes with propensity functions that are expressible enough to test whether a component of the state vector is zero (either directly or in any combination).

This result tells us that there is no algorithm that can fully automatically check whether any MPM is ergodic or not. In our case that means that there is no algorithm that can synthesize a Lyapunov function for any ergodic MPM or state non-ergodicity otherwise. However, since our models are usually based on polynomial propensity functions induced by chemical reaction networks and we anyway use a semi-automatic proof technique, where we choose a Lyapunov function to test beforehand, we are not directly affected by this result.

CHAPTER 4

Model Checking Markov Population Models

In the context of continuous-time Markov chains, properties of interest can be specified using the *continuous stochastic logic* (CSL) [ASSB00, BHHK03]. CSL is a *branching-time* temporal logic inspired by the *computation tree logic* (CTL) [EC82]. It allows to reason about properties of states via atomic propositions that are associated with a state, like the number of certain molecules given in this state. But also properties about which sets of states can be reached at all or within certain time bounds as well as long run properties can be specified. Because the underlying semantics of a Markov population model is given as a CTMC, we can also use CSL to reason about properties of such models, when interpreting CSL formulae on the CTMC semantics. Using CSL, we can express many important measures for biological models, like switching times as presented later.

We consider the complete set of CSL formulae specified in [BHHK03], including the steady-state operator and in certain cases also the unbounded until operator but excluding the next-operator in order to keep the presentation simple. Note that the next operator can be added easily but requires a *rate matrix* formulation of the CTMC that allows explicit self-loops [BHHK03]. The resulting logic can express nested probabilistic properties such as *"the long-run probability is at least 0.4*

that we reach Ψ-states along Φ-states within the time interval [6.5,8.5] with a probability larger than 0.98*"* via

$$\mathsf{S}_{\geq 0.4}(\mathsf{P}_{>0.98}(\Phi\ \mathsf{U}^{[6.5,8.5]}\ \Psi)).$$

In the work [HHWZ09b], the authors have extended results on truncation based analysis of infinite state CTMCs similar to the techniques described in Chapter 2.4.3 such that they were able to do approximate model checking for a subset of CSL. This subset excluded the steady-state operator as well as the unbounded until operator. The respective techniques have been implemented in the model checker Infamy [HHWZ09a]. In this chapter, we combine Geobound and Infamy, such that we can also handle the CSL steady-state operator for Markov population models with an infinite state space. Using a ternary logic [KKLW07, Kli10] allows us to compute safe lower and upper bounds for the respective probabilities. In turn, we can decide exactly whether a certain formula holds, does not hold or whether this cannot be decided on the chosen finite truncation of the current model. We show the applicability of the approach on a number of biological models. The results of this chapter have been published in [SHZ14].

4.1 Ternary-Logic Formulation of CSL

▷**Definition 26**

Definition 26: Ternary Logic

We consider a *ternary logic* [KKLW07, Kli10] with values $\mathbb{B}_3 := \{\top, \bot, ?\}$ which forms a complete lattice with the ordering $\bot < ? < \top$, $\mathbb{B}_3$. We interpret $\sqcap$ as the meet ("and" operator), and $\cdot^c$ as the complement ("not") operation, with the usual semantics. Other operators like $\sqcup$ ("or" operator) can be derived easily. Then, $\top$ and $\bot$ can be interpreted as the values *definitely true* and *definitely false* respectively. The symbol ? denotes the *unknown value*. The respective truth tables of the operators are stated below.

$\sqcup$	$\bot$	?	$\top$
$\bot$	$\bot$	?	$\top$
?	?	?	$\top$
$\top$	$\top$	$\top$	$\top$

$\sqcap$	$\bot$	?	$\top$
$\bot$	$\bot$	$\bot$	$\bot$
?	$\bot$	?	?
$\top$	$\bot$	?	$\top$

	$.^c$
$\bot$	$\top$
?	?
$\top$	$\bot$

Consider a formula over a number of values some of which are ?. If the value of this formula is different from ?, we know that when inserting $\bot$ or $\top$ instead of some of these values, the result would still be the same. This way, in some cases we can obtain a truth value even though we do not known the truth values of some formula parts.

When we want to check whether a CTMC satisfies a property, we need to define what *atomic propositions* from a given set $\mathcal{AP}$ hold in a state. We therefore demand that a CTMC is annotated by a *ternary labeling function*.

Definition 27: Ternary Labeling Function ▷Definition 27

A *ternary labeling function* for state space $\mathcal{S}$ and a set of atomic propositions $\mathcal{AP}$ is a function $L : (\mathcal{S} \times \mathcal{AP}) \to \mathbb{B}_3$.

If an MPM with N population types is the subject of analysis, we implicitly demand that each state state $\mathbf{x} \in \mathcal{S}$ is labeled with all boolean expressions over $\mathbf{x}_i$ and the respective truth values.

Example 9: Ternary Labeling of an MPM ▷Example 9

For example, given an MPM with two population types and a state $\mathbf{x} = \begin{bmatrix} 3 & 4 \end{bmatrix}$, we have

- $L(\mathbf{x}, \mathbf{x}_1 \leq z) = \top$ for all $z \geq 3$,
- $L(\mathbf{x}, \mathbf{x}_1 + \mathbf{x}_2 \leq 7) = \top$,
- $L(\mathbf{x}, \mathbf{x}_1^2 + \mathbf{x}_2^2 < 30) = \top$, and
- $L(\mathbf{x}, \mathbf{x}_1 + \mathbf{x}_2 > 7) = \bot$,

to name just a few.

We consider the logic CSL [BHHK03] *without* the next-operator interpreted over the ternary logic of Definition 26.

▷Definition 28

Definition 28: Syntax of CSL

Let $I = [t, t']$ be an interval with $t, t' \in \mathbb{R}_0^+ \cup \{\infty\}$, $t' = \infty \Rightarrow t = 0$, and $t \leq t'$. Let $p \in [0, 1]$ and $\bowtie \in \{\leq, <, >, \geq\}$. The syntax of state formulae (Φ) and path formulae (ϕ) is:

$$\begin{aligned} \Phi &= \mathit{true} \mid a \in \mathcal{AP} \mid \neg\Phi \mid \Phi \wedge \Phi \mid \mathsf{P}_{\bowtie p}(\phi) \mid \mathsf{S}_{\bowtie p}(\Phi), \\ \phi &= \Phi \,\mathsf{U}^I\, \Phi. \end{aligned}$$

Let $\mathbb{F}$ be the set of all CSL formulae. The truth value $[\![\cdot]\!] : ((\mathcal{S} \cup \mathit{Path}) \times \mathbb{F}) \to \mathbb{B}_3$ of formulae is defined inductively as in the following definition.

▷Definition 29

Definition 29: Ternary Semantics of CSL

The ternary semantics of CSL is defined recursively by the mapping depicted in Figure 4.1.

In order to simplify presentation, we demand the existence of an *initial state* $s_{init} \in \mathcal{S}$, that is, $\pi_{s_{init}}(0) = 1$. This way, a CTMC satisfies a formula Φ if the initial state s_{init} does. Further, we specify two derived operators, the *unbounded until*

$$\Phi_1 \,\mathsf{U}\, \Phi_2 := \Phi_1 \,\mathsf{U}^{[0,\infty)}\, \Phi_2$$

and the *eventually operator*

$$\Diamond^I \Phi := \mathit{true} \,\mathsf{U}^I\, \Phi.$$

Moreover, let $\mathcal{A} \subseteq \mathcal{S}$ be a set of states. For better readability, we use the name of the set $\mathcal{A}$ as an atomic proposition in formulae to characterize that the system is in a state contained in $\mathcal{A}$.

$$
\begin{array}{lcl}
[\![s, true]\!] & := & \top \\
[\![s, a]\!] & := & L(s,a) \\
[\![s, \Phi_1 \wedge \Phi_2]\!] & := & [\![s,\Phi_1]\!] \sqcap [\![s,\Phi_2]\!] \\
[\![s, \neg\Phi]\!] & := & [\![s,\Phi]\!]^c \\
[\![s, \mathsf{P}_{\bowtie p}(\phi)]\!] & := & \begin{cases} \top & \text{if } \mathbf{Pr}_s\{\sigma|[\![\sigma,\phi]\!]{=}\top\}{\bowtie}p \text{ and} \\ & \mathbf{Pr}_s\{\sigma|[\![\sigma,\phi]\!]{\neq}\bot\}{\bowtie}p, \\ \bot & \text{if } \mathbf{Pr}_s\{\sigma|[\![\sigma,\phi]\!]{=}\top\}{\not\bowtie}p \text{ and} \\ & \mathbf{Pr}_s\{\sigma|[\![\sigma,\phi]\!]{\neq}\bot\}{\not\bowtie}p, \\ ? & \text{otherwise,} \end{cases} \\
[\![s, \mathsf{S}_{\bowtie p}(\Phi)]\!] & := & \begin{cases} \top & \text{if } S_L(\Phi){\bowtie}p \wedge S_U(\Phi){\bowtie}p, \\ \bot & \text{if } S_L(\Phi){\not\bowtie}p \wedge S_U(\Phi){\not\bowtie}p, \\ ? & \text{otherwise, where} \end{cases} \\
S_L(\Phi) & := & \lim\limits_{t\to\infty} \mathbf{Pr}_s\{\sigma \mid [\![\sigma@t,\Phi]\!] = \top\} \\
S_U(\Phi) & := & \lim\limits_{t\to\infty} \mathbf{Pr}_s\{\sigma \mid [\![\sigma@t,\Phi]\!] \neq \bot\} \\
[\![\sigma, \Phi_1 \,\mathsf{U}^I\, \Phi_2]\!] & := & \begin{cases} \top & \text{if } \exists t{\in}I.[\![\sigma@t,\Phi_2]\!]{=}\top \text{ and} \\ & \forall t'{<}t.[\![\sigma@t',\Phi_1]\!]{=}\top, \\ \bot & \text{if } \forall t{\in}I.[\![\sigma@t,\Phi_2]\!]{=}\bot \text{ or} \\ & \exists t'{<}t.[\![\sigma@t',\Phi_1]\!]{=}\bot, \\ ? & \text{otherwise.} \end{cases}
\end{array}
$$

Figure 4.1: Ternary semantics of CSL.

4.2 Model Checking CSL Based on Truncation

Model checking CSL formulae without the steady state operator S on a finite CTMC over a ternary logic has already been used before to handle a different abstraction technique for CTMCs [KKLW07, Kli10].

Now we discuss how to model check CSL formulae on infinite CTMCs. To this end, our goal is to operate on a finite truncation instead of the original infinite CTMC. In a nutshell, starting from the initial state, we explore the states of the infinite model until we can safely assume that we have explored enough of them to obtain a result in the following steps that can be used to decide the formula. Then, we remove all transitions from the states at the border and set all of their atomic propositions to ?. In [HHWZ09b] the authors already discussed some variants of such a model exploration. In the following, we give another such technique, which is more efficient, because it needs to explore less states.

4.2.1 Truncation-Based Reachability Analysis

We define the set of successor states $\mathrm{post}(\mathcal{A})$ of a set $\mathcal{A} \subseteq \mathcal{S}$ as the set

$$\mathrm{post}(\mathcal{A}) = \{s \in \mathcal{S} \mid \exists s' \in \mathcal{A}.\mathbf{Q}_{s's} > 0\}.$$

This way, given a CTMC model, a subset of the state space $\mathcal{C}_0$, and a CSL formula of the form

$$\Phi = \mathsf{P}_{\bowtie p}(\Phi_1 \,\mathsf{U}^I\, \Phi_2),$$

we want to compute a *finite truncation* sufficient to evaluate $[\![s, \Phi]\!]$, that is, whether formula Φ holds in state s, for all states $s \in \mathcal{C}_0$.

▷Definition 30

Definition 30: Finite Truncation

Let $\mathscr{C} = (\mathcal{S}, \mathbf{Q}, \pi(0))$ be a CTMC with initial state s_{init} and labeling L, $\mathcal{C} \subseteq \mathcal{S}$ with $s_{init} \in \mathcal{C}$ a *finite* subset of $\mathcal{S}$, and let $\dot{\mathcal{C}} = \mathrm{post}(\mathcal{C}) \setminus \mathcal{C}$. The *finite truncation* of $\mathcal{C}$ is the finite state CTMC $\mathscr{C}|_{\mathcal{C}} = (\mathcal{C} \cup \dot{\mathcal{C}}, \mathbf{Q}^{\mathcal{C}}, \pi(0))$ with labeling $L^{\mathcal{C}}$ where for all $a \in \mathcal{AP}$, $L^{\mathcal{C}}(s, a) = L(s, a)$ if $s \in \mathcal{C}$ and $L^{\mathcal{C}}(s, a) =$? otherwise.

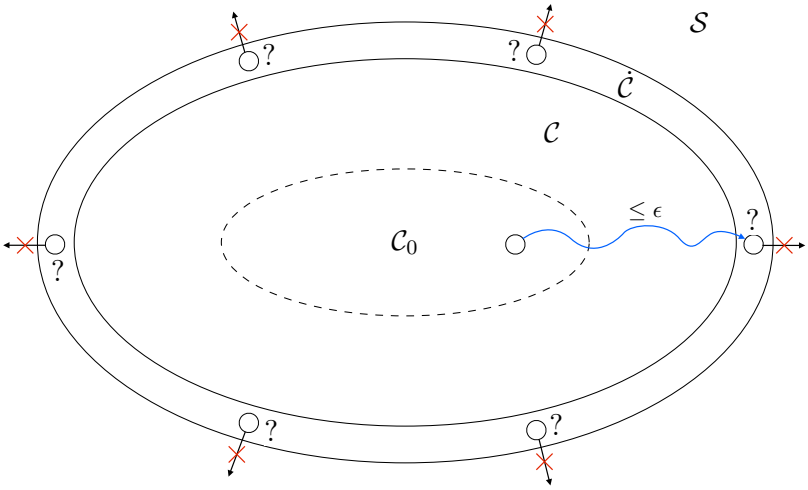


Figure 4.2: Illustration of the relationship between sets $\mathcal{S}$, $\mathcal{C}$, $\mathcal{C}_0$, and $\dot{\mathcal{C}}$.

The infinitesimal generator is defined by $\mathbf{Q}^{\mathcal{C}}_{ss'} = \mathbf{Q}_{ss'}$ if $s \in \mathcal{C}$ and $\mathbf{Q}^{\mathcal{C}}_{ss'} = 0$ otherwise as well as $\mathbf{Q}^{\mathcal{C}}_{ss} = -\sum_{s' \neq s} \mathbf{Q}^{\mathcal{C}}_{ss'}$.

Intuitively, $\mathcal{C}_0$ is the set of all states for which we want to decide the given formula and set $\mathcal{C}$ is the total set of states needed in order to achieve that goal. Figure 4.2 illustrates the relationship between the respective sets. We build the truncation of a model iteratively, using a high-level description of the model as for example given by *transition classes*. We explore the model until for all $s \in \mathcal{C}_0$ the probability to reach states in $\dot{\mathcal{C}}$ is below an accuracy ϵ (blue curve in Figure 4.2), which we may choose as a fixed value or due to the probability bound p.

Algorithm 6 describes how we can obtain a sufficiently large state set $\mathcal{C}$. For $s \in \mathcal{S}$, $s' \in \dot{\mathcal{C}}$ and $t \in \mathbb{R}^+_0 \cup \{\infty\}$ we use $\pi^{(s)}_{s'}(t)$ to denote the probability that at time $t \in \mathbb{R}^+_0$, the CTMC is in state s', under the condition that it was in state s initially. For $t = \infty$, we let π denote the limit for $t \to \infty$. As s' is absorbing, this value exists. Further, for a set of absorbing states $\mathcal{A}$, we let $\xi(s, t, \mathcal{A}) := \sum_{s' \in \mathcal{A}} \pi^{(s)}_{s'}(t)$ denote the

probability to reach set $\mathcal{A}$ within time $t \in \mathbb{R}_0^+ \cup \{\infty\}$. Given a fixed s and t, we can compute $\pi_{s'}^{(s)}(t)$ for all s' at once effectively, and given $\mathcal{A}$ and t we can compute $\xi(s, t, \mathcal{A})$ for all s at once effectively [BHHK03]. We use the notation $\xi_{\mathscr{C}[\mathcal{D}]}(s, t, \mathcal{A})$ to denote that the respective quantity is computed on the CTMC $\mathscr{C}$, where states in $\mathcal{D} \subseteq \mathcal{S}$ are made absorbing.

The algorithm is started on a CTMC $\mathscr{C}$ and a set of states $\mathcal{C}_0$, for which we want to decide a given property expressed as a CSL formula. We also provide the time bound t as well as the accuracy ϵ. With $\widetilde{\mathcal{C}}$ we denote a set of states for which the exploration algorithm may stop immediately, as further exploration is not needed to decide the given property. For $\Phi = \mathsf{P}_{\bowtie p}(\Phi_1 \mathsf{U}^{[0,t]} \Phi_2)$ with $t \in \mathbb{R}_0^+ \cup \{\infty\}$, we could specify $\widetilde{\mathcal{C}}$ as the states which fulfill $\Phi_2 \vee (\neg\Phi_1 \wedge \neg\Phi_2)$.

Algorithm 6 treachability$(\mathscr{C}, \mathcal{C}_0, \widetilde{\mathcal{C}}, t, \epsilon)$

$\mathcal{C} := \mathcal{C}_0$
$\dot{\mathcal{C}} := \mathrm{post}(\mathcal{C}) \setminus \mathcal{C}$
while $\max_{s \in \mathcal{C}_0} \xi_{\mathscr{C}[\dot{\mathcal{C}}]}(s, t, \dot{\mathcal{C}} \setminus \widetilde{\mathcal{C}}) \geq \epsilon$ **do**
    choose s from $\arg\max_{s \in \mathcal{C}_0} \xi_{\mathscr{C}[\dot{\mathcal{C}}]}(s, t, \dot{\mathcal{C}} \setminus \widetilde{\mathcal{C}})$
    while $\xi_{\mathscr{C}[\dot{\mathcal{C}}]}(s, t, \dot{\mathcal{C}} \setminus \widetilde{\mathcal{C}}) \geq \epsilon$ **do**
        choose $\mathcal{A} \subseteq (\dot{\mathcal{C}} \setminus \tilde{\mathcal{C}})$ such that $\xi_{\mathscr{C}[\dot{\mathcal{C}}]}(s, t, \mathcal{A}) \geq \epsilon$
        $\mathcal{C} := \mathcal{C} \cup \mathcal{A}$
        $\dot{\mathcal{C}} := \mathrm{post}(\mathcal{C}) \setminus \mathcal{C}$
    end while
end while
return $\mathcal{C} \cup (\mathrm{post}(\mathcal{C}) \cap \widetilde{\mathcal{C}})$

Let n be the number of states in the final state set $\mathcal{C}$. In the worst case, the state space resembles a linear structure and we will need n time-bounded reachability computations of length in order to arrive at the final set $\mathcal{C}$. Thus, the time complexity is $\mathcal{O}(u \cdot n^2 \cdot t)$ and the space complexity is $\mathcal{O}(n)$ assuming a sparse model as given by an MPM induced by transition classes.

4.2.2 Truncation-Based Steady-State Analysis

We use the state-wise bounds that we retrieve as described in Chapter 3.4 for a finite subset of the state space as computed in Chapter 3.1.

4.2.3 Combining Truncation-Based Analysis Methods for CSL Model Checking

Given a CTMC, we want to check whether it satisfies Φ. This is done in two phases. At first, we construct a finite truncation that is sufficient to check the formula. To this end, we employ an algorithm to determine suitable CTMC truncations. The states explored depend on the specific CSL formula analyzed. The computation works by recursive descent into sub-formulae. The most intricate formulae are the probabilistic operators. After the exploration, we can compute $[\![s_{init}, \Phi]\!]$ on the finite truncation.

Algorithm 7 explore$(\mathscr{C}, \mathcal{C}, \Phi, \epsilon)$

```
1:  switch Φ do
2:      case true:
3:          return C
4:      case a ∈ AP:
5:          return C
6:      case ¬Ψ:
7:          return explore(C, C, Ψ, ε)
8:      case Φ1 ∧ Φ2:
9:          return explore(C, C, Φ1, ε) ∪ explore(C, C, Φ2, ε)
10:     case P_{⋈p}(Φ1 U^{[t,t']} Φ2):
11:         C_t = treachability(C, C, stop(Φ), t, ε)
12:         C_{t'} = treachability(C, C_t, stop(Φ), t' − t, ε)
13:         return explore(C, C_{t'}, Φ1, ε) ∪ explore(C, C_{t'}, Φ2, ε)
14:     case S_{⋈p}(Ψ):
15:         return explore(C, post(C) \ C, Ψ, ε)
```

Algorithm 7 describes the exploration component and returns the finite set of states needed for deciding the specified formula. Given a CTMC $\mathscr{C}$ and a state formula Φ, we call explore$(\mathscr{C}, \{s_{init}\}, \Phi, \epsilon)$ for

a chosen $\epsilon \in (0,1)$. Afterwards, we can use the CSL model checking algorithm for a ternary logic on the model obtained this way. With $\text{stop}(\Phi)$ we denote a set of states for which we can stop the exploration immediately. For nested formulae, this value is computed by a simple precomputation in a recursive manner.

We assume that we have already obtained the steady-state probabilities beforehand. Thus, to obtain the lower bound probabilities of $\mathsf{S}_{\bowtie p}(\Psi)$, we sum up the lower bound steady-state probabilities of states $s \in \mathcal{C}$ with $[\![s, \Psi]\!] = \top$. For the upper probability bound, we sum upper steady-state probabilities of states s with $[\![x, \Psi]\!] \neq \bot$ and add the probability ϵ that limits the steady-state probability outside $\mathcal{C}$. The probabilities computed are the probabilities for all states, because the model is ergodic.

Correctness: Consider a truncation $\mathscr{C}_{\mathcal{C}}$ constructed for the CTMC $\mathscr{C}$ and a state formula Φ. If we obtain the truth value $[\![s, \Phi]\!] \neq ?$ in $\mathscr{C}_{\mathcal{C}}$, then this is also the truth value in $\mathscr{C}$. The correctness is independent of the exploration algorithm, which plays a role for performance and applicability of the approach. If too many states are explored, we may run out of memory, but if too few are explored, we are unable to decide the value in the original model. As a result, the correctness of the algorithm for CSL without steady state follows by giving a simulation relation [Kli10, Definition 3.4.2] between $\mathscr{C}$ and $\mathscr{C}_{\mathcal{C}}$ and [Kli10, Theorem 4.5.2]. The correctness of the steady-state extension follows as we give safe upper and lower bounds in exactly the same way as it is done in [Kli10, Theorem 4.5.2].

Complexity: The space and time complexity of the presented model checking algorithm is that of standard CSL model checking algorithms as for example in [BHHK03] assuming the truncated state space has been computed using the methods of Chapters 4.2.1 and 4.2.2.

4.3 Case Studies

Using several case studies, we assess the effectiveness of our technique. For that, we have combined the tool Geobound [Spi13b] to compute

bounds on steady-state probabilities for Markov population models with the infinite-state model checker Infamy [HHWZ09a] to form the tool chain *GeoInf* [HS13]. To show the efficiency of the approach, we applied GeoInf on a number of models from systems biology. The results were obtained on an Ubuntu 10.04 machine with an Intel dual-core processor at 2.66 GHz equipped with 3 GB of RAM. Instead of the truth value for the formula under considerations, in the result tables we give intervals of the probability measure of the outer formula. We also make use of a derived operator for conditional steady-state measures defined as

$$[\![s, \mathsf{S}_{\bowtie p}(\Phi_1 \mid \Phi_2)]\!] := \begin{cases} \top, & S_L(\Phi_1|\Phi_2) \bowtie p \wedge S_U(\Phi_1|\Phi_2) \bowtie p, \\ \bot, & S_L(\Phi_1|\Phi_2) \not\bowtie p \wedge S_U(\Phi_1|\Phi_2) \not\bowtie p, \\ ? & \text{else}, \end{cases}$$

where

$$\begin{aligned} S_L(\Phi_1 \mid \Phi_2) &:= \frac{S_L(\Phi_1 \wedge \Phi_2)}{S_U(\Phi_2)}, \\ S_U(\Phi_1 \mid \Phi_2) &:= \frac{S_U(\Phi_1 \wedge \Phi_2)}{S_L(\Phi_2)}. \end{aligned}$$

4.3.1 Protein Synthesis

We analyze the MPM encoding protein synthesis, as given in Model 5, with the parameter set

$$\lambda = 1.0, \mu = 5.0, \nu = 1.0, \delta = 0.02.$$

We consider the property that on the long run, given that there are more than 20 proteins, a state with 20 or less proteins is most likely (with a probability of at least 0.9) reached within t time units:

$$\mathsf{S}_{>p}(\mathsf{P}_{>0.9}(\Diamond^{[0,t]} P \leq 20) \mid P > 20).$$

We illustrate this property in Figure 4.3 and give the model checking results in Table 4.1. The shortcut "stop" of Algorithm 7 was not used for the analysis. With "depth" we specify the number of states of the shortest path from the initial state to any other state of the truncation. The run-times of Geobound and Infamy is given in seconds. The rate of

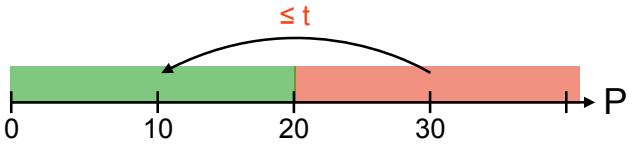


Figure 4.3: Stability property of the protein synthesis model.

decay of proteins depends on the number of proteins existing. For the states on the border of $\mathcal{C}$, we have large rates back to existing states. Because of this, for the given parameters the state space exploration algorithm does not need to explore further states, and the total number n of explored states does not increase with the time bound. To obtain different values of n, we would have needed to choose extremely large time bounds, for which analysis would be on the one hand infeasible and on the other hand would lead to results extremely close to 1. The lower and upper bounds are further away than ϵ. The reason is that we have to divide by $S_U(\Phi_2)$ for the lower and by $S_L(\Phi_2)$ for the upper bound. In turn, this may lead to a much larger error than ϵ as witnessed in the following table. For this model, a choice of $\epsilon = 1e-6$ seems to be appropriate in order to achieve meaningful bounds.

ϵ	t	Depth	Time (s)		n	Probability Bounds
			Geobound	Infamy		$[S_L, S_U]$
	10	8		0.1	217	[0.002, 1.0]
1e–1	20	8	0.9	0.1	217	[0.003, 1.0]
	60	8		0.1	217	[0.004, 1.0]
	10	5		0.3	1531	[0.144, 1.0]
1e–3	20	5	3.2	0.4	1531	[0.259, 0.816]
	60	5		0.7	1531	[0.317, 1.0]
	10	3		34.3	46431	[0.451350, 0.454779]
1e–6	20	3	7.8	67.0	46431	[0.813116, 0.817401]
	60	3		257.5	46431	[0.997642, 1.0]

Table 4.1: Model checking results of the protein synthesis model.

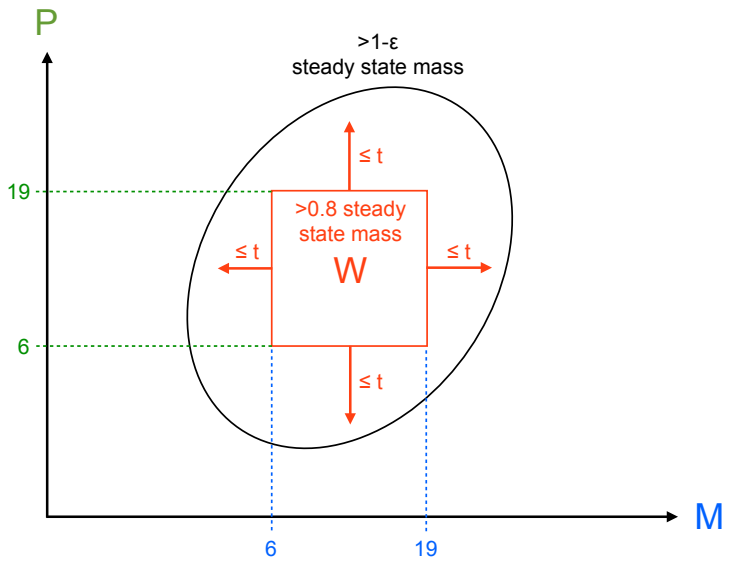


Figure 4.4: CSL property of the gene expression model.

4.3.2 Gene Expression

Next, we analyze the gene expression system formalized in Model 1 with the parameter set

$$\rho = 25.0, \tau = 1.0, \delta_M = 2.0, \delta_P = 1.0.$$

The property of interest is the steady-state probability of leaving a certain set of states W enclosing more than 80% of the steady-state probability mass most likely within t time units, that is,

$$\mathsf{S}_{>p}(\mathsf{P}_{>0.9}(\Diamond^{[0,t]}\neg W) \mid W),$$

where

$$W := M > 5 \wedge M < 20 \wedge P > 5 \wedge P < 20$$

with $\mathsf{S}_{>0.8}(W) = \top$. We illustrate this property in Figure 4.4 and state the model checking results in Table 4.2. Similar to the protein synthesis case study, we see that there is no increase in the number

of states, because the window size already comprises enough states for the transient analysis. Here, a value of $\epsilon = 0.01$ is needed in order to retrieve reasonable bounds.

ϵ	t	Depth	Time (s) Geobound	Time (s) Infamy	n	Probability Bounds $[S_L, S_U]$
0.1	2	24	3.4	5.2	2558	[0.01, 0.2]
	4	24		6.0	2558	[0.3, 0.6]
	8	24		8.5	2558	[0.8, 1.0]
0.05	2	20	6.1	11.9	3663	[0.015, 0.078]
	4	20		15.4	3663	[0.34, 0.46]
	8	20		22.1	3663	[0.90, 1.0]
0.01	2	15	8.5	99.3	11736	[0.015, 0.029]
	4	15		139.9	11736	[0.37, 0.40]
	8	15		219.5	11736	[0.97, 1.0]

Table 4.2: Model checking results of the gene expression model.

4.3.3 Exclusive Switch

The last case study is the exclusive switch as described in Model 3 with the symmetric parameter set

$$\rho_1 = \rho_2 = 0.05, \delta_1 = \delta_2 = 0.005, \beta_1 = \beta_2 = 0.01, \nu_1 = \nu_2 = 0.008.$$

As already depicted in Table 3.2, this system has two attractor regions, that is, two spatial maxima in the steady-state probability distribution over the protein levels, one at $P_1 = 10, P_2 = 0$ and the other one at $P_1 = 0, P_2 = 10$. We are interested in the switching time between these two regions. For this, we estimate the time needed for a 90%-quantile of the steady-state probability mass of one of the two attractors to reach the other attractor region. More precisely, let

$$\begin{aligned} \mathit{start} &:= ||(P_1, P_2) - (10, 0)||_2^2 \leq 4 \\ \mathit{end} &:= ||(P_1, P_2) - (0, 10)||_2^2 \leq 4. \end{aligned}$$

Then, the formula to check is

$$\mathsf{S}_{>p}(\mathsf{P}_{>0.9}(\Diamond^{[0,t]} \mathit{end}) \mid \mathit{start}).$$

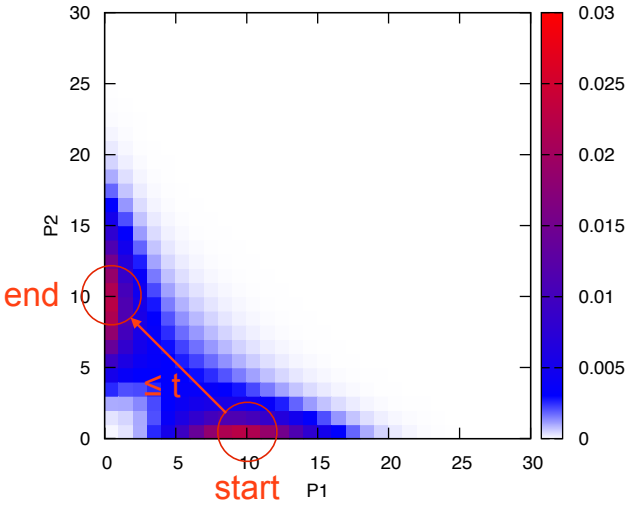


Figure 4.5: CSL property of the exclusive switch model.

We illustrate this formula in Figure 4.5. Note that since the model is symmetric we only have to check one formula, that is, from one attractor to the other. The corresponding results are depicted in Table 4.3. From these results we may conclude that in half of the cases, most likely the switching time between the attractor regions is at most 7800 time units, while in almost all cases the switching time is most likely below 8000 time units, assuming the system has stabilized to a steady state. Again, a value of $\epsilon = 0.01$ is needed in order to retrieve reasonable bounds.

ϵ	t	Depth	Time (s)		n	Probability Bounds
			Geobound	Infamy		$[S_L, S_U]$
0.1	7700	14	5.8	47.9	3414	[0.2, 0.7]
	7800	14		48.0	3414	[0.3, 0.9]
	7900	14		47.6	3414	[0.5, 1.0]
	8000	14		48.1	3414	[0.6, 1.0]
0.05	7700	12	6.9	128.3	4848	[0.26, 0.49]
	7800	12		129.7	4848	[0.43, 0.70]
	7900	12		130.5	4848	[0.64, 0.98]
	8000	12		131.0	4848	[0.83, 1.0]
0.01	7700	8	86.2	1881.6	14806	[0.30, 0.35]
	7800	8		1904.5	14806	[0.50, 0.56]
	7900	8		1930.1	14806	[0.75, 0.82]
	8000	8		1942.9	14806	[0.96, 1.0]

Table 4.3: Model checking results of the exclusive switch model.

CHAPTER 5

Steady State Analysis of Multimodal Markov Chains

For many Markov population models, an efficient computation of the equilibrium distribution is hindered by both a very large or even infinite state space and stiff dynamics. In particular if the distribution is *multimodal*, the main part of the steady state probability mass concentrates on several *attracting regions*. Typically, the locations of such attractors are not known a-priori, but even if their locations are known, the problem is that they are usually separated by low-probability regions which renders the application of standard numerical and simulative methods very difficult. The reason is that switching from one attractor to the other is a rare event and a large number of trajectories is needed to estimate the amount of probability mass of one attractor relative to the mass of the other attractors. Similarly, standard numerical approaches, which are often iterative methods, show slow convergence properties.

In Chapter 3.1, we have seen how to efficiently compute a *finite truncation* $\mathcal{C}$ of the (possibly infinite) state space of a Markov population model. If that truncation is of numerically tractable size, we can directly use the results of Chapter 3.4 to compute state-wise bounds for the equilibrium distribution. However, the problems related to stiffness are still present. For example, the case study that was considered by Milias-Argeitis and Lygeros [MAL11] could not be solved using the tool

PRISM which provides a large number of different numerical methods to obtain the steady state distribution of a Markov chain. Even after a reasonable amount of one million iterations and a relative precision of 1e–15, none of these methods converged. However, since for this example the state space is small (19600 states) it can be solved exactly with direct methods implemented in Matlab using a sparse matrix representation. A three-dimensional extension of that system, the tri-stable toggle switch [LNH+07], containing around 1.5 million states can not be handled by Matlab within a maximum computation time of one day. Here, the number of states corresponds to an appropriate truncation of the infinite state space which is necessary because Matlab (and also PRISM) can only analyze finite systems.

The purpose of this chapter is to develop and analyze a numerical method that approximates the steady state distribution of a large (or infinite) ergodic Markov chain with one or several attractors. We propose an algorithm that overcomes both the problems of largeness of the state space and the problems related to stiffness. Our method provides answers to the following questions:

1. How can we find the attractors of the system, that is, finite regions containing most of the steady state probability?

2. If there is more than one attractor, how can we efficiently compute the accumulated steady state probability of each attracting region?

Note that once the total probabilities of the attractors are known, the probabilities of the individual states can be computed using the approach presented in Chapter 3.4.

To solve problem 1, we use the geometric bounding techniques developed in Chapter 3.1 to first find a region containing all attractors, that is, a single region containing at least 1-ϵ of the total steady state probability mass. In order to separate multiple attractor regions in problem 2, we numerically explore the most likely paths leading from one attractor to the other. This method does *not* explore the whole state space but allows to direct the exploration behavior. Once we have found the attractor regions, we compute their relative strength using a novel approach combining stochastic complementation and inexact matrix-vector multiplication.

We remark that steady state analysis has been widely considered in the area of queuing and computer systems [Ste94] but to the best of the author's knowledge, specific solutions for systems with multimodal distributions as they occur in biological applications have not been developed. A dynamic state space exploration algorithm has been proposed by de Souza e Silva and Ochoa [dSeSO92] for performance and dependability analysis based on finite Markov models. Their method can only be used for the approximation of the steady state probability conditioned on a certain subset. The same applies to Chapter 3.1 where also only a single attractor is considered. In contrast, here we propose a technique to overcome the numerical problems that arise due to the stiffness inherent to multimodal systems, by considering the most likely paths leading from one attractor to any other. In particular, we construct a so called *attractor Markov chain* to obtain the steady state probability of each attractor *relative* to the others. The closest work to ours is that of Milias-Argeitis and Lygeros [MAL11] where the authors consider the same problem but combine stochastic complementation with a *simulative* approach. Here, we focus on numerical solutions and show that they are superior to the simulative approach in [MAL11] both in terms of computation time and accuracy. In addition, we describe how to locate the attractors of the system which was not considered by Milias-Argeitis and Lygeros. The results of this chapter have been published in [SW13].

5.1 Approximation of Attractor Probabilities

In this chapter we propose an approximation algorithm for the efficient computation of the steady state distribution of *multimodal* CTMC, where certain regions of the state space are *attracting regions*, that is, regions corresponding to local maxima of the steady state distribution.

For that, we first concentrate on the problem of approximating the probability of each attractor and assume that the locations of all attractors of the Markov chain are already known. Formally, we partition the state space $\mathcal{S}$ of the Markov chain into n attractors $\mathcal{A}_i$ with $i \in \{1, \ldots, n\}$ and $|\mathcal{A}_i| \in \mathbb{N}$ and the possibly infinite set $\mathcal{I}$ which con-

tains all states that are not part of an attractor, that is,

$$\mathcal{S} = \mathcal{A}_1 \,\dot\cup\, \ldots \,\dot\cup\, \mathcal{A}_n \,\dot\cup\, \mathcal{I}.$$

The goal is to compute the cumulative steady state probabilities p_i with

$$p_i = \sum_{s \in \mathcal{A}_i} \pi_s, \quad \sum_{i=1}^{n} p_i > 1 - \epsilon.$$

We proceed in two steps:

(a) We approximate for each $\mathcal{A}_i$ the conditional steady state distribution $\overline{\pi}^{(i)}$ with

$$\overline{\pi}_s^{(i)} = \frac{\pi_s}{\sum_{s' \in \mathcal{A}_i} \pi_{s'}}$$

for all states $s \in \mathcal{A}_i$.

(b) We approximate the steady state distribution $\widehat{\pi}$ of the aggregated system, that is, the Markov chain that consists of the macro states $\mathcal{A}_1, \ldots, \mathcal{A}_n$, which yields an approximation of $p_1, \ldots, p_n$.

Note that in step (b) we use the fact that ϵ is small and that the probability mass in $\mathcal{I}$ is negligible. Given approximations of p_i and $\overline{\pi}^{(i)}$, the steady state probability of a state $s \in \mathcal{A}_i$ can be approximated as well via

$$\pi_s = p_i \cdot \overline{\pi}_s^{(i)}.$$

Further, we implicitly assume that each Markov chain associated with the (finite) state space $\mathcal{A}_i$ is irreducible such that the corresponding local steady state distribution uniquely exists.

5.1.1 Computation of the Conditional Steady State Distributions of Attractors

For step (a) we remark that the exact conditional distributions $\overline{\pi}^{(i)}$ are the (unique) solutions of

$$\overline{\pi}^{(i)} \cdot \overline{\mathbf{Q}}_{\mathcal{A}_i} = 0 \text{ and } \overline{\pi}^{(i)} \cdot \mathbf{e} = 1$$

where $\overline{\mathbf{Q}}_{\mathcal{A}_i}$ is the stochastic complement of $\mathbf{Q}$ with respect to the set $\mathcal{A}_i$. The problem is that in order to obtain the exact entries of $\overline{\mathbf{Q}}_{\mathcal{A}_i}$,

the whole state space has to be considered (see Definition 22) which is infeasible for large or even infinite Markov chains. Therefore, the behavior outside of $\mathcal{A}_i$ has to be approximated in an appropriate way. By abuse of notation, for a vector $\mathbf{x}$ let $\mathbf{diag}(\mathbf{x})$ denote the diagonal matrix $\mathbf{M}$ with entries $\mathbf{M}_{ij} = \mathbf{x}_i$ if $i = j$ and 0 otherwise. An approximation $\widetilde{\pi}^{(i)} \approx \overline{\pi}^{(i)}$ is given by

$$\widetilde{\pi}^{(i)} \cdot (\mathbf{Q}_{ii} - \mathbf{diag}(\mathbf{Q}_{ii} \cdot \mathbf{e})) = \mathbf{0}, \quad \widetilde{\pi}^{(i)} \cdot \mathbf{e} = 1 \tag{5.1}$$

where $\mathbf{Q}_{ii}$ is the block matrix induced by the partitioning with respect to the attractor sets, that is,

$$\mathbf{Q} = \begin{bmatrix} \mathbf{Q}_{11} & \cdots & \mathbf{Q}_{1n} & \mathbf{Q}_{1\mathcal{I}} \\ \vdots & \ddots & \vdots & \vdots \\ \mathbf{Q}_{n1} & \cdots & \mathbf{Q}_{nn} & \mathbf{Q}_{n\mathcal{I}} \\ \mathbf{Q}_{\mathcal{I}1} & \cdots & \mathbf{Q}_{\mathcal{I}n} & \mathbf{Q}_{\mathcal{I}\mathcal{I}} \end{bmatrix}.$$

Note that the block $\mathbf{Q}_{ij}$ contains all rates of transitions from $\mathcal{A}_i$ to $\mathcal{A}_j$ and that $\mathbf{Q}_{ii} - \mathbf{diag}(\mathbf{Q}_{ii} \cdot \mathbf{e})$ is an infinitesimal generator matrix. Thus, in the corresponding Markov chain the transitions for leaving the set $\mathcal{A}_i$ are redirected to the state from which the set is left. We remark that for solving Equation (5.1), direct methods such as Gaussian elimination can be applied since typically the attractors are of tractable size.

Moreover, we can improve the approximation of the local steady state distribution by considering an enlarged set $\mathcal{A}_i \cup \mathcal{A}'_i$ where $\mathcal{A}'_i$ contains the states reachable from $\mathcal{A}_i$ within a certain number of transitions. For example, in the experimental results shown later, we will choose $\mathcal{A}'_i$ as the set of states reachable from $\mathcal{A}_i$ within 15 steps. Let $\widetilde{\pi}^{(i)}$ denote the obtained approximate distribution on $\mathcal{A}_i \cup \mathcal{A}'_i$. In this case we re-scale $\widetilde{\pi}^{(i)}$ such that it yields a distribution on $\mathcal{A}_i$, that is, for $s \in \mathcal{A}_i$ we use the approximation

$$\overline{\pi}_s^{(i)} \approx \frac{\widetilde{\pi}_s^{(i)}}{\sum_{k \in \mathcal{A}_i} \widetilde{\pi}_k^{(i)}}.$$

5.1.2 Unbounded Reachability Analysis for Markov Population Models

In the following section, we need an efficient way to compute the (unbounded) reachability probability $\mathbf{reach}_{\mathcal{A}}$ to reach some set $\mathcal{A} \subseteq \mathcal{S}$ starting with some initial distribution. The idea is to distribute the probability mass according to the embedded DTMC of the MPM and make states in $\mathcal{A}$ *absorbing*, that is, we ignore all outgoing transitions. Note that we can employ the embedded DTMC (cf. Definition 7 on page 18) since we are not interested in the distribution over time but in the limit. Since again, the algorithm should also be applicable for MPMs with infinite state space, the idea is to use dynamic state space truncation inspired by the transient analysis approach in Algorithm 2. Algorithm 8 provides such a reachability analysis where in the end $\sum_{\mathbf{x}\in\mathcal{A}} p(\mathbf{x})$ approximates $\mathbf{reach}_{\mathcal{A}}$. In detail, after every iteration, states with a probability mass smaller than the threshold δ will be deleted. An iteration step involves the invocation of Algorithm 9, which distributes the current probability distribution along the next-step probabilities encoded in the transition probability matrix of the embedded DTMC. The iterations finally end when there is no significant state left outside of set $\mathcal{A}$, that is, the initial probability mass has either been truncated or absorbed in $\mathcal{A}$. As in Algorithm 2, the total truncation error is accumulated and returned in e.

Algorithm 8 reachability$(\pi(0), \mathbf{Q}, \mathcal{S}, \mathcal{A}, \delta)$

1: $e \leftarrow 0$
2: $p \leftarrow$ new hash map $\mathcal{S} \to [0, 1]$
3: $\forall s$ with $\pi_s(0) > \delta$: $p(s) \leftarrow \pi_s(0)$
4: **while** $\sum_{s\in\mathrm{dom}(p)\setminus\mathcal{A}} p(s) > 0$ **do**
5: $\quad p \leftarrow \mathrm{rstep}(p, \mathbf{Q}, \mathcal{A}, \delta)$
6: $\quad \forall s$ with $p(s) < \delta$: $e \leftarrow e + p(s)$ and remove(p, s)
7: **end while**
8: **return** $[p, e]$

Algorithm 9 rstep$(p, \mathbf{Q}, \mathcal{A})$

for $s \in \mathrm{dom}(p) \setminus \mathcal{A}$ **do**
  for s' with $\mathbf{Q}_{ss'} > 0$ **do**
    $p(s') \leftarrow p(s') - p(s) \cdot \mathbf{Q}_{ss'} \cdot \mathbf{Q}_{ss}^{-1}$
  end for
end for

Note that each step in Algorithm 9 already uses the results of the previous step eliminating the need to store the previous probability distribution in a separate hash map and speeding up convergence. This is justified, since we are not interested in the distribution over time but only the limit. Algorithm 8 can also be used to approximate the reachability of several BSCCs $\mathcal{B}_1, \ldots, \mathcal{B}_n \subseteq \mathcal{S}$ at once. For that, the algorithm needs to be called with $\mathcal{A} = \mathcal{B}_1 \cup \cdots \cup \mathcal{B}_n$ which will yield approximations $\sum_{\mathbf{x} \in \mathcal{B}_i} p(\mathbf{x})$ for the individual $\mathbf{reach}_{\mathcal{B}_i}$ quantities. This is justified since BSCCs can not be left.

5.1.3 Computation of the Steady State Distribution of the Attractor Markov Chain

As discussed in Section 5.1, we first assume the attracting regions $\mathcal{A}_i$ to be given and will show a method to compute them later on. For step (b) we consider the embedded matrix with respect to the partitioning induced by the attractor regions, that is,

$$\mathbf{E} = \begin{bmatrix} \mathbf{E}_{11} & \ldots & \mathbf{E}_{1n} & \mathbf{E}_{1\mathcal{I}} \\ \vdots & \ddots & \vdots & \vdots \\ \mathbf{E}_{n1} & \ldots & \mathbf{E}_{nn} & \mathbf{E}_{n\mathcal{I}} \\ \mathbf{E}_{\mathcal{I}1} & \cdots & \mathbf{E}_{\mathcal{I}n} & \mathbf{E}_{\mathcal{I}\mathcal{I}} \end{bmatrix} = \mathbf{I} + |\mathbf{diag}(\mathbf{Q})|^{-1} \cdot \mathbf{Q}.$$

Extending the stochastic complement to several sets, we define

$$\begin{bmatrix} \mathbf{Q}_{11} & \cdots & \mathbf{Q}_{1n} \\ \vdots & \ddots & \vdots \\ \mathbf{Q}_{n1} & \cdots & \mathbf{Q}_{nn} \end{bmatrix} + \begin{bmatrix} \mathbf{Q}_{1\mathcal{I}} \\ \vdots \\ \mathbf{Q}_{n\mathcal{I}} \end{bmatrix} \cdot \left(\sum_{k=0}^{\infty} \mathbf{E}_{\mathcal{I}\mathcal{I}}^{k}\right) \cdot \begin{bmatrix} \mathbf{E}_{\mathcal{I}1} \cdots \mathbf{E}_{\mathcal{I}n} \end{bmatrix}$$

$$= \begin{bmatrix} \mathbf{Q}_{11}+\mathbf{Q}_{1\mathcal{I}} \cdot \left(\sum_{k=0}^{\infty} \mathbf{E}_{\mathcal{I}\mathcal{I}}^{k}\right) \cdot \mathbf{E}_{\mathcal{I}1} & \cdots & \mathbf{Q}_{1n}+\mathbf{Q}_{1\mathcal{I}} \cdot \left(\sum_{k=0}^{\infty} \mathbf{E}_{\mathcal{I}\mathcal{I}}^{k}\right) \cdot \mathbf{E}_{\mathcal{I}n} \\ \vdots & \ddots & \vdots \\ \mathbf{Q}_{n1}+\mathbf{Q}_{n\mathcal{I}} \cdot \left(\sum_{k=0}^{\infty} \mathbf{E}_{\mathcal{I}\mathcal{I}}^{k}\right) \cdot \mathbf{E}_{\mathcal{I}1} & \cdots & \mathbf{Q}_{nn}+\mathbf{Q}_{n\mathcal{I}} \cdot \left(\sum_{k=0}^{\infty} \mathbf{E}_{\mathcal{I}\mathcal{I}}^{k}\right) \cdot \mathbf{E}_{\mathcal{I}n} \end{bmatrix}$$

$$= \left[\overline{\mathbf{Q}}_{ij}\right]_{i,j\in\{1,\ldots,n\}} = \overline{\mathbf{Q}}.$$

The block $\overline{\mathbf{Q}}_{ii}$ of the complement with respect to all attractor sets should not be confused with the generator matrix $\overline{\mathbf{Q}}_{\mathcal{A}_i}$ which is the stochastic complement of $\mathbf{Q}$ with respect to the attractor set $\mathcal{A}_i$ only. Now we are able to define the *attractor Markov chain*.

▷**Definition 31**

Definition 31: Attractor Markov Chain

The *attractor Markov chain* for a CTMC $(\mathcal{S}, \mathbf{Q}, \pi(0))$ and a partitioning of the state space $\mathcal{S} = \mathcal{A}_1 \dot\cup \ldots \dot\cup \mathcal{A}_n \dot\cup \mathcal{I}$ is the CTMC $(\{\mathcal{A}_1, \ldots, \mathcal{A}_n\}, \widehat{\mathbf{Q}}, \widehat{\pi}(0))$ with $\widehat{\pi}_{\mathcal{A}_i}(0) = \frac{\sum_{s\in\mathcal{A}_i} \pi_s(0)}{\sum_{s'\in\mathcal{A}} \pi_{s'}(0)}$, $\mathcal{A} = \mathcal{A}_1 \dot\cup \ldots \dot\cup \mathcal{A}_n$, and $\widehat{\mathbf{Q}}$ defined by

$$\widehat{\mathbf{Q}} = \begin{bmatrix} \widehat{\mathbf{Q}}_{11} & \cdots & \widehat{\mathbf{Q}}_{1n} \\ \vdots & \ddots & \vdots \\ \widehat{\mathbf{Q}}_{n1} & \cdots & \widehat{\mathbf{Q}}_{nn} \end{bmatrix} = \left[\widehat{\mathbf{Q}}_{ij}\right]_{i,j\in\{1,\ldots,n\}},$$

where

$$\widehat{\mathbf{Q}}_{ij} = \overline{\pi}^{(i)}\, \overline{\mathbf{Q}}_{ij} \cdot \mathbf{e} = \overline{\pi}^{(i)} \cdot \left[\mathbf{Q}_{i\mathcal{I}} \cdot \left(\sum_{k=0}^{\infty} \mathbf{E}_{\mathcal{I}\mathcal{I}}^{k}\right) \cdot \mathbf{E}_{\mathcal{I}j} + \mathbf{Q}_{ij}\right] \cdot \mathbf{e}. \quad (5.2)$$

Rate $\widehat{\mathbf{Q}}_{ij}$ is the aggregated rate of the transition from attractor $\mathcal{A}_i$ to $\mathcal{A}_j$ ignoring the time spent outside the attractor regions. To approximate $\widehat{\mathbf{Q}}$, we first replace $\overline{\pi}_j^{(i)}$ by its approximation from step (a). Then,

the idea is to exploit Algorithm 8 implementing reachability based on inexact matrix-vector multiplications for computing $\widehat{\mathbf{Q}}$ row-wise as illustrated in Algorithm 10.

Algorithm 10 attract$(\mathbf{Q}, \mathcal{S}, \{\mathcal{A}_j\}_{1 \leq j \leq n}, i, \delta)$

1: $\theta \leftarrow \overline{\pi}^{(i)} \cdot \mathbf{Q}_{i\mathcal{I}} \cdot \mathbf{e}$
2: $\begin{bmatrix} m & e \end{bmatrix} \leftarrow$ reachability$(\overline{\pi}^{(i)} \cdot \mathbf{Q}_{i\mathcal{I}}, \mathbf{Q}, \mathcal{S}, \mathcal{A}_1 \,\dot\cup\, \ldots \,\dot\cup\, \mathcal{A}_n, \delta \cdot \theta)$
3: **for** $j \in \{1, \ldots, n\}$ **do**
4: $\quad \widehat{\mathbf{Q}}_{ij} \leftarrow \sum_{r \in \mathcal{A}_j} m(r) + \overline{\pi}^{(i)} \cdot \mathbf{Q}_{ij} \cdot \mathbf{e}$
5: **end for**
6: **return** $\begin{bmatrix} \widehat{\mathbf{Q}}_{i1} & \ldots & \widehat{\mathbf{Q}}_{in} \end{bmatrix}$

Theorem 17: Correctness of Algorithm 10 ▷Theorem 17

Given a CTMC $(\mathcal{S}, \mathbf{Q}, \pi(0))$ and a partitioning

$$\mathcal{S} = \mathcal{A}_1 \,\dot\cup\, \ldots \,\dot\cup\, \mathcal{A}_n \,\dot\cup\, \mathcal{I}$$

of the state space, then Algorithm 10 attract$(\mathbf{Q}, \{\mathcal{A}_j\}_{1 \leq j \leq n}, i, \delta)$ approximates the i-th row of the infinitesimal generator matrix $\widehat{\mathbf{Q}}$ of the corresponding attractor Markov chain.

Proof: Based on a small significance threshold δ, we consider only the relevant summands of Equation (5.2), similar to approximation algorithms for the transient distribution of a Markov chain [HMW09, DHMW09]. In other words, we use inexact matrix-vector multiplications while propagating the *rate-mass* vector $\overline{\pi}^{(i)} \cdot \mathbf{Q}_{i\mathcal{I}}$ through the infinite sum. For that we employ Algorithm 8 which was originally designed to propagate probability distributions. Likewise, Algorithm 10 iterates until the rate mass is absorbed in a set of absorbing states and truncates all states with *insignificant* rate mass. By insignificant rate mass, we mean the total weighted rate mass $\theta = \overline{\pi}^{(i)} \cdot \mathbf{Q}_{i\mathcal{I}} \cdot \mathbf{e}$ that leaves attracting region $\mathcal{A}_i$, as computed in line 1, times the desired threshold δ. Note that we need to weight δ by θ in line 2, since we no longer iterate on a probability distribution with total accumulated probability sum one,

but on a rate mass vector as described above.

In the while-loop of Algorithm 8, we consider those states that are reachable from $\mathcal{A}_i$ within K transitions where K is the number of times the while loop is executed. This corresponds to a truncation of the infinite sum in Equation (5.2) after K summands.

In Algorithm 9, we only consider transitions of states in $\mathcal{I}$. This ensures that the while-loop of Algorithm 8 will terminate after a finite number of steps because eventually the set $\mathcal{I}$ is left and the sets $\mathcal{A}_i$ are absorbing. Note that $\mathbf{Q}_{\mathbf{xy}} \cdot \mathbf{Q}_{\mathbf{xx}}^{-1}$ (Algorithm 9, line 3) corresponds to the entries in $\mathbf{E}_{\mathcal{I}\mathcal{I}}$ and $\mathbf{E}_{\mathcal{I}j}$. Thus, we propagate the vector $\overline{\pi}^{(i)} \cdot \mathbf{Q}_{i\mathcal{I}}$ step-by-step in the embedded Markov chain represented by $\mathbf{E}_{\mathcal{I}\mathcal{I}}$ and $\mathbf{E}_{\mathcal{I}j}$ which yields an approximation of the vector $\overline{\pi}^{(i)} \cdot \mathbf{Q}_{i\mathcal{I}} \cdot (\sum_{k=0}^{K-1} \mathbf{E}_{\mathcal{I}\mathcal{I}}^k)$ where K is the number of times the while-loop has been executed so far. In line 6 of Algorithm 8 we find all relevant states, that is, states that contribute at least a fraction δ of the total rate mass θ. The states that contribute less are dropped. Note that this ensures that among the states reachable from $\mathcal{A}_i$ within K transitions, we consider only those that are relevant. The for-loop in lines 3 and 4 in Algorithm 10 finally adds the last summand $\overline{\pi}_i \cdot \mathbf{Q}_{ij} \cdot \mathbf{e}$ as in Equation 5.2. □

The approximation proposed in Algorithm 10 corresponds to a reachability analysis during which the main part of the rate mass drifts back towards the attractor. The remaining part drifts to the other attractors where the truncation based on δ ensures that the most likely paths between the attractors are explored (they correspond to the subset of $\mathcal{I}$), that is, we do not explore regions holding an negligible portion of the total rate mass θ. More precisely, regions that are rarely visited when switching between attractors will not be included.

Once an approximation of $\widehat{\mathbf{Q}}$ is obtained we can compute the steady state distribution $\widetilde{\pi}$ of the attractor Markov chain using, for instance, a direct solution method such as Gaussian elimination. Note that the attractor Markov chain has only n states. Thus, even if the transition rates in $\widehat{\mathbf{Q}}$ differ widely, an efficient solution is possible.

5.2 Approximation Error Analysis

5.2.1 Error of the Local Steady State Computation

We first consider the error made in step (a), that is, when approximating $\overline{\pi}^{(i)}$. For each state s in $\mathcal{A}_i$ that has a transition to a state outside of $\mathcal{A}_i$ we construct a Markov chain with generator $\widetilde{\mathbf{Q}}^{(s)}$. Matrix $\widetilde{\mathbf{Q}}^{(s)}$ is equal to matrix $\mathbf{Q}_{ii}$ except that from the s-th column we subtract the vector $\mathbf{Q}_{ii} \cdot \mathbf{e}$, that is, we redirect all transitions of state s leaving $\mathcal{A}_i$ back to state s just like we did in Chapter 3.4. Consequently, solving the steady state distributions for all $\widetilde{\mathbf{Q}}^{(s)}$ and taking the component-wise minima and maxima gives bounds on $\overline{\pi}_s$ as in Equation (3.16). It is of course computationally quite expensive to bound the error in this way except if the attractor is very small. The approximation obtained in step (a) yields a solution which lies within these bounds since any redirection induces a distribution that lies in the polyhedral hull spanned by the steady state distributions of $\widetilde{\mathbf{Q}}^{(i)}$ [CS84]. The bounds correspond to the worst case and in practice our approximation is much better than the bounds obtained in the way that we describe above.

5.2.2 Error of the Computation of the Steady State of the Attractor Markov Chain

For step (b), we note that some parts of the total rate mass θ is dropped because of the threshold δ. Consequently, when we approximate the rows of $\widehat{\mathbf{Q}}$ using Algorithm 10, that is,

$$\widehat{\mathbf{Q}} \approx \widetilde{\mathbf{Q}} = \begin{bmatrix} \text{attract}(\mathbf{Q}, \{\mathcal{A}_i\}_{1 \le i \le n}, 1, \delta) \\ \vdots \\ \text{attract}(\mathbf{Q}, \{\mathcal{A}_i\}_{1 \le i \le n}, n, \delta) \end{bmatrix},$$

the resulting matrix $\widetilde{\mathbf{Q}}$ will not be an infinitesimal generator matrix since the row sum will be smaller than zero. The reason is that for the off-diagonal entries we obtain under-approximations as

$$\widehat{\mathbf{Q}}_{ij} = \overline{\pi}^{(i)} \cdot \overline{\mathbf{Q}}_{ij} \cdot \mathbf{e} = \overline{\pi}^{(i)} \cdot \mathbf{Q}_{ij} \cdot \mathbf{e} + \underbrace{\overline{\pi}^{(i)} \cdot \mathbf{Q}_{i\mathcal{I}} \cdot (\sum_{k=0}^{\infty} \mathbf{E}_{\mathcal{I}\mathcal{I}}^{k}) \cdot \mathbf{E}_{\mathcal{I}j} \cdot \mathbf{e}}_{\geq \sum_{r \in \mathcal{A}_j} m(r)} \geq \widetilde{\mathbf{Q}}_{ij},$$

where $m(r)$ refers to the value of state r in hash map m of Algorithm 10. The diagonal entries $\widetilde{\mathbf{Q}}_{ii}$, however, contain the total rate at which attractor $\mathcal{A}_i$ is left. Given that we have the exact local steady states $\overline{\pi}^{(i)}$, we bound the approximation error made during the computation of $\widetilde{\pi}$ (the steady state of $\widehat{\mathbf{Q}}$) as follows. The idea is again similar to Chapter 3.4, that is, we consider the slack rate mass vector $\mathbf{s} = -\widetilde{\mathbf{Q}}\mathbf{e}$ whose i-th entry is the difference between the total rate mass at which macro state $\mathcal{A}_i$ is left and the approximated rate mass that accumulated in the attractors (including $\mathcal{A}_i$). In order to transform $\widetilde{\mathbf{Q}}$ into a generator matrix, the amount s_i has to be distributed among the elements in the i-th row (including the diagonal entry since it is possible to enter $\mathcal{I}$ from $\mathcal{A}_i$ and return to $\mathcal{A}_i$). Bounds for the approximation error can be derived by considering all *extreme cases* where vector $\mathbf{s}$ is added to the j-th column. Formally, we define the matrices $\widetilde{\mathbf{Q}}^{(j)}$ with entries

$$\widetilde{\mathbf{Q}}^{(j)}_{ik} = \begin{cases} \widetilde{\mathbf{Q}}_{ik} & \text{if } k \neq j, \text{ and} \\ \widetilde{\mathbf{Q}}_{ik} + s_i & \text{otherwise} \end{cases}$$

for $1 \leq j \leq n$. Now, if $\widetilde{\pi}^{(j)}$ is the unique steady state of $\widetilde{\mathbf{Q}}^{(j)}$, for the steady state $\widehat{\pi}_i$ of $\widehat{\mathbf{Q}}$ we have [CS84, DHSW11]

$$\widehat{\pi}_i \in [\min_j \ \widetilde{\pi}^{(j)}_i, \max_j \ \widetilde{\pi}^{(j)}_i].$$

Since the number of attractor regions n is small, we can compute the bounds on $\widehat{\pi}_i$ efficiently using direct methods. Multiplication of the above bounds on $\widehat{\pi}_i$ with the conditional steady state probability $\overline{\pi}^{(i)}_s$ yields a bounded approximation of the steady state probability π_s for state $s \in A_i$. Note that the probability of all states in $\mathcal{I}$ is approximated as zero. In order to combine the bounds from steps (a) and (b), Algorithm 10 can be executed twice where $\overline{\pi}^{(i)}$ is replaced by its upper and lower bound, respectively.

5.2.3 Choosing the Truncation Threshold

The accuracy of the approximation depends on the parameter δ. It is possible to dynamically control the amount of rate mass that gets *lost* due to the truncation by repeating a step of the reachability analysis

with a lower value for δ. In this case, one has to determine a priori an upper bound on the rate mass lost per step. We note that for $\delta = 0$, the attractor Markov chain of an ergodic CTMC is always ergodic [Mey89, DHSW11]. However, if δ is chosen too high it might happen that $\widehat{\mathbf{Q}}$ is not ergodic. In such a case the computation has to be repeated using a smaller value for δ. In our experiments we chose $\delta \in [1e-20, 1e-8]$ and we always obtained an ergodic $\widehat{\mathbf{Q}}$. Moreover, we found that the approximation of $\widehat{\pi}$ is accurate even for $\delta = 1e-8$. In fact, choosing $\delta \in [1e-20, 1e-10]$ was also suggested in [DHMW09].

5.3 Locating Attracting Regions

Theorem 8 provides a way to truncate a large system in an appropriate way. The problem of locating attractors is, however, not fully solved since the set $\mathcal{C}$ contains all attractors of the system but it might also contain low probability regions. Moreover, the drift d does not provide enough information to locate attractors. Regions where the drift is close to zero are, of course, good candidates for attractors but since d is an expectation, there might be states with $d(\mathbf{x}) \approx 0$ where the system does resides for a short time span. Similarly, there might be states with high absolute drift and high steady state probability.

A simple way of locating attractors is to generate a long trajectory of the system for example by employing simulation as in Algorithm 1. Then, attractors are those regions where the process resides most of the time, that is, regions that have a small probability of exiting. The problem is that once an attractor is found it is unlikely that the system switches to another attractor rendering this strategy inefficient. If many trajectories are generated that start from different initial states, it is more likely that all attractors are found. It is however not clear how to determine the extents of an attractor and when a simulation run can be finished.

We suggest an alternative strategy where we employ Algorithm 11, a modified version of Algorithm 8, that is, reachability analysis on a dynamically truncated state space. The idea is to perform the analysis from several starting points which can be chosen at random, based on prior knowledge about the system or based on approximations such as the mean field of the Markov chain. It will follow the main part

of the probability mass and terminates as soon as the set of states corresponding to the attracting regions stabilizes, that is, set $\mathcal{D}$ does not change within a reachability step. Again $\delta > 0$ is a small threshold

Algorithm 11 locate($\mathbf{Q}, \mathcal{S}, \mathcal{W}, \delta$)

1: $p \leftarrow$ new hash map $\mathcal{S} \rightarrow [0, 1]$
2: $\forall \mathbf{x} \in \mathcal{W}$: $p(\mathbf{x}) \leftarrow |\mathcal{W}|^{-1}$
3: $\mathcal{D} \leftarrow \emptyset$
4: **while** dom(p) $\neq \mathcal{D}$ **do**
5: $\quad \mathcal{D} \leftarrow$ dom(p)
6: $\quad p \leftarrow$ rstep($p, \mathbf{Q}, \emptyset, \delta$)
7: $\quad \forall \mathbf{x}$ with $p(\mathbf{x}) < \delta$: remove($p, \mathbf{x}$)
8: **end while**
9: **return** The set $\{\mathcal{A}_1, \ldots, \mathcal{A}_n\}$ of BSCCs in set $\mathcal{D}$

which is either fixed during the analysis or chosen in an adaptive way to ensure that the total number of states in $\mathcal{D}$ remains small. Note that in order to guarantee that all attractors are found, we must choose the initial set $\mathcal{W}$ such that $\mathcal{C} \subseteq \mathcal{W}$, where $\mathcal{C}$ is a subset of the state space that contains at least $1 - \epsilon$ of the steady state probability mass as for example determined in Chapter 3.1. Our experimental results, show that even with the high choice $\delta = 1e - 7$ we can locate all attractor regions of the examples that we considered.

5.4 Case Studies

In the following, we apply our approach to two case studies from biology. The first case study was also considered in [MAL11] and since experimental results were reported, we can compare our results to those in [MAL11]. All computations were done on a 2.66 GHz machine with 4 GB of RAM using a single core, that is, a machine with the same specification as in [MAL11]. The two models that we consider are structurally similar (both are genetic switches) but differ in the number of attractors (two and three). Moreover, the attractors of the second model have very different probabilities while for the first model the probability mass distributes nearly equally. We remark that these two models

Method	δ	$\widehat{\pi}_1$	$\widehat{\pi}_2$	Time (s)
Exact	–	0.472719...	0.527280...	463
[MAL11]	–	0.4759	0.5241	290
Ours	1e–5	0.5	0.5	60.1+0.170+0.304
Ours	1e–7	0.47	0.53	60.1+0.170+0.307
Ours	1e–8	0.473	0.527	60.1+0.170+0.312
Ours	1e–9	0.4727	0.5273	60.1+0.170+0.313
Ours	1e–20	0.47272	0.52728	60.1+0.170+0.367

Table 5.1: Results of the multimodal steady state analysis applied to the Michaelis-Menten form of the genetic toggle switch model.

are exemplary for gene regulatory networks with multimodal steady state distributions.

5.4.1 Genetic Toggle Switch

In Table 5.1 we list the experimental results of the methods of the previous sections applied to the Michaelis-Menten form of the genetic toggle switch as described in Model 6. The parameter set is

$$\rho_A = 60.0, \rho_B = 30.0, n_A = 3, n_B = 2, \delta_A = \delta_B = 1.0,$$

as in Chapter 3.7.1 where also the respective geometric bounds have been constructed. We took these bounds to truncate the infinite state space. In order to locate the attractor regions, we determined the starting points by considering the mean field of the model. More precisely, the expected molecule numbers $(\mathbf{x}_1(t), \mathbf{x}_2(t))$ can be approximated as

$$\tfrac{d}{dt}\begin{bmatrix}\mathbf{x}_1 & \mathbf{x}_2\end{bmatrix} = \begin{bmatrix}\rho_A \cdot (1+\mathbf{x}_2^{n_A})^{-1} - \delta_A \cdot \mathbf{x}_1 & \rho_B \cdot (1+\mathbf{x}_1^{n_B})^{-1} - \delta_B \cdot \mathbf{x}_2\end{bmatrix}.$$

We solved $\frac{d}{dt}\mathbf{x} = \mathbf{0}$ using the HOM4PS2 [LLT08] package which took less than one second. This yielded the three (rounded) solutions $(0, 30)$, $(60, 0)$, and $(3, 3)$ which lie inside the geometric bounds given by the set $\mathcal{C}$. We performed a truncation-based transient analysis starting in the two points with truncation threshold of $\delta = 1e-7$. The method converged after 421 steps, that is, when the set of significant states remained unchanged. This took 60.1 seconds of computation time. The

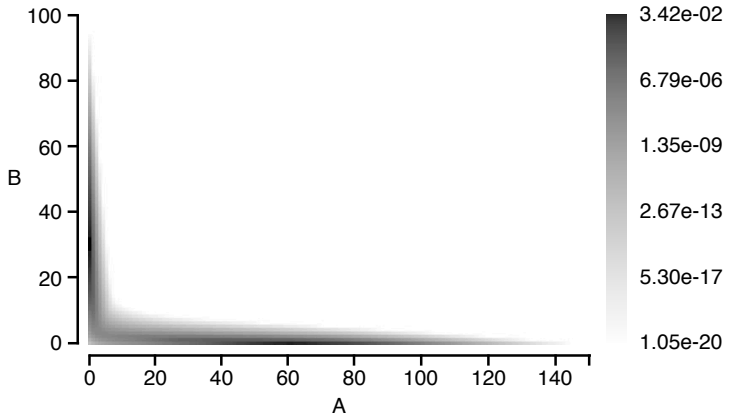


Figure 5.1: Steady state distribution of the Michaelis-Menten form of the toggle switch model on a logarithmic scale. Completely white states are states that have been truncated.

resulting significant sets correspond to our approximation of the attractor regions $\mathcal{A}_1$ and $\mathcal{A}_2$. Note that the probability mass that started in the third point $(3, 3)$ distributed among the sets $\mathcal{A}_1$ and $\mathcal{A}_2$ implying that it corresponds to an unstable fixed point of the mean field. The sets that we derived for $\delta = 1e-7$ are slightly smaller than the regions

$$\mathcal{A}_1 = \{\mathbf{x} \in \mathbb{N}^2 \mid 0 \leq \mathbf{x}_1 \leq 2 \text{ and } 5 \leq \mathbf{x}_2 \leq 60\}, \text{ and}$$
$$\mathcal{A}_2 = \{\mathbf{x} \in \mathbb{N}^2 \mid 14 \leq \mathbf{x}_1 \leq 105 \text{ and } 0 \leq \mathbf{x}_2 \leq 2\}.$$

defined in [MAL11]. Using a threshold of $\delta = 1e-8$ for our method resulted in slightly larger sets. In order to allow a comparison with the results in [MAL11], we use the regions defined above for the approximation of the attractor probabilities.

To obtain a more accurate approximation of $\overline{\pi}^{(1)}$ and $\overline{\pi}^{(2)}$, we used enlarged sets $\mathcal{A}_1$ and $\mathcal{A}_2$ by including those states reachable within 15 steps for the computation as described in Chapter 5.1.1. Then, we constructed the attractor Markov chain using Algorithm 10. Note that the chain only consists of two states, $\mathcal{A}_1$ and $\mathcal{A}_2$. In the second column in Table 5.1 we list the final result of our approximation for different values of the truncation threshold δ of Algorithm 10. Moreover, since the trun-

cated model (containing those states inside the geometric bounds) was of tractable size, we used the direct solution methods implemented in Matlab to compute the *exact solution* circumventing problems related to stiffness. We also list the results given in [MAL11] for comparison. The number of digits corresponds to the computational precision, where the Matlab solution was computed up to the machine precision of 15 digits. In our method, the computation time consists of the time needed to locate the attractor regions (left), to compute the local steady states (middle) and the time needed to compute the attractor probabilities $\widehat{\pi}_i$ using Algorithm 10 (right). The time needed to compute the geometric bounds and to solve the steady state of the mean-field was negligible (below 0.03 seconds each). The final steady state distribution of this model is depicted in Figure 5.1.

Our method is magnitudes faster than the exact solution using Matlab and the simulation based approach in [MAL11]. Note that in contrast to [MAL11], our method allows for an identification of the attractor regions. Also, the precision of our algorithm is much higher than the results presented in [MAL11]. From the table, it can also be seen that the accuracy of the approximation is very good for thresholds $\delta \leq 1e-8$ and the computation time is short even for $\delta = 1e-20$.

5.4.2 Tri-Stable Genetic Switch

In order to show the applicability of our approach in the presence of multiple attractors, we also analyzed a model with three attractors.

Model 8: Tri-Stable Toggle Switch ▷Model 8

The *tri-stable toggle switch* [LNH+07] consists of three protein species $A(1)$, $B(2)$, and $C(3)$ encoded by three genes, where each species represses the production of the other two species. The chemical reactions are given below, where the parameters n_{ij} regulate the repression strength among the proteins and rates ρ_{ij} and δ_i control protein production and degradation for $i, j \in \{A, B, C\}$.

$$\varnothing \xrightarrow{\frac{\rho_{AB}}{1+B^{n_{AB}}}+\frac{\rho_{AC}}{1+C^{n_{AC}}}} A, \qquad A \xrightarrow{\delta_A} \varnothing,$$

$$\varnothing \xrightarrow{\frac{\rho_{BA}}{1+A^{n_{BA}}}+\frac{\rho_{BC}}{1+C^{n_{BC}}}} B, \qquad B \xrightarrow{\delta_B} \varnothing,$$
$$\varnothing \xrightarrow{\frac{\rho_{CA}}{1+A^{n_{CA}}}+\frac{\rho_{CB}}{1+C^{n_{CB}}}} C, \qquad C \xrightarrow{\delta_C} \varnothing$$

For the analysis, we chose the parameter set $\rho_{AB} = \rho_{AC} = 60.0$, $\rho_{BA} = \rho_{BC} = 30.0$, $\rho_{CA} = \rho_{CB} = 50.0$, $\delta_A = \delta_B = \delta_C = 1.0$ and $n_{AB} = n_{AC} = n_{BA} = n_{BC} = n_{CA} = n_{CB} = 3$. First, we computed geometric bounds as described in Chapter 3.1 for $\epsilon = 0.01$. The bounds are depicted in Figure 5.2 where the computation time was 2.641 seconds. Next, we computed the possible steady states of the mean-field by solving $\frac{d}{dt}\mathbf{x} = \mathbf{0}$ for

$$\begin{aligned}\frac{d}{dt}\mathbf{x} &= \left[\frac{\rho_{AB}}{1+\mathbf{x}_2^{n_{AB}}}+\frac{\rho_{AC}}{1+\mathbf{x}_3^{n_{AC}}} \quad \frac{\rho_{BA}}{1+\mathbf{x}_1^{n_{BA}}}+\frac{\rho_{BC}}{1+\mathbf{x}_3^{n_{BC}}} \quad \frac{\rho_{CA}}{1+\mathbf{x}_1^{n_{CA}}}+\frac{\rho_{CA}}{1+\mathbf{x}_2^{n_{CA}}}\right] \\ &- \left[\delta_A \cdot \mathbf{x}_1 \quad \delta_B \cdot \mathbf{x}_2 \quad \delta_C \cdot \mathbf{x}_3\right],\end{aligned}$$

using the HOM4PS2 package [LLT08] (which took less than 0.030 seconds). We got the five (rounded) solutions [60 0 50], [0 30 50], [60 30 0], [2 3 7], and [8 3 2] located within the geometric bounds.

Truncation-based transient analysis with threshold $\delta = 1e{-}7$ revealed that only the first three points correspond to attracting regions. The sets stabilized after 493 iterations and the total computation time was 136.4 seconds. We will refer to these three regions as $\mathcal{A}_1$, $\mathcal{A}_2$, and $\mathcal{A}_3$ where

$$\begin{aligned}\mathcal{A}_1 &\approx \{\mathbf{x} \in \mathbb{N}^3 \mid 10 \leq \mathbf{x}_1 \leq 110, 0 \leq \mathbf{x}_2 \leq 3, 10 \leq \mathbf{x}_3 \leq 95\}, \\ \mathcal{A}_2 &\approx \{\mathbf{x} \in \mathbb{N}^3 \mid 0 \leq \mathbf{x}_1 \leq 3, 5 \leq \mathbf{x}_2 \leq 70, 5 \leq \mathbf{x}_3 \leq 95\}, \text{ and} \\ \mathcal{A}_3 &\approx \{\mathbf{x} \in \mathbb{N}^3 \mid 5 \leq \mathbf{x}_1 \leq 110, 5 \leq \mathbf{x}_2 \leq 70, 0 \leq \mathbf{x}_3 \leq 3\}.\end{aligned}$$

For approximating the local steady states, we enlarged each set $\mathcal{A}_i$ by those states reachable within 15 steps. We ran Algorithm 10 and list the results in Table 5.2. A plot of the full steady state distribution is given in Figure 5.3. The time needed to compute the local steady state was 246 seconds and to compute $\widehat{\pi}$ was below five seconds for any precision. Note that for this model, we could not solve the linear equation using

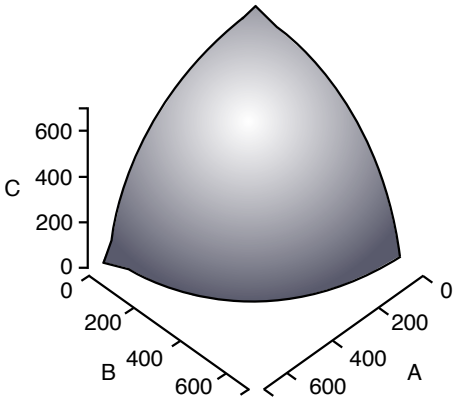


Figure 5.2: Geometric bounds of the tri-stable toggle switch for $\epsilon = 0.01$.

δ	$\widehat{\pi}_1$	$\widehat{\pi}_2$	$\widehat{\pi}_3$	Time (s)
1e–9	0.995	0.001	0.004	2.641+ 136.4 + 246.011 + 1.194
1e–10	0.9952	0.0013	0.0035	2.641+ 136.4 + 246.011 + 1.338
1e–11	0.99516	0.00132	0.00352	2.641+ 136.4 + 246.011 + 1.472
1e–20	0.995165	0.001315	0.003520	2.641+ 136.4 + 246.011 + 4.536

Table 5.2: Results of the multimodal steady state analysis of the tri-stable toggle switch case study.

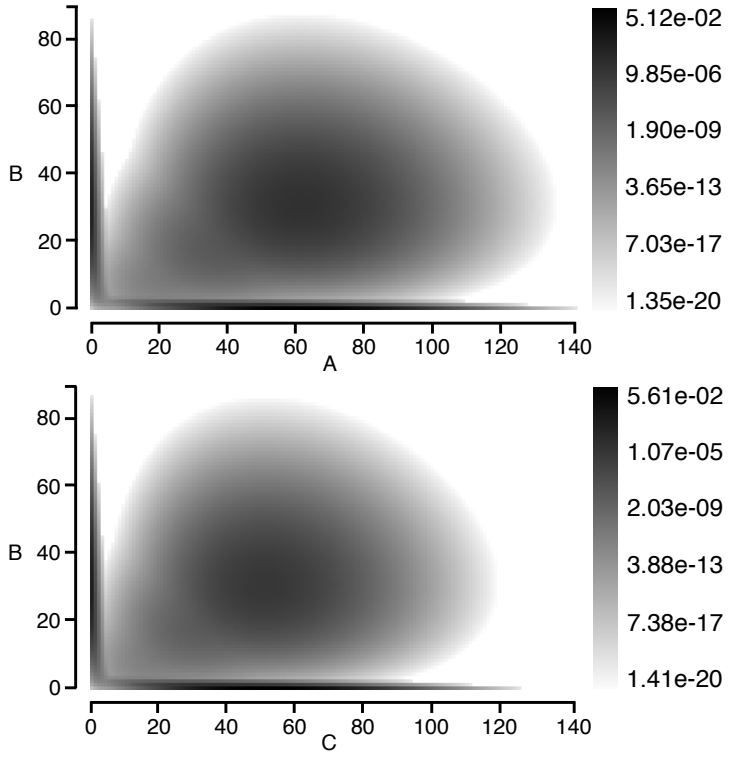


Figure 5.3: Steady state distribution of the tri-stable toggle switch model on a logarithmic scale. Completely white states are states that have been truncated.

Matlab since the corresponding linear equation system has more than 1.5 million equations. But even if the memory requirements could be met, the stiffness of the model would prevent an efficient solution. As opposed to that, our method does not suffer from problems related to stiffness because the time scales are separated during the stochastic complementation. Therefore, we got accurate results and very short computation times as presented in Table 5.2.

CHAPTER 6

On-the-Fly Verification and Optimization of DTA-Properties for Large Markov Chains

In this chapter we consider CTMCs together with specifications given as (single-clock) deterministic timed automata (DTA). We propose algorithms to approximate the probability that the CTMC fulfills the property specified by the DTA, that is, we approximate the *acceptance probability*. DTA can be used to describe *linear-time properties* such as the occurrence of several events where the time between two successive events is smaller or greater than a certain threshold. For instance, it is possible to specify oscillatory behavior or the switching between attracting regions in a CTMC. Such properties are particularly important for applications in systems biology. Also, DTA are in practice often more intuitive and easier to construct than, for instance, logical formulas of the continuous stochastic logic (CSL) [BHHK03].

We focus on CTMCs with very large or most of the time even *infinite* state spaces. Previously developed model-checking algorithms cannot be applied to such chains, because they explore the whole state space and are based on vector-matrix multiplications of the size of the number of states. For the CTMCs that we consider here, it is infeasible to explore the whole state space due to memory and time limitations. In the worst case, the state space is infinite and even a simple static truncation

of the state space does not suffice to make the analysis efficient.

The algorithms that we propose are based on results in [BCH+11], where the computation of the acceptance probability is based on a transient analysis of *augmented CTMCs* (or *column CTMCs*) and the solution of a linear equation system. Here, we also consider column CTMCs but we discretize time and for each time step we consider a dynamic subset of the state space, namely those states that are most relevant in the sense that their probability is greater than a certain threshold. Moreover, instead of deriving the limiting distribution by solving a linear equation system, we construct the column CTMCs in such a way that a simple convergence check for the transient distribution is sufficient to obtain an accurate approximation of the acceptance probability. These improvements allow us to deal with infinite state spaces.

In addition, we investigate the case where the CTMC is parametric as in [BCDS13] and show how the acceptance probability can be optimized. More precisely, we assume that the transition rates of the CTMC may depend on certain parameters, say $\lambda_1, \ldots, \lambda_k$, and extend our algorithm for computing the acceptance probability in such a way that the derivatives with respect to $\lambda_1, \ldots, \lambda_k$ are approximated alongside. Then, we apply a global optimization method to find values for the parameters that maximize the acceptance probability. Such optimizations are particularly interesting for systems where a certain property expressible as a DTA has been observed and the parameters of the CTMC are unknown and have to be estimated accordingly. The results of this chapter have been published in [MNSW13].

6.1 Labeled CTMCs

Similar to the procedure of Chapter 4, in this chapter, we assume the states of a CTMC to be labeled with atomic propositions from a set $\mathcal{AP}$ via a labeling function defined as follows.

Definition 32: Labeling Function

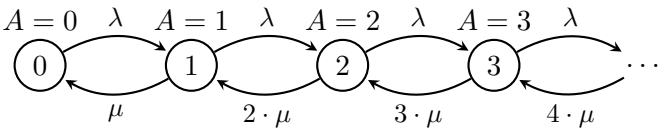
▷Definition 32

A *labeling function* L for the state space $\mathcal{S}$ of a CTMC and a set of atomic propositions $\mathcal{AP}$ is a function $L : \mathcal{S} \to \mathcal{AP}$.

Example 10: Labeled CTMC

▷Example 10

Below we illustrate the M/M/∞ queue from Example 1 with state labels positioned above the nodes assigned by a labeling function. This CTMC serves as our running example for this chapter. We assume the initial distribution given by $\pi_0(0) = 1$. The state labels encode the number of objects of type A, that is, state i is labeled with $A = i$ where i is the number of individuals of type A. Note that there is no bound on the number of objects, that is, the state space of this example is countably infinite. Moreover, independent of the choice of λ and μ, this CTMC is ergodic and thus the limiting distribution π always exists uniquely.



6.2 Single-Clock Deterministic Timed Automata

We use deterministic timed automata with a single clock x to specify the measures of interest for a given CTMC.

▷Definition 33

Definition 33: Clock Constraint and Valuation

A *clock constraint* g over constants $\{c_0, \ldots, c_n\} \subset (\mathbb{N} \cup \{\infty\})$ with $c_0 = 0$ and $c_n = \infty$ is a conjunction $\bigwedge_{i=0}^{k} e_i$ of expressions $e_i = x \bowtie c_j$ where x is a symbol which denotes the (single) clock, $\bowtie \in \{<, \geq\}$, $k \geq 1$, and $0 \leq j < n$. A *clock valuation* $\eta \in \mathbb{R}_0^+$ satisfies an expression $x \bowtie c_j$, written $\eta \models x \bowtie c_j$ iff $\eta \bowtie c_j$. Accordingly, if $g = \bigwedge_{i=0}^{k} e_i$, then $\eta \models g$ iff $\eta \models e_i$ for all $i \leq k$. We use $\mathcal{CC}$ to denote the set of all clock constraints.

To simplify the presentation of our approach we restrict the comparison operators to $<$ and $\geq$. This ensures that we only consider intervals of the form $[a, b)$ with $a \in \mathbb{N}$ and $b \in \mathbb{N} \cup \{\infty\}$. Note that an extension to more complex clock constraints is straightforward. Further, we choose $c_0 = 0$ and $c_n = \infty$ for convenience in order to reduce the number of case distinction in the later presentation. In addition, we want to add that the choice for the constants $c_1, \ldots, c_{n-1}$ to be natural numbers is not as restrictive as it might seem, since time can be scaled by a factor to achieve rational time constraints.

▷Definition 34

Definition 34: Deterministic Timed Automaton

A (single-clock) *deterministic timed automaton* (DTA) [AD94] is a tuple $(\mathcal{M}, \mathcal{AP}, m_0, \mathcal{M}_F, \rightarrow)$ where $\mathcal{M}$ is a set of locations, $\mathcal{AP}$ a set of atomic propositions, $m_0 \in \mathcal{M}$ the initial location, $\mathcal{M}_F \subseteq \mathcal{M}$ a set of final locations, and

$$\rightarrow \subseteq (\mathcal{M} \setminus \mathcal{M}_F) \times 2^{\mathcal{AP}} \times 2^{\mathcal{AP}} \times \mathcal{CC} \times \{\circlearrowleft, \rightsquigarrow\} \times \mathcal{M}$$

an edge relation satisfying the *determinism/non-blocking condition*

$$\begin{aligned} &\forall m \in \mathcal{M} \setminus \mathcal{M}_F, a, a' \in \mathcal{AP}, \eta \in \mathbb{R}_0^+. \\ &\exists!\ (m, \mathcal{A}, \mathcal{A}', g, r, m') \in \rightarrow.\ a \in \mathcal{A} \wedge a' \in \mathcal{A}' \wedge \eta \models g. \end{aligned} \quad (6.1)$$

If $(m, \mathcal{A}, \mathcal{A}', g, r, m') \in \rightarrow$, we also write $m \xrightarrow{\mathcal{A}, \mathcal{A}', g, r} m'$. Informally, in a clock valuation $\eta \in \mathbb{R}_0^+$, the DTA can move from location m to

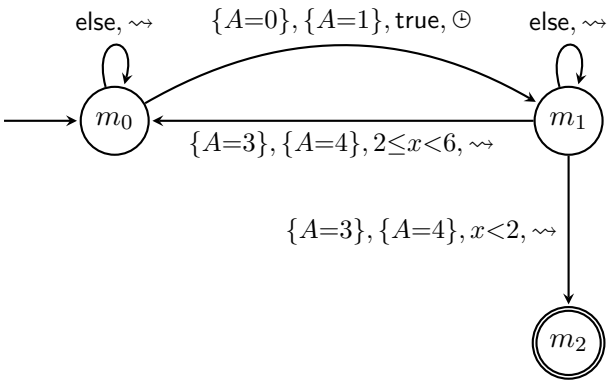


Figure 6.1: Exemplary DTA specification.

location m' along the edge $m \xrightarrow{\mathcal{A},\mathcal{A}',g,r} m'$ if the pre-input symbol is in $\mathcal{A} \subseteq \mathcal{AP}$, the post-input symbol is in $\mathcal{A}' \subseteq \mathcal{AP}$ and if η satisfies the guard $g \in \mathcal{CC}$. Note that this allows us to specify label changes in a more succinct way compared to DTA where only a single input symbol is considered. Taking a transition in the DTA consumes no time. Hence, for current clock valuation η, the new clock valuation η' obtained by executing a transition $m \xrightarrow{\mathcal{A},\mathcal{A}',g,r} m'$ is determined by the *reset action* r. More precisely, if $r = \rightsquigarrow$, then η' equals η, as no time is consumed by the transition. Otherwise, $r = ⏲$ and the clock is reset. In this case, we have $\eta' = 0$. We follow the restriction in [CHKM11] and assume that all final locations in a DTA are absorbing. Further, note that our treatment of DTA allows state-based reasoning in contrast to action-based reasoning.

▷Example 11

Example 11: DTA Specification

In Figure 6.1 we illustrate a DTA that serves as a specification for the CTMC in Example 10. To satisfy the non-blocking condition as stated in Equation (6.1), we use else-transitions as a catch-all for otherwise missing outgoing transitions. Similarly, we simply write true for a transition with $\mathcal{A} = \mathcal{A}' = \mathcal{AP}$ and clock constraint true.
This automaton works as follows. When the number of objects of type A jumps from 0 to 1, the clock is reset and the automaton moves to location m_1. From there, the DTA accepts if the CTMC moves from state 3 to state 4 within the next two time units. Stated differently, the DTA accepts upon entering m_1 if 4 objects of type A become available within the next 2 time units. If, on the other hand, the CTMC manages to take the transition from 3 to 4 only within six time units, the DTA returns to state m_0 waiting for the CTMC to eventually increase the object count from 0 to 1, again. Finally, if the CTMC does not create 4 objects of type A within 6 time units while residing in location m_1, the DTA gets stuck and cannot accept any more. Note that in this example, the set $\mathcal{A}' = \{A = 3\}$ of allowed pre-input labels from m_1 can be replaced by $\mathcal{A}' = \mathcal{AP}$ since the CTMC can only reach $A = 4$ from state 3 and not from state 5 due to the constraints of the previous transition from location m_0 to m_1.

▷Definition 35

Definition 35: Accepting Paths of a DTA

We define the set of all *accepting paths* of a DTA $\mathscr{D} = (\mathcal{M}, \mathcal{AP}, m_0, \mathcal{M}_F, \rightarrow)$ as the subset

$$Paths^{\mathscr{D}}_{acc} \subseteq \bigcup_{n \in \mathbb{N}} \mathcal{M} \times (2^{\mathcal{AP}} \times 2^{\mathcal{AP}} \times \mathbb{R}^+ \times \mathcal{M})^n$$

such that $(m_0, \mathcal{A}_0, \mathcal{A}'_0, t_0, m_1, \ldots, \mathcal{A}_{n-1}, \mathcal{A}'_{n-1}, t_{n-1}, m_n) \in Paths^{\mathscr{D}}_{acc}$

implies that $m_n \in \mathcal{M}_F$ and

$$\forall i < n.\ \exists\, (m_i, \mathcal{A}_i, \mathcal{A}'_i, g_i, r_i, m_{i+1}) \in \rightarrow\ .\ \eta_i + t_i \models g_i,$$

where $\eta_0 = 0$ and $\eta_{i+1} = \eta_i + t_i$ if $r_i = \rightsquigarrow$ and $\eta_{i+1} = 0$ if $r_i = \circledcirc$.

Definition 36: DTA Acceptance of CTMC Paths ▷Definition 36

Let $(\mathcal{S}, \mathbf{Q}, \pi(0))$ be a CTMC labeled with atomic propositions from the set $\mathcal{AP}$ via a labeling function L. Then, a finite CTMC path $(s_0,t_0,s_1,\ldots,t_{k-1},s_k) \in Path^*$, respectively an infinite path $(s_0,t_0,s_1,t_1,\ldots) \in Path^\omega$, is *accepted* by the DTA $\mathscr{D}$ iff there exists an accepting DTA path

$$(m_0, \mathcal{A}_0, \mathcal{A}'_0, t_0, m_1, \ldots, \mathcal{A}_{n-1}, \mathcal{A}'_{n-1}, t_{n-1}, m_n) \in Path^{\mathscr{D}}_{acc}$$

such that $L(s_i) \in \mathcal{A}_i \wedge L(s_{i+1}) \in \mathcal{A}'_i$ for all $i < n$ and $k \geq n$ for finite paths.

We remark that the above definition requires that the DTA and the CTMC change state simultaneously, that is, the DTA can only perform a jump if the CTMC does so. It is possible to extend this definition such that the DTA jumps even though the CTMC remains in a certain state (because some clock constraint became true). The algorithm that we propose in Chapter 6.4 can be modified accordingly. In order to keep our presentation simple we do not consider this extension here.

Example 12: DTA Path Acceptance ▷Example 12

All paths of the CTMC in Example 10 starting with the prefixes

$$(s_0, 1.0, s_1, 0.2, s_2, 0.5, s_3, 0.3, s_2, 0.2, s_3, 0.2, s_4, 1.0, s_5, 1.0)$$

and

$$(s_0, 1.0, s_1, 0.2, s_2, 1.0, s_3, 0.5, s_2, 1.0, s_3, 0.5, s_4, 1.2, s_3, 1.5, s_2, 1.1, s_1, 0.3, s_0, 0.7, s_1, 0.5, s_2, 0.2, s_3, 1.2, s_4, 1.0, s_5, 1.0)$$

are accepted by the DTA from Example 11.

6.3 Region Graph

We begin by defining clock regions before introducing region graphs.

▷Definition 37

Definition 37: Clock Region

Assuming a set $\{c_0, \ldots, c_n\} \subset \mathbb{N} \cup \{\infty\}$ of clock constraint constants with $0 = c_0 < \cdots < c_n = \infty$, we partition the time-line into the *clock regions* $R_i = [c_{i-1}, c_i)$, $i \in \{1, \ldots, n\}$. For $i < n$ we define the *successor clock region* $scr(R_i)$ of R_i as R_{i+1} and $scr(R_n) = R_n$. Moreover, $\mathcal{R}$ is the set of all clock regions. For each clock valuation η, we define $[\eta] = R_i = [c_{i-1}, c_i)$ for $i \in \{1, \ldots, n\}$ such that $c_{i-1} \leq \eta < c_i$. A clock region R_i satisfies a clock constraint g over $\{c_0, \ldots, c_n\}$ if $c_{i-1} \models g$.

Note that checking the lower bound suffices here since all clock valuations $\eta \in [c_{i-1}, c_i)$ satisfy the same clock constraints over constants c_0 to c_n and we only consider the comparison operators $<$ and $\geq$.

▷Definition 38

Definition 38: Region Graph

Let $\mathscr{D} = (\mathcal{M}, \mathcal{AP}, m_0, \mathcal{M}_F, \rightarrow)$ be a DTA with clock region set $\mathcal{R} = \{R_i \mid 1 \leq i \leq n\}$. The *region graph* $G(\mathscr{D})$ of $\mathscr{D}$ is the tuple $(\mathcal{W}, \mathcal{AP}, w_0, \mathcal{W}_F, \dashrightarrow)$ where $\mathcal{W} = \mathcal{M} \times \mathcal{R}$, $w_0 = (m_0, R_1)$, and $\mathcal{W}_F = \mathcal{M}_F \times \mathcal{R}$. The transition relation

$$\dashrightarrow \subseteq \mathcal{W} \times ((2^{\mathcal{AP}} \times 2^{\mathcal{AP}} \times \{\circlearrowleft, \rightsquigarrow\}) \cup \{\delta\}) \times \mathcal{W}$$

is the smallest relation such that

- $(m, R) \overset{\mathcal{A},\mathcal{A}',r}{\dashrightarrow} (m', R')$ if $m \xrightarrow{\mathcal{A},\mathcal{A}',g,r} m'$, $R \models g$, and $R' = \begin{cases} R_1 & \text{if } r = \circledcirc \\ R & \text{otherwise,} \end{cases}$
- $(m, R) \overset{\delta}{\dashrightarrow} (m, R')$ if $R' = scr(R)$.

Intuitively, a region graph is a *finite abstract representation* of a DTA. The state of a DTA is determined by the current location and the clock valuation, which results in an (uncountably) infinite number of possible states. A region graph partitions the DTA states according to their locations and the regions of the clock valuations. The rationale behind this representation is that clock valuations of the same region satisfy the same clock constraints appearing as guards in the DTA.

Example 13: Region Graph ▷Example 13

In Figure 6.2 we illustrate the region graph of the DTA from Example 11, where the edge label true (else) indicates that all (the otherwise missing combinations of) pre- and post-input symbols are accepted.

Definition 39: Product of CTMC and Region Graph ▷Definition 39

For a DTA $\mathscr{D}$ with region graph $G(\mathscr{D}) = (\mathcal{W}, \mathcal{AP}, w_0, \mathcal{W}_F, \dashrightarrow)$ and a CTMC $\mathscr{C} = (\mathcal{S}, \mathbf{Q}, \pi(0))$ labeled by a labeling function $L : \mathcal{S} \to \mathcal{AP}$, the product [CHKM11] of $\mathscr{C}$ and $G(\mathscr{D})$ is defined as the tuple $\mathscr{Z} = \mathscr{C} \otimes G(\mathscr{D}) = (\mathcal{V}, \to, \pi(0)', \mathcal{V}_F)$ where $\mathcal{V} = \mathcal{S} \times \mathcal{W}$, $\mathcal{V}_F = \mathcal{S} \times \mathcal{W}_F$, and $\pi_{(s,w)}(0)' = \pi_s(0)$ if $w = w_0$ and $\pi_{(s,w)}(0)' = 0$, otherwise. Further, $\to \subseteq \mathcal{V} \times ((\mathbb{R}^+ \times \{\circledcirc, \rightsquigarrow\}) \cup \{\delta\}) \times \mathcal{V}$ is defined such that

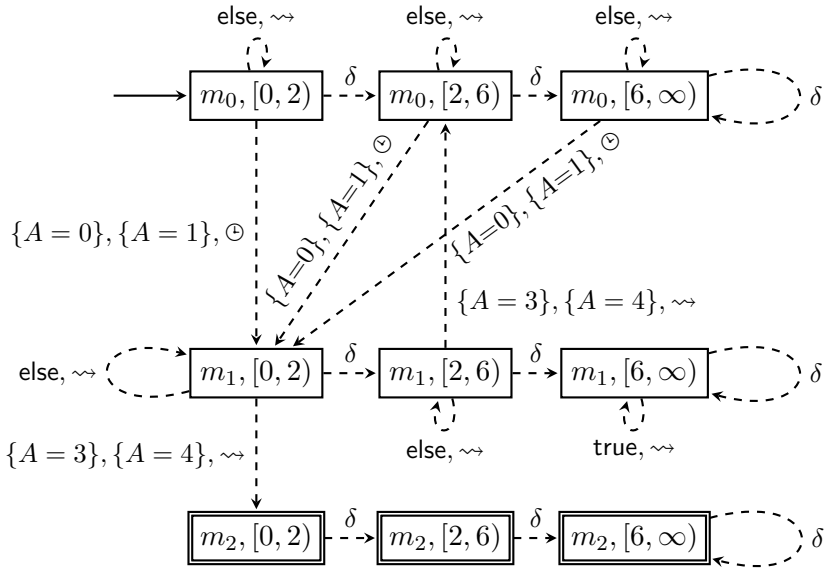


Figure 6.2: Region graph of the DTA from Example 11.

- $(s, w) \xrightarrow{\delta} (s, w')$ iff $w \overset{\delta}{\dashrightarrow} w'$ and
- $(s, w) \xrightarrow{q,r} (s', w')$ iff $q = \mathbf{Q}_{ss'} > 0$ and $w \overset{\mathcal{A},\mathcal{A}',r}{\dashrightarrow} w'$ with $L(s) \in \mathcal{A}$ and $L(s') \in A'$.

In the following, we will use the shorthand notation $\mathscr{Z}$ for $\mathscr{C} \otimes G(\mathscr{D})$. Note that process $\mathscr{Z}$ can be interpreted as a *Markov renewal process* (MRP) [Ger00], that is, a process very similar to a CTMC except that the residence times in the states may not follow an exponential distribution. Here, this occurs when the process moves to a different clock region. We omit a formal definition of Markov renewal processes here as it is not of importance for the algorithm that we propose in the sequel. We refer to [Ger00, DHS09] for more details as well as to [CHKM11] where the more general class of piecewise deterministic Markov processes is considered.

Theorem 18: Acceptance Probability of DTA paths ▷Theorem 18

For a CTMC $\mathscr{C}$ and a DTA $\mathscr{D}$, the probability measure of all paths in $\mathscr{C}$ that are accepted by $\mathscr{D}$ equals the probability of eventually reaching a state $v \in V_F$ in the Markov renewal process $\mathscr{Z} = (\mathcal{V}, \rightarrow, \pi(0), \mathcal{V}_F)$.

Proof: We refer to [CHKM11] for the respective proof. □

We remark that the above theorem is a slight variation of the one by Chen et al. since we do not consider the embedded process of $\mathscr{Z}$. The proof of Theorem 1, however, goes along the same lines as the proof in [CHKM11]. The reason that we do not need embedding here is that, for the approximation algorithm presented in the sequel, a conversion to discrete time does not have any numerical advantages.

In the next section we will see that $\mathscr{Z}$ can be analyzed using several CTMCs (one for each region). Thus, we will approximate the probability measure of all paths in $\mathscr{C}$ that are accepted by $\mathscr{D}$ based on a decomposition of $\mathscr{Z}$.

6.4 Iterative Computation of the Acceptance Probability and its Derivatives

6.4.1 Iterative Steady State Analysis

The iterative algorithm that will be developed in this chapter relies on Algorithm 2 as described in Chapter 2.4.3 for truncation based transient analysis of the product $\mathscr{Z}$. More precisely, this algorithm will be used to compute the probability of acceptance of a DTA by approximating the limiting distributions $\pi = \lim_{t\to\infty} \pi(t)$ of several CTMCs. For this, we keep both, the hash map of the current probabilities p as well as the hash map of the old probabilities p' that are updated in each transient iteration step. During each step of the integration we also check whether $\pi(t)$ has converged by monitoring the maximal difference

$$\max_{s\in \mathrm{dom}(p)\,\cup\,\mathrm{dom}(p')} |p(s) - p'(s)|.$$

Recall that for hash map p (p') holds $p(s) = 0$ ($p'(s) = 0$) if $s \notin \mathrm{dom}(p)$ ($s \notin \mathrm{dom}(p')$). If the maximal difference is small, then convergence is very likely. We can, however, never guarantee that convergence is reached when the system is multimodal as argued in Chapter 2.5.3.

For ergodic multimodal Markov chains, this problem can be avoided by constructing geometric bounds beforehand as described in Chapter 3.1. However, for the case studies that we present in this chapter, such a preprocessing was not necessary and the convergence check explained above was sufficient to obtain an accurate approximation of the limiting distribution.

6.4.2 Acceptance Probabilities

In this section, we consider the Markov renewal process

$$\mathscr{Z} = (\mathcal{V}, \to, \pi(0)', \mathcal{V}_F)$$

and present an algorithm for the approximation of the probability to reach an accepting state, that is, a state in $\mathcal{V}_F$. As shown by Chen et al., this probability is equal to the probability measure of all accepting paths of $\mathscr{C}$ with respect to $\mathscr{D}$ [CHKM11, Theorem 1]. There, the authors

also show that the acceptance probability in $\mathscr{Z}$ can be computed by considering *column CTMCs* as defined below.

Definition 40: Column ▷Definition 40

For $i \in \{1, \ldots, n\}$ where n relates to the number of clock regions, we define the *column* $\mathcal{V}_i \subseteq \mathcal{V}$ of $\mathscr{Z}$ as the set $\mathcal{V}_i = \mathcal{S} \times \mathcal{M} \times \{[c_{i-1}, c_i)\}$ where $\mathcal{S}$ is the state space of $\mathscr{C}$, $\mathcal{M}$ is the state space of $\mathscr{D}$, and $[c_{i-1}, c_i)$ is the i-th clock region of $\mathscr{D}$. Further, we let $\widehat{\mathcal{V}} = \mathcal{S} \times \mathcal{M} \times \{\clockface\}$.

Definition 41: Column CTMC ▷Definition 41

Let $\mathscr{Z}$ be the product of the CTMC $\mathscr{C}$ (labeled with atomic propositions from the set $\mathcal{AP}$ by a labeling function L) and the region graph of the DTA $\mathscr{D}$. For $i \in \{1, \ldots, n\}$ let $\pi^{(i)}$ be a distribution. The *column CTMC* $\mathscr{C}^{(i)} = (\mathcal{S}^{(i)}, \mathcal{AP}, \mathbf{Q}^{(i)}, L^{(i)}, \pi^{(i)})$ is defined by $\mathcal{S}^{(i)} = \widehat{\mathcal{V}} \cup \mathcal{V}_i$, $L^{(i)} : \mathcal{S}^{(i)} \to \mathcal{AP}$ with $L^{(i)}(s, m, r) = L(s)$, and $\mathbf{Q}^{(i)} = \left[\mathbf{Q}^{(i)}_{uv}\right]_{u,v \in \mathcal{S}^{(i)}}$ with

$$\mathbf{Q}^{(i)}_{uv} = \begin{cases} q & \text{if } u \xrightarrow{q,\rightsquigarrow} v \wedge u \in \mathcal{V}_i \wedge v \in \mathcal{V}_i \\ & \text{or } u \xrightarrow{q,\clockface} (s, m, R_1) \wedge u \in \mathcal{V}_i \wedge \\ & v = (s, m, \clockface) \in \widehat{\mathcal{V}} \\ -\sum_{v \neq u} \mathbf{Q}^{(i)}_{uv} & \text{if } u = v, u \in \mathcal{V}_i \\ 0 & \text{otherwise.} \end{cases}$$

The idea behind the column CTMC construction is that for each region of the DTA, in Definition 39, we consider a CTMC that describes the probability flow, within a certain clock region, of the Markov renewal process $\mathscr{Z}$. Obviously, if we do not enter a new clock region, the probability flow inside the i-th clock region is as in the CTMC associated with generator $\mathbf{Q}^{(i)}$. Entering a new clock region occurs if we either

stay longer in the region than $c_i - c_{i-1}$ time units or if a reset transition is taken. Therefore, we accumulate the probability mass of all paths, in which a reset occurs, in the states of the set $\widehat{\mathcal{V}}$. We remark that this is exactly the behavior of the MRP $\mathscr{Z}$, that is, it performs Markovian jumps according to $\mathbf{Q}^{(i)}$ as long as the clock region does not change or a reset occurs.

▷Example 14

Example 14: Column CTMCs

In Figures 6.3, 6.4, and 6.5 we illustrate the column CTMCs of our running examples, that is, the product of the CTMC from Example 10 and the region graph of the DTA from Example 13 where we ignore the initial distributions and assume that states $(k, m_j, [c_{i-1}, c_i))$ and $(k, m_j, \circlearrowleft)$ have label $A = k$ for $k = 0, 1, \ldots$.

The main idea behind the approximation of acceptance probabilities is that we iteratively *run* each column CTMC for time intervals that are equal to the length of the corresponding clock region. After each step of the iteration we move the probability mass of the states in $\mathcal{V}_i$ to the corresponding state copies in $\mathcal{V}_{i+1}$. The probability mass of the states in $\widehat{\mathcal{V}}$ is moved to the corresponding state copies in $\mathcal{V}_1$ (since we have a reset in the MRP). This shift of probability mass can, however, not be done before the successor region and the first region reach the end of their current time interval. Otherwise we would mix probability mass with different clock values. We describe the computation of the acceptance probability in Algorithm 12.

We use vectors $p_1, \ldots, p_n$ for the current probability distributions in the column CTMCs $\mathscr{C}^{(1)}, \ldots, \mathscr{C}^{(n)}$. Initially, only $\mathscr{C}^{(1)}$ is considered since in line 4, the initial distribution $\pi(0)$ is only assigned to states in $\mathcal{S}^{(1)}$. The remaining column CTMCs are considered when probability mass arrives in the corresponding clock regions. The while loop in line 5 is executed until the probability mass in the original CTMC converges, that is, for the convergence check we construct a probability distribution p such that

$$p(s) = \sum_{i=1}^{n} \sum_{m \in \mathcal{M}} p_i(s, m, R_i).$$

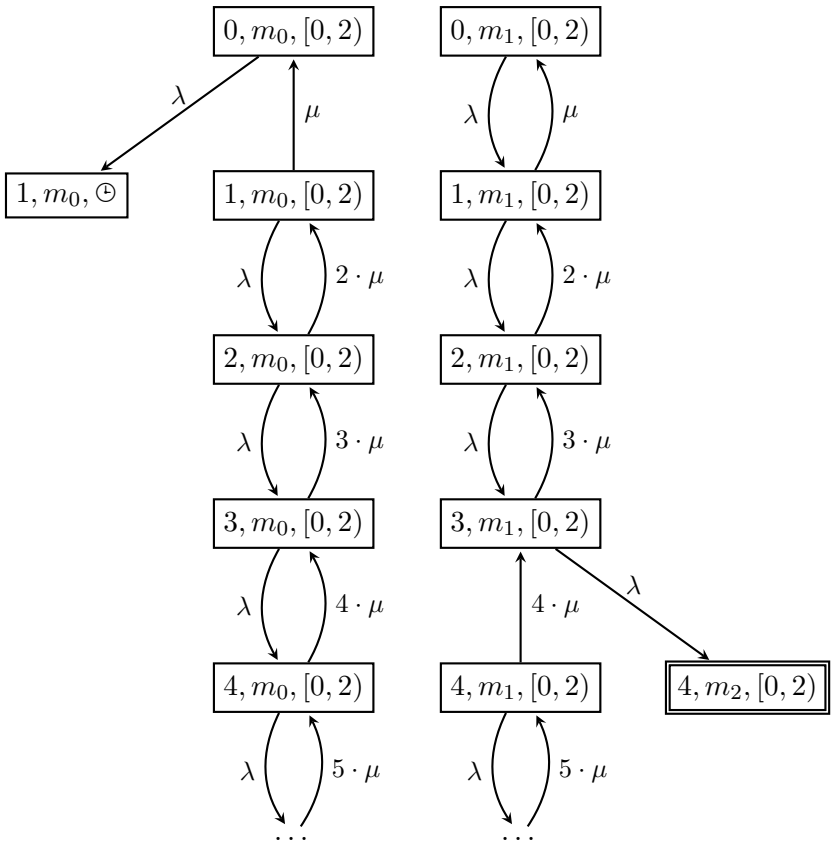


Figure 6.3: Column CTMC $\mathscr{C}^{(1)}$ of the product of the CTMC from Example 10 and the region graph of the DTA in Example 13.

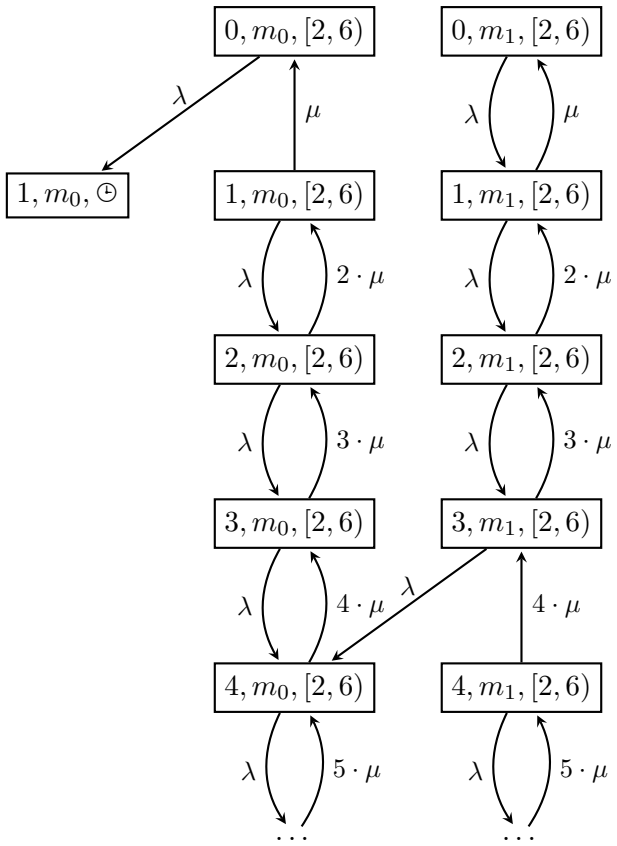


Figure 6.4: Column CTMC $\mathscr{C}^{(2)}$ of the product of the CTMC from Example 10 and the region graph of the DTA in Example 13.

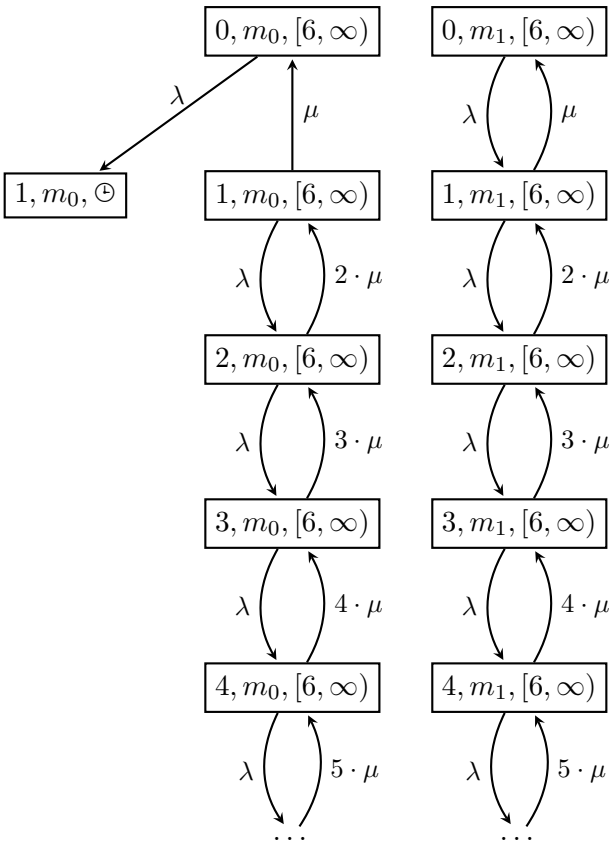


Figure 6.5: Column CTMC $\mathscr{C}^{(3)}$ of the product of the CTMC from Example 10 and the region graph of the DTA in Example 13.

Algorithm 12 accept($\mathscr{C} = (\mathbf{Q}, \mathcal{S}, \pi(0))$, $L : \mathcal{S} \to \mathcal{AP}$, $\mathscr{D} = (\mathcal{M}, \mathcal{AP}, m_0, \mathcal{M}_F, \to)$,$\delta$)

$\Delta_{\max} \leftarrow \max_{1 \le i < n}\{c_i - c_{i-1}\}$
$\forall 1 \le i \le n$: $p_i \leftarrow$ new hash map $\mathcal{S}^{(i)} \to [0,1]$
$\widehat{p} \leftarrow$ new hash map $\widehat{\mathcal{V}} \to [0,1]$
$\forall s \in \mathcal{S}$ with $\pi_s(0) > 0$: $p_1(s, m_0, R_1) \leftarrow \pi_s(0)$
while no convergence **do**
    reset($\widehat{p}$)
    for $i = 1 \ldots n$ **do**
        transient($p_i, \mathbf{Q}^{(i)}, \mathcal{S}^{(i)}, \min\{c_i - c_{i-1}, \Delta_{\max}\}, \delta$, RK4)
    end for
    for $i = n \ldots 1$ **do**
        $\forall v \in \widehat{\mathcal{V}}$: $\widehat{p}(v) \leftarrow \widehat{p}(v) + p_i(v)$ and remove(p_i, v)
        if $i \neq n$ **then**
            for all $(s, m, R_i) \in \mathcal{V}_i$ **do**
                $p_{i+1}(s, m, R_{i+1}) \leftarrow p_{i+1}(s, m, R_{i+1}) + p_i(s, m, R_i)$
                remove($p_i, (s, m, R_i)$)
            end for
        end if
    end for
    $\forall (s, m, R_1) \in \mathcal{V}_1$: $p_1(s, m, R_1) \leftarrow p_1(s, m, R_1) + \widehat{p}(s, m, \clock)$
end while
$p_{acc} \leftarrow \sum_{i=1}^{n} \sum_{v \in \mathcal{V}_F} p_i(v)$
$e \leftarrow 1 - \sum_{i=1}^{n} \sum_{v \in \mathcal{V}_i \cup \widehat{\mathcal{V}}} p_i(v)$
return $\begin{bmatrix} p_{acc} & e \end{bmatrix}$

Note that only the convergence of the original CTMC can be guaranteed (since we assume that the limiting distribution of the original CTMC exists) while the probability distribution of the column CTMCs may not converge in time due to reset cycles in the DTA. Therefore, we check the convergence of p as explained at the end of this chapter which ensures the convergence of the acceptance probability

$$\sum_{i=1}^{n} \sum_{v \in \mathcal{V}_F} p_i(v)$$

in line 21 of Algorithm 12.

Inside the while loop we perform a transient analysis of all column CTMCs for a time equal to the length of the clock region. For this we use truncation based transient analysis as stated in Algorithm 2 which is efficient even if the CTMCs are very large or infinite. It is important to note that these approximations can be parallelized in a straightforward way and indeed, in our implementation, we perform these transient analyses in parallel threads. For the last column CTMC $\mathscr{C}^{(n)}$ the length of the time interval $c_n - c_{n-1}$ is infinite and thus we perform the transient analysis for the time interval $\Delta_{\max} = \max_{1 \leq i < n}\{c_i - c_{i-1}\}$ instead. Note that one can choose any time interval of length at least $\Delta_{\max}$ for that last column CTMC since the probability mass is either moved to $\mathscr{C}^{(1)}$ due to the occurrence of a reset or it remains in $\mathscr{C}^{(n)}$. In the for loop of line 10, we first move the probability mass where a reset occurred to the auxiliary hash map $\widehat{p}$ (line 11) and then we move the remaining probability mass of region R_i to region R_{i+1}. This step corresponds to a δ-transition in the MRP. Note that we have nothing to move in the last region R_n since, if no reset occurs, the probability mass will remain there. By traversing the regions from R_n down to R_1, we ensure that the probability mass is correctly divided among the regions.

If the numerical integration in line 8 was exact, it would hold that

$$\sum_{i=1}^{n} \sum_{v \in \mathcal{V}_i \cup \widehat{\mathcal{V}}} p_i(v) = 1$$

since during the whole procedure probability mass is only moved between the states of the column CTMCs. Therefore,

$$1 - \sum_{i=1}^{n} \sum_{v \in \mathcal{V}_i \cup \widehat{\mathcal{V}}} p_i(v)$$

gives us an estimation of the approximation error similar to the estimation in Algorithm 2 itself. Note that this is only an approximation of the total approximation error since further errors may be introduced due to the numerical integration method (explicit fourth-order Runge-Kutta) and due to the convergence check. If, however, the time step of the numerical integration method as well as the threshold used for the convergence check are small, then the estimation of the approximation error will be accurate because the error that originates from the truncation of insignificant states dominates the other approximation errors.

▷Example 15

Example 15: Approximation of p_{acc}

For the CTMC of Example 10 and the DTA in Example 11 the probability of acceptance is around 0.01535. For instance, if we choose ϵ =1e–15, after five steps of the outer while loop in Algorithm 12, we get $p_{acc} = 0.0152140414$, after eight iterations we get $p_{acc} = 0.0153512141$, and after twelve steps we get $p_{acc} = 0.0153523506$. The total number of significant states in the column CTMCs is about 50 and the running time is less than 1 second.

▷Theorem 19

Theorem 19: Correctness of Algorithm 12

Assuming the numerical integration in line 8 to be exact, Algorithm 12 computes the acceptance probability, that is, the probability of eventually reaching a state $v \in \mathcal{V}_F$ in the Markov renewal process $\mathscr{Z} = \mathscr{C} \otimes G(\mathscr{D}) = (\mathcal{V}, \rightarrow, \pi(0)', \mathcal{V}_F)$.

Proof: For the correctness of Algorithm 12 we use similar arguments as in [BCH+11]. We first note that $\mathscr{Z}$ has the same initial probabilities

as the column CTMCs. We omit an induction on the number of steps in $\mathscr{Z}$ but sketch the proof by considering the three cases for the transitions in $\mathscr{Z}$:

1. Assume that $\mathscr{Z}$ takes a transition of the form

$$(s, m, R_i) \xrightarrow{\delta} (s, m, R_{i+1})$$

for $i < n$. This occurs if $\mathscr{Z}$ remained in R_i for $c_i - c_{i-1}$ time units (and no reset transition was taken). Since in line 8 we integrate the i-th column CTMC for $c_i - c_{i-1}$ time units and afterwards move the probability mass (without reset) of region R_i to R_{i+1}, this transition is correctly mimicked in Algorithm 12. In the case

$$(s, m, R_n) \xrightarrow{\delta} (s, m, R_n)$$

the probability mass remains in region R_n (note that $i \neq n$ in line 12).

2. Consider a transition of the form

$$(s, m, R_i) \xrightarrow{q,\rightsquigarrow} (s', m', R_i).$$

Such transitions are considered during the numerical integration in line 8. Note that the transition rates of $\mathscr{Z}$ and the column CTMCs are identical. Thus, the probability flow in the column CTMCs and in $\mathscr{Z}$ is the same.

3. Assume that $\mathscr{Z}$ takes a transition of the form

$$(s, m, R_i) \xrightarrow{q,\circledcirc} (s', m', R_1).$$

In this case the probability mass is moved to $\widehat{\mathcal{V}}$ during the numerical integration in line 8 (which is due to the structure of the column CTMCs). Then, before the next iteration of the while loop, this probability mass is moved to the first clock region in line 17. Thus, in the next iteration the process continues in the first clock region.

□

6.4.3 Maximization of the Acceptance Probability:

In order to maximize the probability of acceptance when the CTMC is parametric in one or more variables, we need to compute the derivatives of the acceptance probability with respect to those variables. Moreover, these derivative depend on the derivatives of the transient distribution of the CTMC, whose formulae will be developed in the following the lines of [BRT88].

6.4.3.1 Derivatives of the Transient Distribution of a CTMC

Let λ_i be a parameter and assume that the generator matrix $\mathbf{Q}$ is dependent on λ_i. For simplicity, we do not consider the case where the initial conditions are dependent on λ_i. Similar results to those stated below can be derived in that case. Let $\rho_i(t)$ be the vector $\frac{\partial}{\partial \lambda_i}\pi(t)$. Taking the derivative of Chapman-Kolmogorov Equation (2.15) yields

$$\frac{d}{dt}\rho_i(t) = \rho_i(t) \cdot \mathbf{Q} + \pi(t) \cdot \frac{\partial}{\partial \lambda_i}\mathbf{Q} \tag{6.2}$$

with initial condition $\rho_i(t) = \mathbf{0}$. Note that the chain rule is applicable, since $\mathbf{Q}$ as well as $\pi(t)$ depend on λ_i.

Thus, integrating Equations (2.15) and (6.2) simultaneously yields $\pi(t)$ and $\rho_i(t)$. As for $\pi(t)$, during the computation of $\rho_i(t)$ we only consider significant states. In other words, we use the same dynamic state space truncation during the integration of Equation (6.2) that is used for Equation (2.15). If we have several parameters, we consider multiple vectors $\rho_i(t)$ and always truncate the state space according to the threshold δ for the probabilities $p(s)$ for states $s \in \mathrm{dom}(p)$. Recall that we assume $\mathbf{Q}$ to be such that $\pi(t)$ converges. Moreover, we assume that the dependence of $\mathbf{Q}$ on λ_i is such that the derivatives converge as well. In our case studies, the derivatives always converged.

When maximizing (or minimizing) the probability of a certain event, it is useful to be able to approximate not only the first but also the second derivatives. Then, optimization approaches based on gradient descent will perform better since the step size can be adjusted dynamically and thus local optima are found faster. Assume that the generator matrix Q depends on the parameters λ_i and λ_j and that

$\varrho_{ij}(t) = \frac{\partial^2}{\partial\lambda_i\partial\lambda_j}\pi(t)$. Then

$$\frac{d}{dt}\varrho_{ij}(t) = \varrho_{ij}(t)\cdot\mathbf{Q} + \rho_i(t)\cdot\frac{\partial}{\partial\lambda_j}\mathbf{Q} + \rho_j(t)\cdot\frac{\partial}{\partial\lambda_i}\mathbf{Q} + \pi(t)\cdot\frac{\partial^2}{\partial\lambda_i\partial\lambda_j}\mathbf{Q}$$

with initial condition $\varrho_{ij}(0) = \mathbf{0}$. Again, we can integrate $\varrho_{ij}(t)$ simultaneously with $\pi(t)$ and the first order derivatives on the truncated dynamic state space. We remark that if we have k parameters then we need to store $0.5\cdot k\cdot(k+1)$ second order derivatives since we have to consider all possible combinations of two parameters.

6.4.3.2 Derivatives of the Acceptance Probability

Finally, we discuss how to maximize the probability of acceptance. It is straightforward to extend Algorithm 12 such that besides the state probabilities of the column CTMCs also the derivatives with respect to certain parameters are approximated simultaneously as discussed in the previous section. Technically, the first order derivatives require one additional hash maps per parameter λ_i and column CTMC and the second order derivatives require $\frac{k(k+1)}{2}$ additional hash maps for each column CTMC where k is the number of parameters. Then, as a result, the algorithm returns not only the acceptance probability but also its derivatives

$$\frac{\partial}{\partial\lambda_i}p_{acc} = \sum_{m=1}^{n}\sum_{v\in\mathcal{V}_F}\frac{\partial}{\partial\lambda_i}p_m(v) \text{ and} \tag{6.3}$$

$$\frac{\partial^2}{\partial\lambda_i\partial\lambda_j}p_{acc} = \sum_{m=1}^{n}\sum_{v\in\mathcal{V}_F}\frac{\partial^2}{\partial\lambda_i\partial\lambda_j}p_m(v). \tag{6.4}$$

6.4.3.3 Global Optimization

Our global optimization method gets as input a constraint interval for each parameter λ_i and approximates the acceptance probability and the derivatives for particular choices of λ_i. It uses a gradient decent approach to find local maxima and a heuristic for the starting points in the interval in which it tries to find all maxima. Note that, since this is

in general a nonlinear optimization problem, it may still be possible that the global maximum is not found. Moreover, we remark that, since we provide second order derivatives as well, local maxima are found using fewer calls of the approximation algorithm compared to the case where second order derivatives are not approximated since the step size of the gradient descent method can be determined efficiently using the Hessian matrix of the acceptance probability consisting of the second derivatives.

6.5 Case Studies

We consider two case studies from systems biology in the sequel. The *exclusive switch* and the *repressilator*. For that, we implemented Algorithm 12 in C++ and ran experiments on an Intel Core i7 at 2.8 GHz with 8 GB of main memory. For the maximization of the acceptance probability we linked our code to MATLAB's global search routine, which calls our C++ program with different parameters. Our implementation also approximates the first and second order derivatives of the acceptance probability, which is needed for the gradient descent part in global search. The running time of our method depends on the tightness of the intervals that we use as constraints for the unknown parameters as well as on the number of starting points of the global search procedure. We chose intervals that correspond to the order of magnitude of the parameters, that is, if $\lambda_i \in O(10^n)$ for some $n \in \mathbb{Z}$ then we use the interval $[10^{n-1}, 10^{n+1}]$ as constraint for λ_i. For example, if $\lambda_i = 0.1$ then $n = -1$ and we use the interval $[0.01, 1]$. Moreover, for global search we used 10 starting points for both case studies and for our experiments, we chose a probability truncation threshold of $\delta =$1e-15. For both case studies the distribution of the underlying CTMC converges when the acceptance probability converges. We therefore stopped the iteration when the relative difference of the acceptance probability became less than 0.001. Note that in general, one has to check the convergence of the distribution of the underlying CTMC, since the acceptance probability may not alter for several iteration steps even though the distribution of the CTMC has not yet reached equilibrium. For more details, we refer to the discussion in Chapter 2.5.3.

6.5.1 Exclusive Switch

We consider the *exclusive switch* as described in Model 3 with the labeling function L labeling states $\mathbf{x} \in \mathcal{S}$ by their variable valuation, that is, $L(\mathbf{x})$ equals $P_1 = \mathbf{x}_1$, $P_2 = \mathbf{x}_2$, $G = \mathbf{x}_3$, $G.P_1 = \mathbf{x}_4$, $G.P_2 = \mathbf{x}_5$. The reaction rate constants are

$$\rho_1 = 0.2, \rho_2 = 0.3, \delta_1 = \delta_2 = 0.005, \beta_1 = 0.003, \beta_2 = 0.002, \nu_1 = \nu_2 = 0.2$$

and the initial condition is $\pi_{\mathbf{e}_3}(0) = 1$. A plot of the limiting distribution of the CTMC is depicted in Figure 6.6. The two dark regions where most of the probability mass is located are the attractor regions.

6.5.1.1 Switching time

We are interested in the time needed to go from one attractor region to the other and back within T time units. Note that such switching events and times are of particular importance in gene regulatory networks [LLBB07]. In the DTA in Figure 6.7 we define the two attractor regions using bounds on the two protein populations (see boxes in the illustration in Figure 6.6). The DTA accepts paths that switch from one attractor to the other and back within T time units as illustrated by the red dashed arrow in Figure 6.6. Note that in Figure 6.7 we describe subsets of $\mathcal{AP}$ by conjunctions of inequalities. For example, $\{P_1{<}A_1 \wedge P_2{>}A_2\}$ refers to the set of all labels where P_1 is smaller than the threshold A_1 and P_2 is greater than A_2. For the thresholds we used two parameter sets, namely

- $\text{PSet}_1 : (A_1, A_2) = (15, 40), (B_1, B_2) = (25, 30)$ and
- $\text{PSet}_2 : (A_1, A_2) = (10, 50), (B_1, B_2) = (30, 20)$.

Here, A_1, A_2 define the upper left box and B_1, B_2 define the lower right box (second attractor). Moreover, we considered different time bounds $T \in \{200, 500, 1000\}$, chose $T = 1000$, and approximated the acceptance probabilities using Algorithm 12. The corresponding results are listed in Table 6.1.

When we decrease T, the acceptance probability, the run-time, and the average number of significant states becomes smaller. For PSet_2, the acceptance probability is smaller than for PSet_1 since the boxes

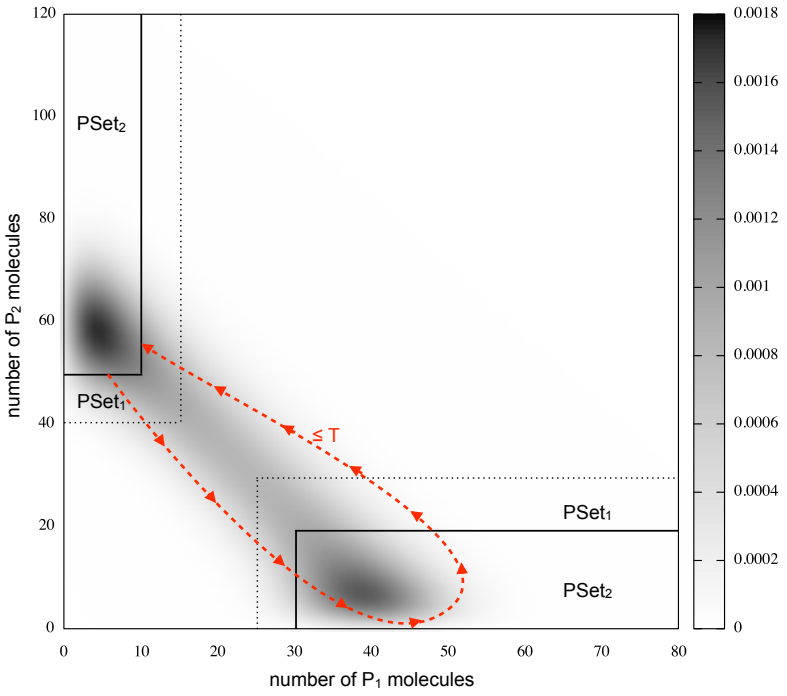


Figure 6.6: Limiting distribution of the exclusive switch model and attractor bounding boxes for parameter sets PSet_1 and PSet_2. The arrow shows an example path of the underlying CTMC that is accepted by the DTA in Figure 6.7 for parameter set 1 assuming the total time that path needed to traverse the attractor regions is not larger than T.

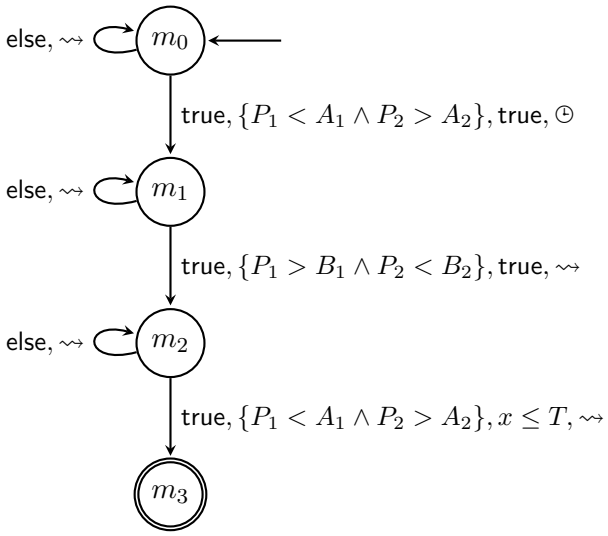


Figure 6.7: DTA specification for the exclusive switch expressing a switching time less than T.

PSet_i	T	p_{acc}	states	time (s)
1	200	0.1154	68658	433
1	500	0.2313	71285	506
1	1000	0.4062	71021	516
2	200	0.004757	69343	593
2	500	0.02522	72983	640
2	1000	0.1069	73062	831

Table 6.1: Results of the exclusive switch case study.

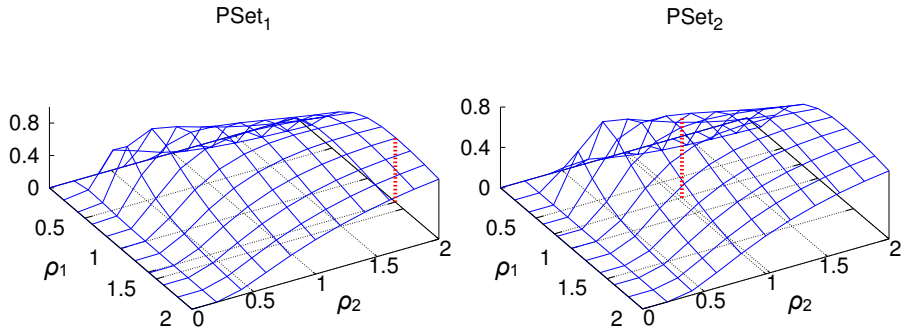


Figure 6.8: Acceptance probabilities for different instances of (ρ_1, ρ_2).

around the attractors are tighter (see Figure 6.6). The average number of significant states is similar for both PSet_1 and PSet_2, but the runtime of the algorithm is slightly larger for PSet_2. This is because it takes longer for the acceptance probability to converge.

6.5.1.2 Optimizing the Switching Time

In this section, we assume that the rate constants ρ_1 and ρ_2 of protein production are parameters and we seek a pair (ρ_1, ρ_2) for which the acceptance probability becomes maximal. We use the technique described in Chapter 6.4.3 to approximate the derivatives of the acceptance probability with respect to ρ_1 and ρ_2 and link our implementation to MATLAB's global search. We remark that the optimization increases the running time significantly since we have to compute acceptance probabilities and derivatives for many instances of (ρ_1, ρ_2). Note that it was not possible to find parameters such that the acceptance probability is very close to one since there will always be paths on which the process stays too long in one of the attractors. Moreover, it is not obvious whether increasing/decreasing ρ_1 and ρ_2 increases/decreases the acceptance probability since the relationship between p_{acc} and ρ_1, ρ_2 is not linear. If we increase the constants, then the attractors are too far away from each other and it becomes unlikely that, once an attractor is entered, the process leaves it. If, on the other hand, ρ_1 and ρ_2 are too small, then reaching the two boxes becomes too unlikely. For PSet_1

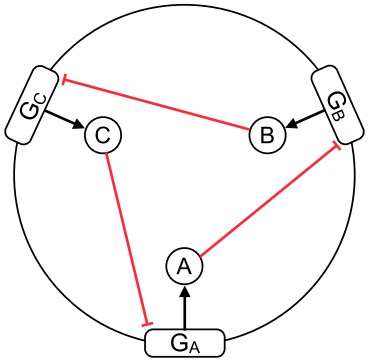


Figure 6.9: Illustration of the repressilator.

we achieve a maximal acceptance probability of $p_{acc} = 0.7990$ with the constants $\rho_1 = 1.3890$ and $\rho_2 = 2$. For PSet_2 we found $\rho_1 = 0.7603$ and $\rho_2 = 1.0344$ with an acceptance probability of $p_{acc} = 0.7865$. Note that for ρ_1 and ρ_2 we chose the constraints $\rho_1, \rho_2 \in [0.01, 2.0]$. In both cases the running time of the optimization was about two days.

In Figure 6.8, we plot for both parameter sets the maximal acceptance probability (dotted red line) and the acceptance probabilities (solid blue grid) for different instances (ρ_1, ρ_2). As mentioned before, the total run-time of the optimization depends on the number and the choice of starting points. The run-time of the gradient descent method which is used for each starting point depends on how far the starting point is from the optimal one. For each starting point it took the gradient descent method on average about 5 hours until convergence with up to ten iterations. In each iteration, the computation of the acceptance probabilities and its derivatives took about half an hour depending on the parameters.

6.5.2 Repressilator

The other case study is the *repressilator* model as described in Model 9.

▷Model 9

Model 9: Repressilator

The repressilator [EL00] is a synthetic, self-regulating, oscillatory gene network consisting of three protein types A, B, and C encoded by three genes G_A, G_B, and G_C, respectively. We illustrate the relationships between the genes in Figure 6.9 and the chemical reactions are as follows.

$$G_A \xrightarrow{\rho_A} G_A + A, \quad G_B \xrightarrow{\rho_B} G_B + B, \quad G_C \xrightarrow{\rho_C} G_C + C,$$

$$A \xrightarrow{\delta_A} \varnothing, \quad B \xrightarrow{\delta_B} \varnothing, \quad C \xrightarrow{\delta_C} \varnothing,$$

$$G_A + C \xrightarrow{\beta_C} \overline{G_A}, \quad G_B + A \xrightarrow{\beta_A} \overline{G_B}, \quad G_C + B \xrightarrow{\beta_B} \overline{G_C}$$

$$\overline{G_A} \xrightarrow{\nu_C} G_A + C, \quad \overline{G_B} \xrightarrow{\nu_A} G_B + A, \quad \overline{G_C} \xrightarrow{\nu_B} G_C + B$$

As long as a gene $G_A(4)/G_B(5)/G_C(6)$ is active, it produces proteins $A(1)/B(2)/C(3)$ at rate $\rho_A/\rho_B/\rho_C$ as modeled in the first row of reactions. The next three reactions model the degradation of proteins $A/B/C$ at rate $\delta_A/\delta_B/\delta_C$. The third line encodes the mutual repression, that is, the proteins A, B, and C bind to the genes G_B, G_C, and G_A at rate β_B, β_C, and β_A suppressing the production of the respective proteins as illustrated in Figure 6.9. In the illustration, a normal arrow means expression and a hooked arrow means repression. The proteins may also unbind from the genes as described in the last three reactions with rates ν_B, ν_C, and ν_A.

We assume symmetric reaction rate constants

$$\rho = \rho_A = \rho_B = \rho_C = 5.0, \delta = \delta_A = \delta_B = \rho_d = 0.1,$$
$$\beta = \beta_A = \beta_B = \beta_C = 1.0, \nu = \nu_A = \nu_B = \nu_C = 1.0,$$

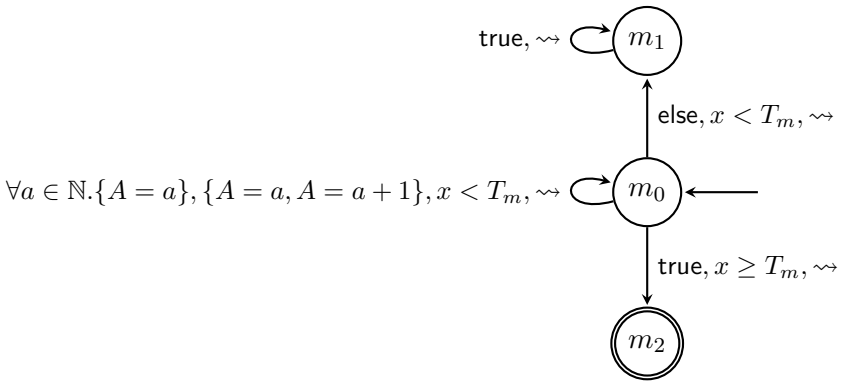


Figure 6.10: DTA specification for the repressilator expressing monotonicity.

and the initial condition $\pi_{\mathbf{0}}(0) = 1$, that is, we have no proteins and all genes start repressed. As in the previous case study, states in the underlying CTMC are labeled by their variable valuation. For the properties we check, it is not important for which chemical species we do the analysis since we have symmetric reaction rate constants. The property expressed by the DTA of Figure 6.10 is related to *monotonicity*, that is, whether the number of molecules of interest is (non-strictly) monotonically increasing along a path until a certain time bound T_m is reached as illustrated in Figure 6.11. This is, for example, of interest during the analysis of noisy oscillations, that is, the higher the probability of monotonicity [BG10], the less probable are deviations in the increasing phase of a period. The initial state m_0 is not left as long the number of molecules stays the same or increases and the time remains below the threshold. If finally time point T_m is reached without any decrease, the DTA accepts by entering state m_2, otherwise a decrease in the number of molecules is captured by a change to state m_1 where the DTA will stay forever and never accept. The acceptance probabilities of the DTA for varying values of T_m are plotted in Figure 6.12. As expected, the probability drops exponentially with increasing time constraint T_m. Still, in about one of ten runs, we get on average a monotonically increasing number of A molecules for at least 5 time

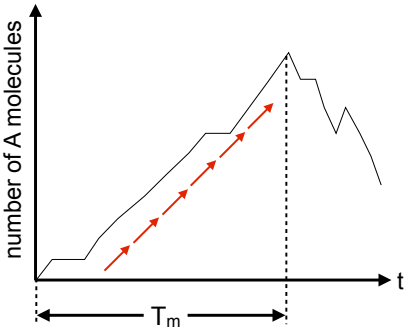


Figure 6.11: Illustration of the monotonicity property.

units. The computation of the acceptance probabilities required only a few minutes for each value of T_m (for example about 2 minutes for $T_m = 20$) and the average number of significant states during each step of the integration was about $250,000$.

We remark that it is trivial to change the DTA to capture (non-strictly) monotonically *decreasing* behavior by changing the post-guard of state m_0 from $\{A = a, A = a + 1\}$ to $\{A = a, A = a - 1\}$ in order to be able to analyze the decreasing phase of a period [BG10] as well.

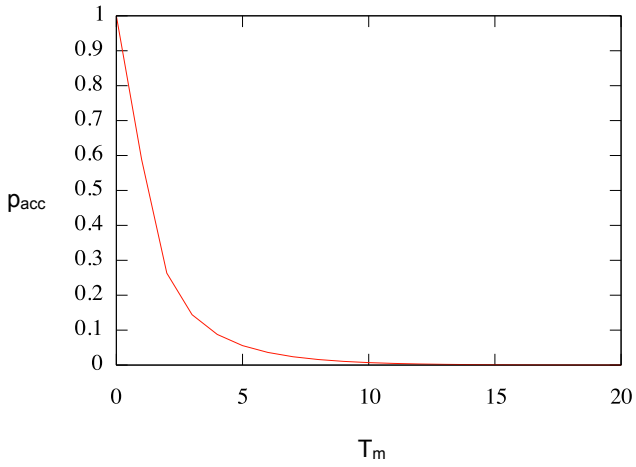


Figure 6.12: Results of the repressilator case study for the monotonicity property.

CHAPTER 7

Characterizing Oscillatory and Noisy Periodic Behavior in Markov Population Models

In the years 1926 respectively 1924, V. Volterra and A. Lotka independently from each other studied the dynamics of predator and prey populations [Vol26, Lot56]. Their key insight was that the amount of both species showed regular *oscillatory fluctuations*, where the predator population followed the prey population. But the phenomenon of oscillation is also present at various granularities and forms in many other systems. Examples are the 24 hour day/night rhythm of living organisms on this planet [BL00] and calcium ion transport between membranes in cells [SMH02]. But oscillations can also be found at macroscopic levels as for example within whole ecospheres like Savannah patches [MWWM07]. Parts of the overview of the literature related to oscillations have been published in [AKS14].

7.1 Continuous-Deterministic Solutions for Noisy Systems

The formalism that was used by Lotka and Volterra were deterministic models, where the expected amount of each species was computed over

time. Those models and the respective analysis techniques are well-understood and are still used widely today to reason about reaction networks in the fields of chemistry and systems biology [FTY11, Kri06, SMH02].

However, as already briefly mentioned in Chapter 2.3.2, recent insights suggest that a completely deterministic approach might not be appropriate in all of the cases since *noise* resulting from low copy numbers of certain populations plays an important role. For example, *circadian clocks*, the basic mechanism behind the 24 hour rhythm, rely on stochastic effects to maintain an oscillatory pattern and to prevent being trapped in an equilibrium state [BL00]. Results like that and others [ARM98, MA97] clearly speak in favor of stochastic modeling. In order to motivate the use of stochastic modeling for certain systems in contrast to traditional techniques based on continuous-deterministic solutions, we will use the *3-way oscillator* model [RMF06, Car06, Car08].

▷**Model 10**

Model 10: 3-Way Oscillator

As its name suggests, the *3-way oscillator* is an oscillatory CRN. The basic mechanism is a positive feedback loop incorporating three species $A(1)$, $B(2)$, and $C(3)$, where A boosts the production of B, B boosts the production of C, and C boosts the production of A, forming a *positive feedback cycle*. The corresponding chemical reactions with conversion rates τ_A, τ_B, τ_C for species A, B, and C are

$$\begin{aligned} A + B &\xrightarrow{\tau_A} 2 \cdot B \\ B + C &\xrightarrow{\tau_B} 2 \cdot C \\ C + A &\xrightarrow{\tau_C} 2 \cdot A \end{aligned}$$

The transition class structure of the 3-way oscillator is

$$\alpha^{(1)}(\mathbf{x}) = \tau_A \cdot \mathbf{x}_A \cdot \mathbf{x}_B, \quad \mathbf{v}^{(1)} = \begin{bmatrix} -1 & 1 & 0 \end{bmatrix},$$

$$\alpha^{(2)}(\mathbf{x}) = \tau_B \cdot \mathbf{x}_B \cdot \mathbf{x}_C, \quad \mathbf{v}^{(2)} = \begin{bmatrix} 0 & -1 & 1 \end{bmatrix},$$

$$\alpha^{(3)}(\mathbf{x}) = \tau_C \cdot \mathbf{x}_C \cdot \mathbf{x}_A, \quad \mathbf{v}^{(3)} = \begin{bmatrix} 1 & 0 & -1 \end{bmatrix}.$$

Depending on the choice of the initial distribution, the extents of the state space varies. More precisely, if for example we fix the initial distribution

$$\pi_{\begin{bmatrix} a & b & c \end{bmatrix}}(0) = 1 \text{ with } \begin{bmatrix} a & b & c \end{bmatrix} \in \mathbb{N}^3$$

the resulting state space is

$$\mathcal{S} = \{ \begin{bmatrix} \mathbf{x}_A & \mathbf{x}_B & \mathbf{x}_C \end{bmatrix} \in \mathbb{N}^3 \mid \mathbf{x}_A + \mathbf{x}_B + \mathbf{x}_C = a + b + c \}$$

with $|\mathcal{S}| = 0.5 \cdot (z+1) \cdot (z+2)$ since the total number of molecules present initially is preserved by all of the three change vectors. A problem of the (undoped) 3-way oscillator in Model 10 is that once a species is depleted, it will never be replenished. From the view of the underlying Markov chain this can be interpreted as follows. Given an initial condition as above and defining $z = a + b + c$, the transient probability mass will eventually be absorbed in the states

$$\begin{bmatrix} z & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & z & 0 \end{bmatrix}, \text{ and } \begin{bmatrix} 0 & 0 & z \end{bmatrix}$$

which have no outgoing transition. In order to prevent such a *deadlock* situation, the three *doping* reactions defined in Model 11 can be used.

Model 11: Doped 3-Way Oscillator ▷**Model 11**

The *doped* 3-way oscillator [Car06] is defined as in Model 10 with three additional chemical reactions

$$\begin{aligned} A &\xrightarrow{\nu_A} B \\ B &\xrightarrow{\nu_B} C \\ C &\xrightarrow{\nu_C} A \end{aligned}$$

The additional transition classes are

$$\alpha^{(4)}(\mathbf{x}) = \nu_A \cdot \mathbf{x}_A, \quad \mathbf{v}^{(4)} = \mathbf{v}^{(1)} = \begin{bmatrix} -1 & 1 & 0 \end{bmatrix},$$

$$\alpha^{(5)}(\mathbf{x}) = \nu_B \cdot \mathbf{x}_B, \quad \mathbf{v}^{(5)} = \mathbf{v}^{(2)} = \begin{bmatrix} 0 & -1 & 1 \end{bmatrix},$$

$$\alpha^{(6)}(\mathbf{x}) = \nu_C \cdot \mathbf{x}_C, \quad \mathbf{v}^{(6)} = \mathbf{v}^{(3)} = \begin{bmatrix} 1 & 0 & -1 \end{bmatrix}.$$

The reachable state space for the initial distribution as specified above is the same as for the undoped 3-way oscillator since the change vectors of the additional reactions are the same as the original ones. Note that the former absorbing states

$$\begin{bmatrix} z & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & z & 0 \end{bmatrix}, \text{ and } \begin{bmatrix} 0 & 0 & z \end{bmatrix}$$

now have outgoing transitions since the doping reactions do not demand the existence of *two* molecules of *different* species anymore. Further, we can now proof that the doped 3-way oscillator is ergodic.

▷Theorem 20

Theorem 20: Ergodicity of the (Doped) 3-Way Oscillator

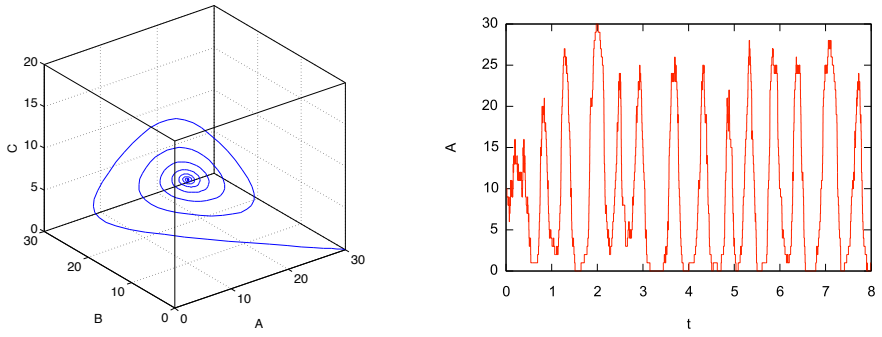
For any initial distribution

$$\pi_{\begin{bmatrix} a & b & c \end{bmatrix}}(0) = 1 \text{ with } \begin{bmatrix} a & b & c \end{bmatrix} \in \mathbb{N}^3,$$

the MPM of the doped 3-way oscillator as specified in Model 11 is *ergodic*.

Proof: Since the state space of the MPM $\mathcal{S} = \{\begin{bmatrix} \mathbf{x}_A & \mathbf{x}_B & \mathbf{x}_C \end{bmatrix} \in \mathbb{N}^3 \mid \mathbf{x}_A + \mathbf{x}_B + \mathbf{x}_C = z\}$ with $z = a + b + c$ is finite, we only need to show irreducibility in order to proof ergodicity. For that, we will show that any state $\begin{bmatrix} \mathbf{x}_A & \mathbf{x}_B & \mathbf{x}_C \end{bmatrix} \in \mathcal{S}$ can reach state $\begin{bmatrix} z & 0 & 0 \end{bmatrix} \in \mathcal{S}$ and vice versa. Starting in state $\begin{bmatrix} \mathbf{x}_A & \mathbf{x}_B & \mathbf{x}_C \end{bmatrix} \in \mathcal{S}$, we can apply reaction 5 (or if possible reaction 2) $\mathbf{x}_B$ times to reach state $\begin{bmatrix} \mathbf{x}_A & 0 & \mathbf{x}_B + \mathbf{x}_C \end{bmatrix}$. Applying reaction 6 (or if possible reaction 3) $\mathbf{x}_B + \mathbf{x}_C$ times brings us to state $\begin{bmatrix} \mathbf{x}_A + \mathbf{x}_B + \mathbf{x}_C & 0 & 0 \end{bmatrix} = \begin{bmatrix} z & 0 & 0 \end{bmatrix}$. In order to get to state $\begin{bmatrix} \mathbf{x}_A & \mathbf{x}_B & \mathbf{x}_C \end{bmatrix}$ from state $\begin{bmatrix} z & 0 & 0 \end{bmatrix}$, we can apply reaction 4 (or if possible reaction 1) $z - \mathbf{x}_A$ times to get to state $\begin{bmatrix} \mathbf{x}_A & z - \mathbf{x}_A & 0 \end{bmatrix}$. From that state, we can apply reaction 5 (or if possible reaction 2) $z - \mathbf{x}_A - \mathbf{x}_B$ times to get to state $\begin{bmatrix} \mathbf{x}_A & \mathbf{x}_B & z - \mathbf{x}_A - \mathbf{x}_B \end{bmatrix}$. Note that in any case $\mathbf{x}_C = z - \mathbf{x}_A - \mathbf{x}_B$ due to the conservation of the total number of molecules. □

Note that for this model, we chose a manual proof to show irreducibility for a wide range of model instances depending on the initial distribution.



(a) Phase plot of the ODE solution. (b) Sample trajectory (observing A).

Figure 7.1: 3-way oscillator

If we were to give a proof for a single model, we could rely on tools like PRISM [KNP11] to automatically compute the reachable state space as well as all (B)SCCs in order to check whether the state space consists of a single (B)SCC.

For further analysis, we will assume the reaction rate constants

$$\tau_A = \tau_B = \tau_C = \nu_A = \nu_B = \nu_C = 1.0.$$

Now, we would like to compare the results of the traditional approach, that is, deriving a continuous-deterministic solution, based on the *law of mass action* of chemistry, with the stochastic view. A phase plot of the deterministic solution with initial concentration $\begin{bmatrix} A & B & C \end{bmatrix} = \begin{bmatrix} 30 & 0 & 0 \end{bmatrix}$ is shown in Figure 7.1a. As can be seen in the plot, the underlying structure is a *damped oscillation*, that is, an initial perturbation caused by the asymmetric initial condition is followed by an oscillatory phase with shrinking amplitude until the equilibrium for all species is reached. Here the equilibrium point is $\begin{bmatrix} 10 & 10 & 10 \end{bmatrix}$. On the other hand, the state space of the MPM induced by the 3-way oscillator's chemical reaction network for initial state

$$\begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \mathbf{x}_3 \end{bmatrix} = \begin{bmatrix} 30 & 0 & 0 \end{bmatrix}$$

is

$$\mathcal{S} = \{ \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \mathbf{x}_3 \end{bmatrix} \in \mathbb{N}^3 \mid \mathbf{x}_1 + \mathbf{x}_2 + \mathbf{x}_3 = 30 \}$$

and due to ergodicity of the model as proofed in Theorem 20 we have for all states $s \in S$ that $\pi_s > 0$. This implies that every state is visited infinitely often which contradicts convergence to any molecule level and further implies everlasting perturbations. Indeed, state $\begin{bmatrix}30 & 0 & 0\end{bmatrix}$ for example has a non-negligible steady state probability of around 0.002 and any simulation run (cf. Figure 7.1b) of the stochastic system almost surely shows never-ending fluctuations. Actually, the deterministic approach reasoning about expected population counts is only justified in the *thermodynamic limit*, when the number of molecules is very high compared to the considered volume. The treatment of a total number of 30 molecules in our case certainly violates that condition and in the following, we restrict to the case where a deterministic treatment is not justified due to stochastic noise.

7.2 Temporal Logic Based Model Checking Approaches

One basic approach that is widely used when analyzing stochastic systems with respect to periodic and oscillatory behavior is *model checking*. More precisely, the property of interest is encoded as a formula in some temporal logic or as some kind of (finite state) automaton and the model checking routine efficiently decides whether the model satisfies the property or not. Model checking based approaches reason about the structure of the models and allow precise statements that hold for sure, unlike simulative approaches which can not give such strong guarantees [BMM09].

In an early model checking approach used to analyze biochemical systems [CRCD+03], the authors make use of the *computation tree-logic* (CTL) [EC82]. For a detailed introduction to CTL, we refer to [BK08]. In the context of reasoning about qualitative aspects of biological models, the idea of requiring an infinite repetition of cyclic behavior was first described in [CRCD+03]. The authors capture that property via the CTL formula

$$\exists\Box\left((P \Rightarrow \exists\Diamond\neg P) \wedge (\neg P \Rightarrow \exists\Diamond P)\right). \tag{7.1}$$

Formula (7.1) demands that there exists at least one path such that whenever some predicate P is satisfied it will be invalid later on, and

vice versa whenever it is not satisfied it will become valid again in the future. Intuitively, that means that there should exist at least one path such that the validity of P alternates forever. CTL model checking was originally applied to *labeled transition systems* (LTS), but a CTMC can be interpreted as a LTS as well by assuming a state transition whenever there is a positive rate between a state and its successor. Using such a construction, one can reason qualitatively about the possible behavior of a CTMC as done in [BMM09] for example. Here, the authors propose the CTL formula

$$\forall\Box(((X_i = k) \Rightarrow \exists\Diamond(X_i \neq k)) \wedge ((X_i \neq k) \Rightarrow \exists\Diamond(X_i = k))), \quad (7.2)$$

in order to query whether a system shows *permanent oscillations*. This formula is similar to Formula (7.1) where the inner formula is strengthened by exchanging the outer $\exists$ by a $\forall$ operator and predicate P is instantiated with $X_i = k$. The intuitive meaning of this formula is that *all* paths should cross a level of k molecules infinitely often. Note that there is an implicit assumption that all oscillations will hit the level of exactly k molecules, that is, the maximum change of the molecule number is one.

The authors are concerned that noise could affect the fluctuations. Thus, they change the previous formula slightly to

$$\forall\Box(((X_i = k) \Rightarrow \exists\Diamond\phi_n) \wedge (\phi_n \Rightarrow \exists\Diamond(X_i = k))) \quad (7.3)$$

with $\phi_n := (X_i > k + n) \vee (X_i < k - n)$. They call Formula (7.3) *noise filtered oscillations permanence* and intuitively, in contrast to the previous formula, the required crossing of molecule level k is extended to a crossing of the interval $(k-n, k+n)$ which resembles a noise band of size n around the desired level k.

So far, we have discussed how qualitative aspects like the permanence of oscillatory behavior are analyzed in the literature. The logic CTL proved to be expressive enough for this task but in order to be able to reason about quantities like the time needed for an oscillation or the probability of a peak, CTL is ill-equipped. Consequently, approaches like [BG10, BMM09] make use of the *continuous stochastic logic* (CSL) which lifts CTL to the continuous-time probabilistic setting, that is, to continuous-time Markov chains. For a detailed introduction to CSL,

we refer the reader to [BHHK03] and Chapter 4. In [BMM09], Ballarini et al. extend their qualitative approach by incorporating time and probability bounds.

$$\mathsf{P}_{=?}[\Diamond\,(\mathsf{P}_{\geq 1}[\Diamond(X_i = k)] \wedge \mathsf{P}_{\geq 1}[\Diamond(X_i \neq k)])] \tag{7.4}$$

More precisely, they use Formula (7.4) to compute for each state s, the probability p_s that oscillations will *not* terminate in the respective state. With the rest of the probability mass $1 - p_s$, the oscillatory pattern will end in state s since either a level of $X_i = k$ can not be reached or left any more. The authors further analyze the specific model structure of their case study, the 3-way oscillator as described in Model 10, to define short-cut predicates $X_i = INV$ holding in states where oscillations terminate in species i. For the initial condition assigning probability one to a state with molecule count $\mathbf{x}_a + \mathbf{x}_B + \mathbf{x}_C = n$, we have that $INV = n$. This way, the probability of termination of oscillation for any species within time T is captured by the formula

$$\mathsf{P}_{=?}[\Diamond^{[0,T]}\,(X_1 = INV \vee X_2 = INV \vee \cdots \vee X_N = INV)]. \tag{7.5}$$

Note that in [BMM09], only three short-cut predicates were used, where we generalized the formula to N species. Moreover, computing the satisfaction probability of formula $\mathcal{S}_{=?}[X_i = k]$ for every molecule level k allows the authors to reason about the long-run probability distribution of molecules of type i. Finally, they use *rewards* introduced in CSLR [CKKP05, KNP02], an extension of CSL, to query the expected perimeter of k molecules around the initial state resembling the *amplitude* of oscillation. We will not elaborate the construction in detail, since the described approach is tailored to the specific case study and can not be used for arbitrary models.

In a follow-up paper [BG10], the authors base their qualitative characterization of oscillatory behavior on several notions of monotonicity. More precisely, a chemical species i is either monotonically increasing or decreasing indicated by boolean flags *inc_i* respectively *dec_i*, or nothing thereof. In order to relax the strict sense of monotonicity, increasing (decreasing) behavior might include up to a maximum of ns steps without an increase (decrease) in the number of molecules of species i. For that, an additional model variable keeps track of the current center of

the noise band. This additional information is provided by the augmentation of the model by an automaton that keeps track of the noise band and the boolean flags. Technically, the PRISM model is composed in parallel with a module encoding the automaton and synchronizing on the chemical reactions to detect changes in the molecule counts. Finally, a CSL formula to query the probability that some species i increases monotonically (modulo noise) until some level k is reached is given by

$$\mathsf{P}_{=?}[inc\_i \ \mathsf{U} \ (X_i = k)]. \tag{7.6}$$

Oscillatory behavior of a species i, more precisely, a single period, is then characterized by a monotonic increase from a current level j to some level $k > j$, followed by a monotonic decrease back to level j, where the noise band ns is used. The authors finally use *linear temporal logic* (LTL) to formalize such an oscillation pattern of amplitude $j - 1$ via

$$\mathsf{P}_{=?}\left[inc\_i \ \mathsf{U} \ (X_i = k \wedge (dec\_i \ \mathsf{U} \ X_i = j))\right]. \tag{7.7}$$

Note that the original formula in [BG10] uses the W operator which behaves like the U operator but is also satisfied if the first sub-formula holds forever. The major difference between LTL and the branching time logics CTL and CSL is that it is a *linear time* logic, that is, the semantics is based on the paths of a model in contrast to the states as in CTL/CSL. More precisely, although there is a path operator in CTL/CSL, the final judgment, whether a path formula is satisfied is done per state (by validating the path formula satisfaction probability against the probability bounds). As a consequence, path formulae can not be nested. Since the semantics of LTL is based on paths, nesting is possible. We will not give a full introduction to LTL but will discuss the intuitive meaning of the presented formulae and refer to [Pnu77] for details. Formula (7.7) queries the probability measure of all paths where species i is monotonically increasing until a level of k molecules is reached, followed by a monotonic decrease until level j.

$$\mathsf{P}_{=?}\left[inc\_i \,\mathsf{U}\, (X_i = k \wedge ((k - ns \leq X_i \leq k + ns) \,\mathsf{U}\, (dec\_i \,\mathsf{U}\, X_i = j)))\right] \tag{7.8}$$

Note that again, the original formula used the W operator instead of the U operator. The authors further relax the requirements of oscillatory

behavior by allowing the fluctuation to stay at the peak, that is, around a molecule level of k (module noise band ns), for an unlimited amount of time as described by Formula (7.8).

7.3 Defining Oscillatory and Noisy Periodic Behavior

Inspired by these results we want to formally approach the problem of defining and analyzing oscillatory behavior for continuous-time Markov chains. Before we can start however, we must first define what quantity we are interested in. For that, we will use the notion of *observation functions*.

▷Definition 42

Definition 42: Observation Function

Given the state space $\mathcal{S} \subseteq \mathbb{N}^N$ of a Markov population model and an *observation weight* $\mathbf{o} \in \mathbb{R}^N$ we define the *observation function* as the function that maps a state $\mathbf{x} \in \mathcal{S}$ to the scalar $\mathbf{x} \cdot \mathbf{o}$.

▷Definition 43

Definition 43: Observed Process

Given a Markov population model $\mathcal{X}$ and a suitable *observation weight* $\mathbf{o} \in \mathbb{R}^N$ we define the *observed process* as the process $\mathcal{X}^{\mathbf{o}}$ with $\mathcal{X}^{\mathbf{o}}(t) = \mathcal{X}(t) \cdot \mathbf{o}$ for all $t \in \mathbb{R}_{\geq 0}$.

7.3.1 Why Fourier Analysis Fails

In the following, we will be interested in sequences of observations at time points $t_1, t_2, \ldots, t_T \in \mathbb{R}_0^+$ with $t_1 < t_2 < \cdots < t_T$ that make up a total observation by summation. More precisely, for a Markov population model $\mathcal{X}$ with state space $\mathcal{S}$ and a complex function $f : \mathcal{S} \times \mathbb{R}_0^+ \to \mathbb{C}$, we define the random variable

$$Y_f = \sum_{i=1}^{T} f(\mathcal{X}(t_i), t_i).$$

Further, we define the matrices

$$\mathbf{P}^{(i,j)} = \left[\mathbf{Pr}\left[\mathcal{X}(t_j) = y \mid \mathcal{X}(t_i) = x\right]\right]_{x,y\in\mathcal{S}}$$

with $i < j$ representing the transition probabilities from time point t_i to t_j, where $t_0 = 0$ and let $\pi(t)$ denote the transient distribution of $\mathcal{X}(t)$. The expectation $\mathbf{Exp}\left[Y_f\right]$ can be expressed by

$$\begin{aligned}
&\mathbf{Exp}\left[Y_f\right] \\
= \;& \textstyle\sum_{x_1,\ldots,x_T\in S} \mathbf{Pr}\left[\mathcal{X}(t_1) = x_1, \ldots, \mathcal{X}(t_T) = x_T\right] \cdot \sum_{i=1}^{T} f(x_i, t_i) \\
= \;& \textstyle\sum_{x_1\in\mathcal{S}} \pi_{x_1}(t_1) \cdot \sum_{x_2\in\mathcal{S}} \mathbf{P}^{(1,2)}_{x_1x_2} \cdot \ldots \cdot \sum_{x_T\in\mathcal{S}} \mathbf{P}^{(T-1,T)}_{x_{T-1}x_T} \cdot \sum_{i=1}^{T} f(x_i, t_i) \\
= \;& \textstyle\sum_{x_1\in\mathcal{S}} \pi_{x_1}(t_1) \cdot \left[f(x_1, t_1) + \sum_{x_2\in\mathcal{S}} \mathbf{P}^{(1,2)}_{x_1x_2} \cdot \left[f(x_2, t_2) + \ldots\right]\right]
\end{aligned}$$

Proof: The proof of the last step for $T = 1$ is trivial. For $T \geq 2$, the proof is done by proving the stronger transformation

$$\begin{aligned}
&\textstyle\sum_{x_2\in\mathcal{S}} \mathbf{P}^{(1,2)}_{x_1x_2} \cdot \ldots \cdot \sum_{x_T\in\mathcal{S}} \mathbf{P}^{(T-1,T)}_{x_{T-1}x_T} \cdot \left(C + \sum_{i=1}^{T} f(x_i, t_i)\right) \\
= \;& C + f(x_1, t_1) + \textstyle\sum_{x_2\in\mathcal{S}} \mathbf{P}^{(1,2)}_{x_1x_2} \cdot \left[f(x_2, t_2) + \ldots\right]
\end{aligned}$$

for a fixed $x_1 \in S$ and a constant $C \in \mathbb{C}$ via induction on T. For $T = 2$ we get

$$\begin{aligned}
&\textstyle\sum_{x_2\in\mathcal{S}} \mathbf{P}^{(1,2)}_{x_1x_2} \cdot \left(C + f(x_1, t_1) + f(x_2, t_2)\right) \\
= \;& \left(C + f(x_1, t_1)\right) \cdot \underbrace{\sum_{x_2\in\mathcal{S}} \mathbf{P}^{(1,2)}_{x_1x_2}}_{=1} + \textstyle\sum_{x_2\in\mathcal{S}} \mathbf{P}^{(1,2)}_{x_1x_2} \cdot f(x_2, t_2) \\
= \;& C + f(x_1, t_1) + \textstyle\sum_{x_2\in\mathcal{S}} \mathbf{P}^{(1,2)} x_1x_2 \cdot f(x_2, t_2).
\end{aligned}$$

And for $T \to T + 1$ we can infer

$$
\begin{aligned}
& \textstyle\sum_{x_2 \in \mathcal{S}} \mathbf{P}^{(1,2)}_{x_1 x_2} \cdot \ldots \cdot \sum_{x_{T+1} \in \mathcal{S}} \mathbf{P}^{(T,T+1)}_{x_T x_{T+1}} \cdot \left(C + \textstyle\sum_{i=1}^{T+1} f(x_i, t_i) \right) \\
= \quad & \textstyle\sum_{x_2 \in \mathcal{S}} \mathbf{P}^{(1,2)}_{x_1 x_2} \cdot \left[C + f(x_1, t_1) + \sum_{x_2 \in \mathcal{S}} \mathbf{P}^{(1,2)}_{x_1 x_2} \cdot [f(x_2, t_2) + \ldots] \right] \\
= \quad & (C + f(x_1, t_1)) \cdot \underbrace{\sum_{x_2 \in \mathcal{S}} \mathbf{P}^{(1,2)}_{x_1 x_2}}_{=1} + \textstyle\sum_{x_2 \in \mathcal{S}} \mathbf{P}^{(1,2)}_{x_1 x_2} \cdot [f(x_2, t_2) + \ldots] \\
= \quad & C + f(x_1, t_1) + \textstyle\sum_{x_2 \in \mathcal{S}} \mathbf{P}^{(1,2)}_{x_1 x_2} \cdot [f(x_2, t_2) + \ldots] .
\end{aligned}
$$

□

In the induction step we used the fact that C and $f(x_1, t_1)$ are constants with respect to all occurring sums. By defining $\mathbf{f}^{(i)} = \left[f(x, t_i) \right]_{x \in \mathcal{S}}$, the last formula can easily be transformed into matrix vector form

$$
\mathbf{Exp}\left[Y_f \right] = \pi(0) \cdot \mathbf{P}^{(0,1)} \cdot \left[\mathbf{f}^{(1)} + \mathbf{P}^{(1,2)} \cdot \left[\mathbf{f}^{(2)} + \mathbf{P}^{(2,3)} \cdot \left[\mathbf{f}^{(3)} + \ldots \right] \right] \right] \cdot \mathbf{e}, \tag{7.9}
$$

which is equivalent to the weighted sum

$$
\sum_{i=1}^{T} \pi(t_i) \cdot \mathbf{f}^{(i)} \cdot \mathbf{e}. \tag{7.10}
$$

Now, if we start the analysis in steady state, that is, $\pi(0) = \pi$, we loose the ability to reason about time-inhomogeneous observations in many cases since Equation (7.10) becomes

$$
\pi \cdot \sum_{i=1}^{T} \mathbf{f}^{(i)} \cdot \mathbf{e},
$$

where the actual time-dependency of the Markov process vanishes and is subsumed by the equilibrium distribution.

Since we are interested in the oscillatory character of a Markov population process, it might be tempting to compute the *expected discrete Fourier transform* of the observations over time. Consequently, given an observed Markov population model $\mathcal{X}^{\mathbf{o}}(t)$ we can define the k-th

frequency component of its *discrete Fourier transform* as

$$\mathcal{X}^{\mathbf{o}}_{(k)} = \sum_{t=0}^{T-1} \mathcal{X}^{\mathbf{o}}(t) \cdot e^{-2 \cdot i \cdot \pi \cdot \frac{k}{T} \cdot t}. \tag{7.11}$$

Defining T functions

$$f_{(k)}(\mathbf{x}, t) = (\mathbf{x} \cdot \mathbf{o}) \cdot e^{-i \cdot 2 \cdot \pi \cdot \frac{k}{T} \cdot t} \tag{7.12}$$

with $k \in \{0, \ldots, T-1\}$ and assuming $\pi(0) = \pi$ gives

$$\mathbf{Exp}\left[\mathcal{X}^{\mathbf{o}}_{(k)}\right] = \mathbf{Exp}\left[Y_{f_k}\right] = \pi \cdot \sum_{t=0}^{T-1} \mathbf{f}^{(t)}_{(k)} \cdot \mathbf{e}$$

with $\mathbf{f}^{(i)}_{(k)} = \left[f_{(k)}(\mathbf{x}, t_i)\right]_{\mathbf{x} \in \mathcal{S}}$.

Thus, when we are interested in the expected Fourier coefficients on the long run, that is, taking steady state distribution as the initial distribution, the time dependency in the computation vanishes and barely any information can be extracted. Note that this result also affects simulative approaches, where long simulation runs are used to sample $\mathcal{X}^{\mathbf{o}}_{(k)}$ in the steady state.

7.3.2 Threshold-Based Definition of Oscillatory Character and Noisy Periods

Due to that result, we are forced to follow a different strategy to analyze oscillatory behavior in Markov population models. For that, we first define what it means for a system to be *oscillatory*. Similar to the CSL-based approach of Ballarini et al. [BMM09, BG10], which defines oscillatory behavior as never-ending fluctuations around a single value, we demand for an oscillatory model that both, a lower and a higher amplitude level, is crossed infinitely often.

Definition 44: Oscillatory MPM ▷**Definition 44**

An MPM $\mathcal{X}$ is called *oscillatory* for observation weights $\mathbf{o}$ and amplitude levels $L, H \in \mathbb{N}$ with $H > L$, if the probability measure of all trajectories of $\mathcal{X}^{\mathbf{o}}$ visiting intervals $(-\infty, L)$, $[L, H)$, and

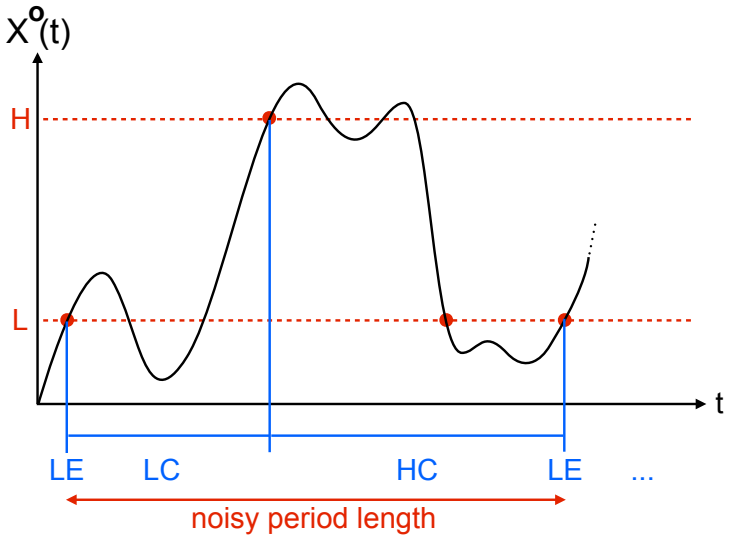


Figure 7.2: Events and phases of a (noisy) period.

$[H, \infty)$ infinitely often is one.

Obviously, an MPM is either *oscillatory* or the probability mass of trajectories with *converging* or *diverging* observations is greater than zero. Assuming a system is oscillatory, we are also interested in the time needed to oscillate once around this interval. We call this duration *noisy period length*. A single period can be split into several events and phases (cf. Figure 7.2). It starts with crossing the lower bound L (event LE) which is succeeded by a phase where the upper bound H has not been reached yet (phase LC). When this bound is finally reached, the period switches into the HC phase which is ended by another crossing of the lower bound L from below. This is indicated by another LE event and the classification pattern repeats.

In order to simplify presentation, we assume that L and H are chosen such that no transition in the MPM may skip the LC phase. Since we only consider bimolecular reactions which do not alter the observation level by more than one, this is the case if we choose $H - L \geq 2$.

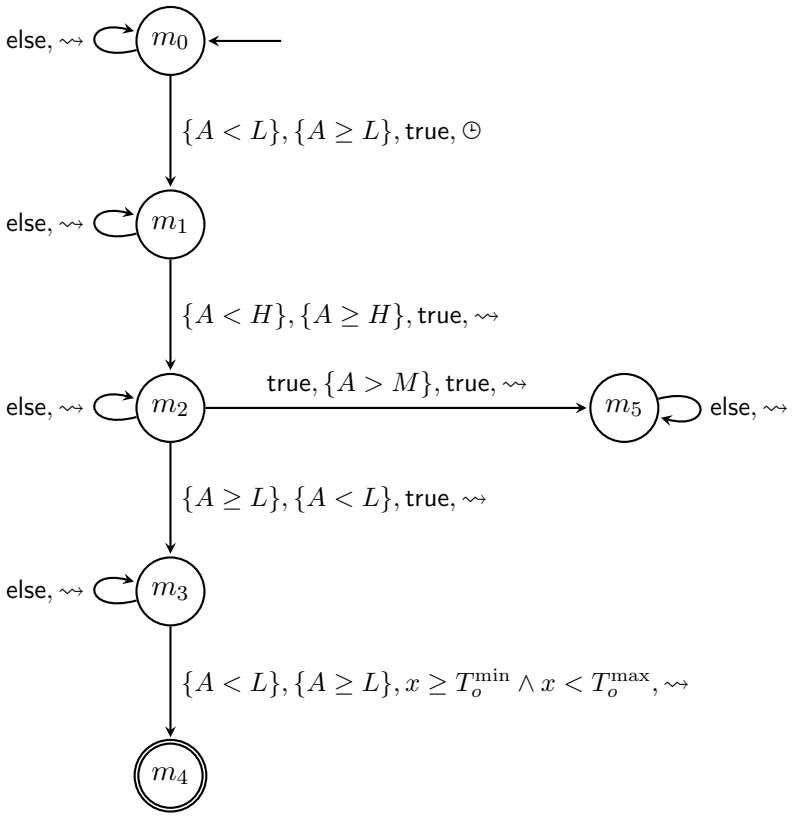


Figure 7.3: DTA specification for the repressilator expressing oscillatory behavior.

Nevertheless, for the case studies later in this chapter, we also choose $H = L+1$, but we made sure that the resulting special cases are handled correctly in the actual implementation of the numerical analysis.

7.3.3 DTA-Based Model Checking Approach

We formalized the above classification pattern for a single noisy period in the DTA in Figure 7.3. The DTA accepts if starting in an LE event, a second LE event is reached within $[T_{\min}, T_{\max}]$ time units. Further, we constrained the maximum observation level to M, that is, we demand

that a maximal molecule number $M = 20$ is not surpassed, enforcing a controlled behavior and preventing extreme spikes.

In order to study that property for the repressilator, we use the same reaction rate constants, algorithm, implementation, and machine as in Chapter 6. Moreover, we use two parameter sets for L and H:

$$\mathrm{PSet}_1 : (L, H) = (6, 10),$$

$$\mathrm{PSet}_2 : (L, H) = (4, 12)$$

and set $M = 20$. We are interested in oscillations with a short period and choose $T_{\min} = 0, T_{\max} = 5$ for the clock constraint in Figure 7.3. This yields relatively small acceptance probabilities, namely $p_{acc} = 0.06018$ for PSet_1 and $p_{acc} = 0.04968$ for PSet_2. Larger values of $T_{\max}$ yield higher acceptance probabilities and long running times for the algorithm. For $T_{\max} = 5$ we used on average about 1,300,000 significant states per iteration for both parameter sets. The running times were about one hour.

Optimization of the Oscillatory Behavior: We also optimized the acceptance probability assuming that the production rate ρ as well as the binding rate β are parameters. The optimization procedure as described in Section 6.4.3 was used. We found that both values decrease for a maximal acceptance probability of $p_{acc} = 0.1357$ (PSet_1) and $p_{acc} = 0.1651$ (PSet_2). While ρ becomes optimal at 2.3333 (PSet_1) and 3.3051 (PSet_2), the optimal value of β reaches the border of the interval that we used as constraint, that is, 0.1 for both parameter sets. The running time for the optimization was about 13 hours for PSet_1 and 10 hours for PSet_2. Note that the lower production rate ρ ensures that the protein populations do not become so large (compared to the original choice of ρ) such that it takes too long to reach the lower threshold L. Moreover, the influence of the binding rate β determines the height of the peak and how far the protein populations of repressed genes goes down. Thus, with a too large β the molecule numbers raise and fall too extremely and a smaller value of β is more beneficial. Note, however, that when β becomes very small, then it becomes more likely that the molecule numbers stay above the lower bound L. Indeed, in another experiment we chose the larger interval $[0.0001, 10]$ as constraint for β and we found the maximal acceptance probability at $\beta \approx 0.8$.

7.4 Period Detector Expansion

Inspecting the DTA in Figure 7.3 reveals, that in fact, no clock reset is needed if the model is started in a state corresponding to an LE event. Consequently, the classification pattern of a noisy period can also be done more efficiently via *finite state automata*. This procedure has been inspired by model checking algorithms for *pathCSL* [BCH$^+$03] and *asCSL* [BCK$^+$07], extensions to standard CSL, where the until operator is replaced by a more general regular-expression-based path operator. The main motivation not to directly apply these approaches was to create a highly specialized algorithm optimized for speed and to be able to cope with infinite state spaces. This was achieved by using truncation based transient analysis and *lazy* product CTMC construction. By lazy, we mean that not the whole product CTMC is built a priori but only as far as it is needed for the computation – trivially a strong requirement to be able to support infinite state spaces. Further, unnecessary computational overhead caused by re-computations introduced by the classical model checking algorithm as experienced in [Spi09] is avoided. In the following we will describe this approach in detail, that is, how to incorporate the period classification pattern into a given MPM using a compositional approach. The material of this chapter has been published in [Spi13a].

Definition 45: Deterministic Finite Automaton ▷Definition 45

A *deterministic (non-blocking) finite automaton (DFA)* on a set $\mathcal{A}$ is a tuple $(\mathcal{M}, m_0, \rightarrow)$, where $\mathcal{M}$ is a finite set of states, $m_0 \in \mathcal{M}$ is the initial state, and $\rightarrow \subseteq \mathcal{M} \times \mathcal{P} \times \mathcal{P} \times \mathcal{M}$ is the transition relation. Here, $\mathcal{P}$ denotes the set of predicates over $\mathcal{A}$. We further demand that

$$\forall m \in \mathcal{M}, x \in \mathcal{A}. \exists! p, p' \in \mathcal{P}, m' \in \mathcal{M}. p(x) \wedge p'(x) \wedge (m, p, p', m') \in \rightarrow,$$

that is, that the transition relation is *deterministic* and *non-blocking*.

▷Definition 46

Definition 46: Product of MPM and DFA

The product $\mathscr{C} \otimes \mathscr{D}$ of an MPM $\mathscr{C} = (\mathcal{S}, \mathbf{Q}, \pi(0))$ and a DFA $\mathscr{D} = (\mathcal{M}, m_0, \rightarrow)$ on $\mathcal{S}$ is an MPM $\mathscr{C}' = (\mathcal{S}', \mathbf{Q}', \pi(0))$ where $\mathcal{S}' = \mathcal{S} \times \mathcal{M}$, $\pi_{\left[\mathbf{x} \quad m\right]}(0)' = \pi_{\mathbf{x}}(0)$ if $m = m_0$ and 0 otherwise, and its infinitesimal generator $\mathbf{Q}'$ is defined by

$$\mathbf{Q}'_{\left[\mathbf{x} \quad m\right]\left[\mathbf{y} \quad m'\right]} = \begin{cases} \mathbf{Q}_{\mathbf{xy}} & \text{if } \mathbf{x} \neq \mathbf{y} \wedge \mathit{eval}(\mathbf{x}, \mathbf{y}, m, m'), \\ E(\left[\mathbf{x} \quad m\right]) & \text{if } \mathbf{x} = \mathbf{y} \wedge m = m', \\ 0 & \text{otherwise,} \end{cases}$$

where

$$E(\left[\mathbf{x} \quad m\right]) = - \sum_{\mathbf{z} \neq \mathbf{x} \vee m'' \neq m} \mathbf{Q}'_{\left[\mathbf{x} \quad m\right]\left[\mathbf{z} \; m''\right]}$$

and predicate $\mathit{eval}(\mathbf{x}, \mathbf{y}, m, m')$ is true iff

$$\exists p, p' \in \mathcal{P}. p(\mathbf{x}) \wedge p'(\mathbf{y}) \wedge (m, p, p', m') \in \rightarrow .$$

For a state $s = [\mathbf{x} \; m]$, we define $s_M = m$.

▷Definition 47

Definition 47: Period Detector Expanded MPM

Given an MPM $\mathscr{C} = (\mathcal{S}, \mathbf{Q}, \pi(0))$ with $\mathcal{S} \subseteq \mathbb{N}^N$, the *period detector expanded MPM* (PDMPM) of $\mathscr{C}$ for observation weights $\mathbf{o}$ is the product MPM $\mathscr{C}' = \mathscr{C} \otimes \mathscr{D}^{PD}$ where $\mathscr{D}^{PD}$ denotes the DFA depicted in Figure 7.4. In order to keep the presentation readable, we have abbreviated some transitions to make the DFA deterministic and non-blocking by the expression else.

The intuition behind the *period detector expanded* MPM construction is that it mimics the original behavior but additionally annotates the state space of the MPM by the information of the DFA in which event or phase of an oscillation the system currently is. Note that we do not need an acceptance condition for the DFA.

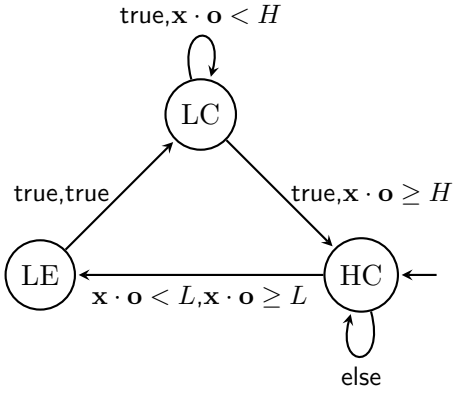


Figure 7.4: Period detector DFA.

Theorem 21: Equivalence of an MPM and its PDMPM ▷Theorem 21

An MPM $\mathscr{C} = (\mathcal{S}, \mathbf{Q}, \pi(0))$ and its product with a DFA $\mathscr{D}$ on $\mathcal{S}$ are F-bisimilar according to [BHHK03] written $\mathscr{C} \sim_F \mathscr{C} \otimes \mathscr{D}$ for $F = \mathbb{R}$ with respect to the labeling functions $L(\mathbf{x}) = \{\mathbf{x}\cdot\mathbf{o}\}$ on $\mathscr{C}$ and $L(\begin{bmatrix}\mathbf{x} & m\end{bmatrix}) = \{\mathbf{x}\cdot\mathbf{o}\}$ for all $\mathbf{x} \in \mathcal{S}$ on $\mathscr{C} \otimes \mathscr{D}$ and fixed observation weights $\mathbf{o}$.

Proof: Let DFA $\mathscr{D} = (\mathcal{M}, m_0, \rightarrow)$ and relations $\mathcal{R}_1$ and $\mathcal{R}_2$ be defined as

$$\begin{aligned}\mathcal{R}_1 &= \{(\mathbf{x}, \begin{bmatrix}\mathbf{x} & m\end{bmatrix}) \mid \mathbf{x} \in \mathcal{S}, m \in \mathcal{M}\} \text{ and}\\ \mathcal{R}_2 &= \{(\begin{bmatrix}\mathbf{x} & m\end{bmatrix}, \begin{bmatrix}\mathbf{x} & m'\end{bmatrix}) \mid \mathbf{x} \in \mathcal{S}, m, m' \in \mathcal{M}\}.\end{aligned}$$

Then, relation $\mathcal{R} = \mathcal{R}_1 \cup \mathcal{R}_1^{-1} \cup \mathcal{R}_2 \cup id(\mathcal{S})$, where $id(\mathcal{S}) = \{(\mathbf{x}, \mathbf{x}) \mid \mathbf{x} \in \mathcal{S}\}$ denotes the identity relation on $\mathcal{S}$, is a *F-bisimulation* relation. For a detailed proof, we refer to the author's Master thesis [Spi09]. □

Theorem 21 ensures, that the MPM and its period detector expanded MPM are equal in the sense that they behave the same with respect to the probability of any observations that can be made starting in any

state. In particular, no oscillations are artificially introduced due to the described extension.

7.5 Analysis of the PDMPM

The following theorem shows how the oscillatory character of an MPM can be checked with the help of period detector expansion.

▷Theorem 22

Theorem 22: Oscillation Property

Given an MPM with observation weights $\mathbf{o}$ and amplitude levels $L, H \in \mathbb{N}$ with $H > L$. If its PDMPM with state space $\mathcal{S}' \subseteq \mathcal{S} \times \{LE, LC, HC\}$ is ergodic and

$$\forall m \in \{LE, LC, HC\}.\exists \mathbf{x} \in \mathcal{S}.\pi_{\begin{bmatrix}\mathbf{x} & m\end{bmatrix}} > 0,$$

where π denotes the steady state probability distribution of the PDMPM, the MPM is oscillatory according to Definition 44.

Proof: Ergodicity implies positive-recurrence of all states and therefore *divergence* is ruled out. The existence of at least one state $\begin{bmatrix}\mathbf{x} & m\end{bmatrix}$ in each phase $m \in \{LE, LC, HC\}$ with positive steady state probability and the construction of the PDMPM imply that each of the observation intervals $(-\infty, L), [L, H), [H, \infty)$ is visited infinitely often contradicting *convergence*. □ □

In case of an oscillatory system, we also want to quantify the time needed for oscillations in the long run. We start by defining the time $\mathcal{L}(t)$ needed for the next two LE events in the PDMPM after time point t. Since the underlying process is stochastic, $\mathcal{L}(t)$ is a random variable as defined in Equation (7.13).

$$\begin{aligned}\mathcal{L}(t) = \min[t_2 : \exists t_1 \geq t, t_1 < t_2.\\ \mathcal{X}_M(t_1) = \mathcal{X}_M(t_2) = LE \wedge \forall t' \in (t_1, t_2).\mathcal{X}_M(t') \neq LE] - t\end{aligned} \quad (7.13)$$

Next, we define the *noisy period length* as the time $\mathcal{L}(t)$ for those states and times where an oscillation just begins, that is, when $\mathcal{X}_M(t) = LE$.

Our final goal is to approximate the cumulative distribution function (CDF)

$$\lim_{t\to\infty} \mathbf{Pr}\left[\mathcal{L}(t) \leq T \mid \mathcal{X}_M(t) = LE\right] \tag{7.14}$$

of the noisy period length on the long run for a given time bound T. Algorithm 13 returns two vectors, cdf and pdf, containing approximations of the cumulative respectively probability distribution of the noisy period length.

Algorithm 13 nperiod$(\mathscr{C} = (\mathcal{S}, \mathbf{Q}, \pi(0)), \mathbf{o}, L, H, \Delta, \alpha)$

let $\mathscr{C}' = (\mathcal{S}' = \mathcal{S} \times \{LE, LC, HC\}, \mathbf{Q}', \pi(0)')$ be the PDMPM of $\mathscr{C}$ for observation weight $\mathbf{o}$
solve $\pi \cdot \mathbf{Q}' = 0$ with $\pi \cdot \mathbf{e} = 1$
$p \leftarrow$ new hash map $\mathcal{S}' \to [0, 1]$
$p(\begin{bmatrix}\mathbf{x} & m\end{bmatrix}) \leftarrow \pi_{\begin{bmatrix}\mathbf{x} & LE\end{bmatrix}}$ if $m = LC$ and 0 otherwise $\forall \mathbf{x} \in \mathcal{S}, m \in \{LE, LC, HC\}$
$p \leftarrow p \cdot (p \cdot \mathbf{e})^{-1}$
$t \leftarrow 0$
$e_a \leftarrow 0$
$\mathrm{cdf}(-\Delta) \leftarrow 0$
while $\sum_{\mathbf{x}} p(\begin{bmatrix}\mathbf{x} & LE\end{bmatrix}) + e_a < \alpha$ **do**
    $[p, e] \leftarrow \mathrm{transient}(\mathbf{Q}', p, \mathcal{S}', \Delta, \delta, RK4)$ where states $[\mathbf{x}\ LE]$ are absorbing
    $\mathrm{cdf}(t) \leftarrow \sum_{\mathbf{x}} p(\begin{bmatrix}\mathbf{x} & LE\end{bmatrix})$
    $e_a \leftarrow e_a + e$
    $\mathrm{pdf}(t) \leftarrow \mathrm{cdf}(t) - \mathrm{cdf}(t - \Delta)$
    $t \leftarrow t + \Delta$
end while
return [cdf, pdf]

▷Theorem 23

Theorem 23: Correctness of Algorithm 13

Given an oscillatory MPM $\mathcal{X}(t)$, Algorithm 13 approximates probability

$$\lim_{t\to\infty} \mathbf{Pr}\left[\mathcal{L}(t) \leq T \mid \mathcal{X}_M(t) = LE\right]$$

by $\mathrm{cdf}(T)$.

Proof: First, we note that the set of paths satisfying $\mathcal{L}(t) \leq T$ is measurable since the problem can be reduced to bounded reachability where respective proofs as in [BHHK03] can be used. Concerning the algorithm, we first compute the steady state distribution π of the PDMPM in line 2 and normalize the sub-distribution with entries $\pi_{[\mathbf{x}\ LE]}$ for $\mathbf{x} \in \mathcal{S}$ corresponding to states in the LE event in lines 4 and 5 which resembles the conditioning in Equation (7.14).

Note that in line 4 we take states in the LC phase instead of the LE event for the initial distribution. The reason is that this way we do not have to distinguish between the first and second LE event. Consequently, if a state s with $M(s) = LE$ is entered, a full period has been performed. This is justified if $H - L \geq 2$ and only a maximal increase of one in the observation level per transition can be made (bimolecular reactions), since the LE event is over in any case after one step. We restricted to that case in order to simplify the presentation. In our implementation, we do have separate annotations for the first and second LE event.

The while-loop from lines 9 to 15 performs a transient analysis using this distribution as the initial distribution and states corresponding to a second LE event are made absorbing. Consequently, after the transient analysis, the total mass in the absorbing states corresponds to the proportion of paths having finished a full oscillatory cycle up to time t. Note that the transient analysis is not exact since we truncate states with probability less than δ as described in Algorithm 2. Therefore, in addition, we compute the accumulated error in e_a (line 12). We stop the iteration as soon as a threshold α of the total initial probability mass minus the accumulated error has been absorbed, e.g. $\alpha = 0.99$. In lines 11 and 14, we keep track of the time and the absorbed mass

which gives the CDF quantized by the time step Δ. Taking finite differences of the CDF in line 13 finally gives an approximation of the probability density function (PDF) of the noisy period length. The algorithm terminates since each period will finally end with probability one due to the construction of the PDMPM, the ergodicity implied by the oscillatory character of the MPM and the states corresponding to the end of an oscillation being made absorbing. More precisely, the sum $\sum_{\mathbf{x}} p(\begin{bmatrix}\mathbf{x} & LE\end{bmatrix}) + e$ will approach one with $t \to \infty$. □

The total error $\theta = (1 - \alpha) + e_a$ corresponds to the probability mass where we can not tell whether it belongs to an LE state or not. If the support of the steady state distribution is infinite and has to be truncated as shown in Chapter 3, error ϵ has to be added to θ as well. The error θ can be controlled by increasing α. A choice of $\delta = 10^{-20}$ usually results in a negligible error e_a [DHMW09]. The time complexity of the steady state computation is $\mathcal{O}(n^3)$, where n is the number of states and the complexity of the CDF/PDF computation using truncation based transient analysis is $\mathcal{O}(u \cdot t \cdot n)$ assuming a sparse model which is given in the case of an MPM induced by a constant number of R transition classes. Here, t is determined by α and corresponds to maximal period length which still occurs with significant probability, u is the maximum exit rate encountered and n is the maximum number of states with significant probability mass until time t. The space complexity is $\mathcal{O}(n)$.

7.6 Case Studies

Finally, we will show the numerical results of applying the presented method to two case studies from systems biology. All computations were performed on an Intel Core i5 2.66 GHz machine with 8 GB of RAM. In all experiments, we used thresholds $\delta = 10^{-20}$, $\alpha = 0.9999$, and chose $\Delta = 0.5 \cdot u^{-1}$, where u is the maximum exit rate over the complete time course.

7.6.1 3-Way Oscillator

First, we will analyze the *(doped) 3-way oscillator* as described in Model 11. We choose initial state $(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3) = (30, 0, 0)$ with prob-

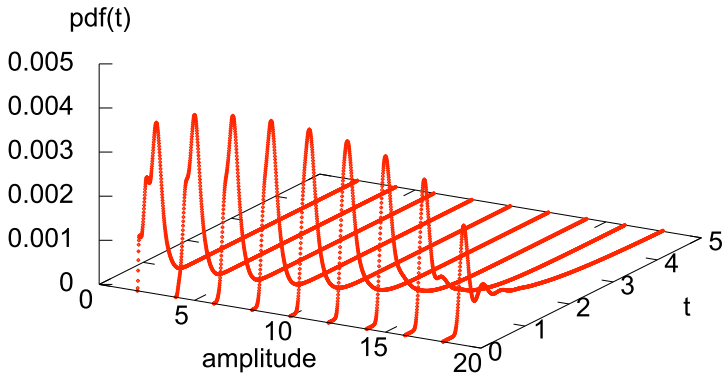


Figure 7.5: Noisy period length PDF of the 3-way oscillator model for several amplitudes.

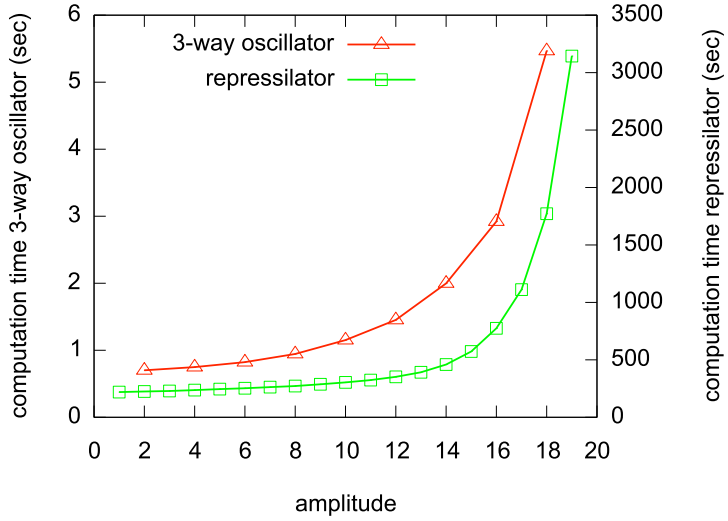


Figure 7.6: Computation times for computing the noisy period length PDF.

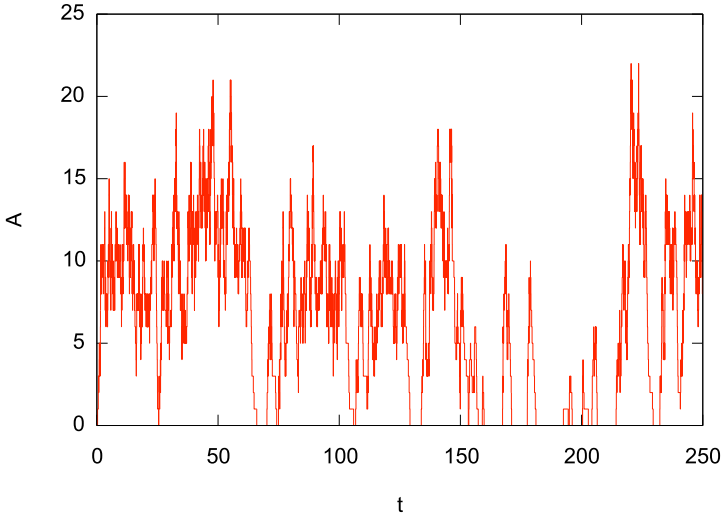


Figure 7.7: Sample trajectory of the repressilator model (observing A).

ability one and rates $\tau_A = \tau_B = \tau_C = \nu_A = \nu_B = \nu_C = 1.0$. As argued in Chapter 7.1, the resulting state space is finite. The mean population counts in steady state are $(\overline{\mathbf{x}_1}, \overline{\mathbf{x}_2}, \overline{\mathbf{x}_3}) = (10, 10, 10)$ and due the symmetric choice of rate constants, we are only interested in species A, that is, $\mathbf{o} = e_1$. As can be seen in the sample trace in Figure 7.1b, the oscillations are around this mean value. Consequently, we took $L = \overline{\mathbf{x}_1} - \frac{a}{2}$ and $H = \overline{\mathbf{x}_1} + \frac{a}{2}$ for the interval bounds and varied the amplitude $a \in \{2, 4, 6, 8, 10, 12, 14, 16, 18\}$.

The system is oscillatory for all those amplitude levels and the results of the noisy period length analysis are depicted in Figure 7.5 with the computation times presented in Figure 7.6. Most likely, the (noisy) period length of the 3-way oscillator is around 0.5 time units and period lengths of 5 or more time units are rare, even in the case of full amplitudes ($a = 18$). This coincides with the observations that can be made from the sample trajectory in Figure 7.1b.

7.6.2 Two-Species Repressilator

The other case study studies the *repressilator* model as described in Model 12 with parameter set $\rho = \rho_A = \rho_B = 10.0$, $\delta = \delta_A = \delta_B = 1.0$,

$\beta = \beta_A = \beta_B = 0.05$, and $\nu = \nu_A = \nu_B = 0.2$.

▷Model 12

Model 12: Two-Species Repressilator

The *two-species* repressilator is the repressilator of Model 9 for two protein types $A(1)$ and $B(2)$ encoded by two genes $G_A(3)$ and $G_B(4)$. The chemical reactions are

$$G_A \xrightarrow{\rho_A} G_A + A, \qquad G_B \xrightarrow{\rho_B} G_B + B,$$

$$A \xrightarrow{\delta_A} \varnothing, \qquad B \xrightarrow{\delta_B} \varnothing,$$

$$G_A + B \xrightarrow{\beta_B} \overline{G_A}, \qquad G_B + A \xrightarrow{\beta_A} \overline{G_B},$$

$$\overline{G_A} \xrightarrow{\nu_B} G_A + B, \qquad \overline{G_B} \xrightarrow{\nu_A} G_B + A,$$

and the reaction rate constants have the same interpretation as in Model 9.

Here, the state space $\mathcal{S} = \mathbb{N}^2 \times \{0,1\}^2$ is not finite for initial state $(0,0,1,1)$. Consequently, geometric bounds for 90% of the steady state probability mass were computed according to Chapter 3.1 which resulted in upper bounds of 32 molecules for both protein species.

Sample traces of the repressilator model as illustrated in Figure 7.7 reveal that unlike the oscillation around a mean value, the repressilator with the specified parameters has a rather peak-like oscillation pattern, that is, periods start at the zero level, reach a maximum peak level and finally return to the zero level again. Therefore, we chose $L = 1$ and $H = L + a$ for varying a with $1 \leq a \leq 19$.

The system is oscillatory for all amplitude values and the results of the noisy period length analysis are depicted in Figure 7.8 with the respective computation times presented in Figure 7.6. The majority of periods have durations of less than 80 time units and the larger the amplitudes, the larger also the period length becomes, since peaks of

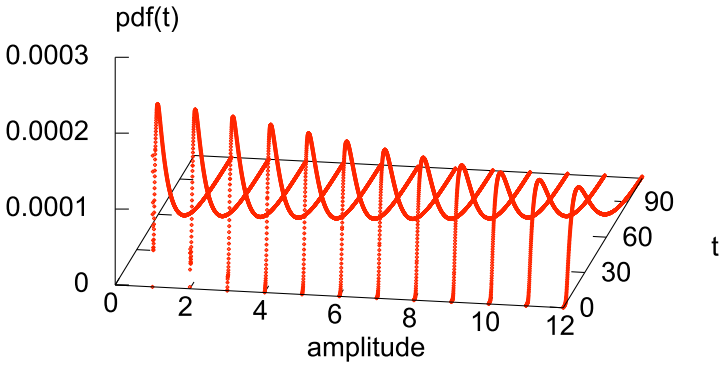


Figure 7.8: Noisy period length PDF of the repressilator model for several amplitudes.

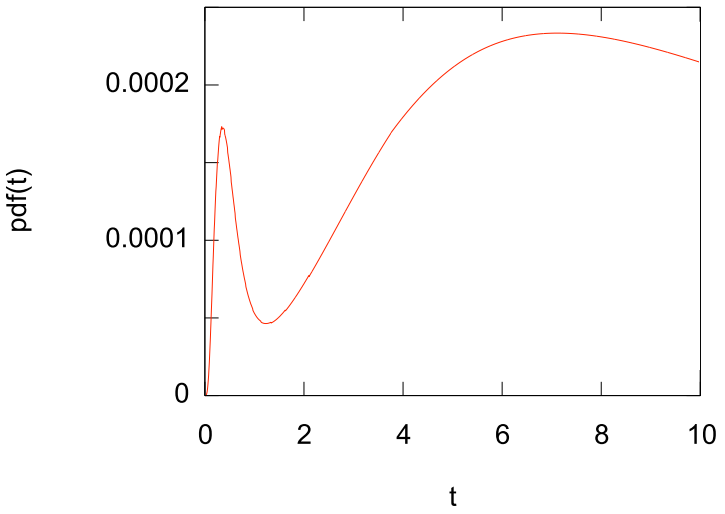


Figure 7.9: Noisy period length PDF of the repressilator model for $a = 1$.

higher amplitudes become more rare. An interesting phenomenon of the repressilator can be witnessed for an amplitude of $a = 1$ as depicted in Figure 7.9 where we set the smallest constraint on the minimal amplitude. While more than 98% of the oscillations have a period length of 1.23 time units or more, a small amount of around 1.87% of the oscillations only lasts for 1.23 time units or less as can be seen by the first peak in probability in Figure 7.9. This bi-modality of the probability distribution can be explained by two effects. On the one hand, the smaller peak within $t \in [0, 1.23]$ occurs since there is little time to build up a significant amount of A molecules. Consequently, the chance of the A molecules repressing gene G_B is small and therefore the amount of B molecules grows as well and finally species B may win the competition to represses its competitor, gene G_A. On the other hand, the degradation rate δ of the molecules is high compared to the gene unbinding rate ν. Thus, it is very likely that all A molecules degrade until the unbinding event happens and the oscillatory cycle ends. Therefore, each oscillation must first cross a kick-start level of molecules in order to perform a longer cycle. However, most of the time this threshold is surpassed and the oscillation is only ended by spontaneous and long enduring repressions by B molecules.

CHAPTER 8

Conclusion

Markov population models (MPMs) can capture a wide variety of systems from many fields like queuing theory and systems biology where the usual interest is on the behavior over time as captured by transient analysis and on the long-run behavior as captured by steady state analysis.

This doctoral thesis focused on the latter, proposing highly efficient methods to elegantly circumvent problems related to infinite state spaces. The key idea was to exploit *Lyapunov functions* in order to construct an algorithm capable of automatically generating symbolic descriptions of finite state space truncations called *geometric bounds*. This algorithm is based on the computation of the global maximum expected change of the Lyapunov function called the *drift*. We empowered symbolic global optimization by solving sets of non-linear equation systems which can be done efficiently using methods like *polyhedral homotopy continuation*. This finally allowed us to to answer questions like *what finite region in the state space entails more than a given lower bound (close to one), of the steady state probability mass*. Moreover, we showed how to lift methods like stochastic complementation to the infinite state setting in order to refine those bounds, that is, to compute even *state-wise bounds* for all states inside that region. We could successfully integrate those

techniques to construct a *model checking* routine for the steady state operator of the *continuous stochastic logic*.

If the system of consideration only had a single attractor, that is, a single peak in the equilibrium distribution, the methods presented in Chapter 3 were already efficient. However, if the steady state distribution was *multimodal*, that is, it had multiple peaks, a fast solution was hindered by the underlying stiffness of the model. In order to combat that problem, we were able to combine truncation-based reachability analysis, aggregation, and stochastic complementation in order to construct the *attractor Markov chain* (AMC), that is, a high-level view of the original model. The decisive advantage of the AMC is that the computational complexity is reduced drastically since this Markov chain has exactly one state per attractor and analysis becomes manageable. This way, information about the strengths of the respective attractors relative to each other can be computed efficiently. In combination with the computation of the local steady state distributions of each attractor, even the full steady state distribution could be approximated accurately.

The final goal was to study the *oscillatory* behavior of MPMs. For that, we developed an approximative algorithm that computes the acceptance probability of an MPM with respect to a specification given as a *deterministic (single-clock) timed automaton*. This class of automata allows for the encoding of a subset of linear-time properties such as oscillatory behavior. It turned out, that the analysis of the underlying Markov renewal process could be improved even further via truncation-based transient analysis of a set of altered MPMs. In order to further speed up the computation time of this approach for the property of oscillation, a specialized algorithm, based on truncation-based transient analysis of the product of the original MPM and a small deterministic finite state automaton, was constructed in order to analyze the *oscillatory properties* of the MPM, that is the probability distribution function of the period length.

During the whole course of this thesis, we evaluated the precision and performance of the developed methods and algorithms using a large variety of models from the application area of systems biology.

Future Work: The extents of usability of the presented techniques are reached when even the finite reduced state space containing only the

significant states grows too large. The main reason for this happening is the *curse of dimensionality*, that is, the exponential blow-up in the computational complexity with respect to the number of population types. While we could easily solve systems with up to three chemical species and several bounded species like genes, experiments with larger models showed that in several cases the drastically increased run-time and/or memory demands rendered the practical usability very difficult. Several techniques like Michaelis-Menten approximation allow the reduction of a model's dimensionality but depend on the validity of several assumptions, like that a fast reacting species is in quasi-equilibrium with respect to a slower species. One promising remedy the author sees is the usage of *hybrid models*, where population types with low variance are modeled deterministically via ordinary differential equations and population types with high variance retain their stochastic nature. For transient analysis, efficient techniques have already been developed and show excellent results [LMW11]. It remains to examine whether the same can be done for steady state analysis as well – for example by following and extending [HKNT96, BHK$^+$11, TV13].

Bibliography

[ACK10] D. F. Anderson, G. Craciun, and T. G. Kurtz. Product-form stationary distributions for deficiency zero chemical reaction networks. *Bulletin of Mathematical Biology*, 72(8):1947–1970, 2010.

[AD94] R. Alur and D. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.

[AKS14] A. Andreychenko, T. Krüger, and D. Spieler. Analyzing oscillatory behavior with formal methods. In *Proceedings of the ROCKS autumn school 2013*, Lecture Notes in Computer Science, 2014.

[And91] W. Anderson. *Continuous-time Markov chains: An applications-oriented approach.* Springer, 1991.

[ARM98] A. Arkin, J. Ross, and H. H. McAdams. Stochastic kinetic analysis of developmental pathway bifurcation in phage lambda-infected escherichia coli cells. *Genetics*, 149(4):1633–1648, 1998.

[ASSB00] A. Aziz, K. Sanwal, V. Singhal, and R. K. Brayton. Model-checking continuous-time Markov chains.

ACM Transactions on Computational Logic, 1(1):162–170, 2000.

[Atk89] K. Atkinson. *An introduction to numerical analysis*. Wiley, 2nd edition, 1989.

[AW87] J. Abate and W. Whitt. Transient behavior of the M/M/1 queue: Starting at the origin. *Queueing Systems*, 2(1):41–65, 1987.

[Ban55] S. Banach. *Théorie des opérations linéaires*. Chelsea Publishing Corporation, 1955.

[BCDS13] L. Brim, M. Ceska, S. Drazan, and D. Safranek. Exploring Parameter Space of Stochastic Biochemical Systems using Quantitative Model Checking. In *Computer Aided Verification (CAV 2013)*, volume 8044 of *LNCS*, pages 107–123. Springer, 2013.

[BCH+03] C. Baier, L. Cloth, B. Haverkort, H. Hermanns, and J.-P. Katoen. Model checking pathCSL. In *Proceedings of the Sixth International Workshop on Performability Modeling of Computer and Communication Systems*, pages 19–22, 2003.

[BCH+11] B. Barbot, T. Chen, T. Han, J.-P. Katoen, and A. Mereacre. Efficient CTMC model checking of linear real-time objectives. In *Proceedings of the 17th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, volume 6605 of *LNCS*, pages 128–142. Springer, 2011.

[BCK+07] C. Baier, L. Cloth, M. Kuntz, M. Siegle, and B. Haverkort. Model checking Markov chains with actions and state labels. *IEEE Transactions on Software Engineering*, 33(4):209–224, 2007.

[BG10] P. Ballarini and M. L. Guerriero. Query-based verification of qualitative trends and oscillations in biochemical systems. *Theoretical Computer Science*, 411(20):2019–2036, 2010.

[BH25] G. E. Briggs and J. B. Haldane. A note on the kinetics of enzyme action. *The Biochemical journal*, 19(2):338–339, 1925.

[BH72] J. R. Bunch and J. E. Hopcroft. *Triangular Factorization and Inversion by Fast Matrix Multiplication.* Department of Computer Science: Technical report. Defense Technical Information Center, 1972.

[BHHK03] C. Baier, H. Hermanns, B. Haverkort, and J.-P. Katoen. Model-checking algorithms for continuous-time Markov chains. *IEEE Transactions on Software Engineering*, 29(6):524–541, 2003.

[BHK+11] M. Balázs, G. Horváth, S. Kolumbán, P. Kovács, and M. Telek. Fluid level dependent Markov fluid models with continuous zero transition. *Performance Evaluation*, 68(11):1149–1161, 2011.

[BK08] C. Baier and J.-P. Katoen. *Principles of Model Checking.* The MIT Press, 2008.

[BKPR06] K. Ball, T. G. Kurtz, L. Popovic, and G. Rempala. Asymptotic analysis of multiscale approximations to reaction networks. *The Annals of Applied Probability*, 16(4):1925–1961, 2006.

[BL00] N. Barkai and S. Leibler. Biological rhythms: Circadian clocks limited by noise. *Nature*, 403:267–268, 2000.

[BL06] K. Bryan and T. Leise. The $25,000,000,000 eigenvector: The linear algebra behind Google. *SIAM Review*, 48(3):569–581, March 2006.

[BMM09] P. Ballarini, R. Mardare, and I. Mura. Analysing biochemical oscillation through probabilistic model checking. In *Proceedings of the 2nd Workshop From Biology to Concurrency and Back*, volume 229 of *Electronic Notes in Theoretical Computer Science*, pages 3–19. Elsevier, 2009.

[BRT88] J. T. Blake, A. L. Reibman, and K. S. Trivedi. Sensitivity analysis of reliability and performability measures for multiprocessor systems. *SIGMETRICS Performance Evaluation Review*, 16(1):177–186, 1988.

[Car06] L. Cardelli. Artificial biochemistry. Technical report, Microsoft Research, 2006.

[Car08] L. Cardelli. Artificial biochemistry. In *In Algorithmic Bioproceses, LNCS*. Springer, 2008.

[CHKM11] T. Chen, T. Han, J.-P. Katoen, and A. Mereacre. Model checking of continuous-time Markov chains against timed automata specifications. *Logical Methods in Computer Science*, 7:1–34, 2011.

[Cin75] E. Cinlar. *Introduction to stochastic processes*. Englewood Cliffs, N.J. Prentice-Hall, 1975.

[CKKP05] L. Cloth, J.-P. Katoen, M. Khattri, and R. Pulungan. Model-checking Markov reward models with impulse rewards. In *Dependable Systems and Networks*, Yokohama, 2005.

[Cou85] P. J. Courtois. Analysis of large Markovian models by parts. applications to queueing network models. In *Messung, Modellierung und Bewertung von Rechensystemen*, pages 1–10. Springer Berlin Heidelberg, 1985.

[CRCD+03] N. Chabrier-Rivier, M. Chiaverini, V. Danos, F. Fages, and V. Schächter. Modeling and querying biomolecular interaction networks. *Theoretical Computer Science*, 325:25–44, 2003.

[CS84] P. J. Courtois and P. Semal. Bounds for the positive eigenvectors of nonnegative matrices and for their approximations by decomposition. *Journal of the ACM*, 31(4):804–825, 1984.

[DEG+99] J. W. Demmel, S. C. Eisenstat, J. R. Gilbert, X. S. Li, and J. W. H. Liu. A supernodal approach to sparse partial

pivoting. *SIAM Journal on Matrix Analysis and Applications*, 20(3):720–755, 1999.

[DHMW09] F. Didier, T. A. Henzinger, M. Mateescu, and V. Wolf. Fast adaptive uniformization of the chemical master equation. In *Proceedings of HIBI*, pages 118–127, Washington, DC, USA, 2009. IEEE Computer Society.

[DHS09] S. Donatelli, S. Haddad, and J. Sproston. Model checking timed and stochastic properties with CSL^{TA}. *IEEE Transactions on Software Engineering*, 35(2):224–240, 2009.

[DHSW11] T. Dayar, H. Hermanns, D. Spieler, and V. Wolf. Bounding the equilibrium distribution of Markov population models. *Numerical Linear Algebra with Applications*, 18(6):931–946, 2011.

[DR78] I. S. Duff and J. K. Reid. An implementation of Tarjan's algorithm for the block triangularization of a matrix. *ACM Transactions on Mathematical Software*, 4(2):137–147, 1978.

[dSeSO92] E. de Souza e Silva and P. M. Ochoa. State space exploration in Markov models. *SIGMETRICS Performance Evaluation Review*, 20(1):152–166, 1992.

[EC82] E. A. Emerson and E. M. Clarke. Using branching time temporal logic to synthesize synchronization skeletons. *Science of Computer Programming*, 2(3):241–266, 1982.

[EL00] M. Elowitz and S. Leibler. A synthetic oscillatory network of transcriptional regulators. *Nature*, 403(6767):335–338, 2000.

[Eng08] S. Englblom. *Numerical solution methods in stochastic chemical kinetics*. PhD thesis, Uppsala University, 2008.

[FG88] B. L. Fox and P. W. Glynn. Computing Poisson probabilities. *Communications of the ACM*, 31(4):440–445, April 1988.

[FTY11] J. E. Ferrell, T. Y.-C. Tsai, and Q. Yang. Modeling the cell cycle: Why do certain circuits oscillate? *Cell*, 144(6):874–885, 2011.

[GCC00] T. S. Gardner, C. R. Cantor, and J. J. Collins. Construction of a genetic toggle switch in Escherichia coli. *Nature*, 403(6767):339–342, 2000.

[Ger00] R. German. *Performance analysis of communication systems with non-Markovian stochastic Petri nets*. John Wiley & Sons, Inc., New York, USA, 2000.

[Gil76] D. T. Gillespie. A general method for numerically simulating the stochastic time evolution of coupled chemical reactions. *Journal of Computational Physics*, 22(4):403 – 434, 1976.

[Gil92] D. T. Gillespie. A rigorous derivation of the Chemical Master Equation. *Physica A*, 188:404–425, 1992.

[Gil01] D. T. Gillespie. Approximate accelerated stochastic simulation of chemically reacting systems. *The Journal of Chemical Physics*, 115(4):1716, 2001.

[GP98] P. Goss and J. Peccoud. Quantitative modeling of stochastic systems in molecular biology by using stochastic Petri nets. *Proceedings of the National Academy of Science, USA*, 95:6750–6755, 1998.

[Gra77a] W. K. Grassmann. Transient solutions in Markovian queueing systems. *Computers & Operations Research*, 4(1):47–53, 1977.

[Gra77b] W. K. Grassmann. Transient solutions in Markovian queues: An algorithm for finding them and determining their waiting-time distributions. *European Journal of Operational Research*, 1(6):396–402, 1977.

[Gra91] W. K. Grassmann. Finding transient solutions in Markovian event systems through randomization. In *W.J. Stew-*

art (editor), Numerical solution of Markov chains, pages 357–371, 1991.

[GW76] R. Görtz and D. F. Walls. Steady state solutions of Master equations without detailed balance. *Zeitschrift für Physik B Condensed Matter*, 25(4):423–427, 1976.

[Hal82] P. R. Halmos. *A Hilbert space problem book*. Graduate Texts in Mathematics. Springer, New York, Berlin, Heidelberg, second edition, 1982.

[HBS+07] M. Hegland, C. Burden, L. Santoso, S. Macnamara, and H. Booth. A solver for the stochastic master equation applied to gene regulatory networks. *Journal of Computational and Applied Mathematics*, 205:708–724, 2007.

[HHWZ09a] E. M. Hahn, H. Hermanns, B. Wachter, and L. Zhang. INFAMY: An infinite-state Markov model checker. In *Proceedings of CAV*, pages 641–647. Springer, 2009.

[HHWZ09b] E. M. Hahn, H. Hermanns, B. Wachter, and L. Zhang. Time-bounded model checking of infinite-state continuous-time Markov chains. *Fundamenta Informaticae*, 95:129–155, 2009.

[Hil10] A. V. Hill. The possible effects of the aggregation of the molecules of haemoglobin on its dissociation curves. *Journal of Physiology*, 40:iv–vii, 1910.

[HJW09] T. A. Henzinger, B. Jobstmann, and V. Wolf. Formalisms for specifying Markovian population models. In *Proceedings of the 3rd International Workshop on Reachability Problems*, volume 5797 of *LNCS*. Springer, 2009.

[HKNT96] G. Horton, V. G. Kulkarni, D. M. Nicol, and K. S. Trivedi. Fluid stochastic Petri sets: Theory, applications, and solution techniques, 1996.

[HMMW10] T. A. Henzinger, L. Mikeev, M. Mateescu, and V. Wolf. Hybrid numerical solution of the chemical master equa-

tion. In *Proceedings of CMSB*, pages 55–65, New York, NY, USA, 2010. ACM.

[HMW09] T. A. Henzinger, M. Mateescu, and V. Wolf. Sliding window abstraction for infinite Markov chains. In *In Proceedings of CAV, volume 5643 of LNCS*, pages 337–352. Springer, 2009.

[HP92] P. G. Harrison and N. M. Patel. *Performance modelling of communication networks and computer architectures (International computer science series)*. Addison-Wesley, Boston, MA, USA, 1st edition, 1992.

[HS07] P. E. Heegaard and W. Sandmann. Ant-based approach for determining the change of measure in importance sampling. In *Proceedings of the Winter Simulation Conference*, pages 412–420. IEEE Press, 2007.

[HS13] E. M. Hahn and D. Spieler. Geoinf. `http://mosi.cs.uni-saarland.de/?page_id=545`, September 2013.

[JH07] T. Jahnke and W. Huisinga. Solving the chemical master equation for monomolecular reaction systems analytically. *Journal of Mathematical Biology*, 54:1–26, 2007.

[JP04] J. Jacod and P. Protter. *Probability Essentials*. Springer, 2nd edition, August 2004.

[Jun06] S. Juneja. Rare-event simulation techniques: An introduction and recent advances. In *Handbook of simulation, Volume 13 of Handbooks in operations research and management science*, pages 291–350. Elsevier, 2006.

[KKLW07] J.-P. Katoen, D. Klink, M. Leucker, and V. Wolf. Three-valued abstraction for continuous-time Markov chains. In *CAV*, volume 4590 of *Lecture Notes in Computer Science*, pages 311–324. Springer, 2007.

[Kli10] D. Klink. *Three-Valued Abstraction for Stochastic Systems*. PhD thesis, RWTH Aachen, 2010.

[KNP02] M. Kwiatkowska, G. Norman, and A. Pacheco. Model checking expected time and expected reward formulae with random time bounds. In *Proceedings of the 2nd Euro-Japanese Workshop on Stochastic Risk Modelling for Finance, Insurance, Production and Reliability*, 2002.

[KNP11] M. Kwiatkowska, G. Norman, and D. Parker. PRISM 4.0: Verification of probabilistic real-time systems. In *Proceedings of CAV*, volume 6806 of *LNCS*. Springer, 2011.

[Kol31] A. Kolmogoroff. Über die analytischen Methoden in der Wahrscheinlichkeitsrechnung. *Mathematische Annalen*, 104:415–458, 1931.

[Kri06] S. Krishna. Minimal model of spiky oscillations in NF-κb signaling. *Proceedings of the National Academy of Sciences of the United States of America*, 103(29):10840–10845, 2006.

[Kul95] V. G. Kulkarni. *Modeling and Analysis of Stochastic Systems*. Chapman & Hall texts in statistical science series. Chapman & Hall, 1995.

[Kur72] T. G. Kurtz. The relationship between stochastic and deterministic models for chemical reactions. *Journal of Chemical Physics*, 57(7):2976 –2978, 1972.

[Lan62] R. Landauer. Fluctuations in bistable tunnel diode circuits. *Journal of Applied Physics*, 33:2209–2216, 1962.

[LK00] A. M. Law and D. M. Kelton. *Simulation Modeling and Analysis*. McGraw-Hill Higher Education, 2000.

[LLBB06] A. Lipshtat, A. Loinger, N. Q. Balaban, and O. Biham. Genetic toggle switch without cooperative binding. *Physical Review Letters*, 96(18):188101, 2006.

[LLBB07] A. Loinger, A. Lipshtat, N. Q. Balaban, and O. Biham. Stochastic simulations of genetic switch systems. *Physical Review E*, 75(2):021904, 2007.

[LLT08] T.L. Lee, T.Y. Li, and C.H. Tsai. HOM4PS-2.0: a software package for solving polynomial systems by the polyhedral homotopy continuation method. *Computing*, 83(2-3):109–133, 2008.

[LMW11] M. Lapin, L. Mikeev, and V. Wolf. SHAVE – Stochastic hybrid analysis of Markov population models. In *Proceedings of HSCC*, New York, NY, USA, 2011. ACM.

[LNH+07] J. Lohmueller, N. Neretti, B. Hickey, A. Kaka, A. Gao, J. Lemon, V. Lattanzi, P. Goldstein, L.K. Tam, M. Schmidt, A.S. Brodsky, K. Haberstroh, J. Morgan, T. Palmore, G. Wessel, A. Jaklenec, H. Urabe, J. Gagnon, and J. Cumbers. Progress toward construction and modelling of a tri-stable toggle switch in E. coli. *Synthetic Biology, IET*, 1(1.2):25–28, 2007.

[Lot56] A.J. Lotka. *Elements of mathematical biology*. Dover Publications, 1956.

[MA97] H. H. McAdams and A. Arkin. Stochastic mechanisms in gene expression. *Proceedings of the National Academy of Sciences of the United States of America*, 94(3):814–819, 1997.

[MAL11] A. Milias-Argeitis and J. Lygeros. Efficient stochastic simulation of metastable Markov chains. In *Conference on Decision and Control and European Control Conference*, pages 2239–2244. IEEE, 2011.

[MdSeSG89] R. R. Muntz, E. de Souza e Silva, and A. Goyal. Bounding availability of repairable systems. *IEEE Transactions on Computers*, 38(12):1714–1723, 1989.

[Mey89] C. D. Meyer. Stochastic complementation, uncoupling Markov chains, and the theory of nearly reducible systems. *SIAM Review*, 31:240–272, 1989.

[Min67] M. L. Minsky. *Computation: finite and infinite machines*. Prentice-Hall, Inc., 1967.

[MK06] B. Munsky and M. Khammash. The finite state projection algorithm for the solution of the chemical master equation. *The Journal of Chemical Physics*, 124(4), 2006.

[ML78] C. Moler and C. Van Loan. Nineteen dubious ways to compute the exponential of a matrix. *SIAM Review*, 20:801–836, 1978.

[MNSW13] L. Mikeev, M. R. Neuhäußer, D. Spieler, and V. Wolf. On-the-fly verification and optimization of DTA-properties for large Markov chains. *Formal Methods in System Design*, 43(2):313–337, 2013.

[MPG29] R. V. Mises and H. Pollaczek-Geiringer. Praktische Verfahren der Gleichungsauflösung. *Zeitschrift für angewandte Mathematik und Mechanik*, 9:58–77, 1929.

[Mun08] B. Munsky. *The finite state projection approach for the solution of the Master Equation and its applications to stochastic gene regulatory networks.* PhD thesis, University of California, 2008.

[MWWM07] K. M. Meyer, K. Wiegand, D. Ward, and A. Moustakas. SATCHMO: A spatial simulation model of growth, competition, and mortality in cycling savanna patches. *Ecological Modelling*, 209(24):377 – 391, 2007.

[Nas04] A. Nasroallah. Monte Carlo simulation of Markov chain steady-state distribution. *Extracta Mathematicae*, 19(2):279–288, 2004.

[Nor98] J. R. Norris. *Markov chains (Cambridge series in statistical and probabilistic mathematics).* Cambridge University Press, 1998.

[Pau04] J. Paulsson. Summing up the noise in gene networks. *Nature*, 427(6973):415–418, 2004.

[Pnu77] Amir Pnueli. The temporal logic of programs. In *Proceedings of SFCS*, pages 46–57. IEEE Computer Society, 1977.

[Pol02] L. Polkowski. *Rough sets: Mathematical foundations.* Physica-Verlag, 2002.

[PW96] J. G. Propp and D. B. Wilson. Exact sampling with coupled Markov chains and applications to statistical mechanics. *Random Structures and Algorithms*, 9(1-2):223–252, 1996.

[QS04] D. Qiu and R. Srikant. Modeling and performance analysis of BitTorrent-like peer-to-peer networks. *SIGCOMM Computer Communication Review*, 34(4):367–378, 2004.

[Ris02] A. Riska. *Aggregate matrix-analytic techniques and their applications.* PhD thesis, College of William and Mary, 2002.

[RMF06] T. Reichenbach, M. Mobilia, and E. Frey. Coexistence versus extinction in the stochastic cyclic Lotka-Volterra model. *Physical Review E*, 74:051907, Nov 2006.

[ROB05] S. Ramsey, D. Orrell, and H. Bolouri. Dizzy: Stochastic simulation of large-scale genetic regulatory networks. *Journal of Bioinformatics and Computational Biology*, 3, 2005.

[RPCG03] M. Rathinam, L. R. Petzold, Y. Cao, and D. T. Gillespie. Stiffness in stochastic chemically reacting systems: The implicit τ-leaping method. *The Journal of Chemical Physics*, 119(24):12784–12794, 2003.

[RT96] V. Ramaswami and P. G. Taylor. Some properties of the rate operators in level dependent quasi-birth-and-death processes with countable number of phases. *Communications in Statistics. Stochastic Models*, 12(1):143–164, 1996.

[SBM07] R. Sidje, K. Burrage, and S. MacNamara. Inexact uniformization method for computing transient distributions of Markov chains. *SIAM Journal on Scientific Computing*, 29(6):2562–2580, 2007.

[Sch21] J. Schur. Über lineare Transformationen in der Theorie der unendlichen Reihen. *Journal für die reine und angewandte Mathematik (Crelle's Journal)*, 1921:79–111, 1921.

[Sen73] E. Seneta. *Non-negative matrices: An introduction to theory and applications*. Wiley, 1973.

[SES02] P. S. Swain, M. B. Elowitz, and E. D. Siggia. Intrinsic and extrinsic contributions to stochasticity in gene expression. *Proceedings of the National Academy of Science of the United States of America*, 99(20):12795–12800, 2002.

[SHZ14] D. Spieler, E. M. Hahn, and L. Zhang. Model checking CSL for Markov population models. In *Proceedings of QAPL*, Electronic Proceedings in Theoretical Computer Science, 2014.

[SMH02] S. Schuster, M. Marhl, and T. Höfer. Modelling of simple and complex calcium oscillations. *European Journal of Biochemistry*, 269(5):1333–1355, 2002.

[Spi09] D. Spieler. Model checking of oscillatory and noisy periodic behavior in Markovian population models. Technical report, Saarland University, 2009. Master thesis available at `http://mosi.cs.uni-saarland.de/?page_id=93`.

[Spi13a] D. Spieler. Characterizing oscillatory and noisy periodic behavior in Markov population models. In *Proceedings of QEST*, pages 106–122, 2013.

[Spi13b] D. Spieler. Geobound. `http://mosi.cs.uni-saarland.de/?page_id=74`, September 2013.

[SS63] J. C. Shepherdson and H. E. Sturgis. Computability of recursive functions. *Journal of the ACM*, 10(2):217–255, 1963.

[Ste94] W. J. Stewart. *Introduction to the Numerical Solution of Markov Chains*. Princeton University Press, 1994.

[SW13] D. Spieler and V. Wolf. Efficient steady state analysis of multimodal markov chains. In *Analytical and Stochastic Modeling Techniques and Applications*, volume 7984 of *Lecture Notes in Computer Science*, pages 380–395. Springer, Berlin Heidelberg, 2013.

[Tar72] R. Tarjan. Depth-first search and linear graph algorithms. *SIAM Journal on Computing*, 1(2), 1972.

[Tri02] K. S. Trivedi. *Probability and statistics with reliability, queuing and computer science applications*. John Wiley and Sons Ltd., Chichester, UK, 2nd edition edition, 2002.

[TV13] M. Telek and M. Vécsei. Analysis of fluid queues in saturation with additive decomposition. In A. Dudin, V. Klimenok, G. Tsarenkov, and S. Dudin, editors, *Modern Probabilistic Methods for Analysis of Telecommunication Networks*, volume 356 of *Communications in Computer and Information Science*, pages 167–176. Springer Berlin Heidelberg, 2013.

[TvO01] M. Thattai and A. van Oudenaarden. Intrinsic noise in gene regulatory networks. *Proceedings of the National Academy of Science of the United States of America*, 98(15):8614–8619, 2001.

[Twe75] R. L. Tweedie. Sufficient conditions for regularity, recurrence and ergodicity of Markov processes. *Mathematical Proceedings of the Cambridge Philosophical Society*, 78:125, 1975.

[vD88] N. M. van Dijk. On the finite horizon Bellman equation for controlled Markov jump models with unbounded characteristics. *Stochastic Processes and their Applications*, 28:141–157, 1988.

[VdV03] H.A. Van der Vorst. *Iterative Krylov methods for large linear systems*. Cambridge monographs on applied and computational mathematics. Cambridge University Press, 2003.

[Ver96] Jan Verschelde. *Homotopy continuation methods for solving polynomial systems*. PhD thesis, Department of Computer Science, K.U.Leuven, Leuven, Belgium, May 1996. Haegemans, Anny and Van de Vel, Hugo (supervisors).

[vMS94] A. P. A. van Moorsel and W. H. Sanders. Adaptive uniformization. *Communications in Statistics - Stochastic Models*, 10(3):619–647, 1994.

[Vol26] V. Volterra. Fluctuations in the abundance of a species considered mathematically. *Nature*, 118:558–560, 1926.

List of Definitions

List of Models

List of Theorems

List of Examples

List of Figures

List of Tables