The Process Algebra for Located Markovian Agents and Scalable Analysis Techniques for the Modelling of Collective Adaptive Systems

Cheng Feng

Doctor of Philosophy
Laboratory for Foundations of Computer Science
School of Informatics
University of Edinburgh
2016

Abstract

Recent advances in information and communications technology have led to a surge in the popularity of artificial Collective Adaptive Systems (CAS). Such systems, comprised by many spatially distributed autonomous entities with decentralised control, can often achieve discernible characteristics at the global level; a phenomenon sometimes termed *emergence*. Examples include smart transport systems, smart electricity power grids, robot swarms, etc. The design and operational management of CAS are of vital importance because different configurations of CAS may exhibit very large variability in their performance and the quality of services they offer. However, due to their complexity caused by varying degrees of behaviour, large system scale and highly distributed nature, it is often very difficult to understand and predict the behaviour of CAS under different situations. Novel modelling and quantitative analysis methodologies are therefore required to address the challenges posed by the complexity of such systems.

In this thesis, we develop a process algebraic modelling formalism that can be used to express complex dynamic behaviour of CAS and provide fast and scalable analysis techniques to investigate the dynamic behaviour and support the design and operational management of such systems. The major contributions of this thesis are:

(i) development of a novel high-level formalism, PALOMA, the Process Algebra for Located Markovian Agents for the modelling of CAS. CAS specified in PALOMA can be automatically translated to their underlying mathematical models called Population Continuous-Time Markov Chains (PCTMCs).

(ii) development of an automatic moment-closure approximation method which can provide rapid Ordinary Differential Equation-based analysis of PALOMA models.

(iii) development of an automatic model reduction algorithm for the speed up of stochastic simulation of PALOMA/PCTMC models.

(iv) presenting a case study, predicting bike availability in stations of Santander Cycles, the public bike-sharing system in London, to show that our techniques are well-suited for analysing real CAS.

Acknowledgements

First and foremost, I want to give many thanks to my supervisor, Prof. Jane Hillston, who has given me tremendous help and invaluable advice over the past three years. Without her inspiring, patient and supportive supervision, I would never have finished this thesis. The enjoyable PhD journey with Prof. Hillston will always be a cherished memory.

I would also like to thank many other colleagues within the QUANTICOL project including Stephen Gilmore, Vashti Galpin, Daniël Reijsbergen, Luca Bortolussi for their kindly help and collaboration on my work. Thanks extend to my great office-mates, Anastasis Georgoulas, Ludovica Luisa Vissat, Natalia Zoń, Yota Katsikouli, Maria Astefanoaei and Alireza Pourranjbar, who provided me a fantastic work environment.

I also need to say many thanks to my family and friends for their endless support throughout the past three years. Special thanks to Lili Liu, who always cheered me up and accompanied me during writing up the thesis.

Finally, I would like to acknowledge that all my PhD work was financially supported by the EU QUANTICOL studentship and the Edinburgh Informatics Global scholarship.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified. A preliminary version of the material in Chapter 3 has been appeared in [Feng and Hillston, 2014] and [Feng et al., 2016b]. Part of the work in Chapter 4 has been appeared in [Feng et al., 2016a]. Chapter 5 is based on the work published in [Feng and Hillston, 2015]. Chapter 6 is an extended version of the material published in [Feng et al., 2016b].

(*Cheng Feng*)

Table of Contents

1 Introduction **1**

1.1 Motivation 1

1.2 Thesis Structure and Contributions 3

2 Background **5**

2.1 Continuous Time Markov Chain 5

2.1.1 Infinitesimal Generator Matrix 6

2.1.2 Transient State Distribution 6

2.1.3 Steady State Distribution 7

2.1.4 State Space Explosion 8

2.2 Population Continuous Time Markov Chain 9

2.2.1 Stochastic Simulation of PCTMCs 11

2.2.2 Fluid Approximation of PCTMCs 13

2.3 High-level CTMC/PCTMC Formalisms 17

2.3.1 Markovian Agent Models 18

2.3.2 Stochastic Process Algebras 21

3 PALOMA: A Process Algebra For Located Markovian Agents **23**

3.1 Syntax 24

3.2 A Motivating Example 28

3.3 Individual-based Semantics 30

3.3.1 The Delay Transition Relation 31

3.3.2 The Probabilistic Transition Relation 33

3.3.3 CTMC 35

3.4 Population-level Semantics 35

3.4.1 PCTMC 37

3.5 Simulation 39

4 Automatic Moment Closure Approximation of PALOMA Models **43**
4.1 The Derivation of Moment ODEs 44
4.2 Moment ODE Reduction 46
4.2.1 Neighbourhood Relation 47
4.2.2 Reduction Method 47
4.3 Moment-closure Method 49
4.4 Case Studies 52
4.4.1 An Epidemiological SIS Model 52
4.4.2 A Wireless Sensor Network Model 55
4.4.3 The City Bike-sharing Model 60
4.5 Summary 60

5 The Speed-up of Stochastic Simulation of PCTMCs **63**
5.1 Related Works 65
5.2 Reduction Proposal Generation 66
5.2.1 Direct Coupling Coefficient 66
5.2.2 Directed Coupling Graph 68
5.2.3 Coupling Propagation 69
5.2.4 Generating Algorithm for Reduction Proposals 70
5.3 Error Control of Reduction Proposals 71
5.4 Searching for the Optimal Reduction Proposal 72
5.5 Evaluation 73
5.5.1 Experiments on the Bike-sharing Example 73
5.5.2 Experiments on the Smart Taxi Example 76
5.6 Summary 78

6 Moment-based Availability Prediction for Bike-sharing Systems **81**
6.1 Introduction 82
6.2 PCTMC with Time-dependent Rates 83
6.3 Markov Queueing Model 84
6.4 PCTMC of Bike-sharing Model 86
6.4.1 A Naive PCTMC Model 86
6.4.2 Directed Contribution Graph with Contribution Propagation . 87
6.4.3 The Reduced PCTMC Model 90
6.5 Reconstructing the Probability Distribution Using the Maximum Entropy Approach 93

6.5.1 Reconstruction Algorithm 94
6.6 Experiments 95
6.6.1 Root Mean Square Error 96
6.6.2 Probability of Making a Correct Recommendation 96
6.6.3 Time Cost 97
6.7 Conclusions 98

7 Conclusions **101**
7.1 Summary 101
7.2 Further Works 102
7.2.1 Enhancing Expressiveness of PALOMA 103
7.2.2 Defining Useful Performance Measures for CAS 103
7.2.3 Statistical Model Checking of CAS 104
7.2.4 Learning Model Parameters From Data 105

Bibliography **107**

List of Figures

2.1 The schematic structure of two Markovian agents in two locations . . 19

3.1 The city bike sharing model 29
3.2 The delay transition relation for PALOMA 32
3.3 The probabilistic transition relation for PALOMA 34
3.4 Structural congruence in PALOMA 36
3.5 The population-level structured operational semantics of PALOMA . 38
3.6 The trajectories of the average number of available bikes in the stations in the agent-based and population-level simulation of the bike-sharing model each with 10000 runs 41

4.1 The correlation graph of a moment variable $\mathbb{E}[x_1x_2x_3x_4x_5^2]$. 48
4.2 The first moment of infected population 54
4.3 The second moment of infected population 55
4.4 The topology of the WSN model 55
4.5 The first moment of number of sensor nodes with different pheromone level 58
4.6 The second moment of number of sensor nodes with different pheromone level 59
4.7 The expected pheromone level in each cell at time 100. 59
4.8 The first three moments of number of available bikes in the central station 61

5.1 The directed coupling graph for the PCTMC with population variables (A,B,C,D), and transitions $(\tau_1,\tau_2,\tau_3,\tau_4)$. Weights on edges are the direct coupling coefficients. 69
5.2 The proportional reduction of simulation time, number of transitions with different acceptable error thresholds in the experiments on the bike-sharing example 75

5.3 The proportional reduction of simulation time, number of transitions with different acceptable error thresholds in the experiments on the smart taxi example. 78

6.1 The time-inhomogeneous Markov queue for station i 85
6.2 The number of bikes in use in 20 minute slots from 06:00 to 22:00 in Santander Cycles, London during weekdays and weekends. 86
6.3 An example directed contribution graph with six stations 88
6.4 The empirical cumulative distribution function of contribution coefficients (x is the value of contribution coefficients) 90

List of Tables

3.1 Important notations in Chapter 3 24
3.2 The bike-sharing model simulation configuration (unit of time in simulation: minute) 39
3.3 Simulation time cost of 10000 runs of the bike-sharing model 39

4.1 The SIS model simulation configuration 53
4.2 Simulation vs. moment analysis of the SIS model 56
4.3 The WSN model simulation configuration 58
4.4 Simulation vs. moment analysis of the WSN model 58
4.5 The bike-sharing model simulation configuration 60
4.6 Simulation vs. moment analysis of the bike-sharing model 62

5.1 The average error (with 99% confidence interval) caused by reduction with different acceptable error thresholds in the experiments on the bike-sharing example. 76
5.2 The average error (with 99% confidence interval) caused by reduction with different acceptable error thresholds in the experiments on the smart taxi example. 78

6.1 The calculated RMSE on the prediction of the number of available bikes 96
6.2 Average score of making a recommendation to the "Will there be a bike?" query with 95% confidence interval 98
6.3 Average score of making a recommendation to the "Will there be a slot?" query with 95% confidence interval 99
6.4 Time cost to make a prediction with 95% confidence interval 100

Chapter 1

Introduction

1.1 Motivation

Collective adaptive systems (CAS) as a broad term, often refers to systems composed of many individuals which adapt their behaviour locally without centralised control whereas they exhibit collective behaviour at the global level. Such systems are extremely common in the natural world. Examples include colonies of ants, flocks of birds and swarms of bees. In the man-made world, there are also many engineered CAS such as smart transport systems, smart electricity power grids, robot swarms and so on. More specifically, CAS can be viewed as being constituted by a large number of spatially distributed simple entities that interact locally with one other as well as the environment. Each entity may have its own properties and objectives. At the individual level, entities perform actions following rules with very little or even no knowledge about the global system. Although there is no centralised control to instruct how individuals should behave, interactions between these entities lead to the emergence of some interesting collective behaviour at the system level. In the natural world, the emergence of collectives often exhibits some interesting properties such as global optimization. For instance, ants can always find the shortest path between their colony and a food source through laying down pheromone trails on the road by each individual ant. In the man-made domain, the designing goals of CAS are often the achievement of more efficient allocation and utilisation of resources. As an illustration, by tracking and predicting the behaviour of passengers, how many taxis and buses are needed at a specific location can be evaluated, thus more efficient deployment of public transport can be achieved.

In comparison with traditional collaborative systems with centralised control, CAS

have advantages on their flexibility (members of CAS can quickly adapt to their changing environment), robustness (failure of individuals will be less likely to influence the performance of the whole system) and self-organization (individuals need very little supervision, thus no top down control required). As a result, with the pervasion of ICT-based systems, application of CAS has become more and more popular, which contributes to an increasing societal importance of CAS modelling, e.g., in the context of smart cities [ter Beek et al., 2014]. Modelling existing CAS can help us better understand the systems, possibly improve them, and predict their future behaviour to provide suggestions to system users for satisfaction enhancement. Modelling new CAS can provide us the chance to conduct a comprehensive analysis of their design and investigate all aspects of their behaviour before they are put into operation. For example, we may want to investigate the behaviour of CAS with respect to their specification in order to make sure that their performance meets the predefined requirements, and some unexpected behaviour like system crash does not show up in the design period. As a result, a framework for quantitative modelling and analysis of CAS is required in order to support comprehensive mathematical experiments on these systems.

There are many challenges for quantitative modelling and analysis of CAS [Hillston, 2013]. First of all, entities in CAS are often spatially distributed, and the interaction between them can be influenced by their localities. Thus, in order to faithfully capture CAS, the spatial arrangement of entities as well as the constraints that the spatial arrangement places on the interaction between entities must be represented. For example, interaction may only be allowed for entities which are co-located or within a certain physical distance of each other whereas in some circumstances, space may be segmented in a way such that even physically close entities are unable to communicate. Furthermore movement can also be a crucial aspect of the behaviour of entities within the system. Therefore it becomes essential to develop modelling formalisms in which space is captured explicitly, and in which the same entity in different locations can be distinguished. This poses significant challenges to both model expression and solution.

Secondly, real CAS usually embody rich forms of interaction between entities. For example, communication between entities can be synchronous and asynchronous, one-to-one and one-to-many, etc. Moreover, interaction between entities can also be influenced by their states, attributes, and possibly the environment. For example, the transmission range of packets in a wireless sensor network can be influenced by the battery levels of nodes and the environment conditions (underwater, forest, etc.). Pro-

viding a mechanism to allow all these types of interaction to be captured is difficult but necessary. Meanwhile, given the scale of CAS, which often rely on large populations of entities in order to meet their objectives, we must also find efficient and scalable mechanisms both to express and to analyse the developed models. This thesis is aimed at addressing the above challenges by providing a high-level formalism and its scalable analysis techniques for the modelling of CAS.

1.2 Thesis Structure and Contributions

The first contribution of this thesis is the development of a novel high-level modelling formalism, PALOMA, the process algebra for located Markovian agents, for the modelling of CAS. Specifically, as CAS are often very complicated, attempts to model them without a high-level formalism are likely to be time-consuming and error-prone. Stochastic process algebras [Clark et al., 2007], which were formal languages originally developed to represent concurrent dynamic systems, are well-suited for constructing models of CAS. PALOMA is a stochastic process algebra tailored for the modelling of CAS. Specifically, the novelty of PALOMA is that it supports explicit spatial representation and provides mechanisms to capture the various interaction patterns between entities in CAS. The underlying mathematical model of a PALOMA model is a Continuous Time Markov Chain (CTMC), which can be further mapped to a Population CTMC (PCTMC) through a counting abstraction. Based on the PCTMC generated via formal semantics, comprehensive quantitative analysis of the system can be conducted.

As CAS usually consist of a large number of entities, classical Markov chain analysis techniques based on linear-algebraic operations are entirely infeasible for analysing the derived PCTMCs from PALOMA models due to their extremely large state space. Among the scalable analysis techniques are the Ordinary Differential Equation (ODE)-based fluid approximation and discrete-event stochastic simulation; both can avoid storing the entire state space of the PCTMC. However, existing fluid approximation methods cannot be directly applied to PALOMA models, and because of the highly-heterogeneous nature of CAS, the derived ODEs for fluid approximation can easily exceed the ability of contemporary ODE solvers. Like most Monte Carlo approaches, the inefficiency of stochastic simulation has also become an obstacle for analysing CAS. Thus, the second contribution of this thesis is the development of a novel automatic fluid approximation methodology which can provide a rapid analysis of the

moments (mean, variance, covariance, skewness, kurtosis, etc.) of the populations of entities in an arbitrary PALOMA model. The third contribution is the development of a novel algorithm that can significantly accelerate stochastic simulation of PCTMCs, especially for CAS models, through automatic model reduction.

The last contribution of the thesis is a case study to show that our scalable analysis techniques can be easily adapted and applied to quantitative modelling and analysis of real CAS. Specifically, we use our scalable analysis techniques for bike availability prediction in stations of Santander Cycles, the public bike-sharing system in London. We show our model can outperform the time-inhomogeneous single station Markov queueing model on several performance metrics for bike availability prediction.

We summarise the structure of the thesis as follows. The next chapter gives the literature review of the mathematical background, existing analysis techniques and high-level formalisms for CTMC/PCTMCs. Chapter 3 presents the concepts, syntax and semantics of our modelling formalism, the process algebra PALOMA. The two novel scalable analysis techniques for fluid approximation and stochastic simulation of PALOMA/PCTMC models are introduced in Chapter 4 and 5, respectively. Chapter 6 gives the case study of bike availability prediction using moment-based fluid approximation. Finally, in chapter 7, the achievements of this thesis are summarised, and the directions of future work are discussed.

Chapter 2

Background

Before introducing our modelling formalism for CAS and its associated analysis techniques, we give a literature review of the background material in this chapter. Specifically, we will start with the introduction of continuous-time Markov chains (CTMCs), a mathematical framework which is widely studied in many literatures, e.g. ([Norris, 1998, Anderson, 2012]). The numerical analysis techniques and the state space explosion problem for CTMCs will be discussed. We then discuss a subset of CTMCs, population CTMCs (PCTMCs), which can achieve a compact representation of the underlying system being modelled, and is also the mathematical framework of our modelling formalism in this thesis. We will also review the common analysis techniques for PCTMCs. Having described the basic mathematical framework, we finally discuss high-level modelling formalisms that can help modellers to construct CTMC or PCTMC models for specific systems in a more intuitive manner.

2.1 Continuous Time Markov Chain

A continuous-time Markov chain (CTMC) is a continuous-time stochastic process which satisfies the Markov property. Specifically, a CTMC is characterised by a random variable $X(t)$, indexed over continuous-time t, and a countable discrete state space $\mathcal{S}$ such that $X(t) \in \mathcal{S}$ for all t. Moreover, with the Markov property, the future behaviour of a CTMC depends only on its current state, not on its historical behaviour. Formally, letting $\mathcal{F}_{X(s)}$ denote the history of X up to time s, and $j \in \mathcal{S}$, $0 \leq s \leq t$, the **Markov property** indicates $X(t)$ satisfies:

$$P\{X(t) = j \mid \mathcal{F}_{X(s)}\} = P\{X(t) = j \mid X(s)\}. \tag{2.1}$$

We also say a CTMC is **time-homogeneous** if the behaviour of the process does not depend on when it is observed. Thus, for all $0 \leq s \leq t$, it follows that

$$P\{X(t) = j \mid X(s)\} = P\{X(t-s) = j \mid X(0)\}. \tag{2.2}$$

2.1.1 Infinitesimal Generator Matrix

The dynamic behaviour of a CTMC is represented by the transitions between these states, and times spent in the states, which are also called *sojourn times*. Specifically, if a state $i \in \mathcal{S}$ is entered at time t and the next state transition takes place at time $t + T_i$, T_i is the sojourn time in state i. By the Markov property, at any time point t, the distribution of the time until the next change of state is independent of the time of the previous change of state. In other words, T_i satisfies the memoryless property, and is therefore exponentially distributed (since the exponential random variable is the only continuous random variable with this property). Hence at time t, the probability that there is a state transition in the interval $(t, t+h)$ is $q_i \cdot h + o(h)$ as $h \to 0$, where q_i is the parameter of the exponential distribution of the sojourn time in state i and $\mathbb{E}[T_i] = 1/q_i$. Suppose that if the process jumps out of state i, it will enter state j with probability p_{ij}. Then, for $i \neq j$, $i, j \in \mathcal{S}$, we have:

$$P(X(t+h) = j \mid X(t) = i) = q_{ij} \cdot h + o(h) \tag{2.3}$$

where $q_{ij} = q_i \cdot p_{ij}$ is the transition rate from state i to state j, which is also exponentially distributed according to the decomposition principle of exponential distributions.

With the definition of the transition rates between states, then a time-homogeneous CTMC with n states can be characterised by a $n \times n$ **infinitesimal generator matrix** $\mathbf{Q}$. The element in the jth column of the ith row of $\mathbf{Q}$ is the transition rate from state i to state j, i.e., q_{ij} $(i \neq j)$. The diagonal elements are chosen to ensure that the sum of the elements in every row is zero, i.e., $q_{ii} = -\sum_{j \neq i} q_{ij}$.

2.1.2 Transient State Distribution

For a CTMC with n states, given its initial state distribution $\pi_0 = \{\pi_1(0), \ldots, \pi_n(0)\}$, we are often interested in the transient state distribution of the CTMC at time t, denoted as $\pi(t)|_{\pi_0} = \{\pi_1(t), \pi_2(t), \ldots, \pi_n(t)\}$, where $\pi_i(t)$ is the transient state probability of the CTMC being in state i at time t. The transient state probability distribution is calculated as

$$\pi(t)|_{\pi_0} = \pi_0 \cdot \mathbf{P}(t) \tag{2.4}$$

where $\mathbf{P}(t)$ is a $n \times n$ transition probability matrix with each element

$$P_{ij}(t) = P(X(t) = j | X(0) = i). \tag{2.5}$$

The evolution of $\mathbf{P}(t)$ for a time-homogeneous CTMC satisfies the Kolmogorov's forward equations:

$$\frac{\mathrm{d}\,\mathbf{P}(t)}{\mathrm{d}\,t} = \mathbf{P}(t) \cdot \mathbf{Q} \tag{2.6}$$

with initial condition $\mathbf{P}(0) = \mathbf{I}$ is the $n \times n$ identity matrix. Therefore, the transient state distribution of the CTMC can be calculated by the following set of ordinary differential equations (ODEs):

$$\frac{\mathrm{d}\,\pi(t)}{\mathrm{d}\,t} = \pi(t) \cdot \mathbf{Q} \tag{2.7}$$

with $\pi(0) = \pi_0$. The above ODEs can be solved using standard numerical simulation algorithms such as the Runge-Kutta algorithm. Another possibility is to analytically calculate the transient state distribution using a matrix exponential of Equation 2.6 such that:

$$\pi(t) = \pi_0 \cdot \exp(\mathbf{Q}t) = \pi_0 \cdot \sum_{k=0}^{\infty} \frac{(\mathbf{Q}t)^k}{k!} \tag{2.8}$$

The computation of the above equation is, however, unstable due to the coexistence of both positive and negative elements in $\mathbf{Q}$. Thus, an improved algorithm is to use uniformisation [Grassmann, 1977, Stewart, 2009], which transforms the CTMC into a discrete time Markov chain (DTMC). Specifically, the original CTMC is scaled by the fastest transition rate γ, where

$$\gamma \geq \max_{i \in \mathcal{S}} |q_{ii}|, \tag{2.9}$$

so that transitions occur at the same rate in every state. The transient state distribution can be then expressed as

$$\pi(t) = \sum_{k=0}^{\infty} \pi_0 \cdot (\mathbf{Q}')^k \frac{(\gamma t)^k}{k!} \exp(-\gamma t) \tag{2.10}$$

where $\mathbf{Q}' = \mathbf{I} + \mathbf{Q}/\gamma$. Since $\mathbf{Q}'$ only contains positive elements, numerical evaluation of the above equation is stable.

2.1.3 Steady State Distribution

Apart from the transient behaviour of a CTMC, modellers may also be concerned with behaviour of the CTMC over the long term, which we call the steady state distribution. Studying the steady state distribution of a CTMC is very useful because it will

overcome any bias introduced by choosing the initial state distribution of the CTMC. Specifically, a CTMC is said to be **irreducible** if and only if all states in $\mathcal{S}$ can be reached from all other states in a finite number of transitions, otherwise the CTMC is said to be **reducible**. Furthermore, the state probability distribution for an irreducible, time-homogeneous CTMC with a finite state space will always converge for large values of t, and the converged steady state distribution is independent of the chosen initial state distribution.

More specifically, let $\pi = \{\pi_1, \pi_2, \ldots, \pi_n\}$ denote the steady state distribution of an irreducible, time-homogeneous CTMC with n states, where

$$\pi_i = \lim_{t\to\infty} P(X(t) = i | X(0) = j) \quad \forall j \in \mathcal{S}, \tag{2.11}$$

then π takes the solution of the following system of linear equations:

$$-q_{ii} \cdot \pi_i + \sum_{j\neq i} q_{ji} \cdot \pi_j = 0 \tag{2.12}$$

where

$$\sum_{i\in\mathcal{S}} \pi_i = 1. \tag{2.13}$$

Equivalently, Equation 2.12 can also be written as a matrix equation:

$$\pi \mathbf{Q} = \mathbf{0} \tag{2.14}$$

2.1.4 State Space Explosion

It can be seen that all the above transient and steady-state analysis techniques of CTMCs rely on the $n \times n$ infinitesimal generator matrix $\mathbf{Q}$, and the n-dimensional probability vector $\pi(t)$ or π, where n is the size of the state space. Unfortunately, even simple CTMC models can have a very large number of states, thus the size of the generator matrix and/or the probability vector often exceeds what can be handled in the memory of traditional computers. This problem is known as **state space explosion**.

There has been much research work aiming to enable modellers to evaluate CTMCs which suffer from state space explosion. For example, several techniques have been proposed to represent the generator matrix in a more compact form in memory, such as Multi-Terminal Binary Decision Diagrams [Fujita et al., 1997], Edge-Valued Decision Diagrams [Ciardo and Siminiceanu, 2002], and the Kronecker product approaches [Plateau and Atif, 1991, Ciardo and Miner, 1999]. Further implementation optimisations can also be applied, such as using disks to store the infinitesimal generator matrix [Deavours and Sanders, 1998] and parallelizing the computation

[Knottenbelt and Harrison, 1999]. All the above techniques are useful to mitigate the state space explosion problem. However, for most large CTMCs, they are still not scalable enough.

Another common approach to reducing state space is through state aggregation [Gilmore et al., 2001, Buchholz, 1994]. The idea is to aggregate symmetric states which exhibit the same count of indistinguishable agents into superstates. Intuitively, this means we shift the modelling view from individuals to populations. In general, the resulting aggregated population model must satisfy a lumpability condition which means that it has to be a CTMC such that all its transition rates can be defined as functions on the aggregated state space. While individuality is sacrificed in the aggregated model, interactions between individuals are still captured. This implies that the state changes in the original CTMC are represented as changes of population sizes. The resulting CTMC is often called a population CTMC (PCTMC) as described in Section 2.2, which will be the focus of this thesis. Although the resulting PCTMC usually still suffers from state space explosion, state aggregation is still an important step since many efficient approaches to dealing with state space explosion can only be applied on PCTMCs.

A straightforward way to overcome state space explosion is to use stochastic simulation, in which the state space of the CTMC can be explored on-the-fly. By doing this, we can find the states step-by-step as the simulation progresses, thus avoiding the construction of the whole state space at once. However, in order to obtain accurate estimates of the state distribution, a large number of simulation traces have to be evaluated which is often computationally expensive.

For models with large populations, a generally more efficient approach to tackling state space explosion is the use of fluid approximation. Here the key idea is to approximate the behaviour of a discrete event system which jumps between discrete states by a continuous system which moves smoothly over a continuous state space.

Since stochastic simulation algorithms and fluid approximation methods are usually applied on population models, we delay their detailed reviews to Sections 2.2.1 and 2.2.2, respectively.

2.2 Population Continuous Time Markov Chain

A population continuous time Markov chain (PCTMC) is a CTMC whose states are captured by a numerical population vector, and transitions between states are defined

by changes in some of the populations, with rates expressed as functions of populations. The analysis of interest for such models is often the evolution of different populations over time. Formally, a PCTMC can be represented as a tuple $\mathcal{P} = (\mathbf{X}, \mathcal{T}, \mathbf{X}_0)$, where:

- $\mathbf{X} = (x_1, ..., x_n) \in \mathbb{Z}^n_{\geq 0}$ is an integer vector with the ith ($1 \leq i \leq n$) component representing the current population level of an agent type S_i. Each x_i takes values in a finite domain $\mathcal{D}_i \in \mathbb{Z}_{\geq 0}$. Hence, $\mathcal{D} = \prod_{i=1}^{n} \mathcal{D}_i$ is the state space of the model.

- $\mathcal{T} = \{\tau_1, ..., \tau_m\}$ is the set of transitions, of the form $\tau = (r_\tau(\mathbf{X}), \mathbf{d}_\tau)$, where:

  1. $r_\tau(\mathbf{X}) : \mathbb{Z}^n_{\geq 0} \to \mathbb{R}_{\geq 0}$ is the rate function, associating with each transition the rate of an exponential distribution, depending on the global state of the model.

  2. $\mathbf{d}_\tau = (d^1_\tau, \ldots, d^n_\tau) \in \mathbb{Z}^n$ is the update vector which gives the net change for each element of $\mathbf{X}$ caused by transition τ.

- $\mathbf{X}_0 \in \mathbb{Z}^n_{\geq 0}$ is the initial state of the model.

For convenience, transitions in PCTMCs can be easily expressed in the chemical reaction style with S_i being a specific molecular species, $r_\tau(\mathbf{X})$ being the reaction propensity function and $\mathbf{d}_\tau$ capturing the consumed and produced population of species by the reaction:

$$\underline{N_1} S_1 + \ldots + \underline{N_n} S_n \quad \longrightarrow_\tau \quad \overline{N_1} S_1 + \ldots + \overline{N_n} S_n \quad \text{at rate } r_\tau(\mathbf{X})$$

where the net change on the population of agent type/species S_i due to transition τ is given by $d^i_\tau = \overline{N_i} - \underline{N_i}$ ($1 \leq i \leq n$).

Given a PCTMC $\mathcal{P}$, it is straightforward to define its infinitesimal generator matrix $\mathbf{Q}$ by the $\mathcal{D} \times \mathcal{D}$ matrix where each element:

$$q_{\mathbf{x},\mathbf{x}'} = \sum \{ r_\tau(\mathbf{X}) \mid \tau \in \mathcal{T} \wedge \mathbf{X}' = \mathbf{X} + \mathbf{d}_\tau \} \tag{2.15}$$

Clearly, the size of the state space of $\mathcal{P}$ can easily become intractable for the traditional analysis techniques for CTMCs such as introduced in Section 2.1.2 and 2.1.3. Thus, in most cases, stochastic simulation and fluid approximation are the only scalable analysis techniques for PCTMCs.

2.2.1 Stochastic Simulation of PCTMCs

The most straightforward way to analyse a PCTMC is to use stochastic simulation to numerically compute individual realisations of the stochastic process. The idea is simply Monte Carlo: if we sample enough realisations of the stochastic process, then the estimates of statistical properties of the stochastic process will eventually converge to their true values. Specifically, given a PCTMC $\mathcal{P} = (\mathbf{X}, \mathcal{T}, \mathbf{X}_0)$ and the end time of simulation t_e, a trace of the PCTMC $\mathbf{X}(t)$ for $t \leq t_e$ is often calculated by Gillespie's algorithm [Gillespie, 1977] which is also called the direct method, or the standard stochastic simulation algorithm (SSA):

Algorithm 2.1 The SSA of PCTMCs

Require: $\mathcal{P} = (\mathbf{X}, \mathcal{T}, \mathbf{X}_0)$, t_e

1: Set $t = 0$, $\mathbf{X} = \mathbf{X}_0$,
2: **while** $t \leq t_e$ **do**
3: Generate two random numbers α, β uniformly distributed in $(0,1)$
4: Compute $r = \sum_{\tau \in \mathcal{T}} r_\tau(\mathbf{X})$
5: Compute the time when the next transition fires as $t + h$, where $h = \frac{1}{r}\ln[1/\alpha]$
6: **if** $t + h > t_e$ **then**
7: **break**
8: **end if**
9: Compute which transition fires at time $t + h$ by finding τ_j such that

$$\beta \geq \frac{1}{r}\sum_{i=1}^{j-1} r_{\tau_i}(\mathbf{X}) \quad \text{and} \quad \beta < \frac{1}{r}\sum_{i=1}^{j} r_{\tau_i}(\mathbf{X})$$

10: Set $t = t + h$, $\mathbf{X} = \mathbf{X} + \mathbf{d}_{\tau_j}$
11: **end while**

We can repeatedly apply the above algorithm to compute a large number of traces so as to estimate some statistical properties of a PCTMC such as the distribution or moments of the population vector $\mathbf{X}$ at any time point $t \leq t_e$.

In principle, the standard SSA is able to simulate all PCTMCs. However in practice, due to the fact that the cost of SSA increases with the population size and the number of reactions, the inefficiency of SSA has clearly become an obstacle for many realistic models. As a result, numerous approaches have been proposed to improve the efficiency of SSA, including the optimized direct method [Cao et al., 2004], the

next reaction method [Gibson and Bruck, 2000], and the composition rejection algorithm [Slepoy et al., 2008]. However, all these approaches are exact methods which means they have to simulate every transition event, thus their efficiency is limited by the number of transitions in the PCTMC.

In order to overcome the restriction of exact simulation methods, many approximate methods have been proposed, among which the tau-leaping methods [Gillespie, 2001, Cao et al., 2006] are widely used. Specifically, the tau-leaping algorithm speeds up SSA by firing multiple transitions during a selected time interval instead of firing one transition at each step in SSA, given that the transition rates remain relatively constant during the selected time interval. Other approximate approaches mostly focus on exploiting the presence of different time scales in the model. The common idea behind these approaches is to construct abstracted models, by decomposing a model into a fast and a slow subsystem (in some cases, even more time scales can be considered, but the general decomposition idea is the same). The fast subsystem is assumed to reach an equilibrium state at a time scale which is much faster than the time scale of the slow subsystem. Hence, the fast subsystem needs not to be simulated once it reaches its equilibrium state, and the system dynamics are dominated by the slow subsystem which can be simulated solely based on the equilibrium state of the fast subsystem. According to the detailed decomposition methods, the approaches can be further divided into two categories, namely the Quasi-Steady-State approaches [Rao and Arkin, 2003, Mastny et al., 2007, Pu et al., 2011] and the Quasi-Equilibrium approaches [Cao et al., 2005, Bortolussi and Paškauskas, 2014]. In the Quasi-Steady-State approaches, populations are partitioned into fast and slow, and transitions are separated accordingly. The fast-changing populations are approximated with a near stationary distribution over a long period by their quasi-steady state, and the dynamics of slow-changing populations are simulated upon the quasi-steady state. Alternatively, the Quasi-Equilibrium approaches start by partitioning the transitions into fast and slow, and then separating the system into a virtual fast process which only consists of fast transitions and a slow process which only consists of slow transitions. The virtual fast process reaches a stochastic quasi-equilibrium state very quickly between consecutive slow transitions. Then, the rate of slow transitions can be approximated based on the quasi-equilibrium distribution of the virtual fast process. For approaches in both categories, a common downside is that the identification of fast and slow subsystems is usually a manual process, and requires expert knowledge of the dynamic behaviour of the model. This process is expensive and error-prone which significantly hinders

the usage of these approaches. Recently, some work has been done to automate the separation process by obtaining the knowledge of the time-scales through some experimental simulation runs of the entire model [Wu et al., 2012, Bortolussi et al., 2015b]. Multi-time scale separation is not in conflict with the tau-leaping algorithm. Instead, a combination of them can actually give further acceleration [Cao and Petzold, 2008].

Given various acceleration algorithms, sometimes it may also be difficult to decide which algorithm is most efficient with respect to a particular model. Indeed, even for the same model, the performance of a simulation algorithm can vary during simulation runs. Therefore, some adaptive algorithms have been proposed, among which the *adaptive simulator* [Helms et al., 2015] is a generic adaptation scheme that is realised as a wrapper for other simulation algorithms. More specifically, based on machine learning methods, the adaptive simulator can automatically pick the best simulation algorithm from the wrapped ones according to the current state of the model during simulation. Thus it can achieve a better performance compared with the realisation of a single simulation algorithm. The adaptive simulator is implemented as a plug-in of JAMES II [Himmelspach and Uhrmacher, 2007], a well-known Java framework for modelling and simulation.

2.2.2 Fluid Approximation of PCTMCs

Even with so many acceleration algorithms, stochastic simulation is still costly when applied to large PCTMCs. As a result, fluid approximation which describes the moments (mean, variance, covariance, skewness, kurtosis, etc.) of population dynamics in a PCTMC using a system of coupled ODEs, becomes a more appealing method to analyse large PCTMCs since the associated evaluation cost remains almost unaffected as the populations grow.

For example, let $\mathcal{P} = (\mathbf{X}, \mathcal{T}, \mathbf{X}_0)$ be a PCTMC, and $\mathcal{P}^{(N)} = (\mathbf{X}^{(N)}, \mathcal{T}^{(N)}, \mathbf{X}_0^{(N)})$ be the corresponding normalised version, where N is the normalising factor which is usually setted to the initial total population size $\sum_{i=1}^{n} x_i(0)$; $\mathbf{X}^{(N)}$ is the normalised state vector in which $x_i^{(N)} = x_i/N$, $\forall(1 \leq i \leq n)$; $\mathcal{T}^{(N)}$ is the transition set in which each transition's rate function is density dependent such that $\tau = (r_\tau(\mathbf{X}^{(N)}), \mathbf{d}_\tau)$; $\mathbf{X}_0^{(N)}$ is the normalised initial state. Then, for each $\tau \in \mathcal{T}^{(N)}$, if there exists a Lipschitz continuous and bounded function g_τ, such that $r_\tau(\mathbf{X}^{(N)}) = N \cdot g_\tau(\mathbf{X}^{(N)})$, it is certain that the behaviour of the PCTMC becomes deterministic and converges to the solution of a mean-field differential equation [Kurtz, 1981, Bortolussi et al., 2013], which takes the

following form:

$$\frac{\mathrm{d}\mathbf{X}^{(N)}(t)}{\mathrm{d}t} = \sum_{\tau\in\mathcal{T}^{(N)}} \mathbf{d}_\tau \cdot r_\tau(\mathbf{X}^{(N)}(t)) \quad \text{with} \quad \mathbf{X}^{(N)}(0) = \mathbf{X}_0^{(N)} \quad \text{as} \quad N\to\infty \tag{2.16}$$

If populations are in the order of tens or hundreds of thousands, mean-field equations are generally very accurate [Bortolussi et al., 2013]. When populations are smaller, the mean-field equations can still occasionally give a good approximation of the evolution of average values of populations by ignoring any covariance between the individual population variables [Guenther et al., 2012]. However, in many cases it can also lead to high errors due to the groundless independence assumption between population variables.

As a result, a more robust approach is to derive ODEs for the moments of the population variables which also accounts for the dependence between populations. Specifically, let $M : \mathbb{R}^n_{\geq 0} \to \mathbb{R}$ be a moment function, then the moment described by M evolves according to the following differential equation [Engblom, 2006]:

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbb{E}[M(\mathbf{X}(t))] = \sum_{\tau\in\mathcal{T}} \mathbb{E}[(M(\mathbf{X}(t)+\mathbf{d}_\tau) - M(\mathbf{X}(t)))r_\tau(\mathbf{X}(t))] \tag{2.17}$$

with $\mathbb{E}[M(\mathbf{X}(0))] = M(\mathbf{X}_0)$. For example, if we set $M(\mathbf{X}(t)) = x_i(t)$, $M(\mathbf{X}(t)) = x_i^2(t)$, $M(\mathbf{X}(t)) = x_i(t)x_j(t)$, we get the following ODEs to describe the first moment, the second moment and the second-order joint moment respectively, of population variables in an arbitrary PCTMC:

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}\mathbb{E}[x_i] &= \sum_{\tau\in\mathcal{T}} \mathbb{E}[(x_i + d_\tau^i - x_i)r_\tau(\mathbf{X})] = \sum_{\tau\in\mathcal{T}} d_\tau^i \mathbb{E}[r_\tau(\mathbf{X})] \\
\frac{\mathrm{d}}{\mathrm{d}t}\mathbb{E}[{x_i}^2] &= \sum_{\tau\in\mathcal{T}} \mathbb{E}[((x_i + d_\tau^i)^2 - {x_i}^2)r_\tau(\mathbf{X})] \\
&= 2\sum_{\tau\in\mathcal{T}} d_\tau^i \mathbb{E}[x_i \times r_\tau(\mathbf{X})] + \sum_{\tau\in\mathcal{T}} {d_\tau^i}^2 \mathbb{E}[r_\tau(\mathbf{X})] \\
\frac{\mathrm{d}}{\mathrm{d}t}\mathbb{E}[x_i x_j] &= \sum_{\tau\in\mathcal{T}} \mathbb{E}[((x_i + d_\tau^i)(x_j + d_\tau^j) - x_i x_j)r_\tau(\mathbf{X})] \\
&= \sum_{\tau\in\mathcal{T}} d_\tau^i \mathbb{E}[x_j \times r_\tau(\mathbf{X})] + \sum_{\tau\in\mathcal{T}} d_\tau^j \mathbb{E}[x_i \times r_\tau(\mathbf{X})] + \sum_{\tau\in\mathcal{T}} d_\tau^i \times d_\tau^j \mathbb{E}[r_\tau(\mathbf{X})]
\end{aligned}$$

where we use x_i as short for $x_i(t)$, and $r_\tau(\mathbf{X})$ as short for $r_\tau(\mathbf{X}(t))$ for convenience. The above system of ODEs does not necessarily have a solution since the dynamics of lower-order moments can depend on higher-order moments. For example, if we let $r_\tau(\mathbf{X}) = c\, x_i x_j$, then an infinite number of ODEs are required to describe moment dynamics. In order to deal with this problem, various moment-closure methods are proposed to truncate the system of ODEs at a certain order of moment.

2.2.2.1 Moment Closure Methods

A generic closure method is to use the Taylor expansion to approximate moment variables, and then close the moment ODEs by assuming the moments of order higher than a certain threshold around the mean to be zero, which is the so-called moment expansion and central moment truncation method [Ale et al., 2013]. Specifically, let $\mathbb{E}[f(\mathbf{x})] = \mathbb{E}[f(x_1,\ldots,x_d)]$ denote an arbitrary moment variable in the right hand side of a moment ODE which is a factor of d population variables, $\mu = (\mu_1,\ldots,\mu_d)$ denote their means, then by doing a Taylor expansion of $f(x_1,\ldots,x_d)$ about the mean μ, we can rewrite the moment variable as follows:

$$\begin{aligned}
\mathbb{E}[f(\mathbf{x})] &= \sum_{n_1=0}^{\infty}\cdots\sum_{n_d=0}^{\infty}\frac{\mathbb{E}[(x_1-\mu_1)^{n_1}\cdots(x_d-\mu_d)^{n_d}]}{n_1!\cdots n_d!}\cdot\frac{\partial^{n_1+\ldots+n_d}f(\mu_1,\ldots,\mu_d)}{\partial x_1^{n_1}\cdots\partial x_d^{n_d}} \\
&= f(\mu_1,\ldots,\mu_d)+\sum_{i=1}^{d}\frac{\partial f(\mu_1,\ldots,\mu_d)}{\partial x_i}\mathbb{E}[(x_i-\mu_i)]+ \\
&\quad \frac{1}{2!}\sum_{i=1}^{d}\sum_{j=1}^{d}\frac{\partial^2 f(\mu_1,\ldots,\mu_d)}{\partial x_i\partial x_j}\mathbb{E}[(x_i-\mu_i)(x_j-\mu_j)]+ \\
&\quad \frac{1}{3!}\sum_{i=1}^{d}\sum_{j=1}^{d}\sum_{k=1}^{d}\frac{\partial^3 f(\mu_1,\ldots,\mu_d)}{\partial x_i\partial x_j\partial x_k}\mathbb{E}[(x_i-\mu_i)(x_j-\mu_j)(x_k-\mu_k)]+ \\
&\quad \cdots
\end{aligned}$$

Then, if we assume for any central moment whose order is larger than a specific value $\overline{m}$, is equal to zero (i.e., setting terms of the above Taylor expansion corresponding to $\sum_{i=1}^{d} n_i > \overline{m}$ to 0), the moment variable $\mathbb{E}[f(\mathbf{x})]$ will only contain moment variables whose orders are less than or equal to $\overline{m}$. Consequently, the system of moment ODEs can be closed at order $\overline{m}$. The merit of this closure method is that it can deal with any form of moment variables, however it also has a drawback that the derivatives in the Taylor expansion have to be manually calculated. Thus the method is not suitable for integrating into tools for automatic moment closure approximation.

For PCTMCs with only polynomial rates (thus moment variables are also in the form of polynomials), the most straightforward method for moment closure is to make a particular distribution assumption of the population variables in the PCTMC. For example, the normal moment closure assumes that the population variables at each point in time are approximately multivariate normal and therefore all third and higher-order moments can be expressed in terms of means and covariances. This relationship is captured by Isserlis' theorem [Isserlis, 1918]: For $\mathbf{x}$ multivariate normal with mean

μ and covariance matrix σ_{ij}, we have

$$\begin{aligned}\mathbb{E}[(\mathbf{x}-\mu)^{(\mathbf{m})}] &= \mathbb{E}[(x_1-\mu_1)^{(m_1)}\cdots(x_n-\mu_n)^{(m_n)}] = 0 \quad \text{if } o(\mathbf{m}) \text{ is odd}\\ \mathbb{E}[(\mathbf{x}-\mu)^{(\mathbf{m})}] &= \sum\prod \mathbb{E}[(x_i-\mu_i)(x_j-\mu_j)] \quad \text{if } o(\mathbf{m}) \text{ is even}\end{aligned}$$

where the notation $\sum\prod$ means summing over all distinct ways of partitioning $1,\ldots,n$ into pairs of i,j, $o(\mathbf{m}) = m_1+\ldots+m_n$. For example, we can approximate

$$\mathbb{E}[x_i x_j^2] \approx 2\mathbb{E}[x_j]\mathbb{E}[x_i x_j] + \mathbb{E}[x_i]\mathbb{E}[x_j^2] - 2\mathbb{E}[x_i]\mathbb{E}[x_j]^2$$

if multivariate normal distribution for population variables is assumed, which yields $\mathbb{E}[(x_i-\mu_i)(x_j-\mu_j)^2] = 0$.

Similarly, the lognormal closure [Keeling, 2000, Singh and Hespanha, 2006] assumes that the population variables follow a multivariate lognormal distribution. Thus, suppose we want to approximate moment variable $\mathbb{E}[\mathbf{x}^{(\mathbf{m})}]$ using only moment variables of order up to $o(\mathbf{m})-1$. Then, let $\mathbb{M} = \{\mathbf{m_1},\ldots,\mathbf{m_k}\}$ be the moment order set containing all the moments up to the order $o(\mathbf{m})-1$, we can approximate $\mathbb{E}[\mathbf{x}^{(\mathbf{m})}] \approx \prod_{p=1}^{k}(\mathbb{E}[\mathbf{x}^{\mathbf{m_p}}])^{\gamma_p}$, where $\gamma = (\gamma_1,\ldots,\gamma_k)$ is the unique solution to the following system of linear equations

$$\mathbf{C}^{(\mathbf{m})}_{(\mathbf{m}_s)} = \sum_{p=1}^{k}\gamma_p \mathbf{C}^{(\mathbf{m}_p)}_{(\mathbf{m}_s)} \quad \forall s = \{1,\ldots,k\} \quad \text{where} \quad \mathbf{C}^{(\hat{\mathbf{m}})}_{(\check{\mathbf{m}})} = \mathrm{C}^{(\hat{m}_1)}_{(\check{m}_1)}\mathrm{C}^{(\hat{m}_2)}_{(\check{m}_2)}\ldots\mathrm{C}^{(\hat{m}_n)}_{(\check{m}_n)}$$

where $\hat{\mathbf{m}}$ and $\check{\mathbf{m}}$ are vectors with size n, C^l_h is defined as:

$$\mathrm{C}^l_h = \begin{cases} \frac{l!}{(l-h)!h!} & \text{if } l \geq h \\ 0 & \text{if } l < h \end{cases}$$

For instance, if $\overline{m} = 2$, using the lognormal closure, we can approximate:

$$\mathbb{E}[x_i {x_j}^2] \approx \frac{\mathbb{E}[{x_j}^2]\mathbb{E}[x_i x_j]^2}{\mathbb{E}[x_i]\mathbb{E}[x_j]^2}$$

Compared with the normal moment closure method, the lognormal closure has the advantage that probabilities are only defined for position values, thus it is generally preferable for PCTMC models [Keeling, 2000].

Apart from the above two closures, there exist further closure methods based on other distribution assumptions such as the beta-binomial closure [Krishnarajah et al., 2005], Possion closure [Nåsell, 2003], etc. Which distribution assumption is the most valid will depend greatly on the problem being investigated.

There are also many other different closure techniques. For example, pair approximation which approximates the density of triplets (third-order moments) by counting certain link densities (second-order moments) that form the triplet, is typically used in spatial epidemic models [Keeling, 1999, Hiebeler, 2006] and self-organization of adaptive networks such as opinion formation [Nardini et al., 2008]. The maximum-entropy moment-closure makes no assumption about the correlation structure of population variables and chooses the distribution of maximum entropy subject to the constraints based on the knowledge of some lower dimensional marginals [Singer, 2004, Rangan and Cai, 2006]. Both the pair approximation and maximum-entropy moment-closure have the disadvantage that they require one ODE to capture the density or the population in each possible value, and thus are not scalable for PCTMCs.

2.3 High-level CTMC/PCTMC Formalisms

Although the CTMC is a powerful modelling formalism, the construction of CTMC models through the generator matrices is highly complicated and error prone. As a consequence, many high-level formalisms such as stochastic process algebras [Clark et al., 2007] and stochastic Petri nets [Balbo, 2001] have been developed to provide intuitive specification of CTMC models. The process algebraic descriptions are appealing for modelling complex systems since they offer the benefits of describing a system in a modeller-friendly compositional approach. Some formalisms have also been specifically suggested for PCTMC models, and each is suitable for describing a particular set of systems. For example, Grouped PEPA (GPEPA) [Hayden and Bradley, 2010] is a simple syntactic extension of PEPA [Hillston, 1996] which allows for the straightforward derivation of PCTMC models that can be analysed using fluid approximation techniques. GPEPA is particularly suitable for modelling systems with agents which compete for limited shared resources. The Bio-PEPA process algebra [Ciocchetta and Hillston, 2009] can be used to derive PCTMC models specifically for systems biology. ML-Rules [Helms et al., 2014] is a rule-based language developed for supporting the multi-level modelling of cell biological systems whose underlying mathematical framework is also a PCTMC. The stochastic Concurrent Constraint Programming (sCCP) formalism [Bortolussi, 2006] is primarily conceived to construct PCTMC models for bio-chemical processes. A noteworthy formalism is the Markovian agents modelling framework [Cerotti et al., 2010] which is particularly designed to define PCTMCs for describing systems composed by spa-

tially distributed interactive agents. In the remainder of this section, we will review the Markovian agents modelling framework and stochastic process algebras which we believe can provide us a useful insight into the problem of designing a high-level modelling formalism for capturing CAS.

2.3.1 Markovian Agent Models

In this section, we give an overview of the multi-class multi-message Markovian agents model (M^2MAM) which is a mathematical framework that could be a good start point for specifying CAS. M^2MAM is a modelling framework proposed by Cerrotti *et al.* for the modelling of collective systems comprised of populations of agents which are spatially distributed [Cerotti et al., 2010]. Several case studies have demonstrated that this is a powerful and useful framework [Gribaudo et al., 2008, Bruneo et al., 2012, Cerotti et al., 2008].

Specifically, a M^2MAM consists of a collection of Markov agents (MAs) spread over space which is represented by a finite set of locations. Each MA has a location attribute and can be denoted by a finite state machine in which two types of transitions can happen: *local* transitions and *induced* transitions. Local transitions occur whenever the MA changes its state spontaneously with a delay governed by an exponential distribution. Local transitions can also possibly emit messages that can cause the occurrence of MAs' induced transitions in the same or other locations. This enables location-based communication between MAs in the M^2MAM. The reception of an incoming message is governed by the perception function, which depends on both the locations and the states of the sender and receiver MAs. When a MA receives a message, it can either ignore or accept it. In the first case, nothing will happen whereas in the second case, the agent will change its state immediately by performing an induced transition.

Following [Cerotti et al., 2010], we use $\mathrm{MA}^c(\ell)$ to denote a MA of class c in location ℓ. A $\mathrm{MA}^c(\ell)$ can be defined as a tuple $\{Q^c(\ell), \Lambda^c(\ell), G^c(\ell,m), A^c(\ell,m), \pi_0^c(\ell)\}$, in which:

- $Q^c(\ell) = [q_{ij}^c(\ell)]$ is a $n_c \times n_c$ matrix, in which each element $q_{ij}^c(\ell)$ represents the rate of the local transition from state i to state j, with $q_{ii}^c(\ell) = -\sum_{j \neq i}^{n_c} q_{ij}^c(\ell)$ where n_c is the number of states of a MA of class c.
- $\Lambda^c(\ell) = [\lambda_i^c(\ell)]$ is a vector, in which each element $\lambda_i^c(\ell)$ denotes the rate of a self-jump transition which reenters the same state i, for a MA of class c.

- $G^c(\ell,m) = [g^c_{ij}(\ell,m)]$ is a $n_c \times n_c$ matrix in which each element $g^c_{ij}(\ell,m)$ describes the probability of $\mathrm{MA}^c(\ell)$ generating a message of type m during a local transition from state i to state j.

- $A^c(\ell,m) = [a^c_{ij}(\ell,m)]$ is a $n_c \times n_c$ matrix, in which each element $a^c_{ij}(\ell,m)$ $(i \neq j)$ describes the acceptance probability of messages of type m for $\mathrm{MA}^c(\ell)$, with induced transition from state i to state j whereas $a^c_{ii}(\ell,m)$ denotes the probability of dropping this message, and $a^c_{ii}(\ell,m) = 1 - \sum_{j\neq i} a^c_{ij}(\ell,m)$.

- $\pi^c_0(\ell)$ is the initial state probability distribution of an agent of class c in location ℓ.

Figure 2.1 illustrates the schematic structure of two Markovian agents in two locations ℓ and ℓ', where nodes denote states, solid lines between states denote local transitions, dashed lines denote induced transitions, dotted lines denote message emissions and receptions.

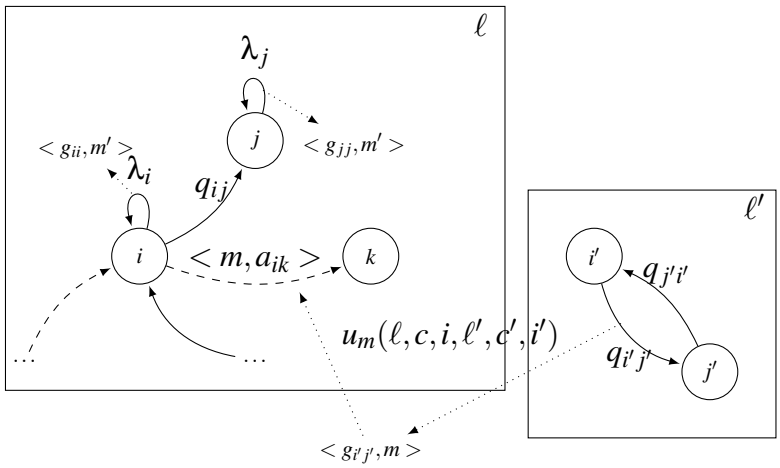

Figure 2.1: The schematic structure of two Markovian agents in two locations

2.3.1.1 Model Analysis

We next introduce how to analyse a M^{2}MAM according to [Cerotti et al., 2010]. Specifically, let $p^c_i(\ell,t)$ denote the density of agents of class c in state i, in location ℓ at time t. In M^{2}MAMs, the density of agents of different classes in a location is assumed to remain constant, this means that the value of $\sum_{i=1}^{n_c} p^c_i(\ell,t) = P^c(\ell)$ is invariant. Furthermore, we use a vector $p^c(\ell,t) = [p^c_i(\ell,t)]$ to denote the state density distribution

of agents of class c in location ℓ and at time t. The analysis of interest is the transient evolution of $p^c(\ell,t)$. It can be computed by solving a set of coupled ODEs.

First of all, the total rate at which messages of type m are generated by a MA of class c in state j and location ℓ can be computed by:

$$\beta_j^c(\ell,m) = \lambda_j^c(\ell) g_{jj}^c(\ell,m) + \sum_{k \neq j} q_{jk}^c(\ell) g_{jk}^c(\ell,m) \tag{2.18}$$

where the first term on the right hand side of the above equation gives the rate at which messages of type m are generated by the MA in state j by a self-jump transition, whereas the second term denotes the rate of message generation by the MA during a local transition from state j to another state.

With $\beta_j^c(\ell,m)$, we can compute $\gamma_{ii}^c(\ell,m,t)$, the total reception rate of messages of type m by a MA of class c in state i and location ℓ, at time t. The rate $\gamma_{ii}^c(\ell,m,t)$ is contributed by all the messages of type m generated by MAs of all classes in all states and all locations, as long as they can be perceived by the receiver MA. Thus, $\gamma_{ii}^c(\ell,m,t)$ is obtained by the following equation:

$$\gamma_{ii}^c(\ell,m,t) = \sum_{\ell' \in \mathcal{L}} \sum_{c'=1}^{C} \sum_{j=1}^{n_{c'}} u_m(\ell,c,i,\ell',c',j) \beta_j^{c'}(\ell',m) p_j^{c'}(\ell',t) \tag{2.19}$$

where $\mathcal{C} = \{1,\ldots,C\}$ is the set of agent classes in the model, $\mathcal{L}$ is the location set over which the MAs are spread, $u_m(\ell,c,i,\ell',c',j)$ is the perception function of message m, whose value represents the probability that an agent of class c, in state i, and in location ℓ perceives a message m sent by an agent of class c' in state j and in position ℓ'.

Finally, we use a diagonal matrix, $\Gamma^c(\ell,m,t) = diag(\gamma_{ii}^c(\ell,m,t))$ to collect the rates in Equation 2.19, and the infinitesimal generator matrix $K^c(\ell,t)$ for the CTMC of agents of class c in location ℓ at time t can be obtained by:

$$K^c(\ell,t) = Q^c(\ell) + \sum_m \Gamma^c(\ell,m,t)[A^c(\ell,m) - I] \tag{2.20}$$

where I is the identity matrix, the first term on the right hand side of the above equation is the infinitesimal generator matrix of the CTMC for local transitions, and the second term gives the infinitesimal generator matrix for induced transitions.

Shifting to a mean field view, the evolution of $p^c(\ell,t)$ can be studied by solving the following ODEs:

$$\frac{\mathrm{d}p^c(\ell,t)}{\mathrm{d}t} = p^c(\ell,t) K^c(\ell,t) \quad \forall(\ell,c) \tag{2.21}$$

with initial conditions $p^c(\ell,t_0) = P^c(\ell)\pi_0^c(\ell) \quad \forall(\ell,c)$.

2.3.2 Stochastic Process Algebras

Process algebras are a family of formal languages which were originally designed for the modelling of systems characterised by communication and concurrency [Milner, 1989]. These systems consist of individual agents which act independently or communicate with others in order to access some shared resources to achieve their goals. Process algebras provide a straightforward way to define, interpret and analyse these systems. By using the various operators (prefix, choice, parallel composition, etc.) in the process algebras, the modeller can specify the distinct agents in the system, the action sequences they perform, and the components can be further composed to build the model for complex systems. Since CAS are usually very complex in a global view, but much simpler in an individual view, we believe it is a rather advantageous strategy to use process algebras to specify CAS in a bottom-up manner.

Classic process algebras such as CCS [Milner, 1989] and CSP [Hoare, 1985] allow the modeller to reason about the qualitative behaviour of the modelled system, such as whether or not there is a deadlock in the system or a particular system state can be reached. In order to investigate the quantitative behaviour of systems with different characteristics, classic process algebras are extended to many different versions.

An important extension of classic process algebras is the stochastic process algebras (SPAs) which associate a random duration that is governed by a negative exponential distribution, with the firing of each action. This allows the model to capture the randomness of systems in the real world. Moreover, based on the rules of the structural operational semantics (SOS rules) for SPAs in the style of [Plotkin, 2004], models specified by SPAs can be translated to a graph which is called the labelled transition system (LTS). In the LTS, the nodes are system states and the edges are the transitions between states. The LTS is further used to generate the underlying CTMC which can be numerically solved for performance evaluation of the modelled system. Among the well-known examples of SPAs are PEPA [Hillston, 1996], stochastic π-*calculus* [Priami, 1995], TIPP [Gotz et al., 1992], EMPA [Bernardo and Gorrieri, 1998], etc. As the underlying mathematical framework is CTMCs, SPA models also suffer from the state-space explosion problem. Indeed, because of the compositionality, very simple SPA models can have very large state spaces.

Recently, by shifting view of the system from individual components to populations, fluid approximation techniques for large CTMCs have been successfully applied to SPA models. For example, [Hillston, 2005] showed how to translate a PEPA model

to a system of ODEs describing the evolution of the mean population of processes by fluid-flow approximation in an informal way. Later, [Tribastone et al., 2012] formally justified the fluid-flow approximation using Kurtz's fluid limit theorem [Kurtz, 1981]. Furthermore, [Hayden and Bradley, 2010] showed how to derive ODEs describing higher moments of PEPA models. [Cardelli, 2008a, Cardelli, 2008b] also presented translations from some SPA models, namely Chemical Parametric Form (CPF), a subset of stochastic π-*calculus* and stochastic interacting processes, to systems of chemical reactions for mean-field ODE analysis. Mean-field ODE analysis is also supported by Bio-PEPA models [Ciocchetta and Hillston, 2009].

There have also been some SPAs which encompass some spatial modelling such as stochastic π-*calculus*, Bio-PEPA, stochastic Bio-Ambients [Brodo et al., 2007] and stochastic bigraphs [Krivine et al., 2008]. But in each of these space is abstractly represented and most focus on a containment relationship between locations. There are also SPAs in which space is explicitly captured. For example, PALPS (Process Algebra with Location for Population Systems) [Antonaki and Philippou, 2012] is a spatial process algebra specifically designed for building ecological models. In PALPS, each process has explicit location attributes in a two-dimensional space. Each location also has its specific attributes, for example the number of processes in the location. Processes can choose whether or not to perform an action depending on the state of its location or nearby locations. SpacePi [John et al., 2008a] which associates each process with a n-dimensional space attribute, is tailored for the modelling of molecular biological systems in which processes' communication is constrained by their distance. The attributed Pi calculus [John et al., 2008b] supports more general representation of space by defining attributes for processes, and attribute-based communication which is an important feature of CAS, is also supported. However, models specified by the attributed Pi calculus can only be analysed by discrete-event simulation, thus the language suffers from the lack of scalable analysis techniques to model large scale systems like CAS. CARMA [Bortolussi et al., 2015a], inspired by our modelling formalism and SCEL [Nicola et al., 2014], is a more general and powerful SPA particularly tailored for the modelling of CAS in which both explicit spatial representation and attribute-based communication are supported. However, currently, only the discrete semantics of CARMA has been introduced. Thus, the work in the remainder of this thesis can also be a guide for developing scalable analysis techniques for CARMA models.

Chapter 3

PALOMA: A Process Algebra For Located Markovian Agents

The Process Algebra for Located Markovian Agents (PALOMA) is a stochastic process algebra which is designed to capture models within the M^2MAM framework. Through its high-level specification, PALOMA provides an intuitive approach to constructing models within the M^2MAM framework by circumventing the rather cumbersome matrix specification of the models [Feng and Hillston, 2014]. The language has been extended in [Feng et al., 2016a] by supporting unicast communication between agents (which is not supported within the M^2MAM framework) in order to enhance the expressiveness of the language in capturing the synchronized interaction patterns between agents in CAS.

PALOMA is equipped with both individual-based and population-level semantics. The individual-based semantics provides the theoretical foundation for automatic derivation of executable models for agent-based simulation in which each agent is simulated as a stochastic process whose events are individually scheduled. However, with the population-level semantics, we are able to map PALOMA models into PCTMCs, for which the simulation can be undertaken at the population-level using Gillespie's algorithm [Gillespie, 1977]. Population-level simulation is more efficient compared with agent-based simulation when the model contains many symmetric agents (agents that are indistinguishable from each other) which is often the case for CAS.

In the remainder of this chapter, we will first introduce the syntax of PALOMA. Then, we will give an example model which captures a city bike-sharing system to illustrate the expressiveness of PALOMA in capturing collective adaptive behaviours. The formal semantics of PALOMA will be given afterwards. Lastly, we will also show

Symbol	Meaning
α	An action type
$!\alpha$	A broadcast message typed α
$!!\alpha$	A unicast message typed α
π	An action
S	An agent state
ℓ	A location
ℓ_s	The location of the sender agent of a message
ℓ_r	The location of the receiver agent of a message
$S(\ell)$	An agent in state S and location ℓ
P, Q	PALOMA components
Σ_w	The sum of the weights of all potential receivers of a unicast message
E	The ether element
Sys	A snapshot of the system in a PALOMA model
ξ	The numerical count matrix of agent types in a PALOMA model
$\mathbf{E}_{i,j}$	A numerical matrix of the same size as ξ, in which only the element in the ith row, jth column is 1, all other elements are 0.

Table 3.1: Important notations in Chapter 3

some results from both agent-based and population-level simulation of the city bike-sharing model. The important notations in this chapter are summarised in Table 3.1.

3.1 Syntax

PALOMA supports the construction of formal models of CAS in which agents are distributed over a discrete set of named locations, $\mathcal{L}$. Agents are parametrised by a *location*, denoted by ℓ, $\ell \in \mathcal{L}$. Each individual agent specifies a finite state machine, and the language is *conservative* in the sense that no agents are created or destroyed during the evolution of the model. There is a finite set of action types $\mathcal{A}$, and actions

may be undertaken spontaneously or may be induced by a message of the same type, sent by another agent in the system. All spontaneous actions are assumed to have a duration governed by an exponential distribution and characterised by a rate r. A model consists of a number of agents composed in parallel. There is no direct communication between agents, for example in the style of CSP [Hoare, 1985] or PEPA [Hillston, 1996], but communication between agents is achieved through broadcast and unicast message passing.

The language has the following grammar:

$$\begin{aligned} \pi &::= !(\alpha,r)@\mathbf{IR}\{\overrightarrow{\ell}\} \mid ?(\alpha,p)@\mathbf{Pr}\{v\} \mid !!(\alpha,r)@\mathbf{IR}\{\overrightarrow{\ell}\} \mid ??(\alpha,p)@\mathbf{Wt}\{v\} \mid (\alpha,r) \\ S(\ell) &::= \pi.S(\ell) \mid S(\ell)+S(\ell) \mid C \\ P &::= S(\ell) \mid P \parallel P \end{aligned}$$

where the first line defines *actions*, the second line defines *agents*, the last line produces *model components*. Specifically, PALOMA supports the following operators:

Prefix: $\pi.S(\ell)$ denotes an agent which performs an action π and behaves as $S(\ell)$ subsequently. π can be one of the following five classes:

- *Spontaneous action with broadcast message emission:* $!(\alpha,r)@\mathbf{IR}\{\overrightarrow{\ell}\}$ describes a spontaneous action α, $\alpha \in \mathcal{A}$ with rate r. During the occurrence of the action, a broadcast message $!\alpha$ is emitted. The *influence range* of the broadcast is defined by the location list $\overrightarrow{\ell}$, which gives the locations in which agents can *potentially* be influenced by this message. $\overrightarrow{\ell}$ can be defined both statically and dynamically. For example, $\overrightarrow{\ell} = [\ell_1, \ell_2, \ell_3]$ means that the influence range of the broadcast is locations ℓ_1, ℓ_2 and ℓ_3, whereas $\overrightarrow{\ell} = range(d)$ denotes that the influence range is a set of locations whose distance from the location of the sender agent is less than a specific threshold d. Some other frequently used definitions of influence range are $\overrightarrow{\ell} = local$ and $\overrightarrow{\ell} = all$, which represent that the influence range of the broadcast message is restricted to the location of the sender agent or consists of all the locations in the model, respectively. Note that the spontaneous action with broadcast message emission is non-blocking. The action is executed even if no agent is able to receive the message.

- *Spontaneous action with unicast message emission:* $!!(\alpha,r)@\mathbf{IR}\{\overrightarrow{\ell}\}$ also describes a spontaneous action of type α, rate r and influence range $\overrightarrow{\ell}$. The difference is that here the emitted message $!!\alpha$ is a unicast, meaning that at most one agent can receive the message. Moreover, the action is blocking, which means

that the action can only be executed if there are one or more potential receivers currently in the system.

- *Spontaneous action without message emission:* (α, r) denotes a spontaneous action α with a rate r. No message is sent out during the firing of this action. Thus it remains an individual action solely of this agent.

- *Induced action by a broadcast message:* $?(\alpha, p)@\mathbf{Pr}\{v\}$ describes an action α which will be triggered *immediately* after receiving and accepting a broadcast message $!\alpha$. Whether an agent can receive a broadcast message is decided by two factors. Firstly, the agent must be located within the influence range of the message; otherwise, the message will be ignored. Secondly, an implicit function $g(v)$ gives the probability that the message will be received by the agent given that it is within the influence range of the broadcast, where $v \in \mathbb{R}$ is a real number defined by the following grammar:

$$v \quad ::= \quad c \quad | \quad dist(\ell_1, \ell_2) \quad | \quad |S(\ell)| \quad | \quad v\,(op)\,v$$

in which c is a constant real number, $dist(\ell_1, \ell_2)$ is the distance between two locations, $|S(\ell)|$ is the number of agents in state S at location ℓ, $(op) \in \{+, -, \times, \div\}$ is a basic arithmetic operator. The function $g(v)$ is simply used to ensure the value domain of the reception probability is proper:

$$g(v) = \begin{cases} 1 & \text{if } v \geq 1 \\ v & \text{if } 0 < v < 1 \\ 0 & \text{if } v \leq 0 \end{cases}$$

The reception probability can be a static value, for example, $v = 0.5$ means that the agent has 50% chance of receiving the message. It can also be a dynamic value, for instance, $v = 1/|S(\ell)|$ denotes that the message reception probability is dependent on the current number of agents in state S at location ℓ in the system.

Once the message has been received, the agent decides whether to accept it. Here, a constant value $p \in [0, 1]$ encodes the probability that the agent will accept the message. This can be thought of as the agent choosing to respond to a spontaneous action of the given type with probability p. The definition of v and p support a rich set of possible interaction patterns between agents which will be illustrated by the example in the next section.

- *Induced action by a unicast message:* $??(\alpha, p)@\mathbf{Wt}\{v\}$ describes an action α which will be triggered immediately after receiving and accepting a unicast message $!!\alpha$. Similarly, a unicast can be potentially received only if the agent is within the influence range of the message. Furthermore, an implicit function

$$w(v) = \begin{cases} v & \text{if } v > 0 \\ 0 & \text{if } v \leq 0 \end{cases}$$

gives the *weight* of the agent to be the receiver of this unicast message, where v follows the same definition as previously. The weights are used to resolve between several potential receiver agents: suppose there are n agents denoted by $S_1(\ell_1), S_2(\ell_2), ..., S_n(\ell_n)$, which can potentially receive the unicast message, with weights $w(v_1), w(v_2), ..., w(v_n)$, respectively. Then, the probability that agent $S_1(\ell_1)$ receives the message is $w(v_1)/\Sigma_w$, where Σ_w denotes $\sum_{i=1}^{n} w(v_i)$, the sum of the associated weights of all potential receivers. Note that if there is no potential receiver, the message cannot be sent since the corresponding spontaneous action with unicast message emission is blocked. The value $p \in [0, 1]$ is a distinct probability deciding whether a received message is accepted or not. Note that if the selected agent does not accept the unicast message, the message is discarded; it cannot be passed to any other potential receiver agent.

Choice: Alternative behaviours are represented by the standard choice operator, $+$. For example, agent $\pi_1.S(\ell) + \pi_2.S'(\ell')$ can either perform an action π_1 and behave as $S(\ell)$ or perform an action π_2 and behave as $S'(\ell')$ depending on which action first completes. Specifically, a choice between spontaneous actions is resolved via the race policy, based on their corresponding rates. For instance, if π_1 and π_2 are both spontaneous actions with rate r_1 and r_2, then the probability to perform action π_1 is $r_1/(r_1 + r_2)$ whilst the probability to perform action π_2 is $r_2/(r_1 + r_2)$. When there is a choice between a spontaneous action and an induced action, the induced action will be fired immediately after a corresponding message is accepted, taking precedence over any enabled spontaneous actions. A choice between induced actions will be decided by which message is accepted first, and the corresponding induced action will be fired. Here, we assume that there is never a choice between two induced actions by the same message within a single agent. Moreover, we impose that there is at most one message currently in the system at any given time. Thus, there will never be a chance that two induced actions within a single agent are going to fire simultaneously.

Constant: $C = S(\ell)$ gives the constant C the behaviour of agent $S(\ell)$. This is how we assign behaviour to agents.

Parallel: $P \parallel Q$ denotes parallel composition of components. For convenience, we also introduce a notation $P[N]$ which indicates N copies of components P in parallel, i.e.,

$$\underbrace{P \parallel P \parallel \ldots \parallel P}_{N}$$

Unlike previous operators which are used to define behaviour for an agent type, this higher level operator is used to specify model components.

3.2 A Motivating Example

In this section, we show a motivating example of a city bike-sharing model to illustrate how the language may be used to describe CAS.

Specifically, we consider a model of a bike sharing service, where we assume a city with m parking stations, each one with its location $\ell_i \in \mathcal{L} = \{\ell_1, \ldots, \ell_m\}$, a number of available bikes N_{b_i}, and a number of available parking slots N_{s_i} (for $i = 1, \ldots, m$). We also assume that we have n users of the bike sharing service: at any time, each user is positioned in one location and can be in one of the two states *Pedestrian* and *Biker*. In each of those states, the user can move around the city (with speed depending on the state) according to preferences modelled by two probability transition matrices Q_b and Q_p of size $m \times m$ for the biker and the pedestrian state, respectively. The user becomes a Biker or a Pedestrian after borrowing and returning a bike from/to a station.

The model is shown in Figure 3.1, in which $Slot(\ell_i)$ and $Bike(\ell_i)$ denote an available slot and bike in the station at location ℓ_i, respectively. Both $Slot(\ell_i)$ and $Bike(\ell_i)$ are passive since they cannot make any spontaneous action. They can only be induced to make a *return* (a bike is returned to this station) or *borrow* (a bike is borrowed from this station) action by a unicast message, and when this happens they switch role. The weight for the agents to receive a *borrow* or *return* message is 1, which means that each bike or slot has equal probability to be taken.

$Station(\ell_i)$ denotes the parking station at location ℓ_i. A parking station performs both a $BikeAvailable_i$ and a $SlotAvailable_i$ self-jump spontaneous actions with broadcast message emission at the rate of γ. In reality, this can be regarded as updating the bike and slot availability of the station on the website or smart phone application. The influence range of the broadcast messages is defined by the function $range(d)$, which

$$
\begin{aligned}
Slot(\ell_i) &= ??(return,1)@\mathbf{Wt}\{1\}.Bike(\ell_i)\\
Bike(\ell_i) &= ??(borrow,1)@\mathbf{Wt}\{1\}.Slot(\ell_i)\\
\\
Station(\ell_i) &= !(SlotAvailable_i,\gamma)@\mathbf{IR}\{range(d)\}.Station(\ell_i)\\
&\quad +!(BikeAvailable_i,\gamma)@\mathbf{IR}\{range(d)\}.Station(\ell_i)\\
\\
Pedestrian(\ell_i) &= (seekb,r_{seekb}).SeekBike(\ell_i)+\sum_{j\neq i}(walk_{ij},Q_p(i,j)).Pedestrian(\ell_j)\\
SeekBike(\ell_i) &= \sum_{j=1}^{m}?(BikeAvailable_j,1)@\mathbf{Pr}\{v_1\}.Walk2Station_j(\ell_i)\\
&\quad \text{where: } v_1=\theta_0+\theta_1\frac{d-dist(\ell_i,\ell_j)}{d}+\theta_2\frac{|Bike(\ell_j)|}{|Bike(\ell_j)|+|Slot(\ell_j)|}\\
Walk2Station_j(\ell_i) &= (W2S_{ij},w2s_{ij}).BorrowBike(\ell_i)\\
BorrowBike(\ell_i) &= !!(borrow,o)@\mathbf{IR}\{local\}.Biker(\ell_i)\\
Biker(\ell_i) &= (seeks,r_{seeks}).SeekSlot(\ell_i)+\sum_{j\neq i}(ride_{ij},Q_b(i,j)).Biker(\ell_j)\\
SeekSlot(\ell_i) &= \sum_{j=1}^{m}?(SlotAvailable_j,1)@\mathbf{Pr}\{v_2\}.Ride2Station_j(\ell_i)\\
&\quad \text{where: } v_2=\theta_0+\theta_1\frac{d-dist(\ell_i,\ell_j)}{d}+\theta_2\frac{|Slot(\ell_j)|}{|Bike(\ell_j)|+|Slot(\ell_j)|}\\
Ride2Station_j(\ell_i) &= (R2S_{ij},r2s_{ij}).ReturnBike(\ell_i)\\
ReturnBike(\ell_i) &= !!(return,o)@\mathbf{IR}\{local\}.Pedestrian(\ell_i)\\
\\
\ldots &\parallel Pedestrian(\ell_i)[N_{pd_i}] \parallel Biker(\ell_i)[N_{br_i}] \parallel Slot(\ell_i)[N_{s_i}] \parallel Bike(\ell_i)[N_{b_i}] \parallel Station(\ell_i) \parallel \ldots
\end{aligned}
$$

Figure 3.1: The city bike sharing model

means that only agents in locations whose distance to the location of the sender station is less than d can potentially be influenced by this message.

$Pedestrian(\ell_i)$ denotes a user in *Pedestrian* state at location ℓ_i. She can travel from location ℓ_i to location ℓ_j at the rate of $Q_p(i,j)$ by doing a spontaneous action $walk_{ij}$ without message emission. She may also seek a bike at the rate of r_{seekb}, and goes into the *SeekBike* state.

The user in the *SeekBike* state at location ℓ_i can do a $BikeAvailable_j$ action induced by a broadcast message sent by a station agent in location ℓ_j and goes to the $Walk2Station_j(\ell_i)$ state, which represents the user walking from location ℓ_i to the bike station in location ℓ_j. The probability of receiving a bike available message from the station in location ℓ_j is defined by v_1, where

$$v_1 = \theta_0 + \theta_1 \frac{d - dist(\ell_i, \ell_j)}{d} + \theta_2 \frac{|Bike(\ell_j)|}{|Bike(\ell_j)| + |Slot(\ell_j)|}.$$

In reality, this can be interpreted as follows: the users will check the bike availability in nearby stations on the website or the smartphone application before deciding where to rent a bike, and they tend to rent a bike from a closer bike station with more available bikes, and θ_1, θ_2 are associated weights of those factors, θ_0 is the noise term.

The user in the $Walk2Station_j(\ell_i)$ state can do a spontaneous action $W2S_{ij}$ at the rate of $w2s_{ij}$, where $1/w2s_{ij}$ is the expected time to walk from ℓ_j to the bike station in location ℓ_i. Then, the borrow bike action *borrow* is fired at the rate of o. Meanwhile, a unicast message *borrow* is sent out, and the user becomes a *Biker*. A user agent in the *Biker* state can perform actions and become a *Pedestrian* again in a similar fashion.

Finally, the last line in the model gives the initial population of agents in the system, where N_{pd_i}, N_{br_i}, N_{s_i} and N_{b_i} are numbers indicating the initial count of pedestrians, bikers, available slots and bikes in location i or station i, respectively.

3.3 Individual-based Semantics

The individual-based semantics provides the basic rules for agent-based simulation of PALOMA models. Concretely, the individual-based semantics proceeds in sequences of alternating steps. This can be regarded as a semi-Markov process: the first step, corresponding to the spontaneous actions, determines a delay, activates an action and sends out a message (except spontaneous actions without message emission) to *occupy* the environment, whilst the second step is probabilistic and determines what the next state will be, as each possible induced action decides whether to fire or not. More

specifically, in order to make sure these two steps alternate correctly, we associate an *ether* element with the system, which provides the environment for *all* agents. The ether element has a distinguished empty state E_0.

Correspondingly, we define two transition relations $\longrightarrow_d$ and $\longrightarrow_{\mathcal{P}}$, which are the delay transition relation and the probabilistic transition relation, respectively.

3.3.1 The Delay Transition Relation

The rules for the delay transition relation ($\longrightarrow_d$) are given in Figure 3.2. As shown in rules DeBrA and DeUnA, a spontaneous action with message emission can only be initiated if the ether is currently empty, and no probabilistic transitions are enabled ($\not\rightarrow_{\mathcal{P}}$). The resulting local state records that the ether contains the message $!\alpha$ or $!!\alpha$, the location of sender agent ℓ_s, the influence range of the broadcast or unicast $\overrightarrow{\ell}$, and the snapshot of the system state Sys after the firing of the spontaneous action, which contains all the information that is needed to evaluate the reception probability of the message in an induced action. More specifically, we can think the snapshot Sys captures the environment of the agents and can modulate the interaction between agents by shaping communication probabilities. The continuation is subject to a probabilistic resolution. Note that for the spontaneous action with unicast message emission, it can be triggered only if there is at least one potential receiver agent in the system ($\exists??(\alpha,p)@\mathbf{Wt}\{v\}.S'(\ell'),\ell_r \in \overrightarrow{\ell}$, where ℓ_r denotes the location of the receiver agent). Any agent awaiting probabilistic resolution is denoted $S(\ell)^{\mathcal{P}}$. The rule of a spontaneous action without message emission, as defined in DeNoMsgA, is just a special case of spontaneous action with broadcast message emission. Specifically, the emitted broadcast message has an empty influence range $\emptyset$. In this case the message will propagate, without impacting any other agents, except to put them into the trivial probabilistic state.

If the ether contains a message of any type, then all agents will immediately witness the ongoing action, enter a probabilistic state and await resolution (rule DeBlock). This means all actions will be blocked until the current message has been fully disseminated and probabilistically resolved, which ensures that only one spontaneous action can be in progress at a time.

Choice behaves as we would anticipate (rule DeChoice). When the ether is empty, then one of the enabled spontaneous actions can be chosen to fire. However, when the ether is occupied by a message, then all actions must be blocked, and the agent will go

DeBrA

$$E_0, !(\alpha,r)@\mathbf{IR}\{\overrightarrow{\ell}\}.S(\ell) \xrightarrow{(\alpha,r)}_d [!\alpha,\ell_s,\overrightarrow{\ell},\mathrm{Sys}],S(\ell)^{\mathcal{P}} \quad (\not\rightarrow_{\mathcal{P}})$$

DeUnA

$$E_0, !!(\alpha,r)@\mathbf{IR}\{\overrightarrow{\ell}\}.S(\ell) \xrightarrow{(\alpha,r)}_d [!!\alpha,\ell_s,\overrightarrow{\ell},\mathrm{Sys}],S(\ell)^{\mathcal{P}} \quad \left(\not\rightarrow_{\mathcal{P}} \wedge \exists ??(\alpha,p)@\mathbf{Wt}\{v\}.S'(\ell'),\ell_r \in \overrightarrow{\ell}\right)$$

DeNoMsgA

$$E_0,(\alpha,r).S(\ell) \xrightarrow{(\alpha,r)}_d [!\alpha,\ell_s,\emptyset,\mathrm{Sys}],S(\ell)^{\mathcal{P}} \quad (\not\rightarrow_{\mathcal{P}})$$

DeBlock

$$[\hat{\alpha},\ell_s,\overrightarrow{\ell},\mathrm{Sys}],\pi.S(\ell) \xrightarrow{(\alpha,r)}_d [\hat{\alpha},\ell_s,\overrightarrow{\ell},\mathrm{Sys}],\left(\pi.S(\ell)\right)^{\mathcal{P}} \quad \left(\hat{\alpha} \in (!\alpha,!!\alpha)\right)$$

DeChoice

$$\frac{E_0,S_1(\ell) \xrightarrow{(\alpha,r)}_d E,S_1'(\ell')^{\mathcal{P}}}{E_0,S_1(\ell)+S_2(\ell) \xrightarrow{(\alpha,r)}_d E,S_1'(\ell')^{\mathcal{P}}} \qquad \frac{E_0,S_2(\ell) \xrightarrow{(\alpha,r)}_d E,S_2'(\ell')^{\mathcal{P}}}{E_0,S_1(\ell)+S_2(\ell) \xrightarrow{(\alpha,r)}_d E,S_2'(\ell')^{\mathcal{P}}}$$

$$\frac{E,S_1(\ell) \xrightarrow{(\alpha,r)}_d E,S_1(\ell)^{\mathcal{P}} \qquad E,S_2(\ell) \xrightarrow{(\alpha,r)}_d E,S_2(\ell)^{\mathcal{P}}}{E,S_1(\ell)+S_2(\ell) \xrightarrow{(a,r)}_d E,\left(S_1(\ell)+S_2(\ell)\right)^{\mathcal{P}}}$$

DeParallel

$$\frac{E_1,P \xrightarrow{(\alpha,r)}_d E',(P')^{\mathcal{P}} \qquad E_2,Q \xrightarrow{(\alpha,r)}_d E',(Q')^{\mathcal{P}}}{\left(E_1,P\right) \parallel \left(E_2,Q\right) \xrightarrow{(a,r)}_d E',\left(P' \parallel Q'\right)^{\mathcal{P}}}$$

DeConstant

$$\frac{E,S(\ell) \xrightarrow{(\alpha,r)}_d E',S'(\ell')}{E,C \xrightarrow{(\alpha,r)}_d E',S'(\ell')} \quad C = S(\ell)$$

Figure 3.2: The delay transition relation for PALOMA

into the probabilistic state. We assume that within a choice both elements are in the same location as they correspond to a single agent.

Parallel agents must agree on the single spontaneous action to take place, and consequently update the ether in the same way (rule DeParallel).

The rule DeConstant is straightforward. It just gives the constant C the delay behaviour of $S(\ell)$ if $C = S(\ell)$ is defined.

3.3.2 The Probabilistic Transition Relation

A spontaneous action is deemed to be complete when all agents have moved to a probabilistic state. In this case a probabilistic resolution must be made to determine the next state. This is defined by the probabilistic transition relation, which will clear the ether and create the opportunity for the next spontaneous action. More specifically, probabilistic resolutions are determined by a second transition relation $\longrightarrow_{\mathcal{P}}$, shown in Figure 3.3.

Specifically, the rule PrBr shows the probabilistic resolution of an induced action influenced by a matching broadcast message. There are two different resolution outcomes according to whether the message is actually received and accepted by the agent or not. Note that all the information that is needed to compute the reception probability $g(v)$ is provided within the ether element. In either case the ether is emptied when the probabilistic resolution is made. The probabilistic resolution of an induced action influenced by a matching unicast message follows a similar pattern (rule PrUn).

For other states in which no action can be induced, the probabilistic resolution is trivial. It will simply clear the ether and return the agent to an active state again (rule PrTr).

PrChoice gives the rules for probabilistic resolution of a choice between actions. Since we assume that there is never a choice between two induced actions of the same type within a single agent (see Page 27), at most one action can be potentially induced within an agent and the probabilistic resolution for the rest of the actions must be trivial. Therefore, if an action is induced within an agent, the agent will simply go to the corresponding subsequent state. Otherwise if no action is induced, the agent will clear the ether and the probabilistic resolution is completed.

Parallel agents undergo probabilistic resolution for a broadcast message independently and their probabilities are multiplied (rule PrBrParallel). For probabilistic resolution for a unicast message, at most one agent can actually receive the message. Thus,

PrBr

$$[!\alpha,\ell_s,\overrightarrow{\ell},\mathrm{Sys}],\left(?(\alpha,p)@\mathbf{Pr}\{v\}.S(\ell)\right)^{\mathcal{P}}\begin{cases}\xrightarrow{(\alpha,g(v)\times p)}_{\mathcal{P}} & E_0,S(\ell)\\ \xrightarrow{(\alpha,1-g(v)\times p)}_{\mathcal{P}} & E_0,?(\alpha,p)@\mathbf{Pr}\{v\}.S(\ell)\end{cases}\quad(\ell_r\in\overrightarrow{\ell})$$

PrUn

$$[!!\alpha,\ell_s,\overrightarrow{\ell},\mathrm{Sys}],\left(??(\alpha,p)@\mathbf{Wt}\{v\}.S(\ell)\right)^{\mathcal{P}}\begin{cases}\xrightarrow{(\alpha,\frac{w(v)}{\Sigma_w}\times p)}_{\mathcal{P}} & E_0,S(\ell)\\ \xrightarrow{(\alpha,1-\frac{w(v)}{\Sigma_w}\times p)}_{\mathcal{P}} & E_0,??(\alpha,p)@\mathbf{Wt}\{v\}.S(\ell)\end{cases}\quad(\ell_r\in\overrightarrow{\ell})$$

PrTr

$$[\hat{\alpha},\ell_s,\overrightarrow{\ell},\mathrm{Sys}],\left(\pi.S(\ell)\right)^{\mathcal{P}}\xrightarrow{(\alpha,1)}_{\mathcal{P}}E_0,\pi.S(\ell)\quad\left(\hat{\alpha}=\begin{cases}!\alpha\wedge\pi\neq?(\alpha,p)@\mathbf{Pr}\{v\}\\ !!\alpha\wedge\pi\neq??(\alpha,p)@\mathbf{Wt}\{v\}\end{cases}\vee\ell_r\notin\overrightarrow{\ell}\right)$$

PrChoice

$$\frac{E,S_1(\ell)^{\mathcal{P}}\xrightarrow{(\alpha,p)}_{\mathcal{P}}E_0,S_1'(\ell')}{E,\left(S_1(\ell)+S_2(\ell)\right)^{\mathcal{P}}\xrightarrow{(\alpha,p)}_{\mathcal{P}}E_0,S_1'(\ell')}\qquad\frac{E,S_2(\ell)^{\mathcal{P}}\xrightarrow{(\alpha,p)}_{\mathcal{P}}E_0,S_2'(\ell')}{E,\left(S_1(\ell)+S_2(\ell)\right)^{\mathcal{P}}\xrightarrow{(\alpha,p)}_{\mathcal{P}}E_0,S_2'(\ell')}$$

$$\frac{E,S_1(\ell)^{\mathcal{P}}\xrightarrow{(\alpha,p)}_{\mathcal{P}}E_0,S_1(\ell)\qquad E,S_2(\ell)^{\mathcal{P}}\xrightarrow{(\alpha,1)}_{\mathcal{P}}E_0,S_2(\ell)}{E,\left(S_1(\ell)+S_2(\ell)\right)^{\mathcal{P}}\xrightarrow{(\alpha,p)}_{\mathcal{P}}E_0,S_1(\ell)+S_2(\ell)}$$

$$\frac{E,S_1(\ell)^{\mathcal{P}}\xrightarrow{(\alpha,1)}_{\mathcal{P}}E_0,S_1(\ell)\qquad E,S_2(\ell)^{\mathcal{P}}\xrightarrow{(\alpha,p)}_{\mathcal{P}}E_0,S_2(\ell)}{E,\left(S_1(\ell)+S_2(\ell)\right)^{\mathcal{P}}\xrightarrow{(\alpha,p)}_{\mathcal{P}}E_0,S_1(\ell)+S_2(\ell)}$$

PrBrParallel

$$\frac{E,P^{\mathcal{P}}\xrightarrow{(\alpha,p)}_{\mathcal{P}}E_0,P'\qquad E,Q^{\mathcal{P}}\xrightarrow{(\alpha,q)}_{\mathcal{P}}E_0,Q'}{E,\left(P\parallel Q\right)^{\mathcal{P}}\xrightarrow{(\alpha,p\times q)}_{\mathcal{P}}\left(E_0,P'\right)\parallel\left(E_0,Q'\right)}$$

PrUnParallel

$$\frac{E,P^{\mathcal{P}}\xrightarrow{(\alpha,p)}_{\mathcal{P}}E_0,P'\qquad E,Q^{\mathcal{P}}\xrightarrow{(\alpha,1)}_{\mathcal{P}}E_0,Q}{E,\left(P\parallel Q\right)^{\mathcal{P}}\xrightarrow{(\alpha,p)}_{\mathcal{P}}\left(E_0,P'\right)\parallel\left(E_0,Q\right)}$$

$$\frac{E,P^{\mathcal{P}}\xrightarrow{(\alpha,1)}_{\mathcal{P}}E_0,P\qquad E,Q^{\mathcal{P}}\xrightarrow{(\alpha,q)}_{\mathcal{P}}E_0,Q'}{E,\left(P\parallel Q\right)^{\mathcal{P}}\xrightarrow{(\alpha,q)}_{\mathcal{P}}\left(E_0,P\right)\parallel\left(E_0,Q'\right)}$$

PrConstant

$$\frac{E,S(\ell)\xrightarrow{(\alpha,r)}_{\mathcal{P}}E_0,S'(\ell')}{E,C\xrightarrow{(\alpha,r)}_{\mathcal{P}}E_0,S'(\ell')}\quad C=S(\ell)$$

Figure 3.3: The probabilistic transition relation for PALOMA

if a non-trivial probabilistic resolution happens within a component, the other components in parallel must stay in their current states after probabilistic resolution (rule PrUnParallel).

Like the constant rule in the delay transition relation, the rule PrConstant just gives C the probabilistic behaviour of $S(\ell)$ if $C = S(\ell)$ is defined.

3.3.3 CTMC

With the basis of the above individual-based semantic rules, a PALOMA model can be represented as a labelled transition system $(\mathcal{S}, \mathcal{A}, \{\xrightarrow{(\alpha,r)} | (\alpha, r) \in \mathcal{A}\})$, where $\mathcal{S}$ is the set of system states, $\mathcal{A}$ is the set of actions, and the transition relation $\xrightarrow{(\alpha,r)} \in \mathcal{S} \times \mathcal{S}$ is given by the combination of the rules in Figure 3.2 and 3.3 as follows:

$$\mathcal{S}_0 \xrightarrow{(\alpha,r)}_d \mathcal{S}_1^{\mathcal{P}}, \quad \mathcal{S}_1^{\mathcal{P}} \xrightarrow{(\alpha,p)}_{\mathcal{P}} \mathcal{S}_2 \quad \Longrightarrow \quad \mathcal{S}_0 \xrightarrow{(\alpha, r \times p)} \mathcal{S}_2$$

More specifically, using the above combination rule, we eliminate all probabilistic states because no time elapses in those states whereas states in a CTMC must have an exponentially distributed sojourn time.

The generated labelled transition system can be further mapped onto a CTMC. However, since the state space of the CTMC is often extremely large, in most cases, it is only possible to explore the state space on-the-fly through agent-based simulation in order to analyse the CTMC.

3.4 Population-level Semantics

The population-level semantics provides the theoretical foundation for population-level stochastic simulation of PALOMA models. Specifically, as agents in the same state and location are indistinguishable in a PALOMA model, we aggregate those indistinguishable agents through a counting abstraction. Furthermore, with the semantics which lifts the transitions within the model to the population level, we map the PALOMA model to a PCTMC whose the state space is much smaller than the original individual-based CTMC. As a result, the computational cost of simulating the model can be reduced compared with the agent-based simulation.

Concretely, in order to define the population-level semantics for PALOMA, for an arbitrary model, we first construct a state vector $\mathbf{S}$ whose size is $|\mathbf{S}|$, where each element S_i denotes a specific agent state that appears in the model. Then, a location vector

$\mathbf{L}$ whose size is $|\mathbf{L}|$ is also constructed, in which each element ℓ_i denotes a specific location that appears in the model (Both $\mathbf{S}$ and $\mathbf{L}$ can be initialized by simply traversing the model definition and listing all the distinct states and locations that appear on the left hand side or the right hand side of a defining equation in the specification of the model). Furthermore, we use a $|\mathbf{S}| \times |\mathbf{L}|$ numerical matrix $\xi(t)$ to represent the current count of agents in all possible states and locations. Specifically, the element $\xi_{i,j}(t)$ in the ith row and jth column of the matrix denotes the current number of agents in state S_i at location ℓ_j at time t (we set $\xi_{i,j}(t) = -1$ if $S_i(\ell_j)$ does not exist). For convenience, we will use ξ as short for $\xi(t)$, $\xi_{i,j}$ as short for $\xi_{i,j}(t)$ hereafter.

Furthermore, we define a structural congruence in Figure 3.4 which allows us to define the population-level semantics in a more compact and straightforward way. Moreover, we also define a function:

$$\mathrm{pf}(S_i(\ell_j)) \quad = \quad \{\pi_n.S_{i_n}(\ell_{j_n}) \mid S_i(\ell_j) = \sum_{n \in N} \pi_n.S_{i_n}(\ell_{j_n})\}$$

which gives the set of prefixes of a specific agent type (we say two agents are of the same type if and only if they are in the same state and location).

Now, we formally define the population-level structured operational semantics with rules for the derivation of population-level transitions for PALOMA in Figure 3.5. Specifically, the rule PbNoMsg infers a population-level transition from a spontaneous action with no message emission of a single agent with rate r. The idea is that if there are $\xi_{i,j}$ copies of an agent type $S_i(\ell_j)$ at any time instant, then the total rate at which a spontaneous action with no message emission in the premise fires is $r \times \xi_{i,j}$. After firing the transition, the number of agents $S_i(\ell_j)$ in the system will decrease by one whereas the number of agents $S'_i(\ell'_j)$ will increase by one (we refer the meaning of $\mathbf{E}_{i,j}$ to Table 3.1).

$$\begin{aligned}
S(\ell)[n] &\equiv \underbrace{S(\ell) \parallel \ldots \parallel S(\ell)}_{\text{n copies}} \\
S(\ell) &\equiv S(\ell)[1] \\
S(\ell)[n_1] \parallel S(\ell)[n_2] &\equiv S(\ell)[n_1 + n_2] \\
S(\ell) \parallel S'(\ell') &\equiv S'(\ell') \parallel S(\ell) \\
\Big(S(\ell) \parallel S'(\ell')\Big) \parallel S''(\ell'') &\equiv S(\ell) \parallel \Big(S'(\ell') \parallel S''(\ell'')\Big)
\end{aligned}$$

Figure 3.4: Structural congruence in PALOMA

The rule PbBrCombo infers a set of population-level transitions from a spontaneous action with broadcast message emission coupled with all its potential receivers. Specifically, suppose there are $\xi_{i,j}$ copies of $S_i(\ell_j)$ which can do a spontaneous action with broadcast message emission at rate r at any time instant, then the total emission rate of the broadcast message $!\alpha$ from those agents is $r \times \xi_{i,j}$. For an agent in state S_m and location ℓ_n which is within the influence range of the broadcast message ($\ell_n \in \overrightarrow{\ell}$), assume the probabilities of receiving and accepting the message are $g(v)$ and p, respectively. Suppose a broadcast message $!\alpha$ is sent out, as agents choose to receive and accept the broadcast message independently, then at the population level, the number of agents in state S_m and location ℓ_n who actually fire the corresponding induced action caused by the broadcast message is a random variable following a Binomial distribution, $K \sim Binomial(\xi_{m,n}, g(v) \times p)$. Hence, the probability that k $\big(k \in \Omega(K)\big)$ copies of $S_m(\ell_n)$ fire their corresponding induced action is $\Pr(k; \xi_{m,n}, g(v) \times p)$, where

$$\Pr(k; \xi_{m,n}, g(v) \times p) = \binom{\xi_{m,n}}{k} (g(v) \times p)^k (1 - g(v) \times p)^{\xi_{m,n} - k} \quad k \in (0, 1, \ldots, \xi_{m,n})$$

Moreover, there can be agents of multiple types in different states or locations which have actions induced by the same broadcast message. Again, as agents choose to receive and respond to the message independently, the probabilities of their outcomes can be multiplied to obtain the combined probability. Therefore, for agents of a type $S_i(\ell_j)$ which can do a spontaneous action with broadcast message emission at rate r at any time instant, suppose there are Z different agent types, denoted as $S_{m_z}(\ell_{n_z})$ $\big(z \in (1, 2, \ldots, Z)\big)$, which have an induced action coupled with the message, we can infer $\prod_z (\xi_{m_z,n_z} + 1)$ population-level transitions from the spontaneous action with broadcast message emission coupled with its all potential receivers. Moreover, for a specific transition in which for $z \in (1, 2, \ldots, Z)$, there are k_z copies of agents $S_{m_z}(\ell_{n_z})$ who fire the corresponding induced actions, its rate is $r \times \xi_{i,j} \times \prod_z \Pr\big(k_z; \xi_{m_z,n_z}, p_z \times g(v_z)\big)$. After firing the transition, one copy of $S_i(\ell_j)$ goes to $S_i'(\ell_j')$, for $z \in (1, 2, \ldots, Z)$, k_z copies of $S_{m_z}(\ell_{n_z})$ go to $S_{m_z'}(\ell_{n_z'})$.

The rule PbUnPair infers a population-level transition from an induced action coupled with a spontaneous action with unicast message emission. There is no explanation since the inference is rather straightforward.

3.4.1 PCTMC

With the population-level semantics, any PALOMA model can be mapped to a PCTMC represented as a tuple $\mathcal{P} = (\mathbf{X}, \mathcal{T}, \mathbf{X}_0)$, where:

PbNoMsg

$$\frac{(\alpha,r).S_{i'}(\ell_{j'}) \in \mathrm{pf}(S_i(\ell_j))}{\xi \xrightarrow{(\alpha,r\times\xi_{i,j})}_* \xi - \mathbf{E}_{i,j} + \mathbf{E}_{i',j'}}$$

PbBrCombo

$$\frac{!(\alpha,r)@\mathbf{IR}\{\overrightarrow{\ell}\}.S_{i'}(\ell_{j'}) \in \mathrm{pf}(S_i(\ell_j)) \qquad \forall z\ ?(\alpha,p_z)@\mathbf{Pr}\{v_z\}.S_{m'_z}(\ell_{n'_z}) \in \mathrm{pf}(S_{m_z}(\ell_{n_z})) \quad \ell_{n_z} \in \overrightarrow{\ell}}{\xi \xrightarrow{\left(\alpha,\, r\times\xi_{i,j}\times\prod_z \Pr\left(k_z;\xi_{m_z,n_z},\, p_z\times g(v_z)\right)\right)}_* \xi - \mathbf{E}_{i,j} + \mathbf{E}_{i',j'} + \sum_z k_z(\mathbf{E}_{m'_z,n'_z} - \mathbf{E}_{m_z,n_z}) \quad \forall k_z \in (0,1\ldots\xi_{m_z,n_z})}$$

PbUnPair

$$\frac{!!(\alpha,r)@\mathbf{IR}\{\overrightarrow{\ell}\}.S_{i'}(\ell_{j'}) \in \mathrm{pf}(S_i(\ell_j)) \quad ??(\alpha,p)@\mathbf{Wt}\{v\}.S_{m'}(\ell_{n'}) \in \mathrm{pf}(S_m(\ell_n)) \quad \ell_n \in \overrightarrow{\ell}}{\xi \xrightarrow{(\alpha,r\times\xi_{i,j}\times p\times\frac{w(v)}{\Sigma_w}\times\xi_{m,n})}_* \xi - \mathbf{E}_{i,j} + \mathbf{E}_{i',j'} - \mathbf{E}_{m,n} + \mathbf{E}_{m',n'}}$$

Figure 3.5: The population-level structured operational semantics of PALOMA

- $\mathbf{X} = (x_1, ..., x_n)$ maps all the non-negative elements in the population numerical matrix ξ to a vector format, where each vector element x_i is the count variable of a specific agent type. Note that we use $\mathbf{X}$ as short for $\mathbf{X}(t)$ to represent the current state of the model at a time instant t.

- $\mathcal{T} = \{\tau_1, ..., \tau_m\}$ is the set of population-level transitions inferred by the rules in Figure 3.5, of the form $\tau = (r_\tau(\mathbf{X}), \mathbf{d}_\tau)$ in which the the action type α is omitted,

  1. $r_\tau(\mathbf{X}) \in \mathbb{R} \geq 0$ is the rate function of transition τ. Specifically,

$$r_\tau(\mathbf{X}) = \begin{cases} r \times x_i & \text{if } \tau \text{ is inferred by rule PbNoMsg} \\ r \times x_i \times \prod_z \Pr\left(k_z; x_z,\, p_z \times g(v_z)\right) & \text{if } \tau \text{ is inferred by rule PbBrCombo} \\ r \times x_i \times p \times \frac{w(v)}{\Sigma_w} \times x_j & \text{if } \tau \text{ is inferred by rule PbUnPair} \end{cases}$$

  2. $\mathbf{d}_\tau \in \mathbb{Z}^n$ is the update vector which gives the net change for each element of $\mathbf{X}$ caused by transition τ. Intuitively, following the mapping rule from ξ to $\mathbf{X}$, $\mathbf{d}_\tau$ also maps the associated update matrix (e.g., the update matrix is $-\mathbf{E}_{i,j} + \mathbf{E}_{i',j'}$ if τ is inferred by rule PbNoMsg) into a vector format.

- $\mathbf{X}_0$ is the initial state of the model.

	Scenario I	Scenario II	Scenario III
m	8	8	8
$N_{pd_i}\ \big(i \in (1,2,\ldots,m)\big)$	10	20	30
$N_{br_i}\ \big(i \in (1,2,\ldots,m)\big)$	10	20	30
$N_{s_i}\ \big(i \in (1,2,\ldots,m)\big)$	5	10	15
$N_{b_i}\ \big(i \in (1,2,\ldots,m)\big)$	5	10	15
Stop time of a simulation run	100	100	100
Number of simulation runs	10,000	10,000	10,000

Table 3.2: The bike-sharing model simulation configuration (unit of time in simulation: minute)

	Scenario I	Scenario II	Scenario III
agent-based simulation	4.3hrs	5.4hrs	7.9hrs
population-level simulation	3.3hrs	3.5hrs	3.7hrs

Table 3.3: Simulation time cost of 10000 runs of the bike-sharing model

With the tuple $\mathcal{P}$, any PALOMA model can be simulated at the population-level using Gillespie's algorithm as described in Algorithm 2.1.

3.5 Simulation

In this section, we show some experiments on the city bike-sharing example using both agent-based simulation and population-level simulation. The goal is twofold. First, we validate the equivalence of agent-based simulation and population-level simulation in capturing the system dynamics of a PALOMA model. Second, we compare the efficiency of the two level simulations.

Specifically, we consider three simulation scenarios which differ in their size of agent populations. Table 3.2 gives the simulation configuration of the three scenarios (only the values of key parameters are listed, the values of other parameters are kept the same in all the scenarios). Figure 3.6 gives the trajectories of the mean number of available bikes in stations over 10,000 simulation runs for each scenario. It can be seen that the trajectories from agent-based simulation overlap with the population-

level simulation. The simulation time cost is given in Table 3.3 in which we can see that the simulation cost of agent-based simulation grows much faster than population-level simulation as the population size increases, which confirms that the population-level simulation is more efficient in case of large population sizes.

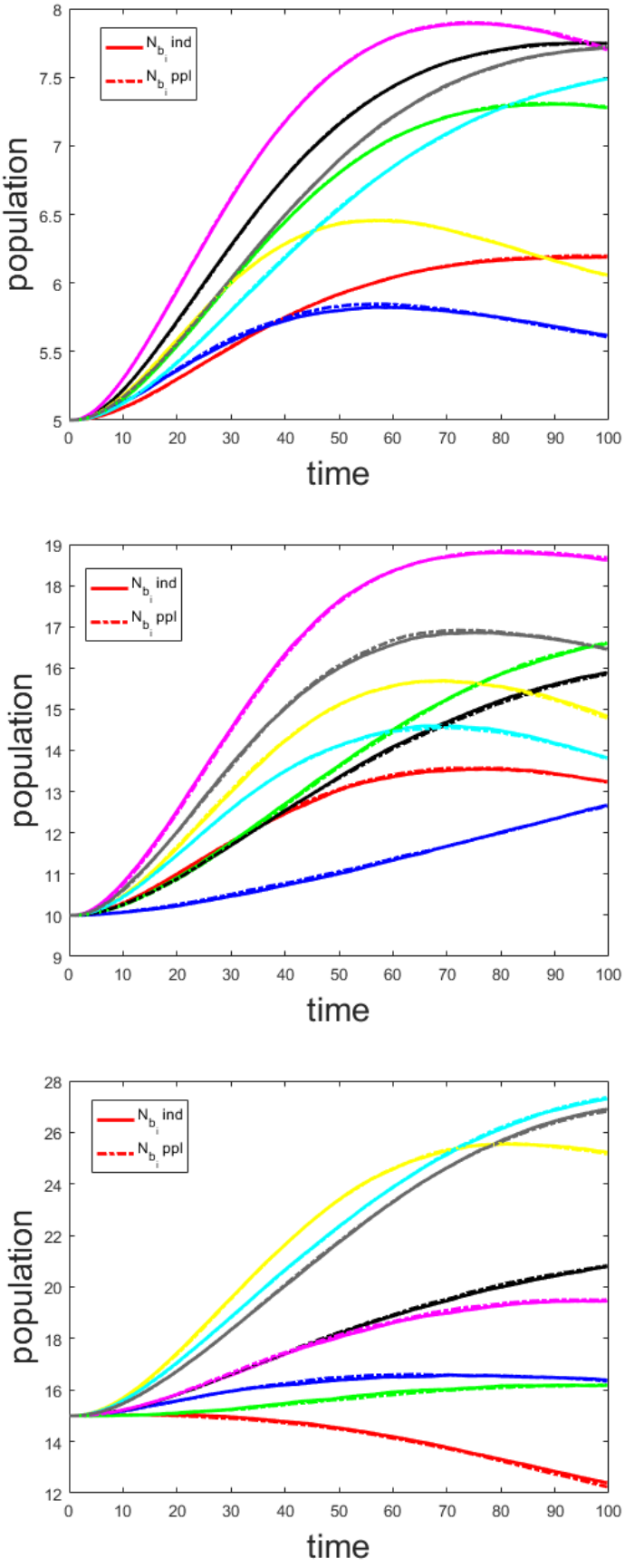

Figure 3.6: The trajectories of the average number of available bikes in the stations in the agent-based and population-level simulation of the bike-sharing model each with 10000 runs

Chapter 4

Automatic Moment Closure Approximation of PALOMA Models

In the previous chapter, we showed that by using a counting abstraction and simulating a PALOMA model at the population level, the computational cost of analysing the model can be reduced compared with agent-based simulation. However, since CAS usually consist of a massive number of entities, stochastic simulation of the associated PALOMA model can still be computationally too expensive (the simulation cost of the bike-sharing example is already very expensive, not to mention CAS with thousands of entities). Moreover, deriving performance measures from stochastic simulation often requires us to simulate the model a large number of times, and then obtain the measures of interest such as mean, variance and covariance of populations from the trajectories of those simulation runs. This means that analysing large-scale CAS may become extremely inefficient or even impractical.

In this chapter, we propose an approach which makes an approximation of the system as a set of ordinary differential equations (ODEs). Unlike earlier fluid approximation techniques of process algebras such as PEPA [Hillston, 2005, Tribastone et al., 2012], our approach is not limited to the expectation or first moment characterisation of system behaviour. Importantly, our moment-closure-based approach also incorporates higher order moments supporting the analysis of the compliance of a system to service level agreements and other performance requirements. Specifically, we will describe the evolution of the moments (mean, variance, covariance, skewness, kurtosis, etc.) of population variables in an arbitrary PALOMA model by a set of coupled ODEs. The obtained ODEs are not, in general, amenable to analytical solution but can nevertheless be solved efficiently by numerical simulation. What's

more, the structure of the set of ODEs is independent of the number of agents in the model, making the approach scalable even in the face of very large populations.

When representing a PALOMA model as moment ODEs, some problems can also arise. First of all, the dynamics of lower-order moments often depend on higher-order moments. Thus, an infinite number of ODEs are required to describe the system. In order to deal with this problem, we propose an automatic moment-closure approach to truncate the system at a certain order of moment. Secondly, the number of ODEs characterising the evolution of the moments, whilst independent of the size of the populations of agents, does depend on the number of locations and local state space of each agent type. In some circumstances this can lead to a prohibitive number of ODEs, or to a slow-down in analysis. Thus we also propose a model reduction technique which generates a reduced set of ODEs on the basis of a formally defined neighbourhood relation that is defined at the level of PCTMC description and can be automatically applied. We demonstrate through three examples that this can significantly improve the efficiency and scalability of moment-closure analysis whilst still retaining high accuracy. All the techniques reported in this chapter are implemented in a tool which is freely available for download[1].

Part of the work in this chapter has been published in a paper in Journal ACM Transactions on Modelling and Computer Simulation [Feng et al., 2016a].

4.1 The Derivation of Moment ODEs

We have mentioned that the evolution of the moments of the underlying population-level stochastic process of an arbitrary PCTMC model can be approximated by the following system of ODEs [Engblom, 2006]:

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbb{E}[M(\mathbf{X})] = \sum_{\tau\in\mathcal{T}} \mathbb{E}[(M(\mathbf{X}+\mathbf{d}_\tau) - M(\mathbf{X}))r_\tau(\mathbf{X})] \tag{4.1}$$

where $M(\mathbf{X})$ denotes the moment to be calculated, $\mathbf{d}_\tau$ and $r_\tau(\mathbf{X})$ represent the update vector and rate of a population-level transition τ, respectively. For instance, if we substitute $M(\mathbf{X})$ with x_i, ${x_i}^2$ and $x_i x_j$, we get the following ODEs to describe the first moment, second moment and second-order joint moment respectively, of population

[1] https://github.com/cfeng783/paloma/wiki#the-paloma-eclipse-plug-in

variables in an arbitrary PALOMA model:

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}t}\mathbb{E}[x_i] &= \sum_{\tau\in\mathcal{T}} \mathbb{E}[(x_i + d_\tau^i - x_i) r_\tau(\mathbf{X})] = \sum_{\tau\in\mathcal{T}} \mathbb{E}[d_\tau^i \cdot r_\tau(\mathbf{X})] \\
\frac{\mathrm{d}}{\mathrm{d}t}\mathbb{E}[{x_i}^2] &= \sum_{\tau\in\mathcal{T}} \mathbb{E}[((x_i + d_\tau^i)^2 - {x_i}^2) r_\tau(\mathbf{X})] \\
&= 2\sum_{\tau\in\mathcal{T}} \mathbb{E}[d_\tau^i \cdot x_i \cdot r_\tau(\mathbf{X})] + \sum_{\tau\in\mathcal{T}} \mathbb{E}[{d_\tau^i}^2 \cdot r_\tau(\mathbf{X})] \\
\frac{\mathrm{d}}{\mathrm{d}t}\mathbb{E}[x_i x_j] &= \sum_{\tau\in\mathcal{T}} \mathbb{E}[((x_i + d_\tau^i)(x_j + d_\tau^j) - x_i x_j) r_\tau(\mathbf{X})] \\
&= \sum_{\tau\in\mathcal{T}} \mathbb{E}[d_\tau^i \cdot x_j \cdot r_\tau(\mathbf{X})] + \sum_{\tau\in\mathcal{T}} \mathbb{E}[d_\tau^j \cdot x_i \cdot r_\tau(\mathbf{X})] + \sum_{\tau\in\mathcal{T}} \mathbb{E}[d_\tau^i \cdot d_\tau^j \cdot r_\tau(\mathbf{X})]
\end{aligned}$$

where d_τ^i is the ith element in $\mathbf{d}_\tau$ representing the update on the population variable x_i caused by transition τ. Note that if we derive the moment ODEs directly from the PCTMC, we may get some *Binomial* probability mass functions in $r_\tau(\mathbf{X})$ from transitions inferred by rule PbBrCombo. However, since we treat the populations of agents as continuous variables in moment-closure approximation, there will be no meaning for such functions in moment ODEs. Hence, in order to avoid these *Binomial* probability mass functions, we first do a transition combination for the PCTMC before deriving the moment ODEs. Specifically, consider the following set of population-level transitions inferred by the rule PbBrCombo:

$$\begin{aligned}
x_i, x_j, x_m, x_n &\xrightarrow{r\times x_i\times \binom{x_m}{0}(p\times g(v))^0(1-p\times g(v))^{x_m}} 0 \quad x_i-1, x_j+1, x_m-0, x_n+0 \\
&\vdots \\
x_i, x_j, x_m, x_n &\xrightarrow{r\times x_i\times \binom{x_m}{k}(p\times g(v))^k(1-p\times g(v))^{x_m-k}} k \quad x_i-1, x_j+1, x_m-k, x_n+k \\
&\vdots \\
x_i, x_j, x_m, x_n &\xrightarrow{r\times x_i\times \binom{x_m}{x_m}(p\times g(v))^{x_m}(1-p\times g(v))^0} x_m \quad x_i-1, x_j+1, x_m-x_m, x_n+x_m
\end{aligned}$$

It is possible to combine the above set of transitions for each possible update into a single population-level transition:

$$x_i, x_j, x_m, x_n \xrightarrow{r\times x_i} x_i-1, x_j+1, x_m - p\times g(v)\times x_m, x_n + p\times g(v)\times x_m$$

because

$$p\times g(v)\times x_m = \sum_{k=0,\ldots,x_m} k \times \binom{x_m}{k} (p\times g(v))^k (1-p\times g(v))^{x_m-k}.$$

Intuitively, this means that we get a combined population-level transition for a set of population-level transitions inferred from PbBrCombo where the rate of the combined

transition is the rate of the spontaneous action with broadcast message emission, the update vector of the combined transition is the *expected number* of agents to actually change their state. Equivalently, we can think that we rewrite the rule PbBrCombo as follows:

$$\frac{!(\alpha,r)@\mathbf{IR}\{\overrightarrow{\ell}\}.S_{i'}(\ell_{j'}) \in \mathrm{pf}(S_i(\ell_j)) \qquad \forall z\ ?(\alpha,p_z)@\mathbf{Pr}\{v_z\}.S_{m'_z}(\ell_{n'_z}) \in \mathrm{pf}(S_{m_z}(\ell_{n_z})) \quad \ell_{n_z} \in \overrightarrow{\ell}}{\xi \xrightarrow{(\alpha,\, r\times\xi_{i,j})}_{*} \xi - \mathbf{E}_{i,j} + \mathbf{E}_{i',j'} + \sum_z p_z \times g(v_z) \times \xi_{m_z,n_z} \times (\mathbf{E}_{m'_z,n'_z} - \mathbf{E}_{m_z,n_z})}$$

where $\forall z\ ?(\alpha,p_z)@\mathbf{Pr}\{v_z\}.S_{m'_z}(\ell_{n'_z}) \in \mathrm{pf}(S_{m_z}(\ell_{n_z}))$, $\ell_{n_z} \in \overrightarrow{\ell}$ traverses all agent types that are potential receivers of the broadcast message α, $\sum_z p_z \times g(v_z) \times \xi_{m_z,n_z} \times (\mathbf{E}_{m'_z,n'_z} - \mathbf{E}_{m_z,n_z})$ is the expected updates of the populations of those agent types after the spreading of the broadcast message.

After transition combination, we are able to circumvent the intractable *Binomial* probability mass functions, and a set of moment ODEs can be derived for fluid moment-closure approximation for an arbitrary PALOMA model.

4.2 Moment ODE Reduction

Describing the evolution of expected population-level dynamics by moment ODEs can dramatically improve our ability to analyse large scale CAS. However, due to the spatially distributed nature of CAS, it is likely that there will be a large number of population variables in the derived PCTMC model. Thus when higher order moments are required, a problem we call *ODE explosion* (so many coupled ODEs that traditional machines do not have enough memory and computational power to numerically simulate them) may emerge. Specifically, consider a PCTMC derived by a PALOMA model in which there are n elements in the population vector $\mathbf{X}$. Suppose we want to approximate the second-order moments of the population variables, then there will be n ODEs to describe the evolution of all $\mathbb{E}[{x_i}^2]$, and $(n^2-n)/2$ ODEs for all $\mathbb{E}[x_i x_j]$. Clearly, the problem of ODE explosion is mostly caused by the number of ODEs for joint moments, as their number grows exponentially as the order of moments increases. Therefore, in order to deal with the ODE explosion problem, in this section, we introduce an algorithm for the reduction of ODEs to describe the evolution of joint moments. Specifically, our algorithm is based on the neighbourhood relation between the population variables in the PCTMC. The definition of the neighbourhood relation will be given in the next subsection.

4.2.1 Neighbourhood Relation

Here, we introduce the neighbourhood relation between population variables in an arbitrary PCTMC which will be the basis of moment ODE reduction.

Concretely, we say two population variables x_i, x_j are one-hop neighbours if one of them can directly influence the evolution of the other. Formally, we define:

$$(x_i, x_j) \in \mathcal{R}^{(1)} \iff \exists \tau, \quad (d_\tau^i \neq 0 \wedge \delta_\tau^j = 1) \vee (d_\tau^j \neq 0 \wedge \delta_\tau^i = 1)$$

where δ_τ^j is an indicator equal to 1 if and only if x_j is updated after transition τ ($d_\tau^j \neq 0$) or x_j appears in the rate function ($r_\tau(\mathbf{X})$) of τ. Intuitively, this means that there exists a transition, in which one of the two population variables is updated, and the other is also involved. Moreover, we can infer two-hop neighbours by:

$$\exists k \notin \{i, j\} \quad (x_i, x_j) \notin \mathcal{R}^{(1)} \wedge (x_i, x_k) \in \mathcal{R}^{(1)} \wedge (x_k, x_j) \in \mathcal{R}^{(1)} \Rightarrow (x_i, x_j) \in \mathcal{R}^{(2)}.$$

More generally, $\mathcal{R}^{(n)}$ is the smallest relation that satisfies

$$\exists k \notin \{i, j\} \quad (x_i, x_j) \notin \mathcal{R}^{(1)}, \ldots, \mathcal{R}^{(n-1)} \wedge (x_i, x_k) \in \mathcal{R}^{(1)} \wedge (x_k, x_j) \in \mathcal{R}^{(n-1)} \Rightarrow (x_i, x_j) \in \mathcal{R}^{(n)}.$$

In general, with a higher-hop neighbourhood relation, the evolution of a population variable will be less likely to influence the other. Thus, the neighbourhood relation gives a coarse approximation of the dependence between the population variables in a PCTMC.

Based on the neighbourhood relation between the population variables, we further propose the following assertion:

Assertion 1. *Let x_i and x_j be two population variables in a PCTMC, then x_i and x_j can be treated as independent of each other if*

$$(x_i, x_j) \in \mathcal{R}^{(d')} \wedge d' > d$$

where $d \geq 0$ is a threshold chosen by the modeller.

Note when $d = 0$, it means that we assume all the population variables are independent; when $d = \infty$, it means all the population variables are treated as inter-dependent.

4.2.2 Reduction Method

After the identification of independent population variables in the PCTMC using the neighbourhood relation, we can construct a *correlation graph* for each distinct moment variable in the derived moment ODEs. The definition of a correlation graph is given below:

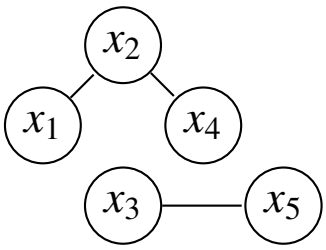

Figure 4.1: The correlation graph of a moment variable $\mathbb{E}[x_1x_2x_3x_4x_5^2]$.

Definition 1. *The correlation graph $\mathcal{G}$ of a moment variable $\mathbb{E}[\mathbf{x}^{\mathbf{m}}] = \mathbb{E}[x_1^{m_1}\cdots x_n^{m_n}]$ is a graph, in which there is a node for each population variable x_i that appears in the expression of the moment variable. Moreover, there is an edge $Edge(x_i,x_j)$ between two nodes if and only if $(x_i,x_j) \in \mathcal{R}^{(d')} \wedge d' \leq d$.*

Furthermore, we say two nodes in $\mathcal{G}$ are connected if and only if there exists a path between them. Therefore, the correlation graph of a moment variable can consist of one or more *correlation islands*. The correlation islands are defined as follows:

Definition 2. *A correlation island I is a subgraph of a correlation graph $\mathcal{G}$ such that:*

$$\begin{aligned} \forall x_i, x_j \in I \quad &\longrightarrow \quad x_i \text{ and } x_j \text{ are connected} \\ \forall x_i \in I, x_j \notin I \quad &\longrightarrow \quad x_i \text{ and } x_j \text{ are not connected} \end{aligned}$$

Figure 4.1 illustrates the correlation graph of a moment variable $\mathbb{E}[x_1x_2x_3x_4x_5^2]$ which consists of two correlation islands. Each correlation island in a correlation graph represents a decoupled moment variable with a lower order than the moment variable represented by the correlation graph. Thus, with the identification of correlation islands in a correlation graph, we can decouple the moment variable for ODE reduction. Specifically, let $\mathbb{E}[\mathbf{x}^{\mathbf{m}}] = \mathbb{E}[x_1^{m_1}\cdots x_n^{m_n}]$ be an arbitrary moment variable that appears on the left hand side of a moment ODE, $\mathcal{G}$ be the corresponding correlation graph, and I be a correlation island in $\mathcal{G}$. We can approximate $\mathbb{E}[\mathbf{x}^{\mathbf{m}}]$ by the following formula:

$$\mathbb{E}[\mathbf{x}^{\mathbf{m}}] \approx \prod_{I \in \mathcal{G}} \mathbb{E}[\prod_{x_i \in I} x_i^{m_i}] \tag{4.2}$$

Furthermore, according to the above formula, for a moment variable $\mathbb{E}[\mathbf{x}^{\mathbf{m}}]$ which appears on the left hand side of a moment ODE, if its correlation graph consists of more than one correlation island, then this moment ODE can be eliminated since it can be approximated by the product of moment variables with lower orders.

Clearly, with a smaller value of reduction threshold d, more population variables in the PCTMC will be approximated as independent of each other. As a result, we can use fewer ODEs to describe the joint moments for the population variables in the PCTMC. But, in the meantime, a larger amount of error can also be introduced by the independence approximation. Therefore, the reduction threshold d is a factor to control the trade-off between the efficiency and accuracy of moment-closure approximation. Thus, starting from $d = 0$ where all population variables are treated as independent of each other, we can find the optimal value of the reduction threshold d for the moment-closure approximation of an arbitrary PCTMC whenever increasing the value of d will not make any observable difference in the results on the evolution of required moments.

In our case studies, we will show that this reduction method can substantially reduce the number of moment ODEs, but still retains very good accuracy compared with the moment-closure approximation without any reduction.

4.3 Moment-closure Method

In this section, we formally explain our automatic moment closure method for an arbitrary PALOMA model. Specifically, for an arbitrary PALOMA model, we derive its moment ODEs up to order $\overline{m}$ according to Equation 4.1, where $\overline{m}$ is the highest order of moments required by the modeller. Then, let $\mathbb{E}[\mathbf{x}^{\mathbf{m}}] = \mathbb{E}[x_1^{m_1} \cdots x_n^{m_n}]$ denote a moment variable that appears on the right hand side of a moment ODE, whose moment order is $o(\mathbf{m}) = m_1 + \ldots + m_n$ $\big($note that we restrict all the moment variables to this product form by letting $\mathbb{E}[f(\mathbf{X})] \approx \mathbb{E}[f_1(\mathbf{X})]/\mathbb{E}[f_2(\mathbf{X})]$ if $f(\mathbf{X}) = f_1(\mathbf{X})/f_2(\mathbf{X})\big)$. Then if there exists a moment variable $\mathbb{E}[\mathbf{x}^{\mathbf{m}}]$ and $o(\mathbf{m}) > \overline{m}$, the ODE system is not closed. Therefore, we need to approximate $\mathbb{E}[\mathbf{x}^{\mathbf{m}}]$ with moment variables whose order is less than or equal to $\overline{m}$, which is the so-called *moment closure approximation*.

For the purpose of moment closure, the first thing we do is to utilise the correlation graph of moment variables introduced in the previous section. Specifically, for an arbitrary moment variable which appears on the right hand side of a moment ODE, if its correlation graph consists of more than one correlation island, we approximate it using Equation (4.2).

Secondly, if the derived moment ODEs only contain moment variables whose orders are no higher than $\overline{m}+1$ (these are cases when the probability of receiving a broadcast message and the weight of receiving a unicast message are constants), we apply the lognormal closure method [Singh and Hespanha, 2006] to close the system of mo-

ment ODEs at order $\overline{m}$. Specifically, the lognormal closure method assumes that the dependence of a higher-order moment on lower order ones is consistent with the population being joint lognormally distributed. Compared with moment closure methods with other probability distribution assumptions on population variables, the lognormal closure has the advantage of probabilities only being defined for positive values, thus it is most suitable for PALOMA models. The details for deriving lognormal closure formulas for an arbitrary order have been introduced in Section 2.2.2. For illustration, if we set $\overline{m} = 2$, using the lognormal closure technique, we can approximate all third order moment variables by:

$$\begin{aligned}
\mathbb{E}[{x_i}^3] &\approx \Big(\frac{\mathbb{E}[{x_i}^2]}{\mathbb{E}[x_i]}\Big)^3 \\
\mathbb{E}[x_i {x_j}^2] &\approx \frac{\mathbb{E}[{x_j}^2]\mathbb{E}[x_i x_j]^2}{\mathbb{E}[x_i]\mathbb{E}[x_j]^2} \\
\mathbb{E}[x_i x_j x_k] &\approx \frac{\mathbb{E}[x_i x_j]\mathbb{E}[x_i x_k]\mathbb{E}[x_j x_k]}{\mathbb{E}[x_i]\mathbb{E}[x_j]\mathbb{E}[x_k]}
\end{aligned}$$

In case when the derived moment ODEs also contain moment variables that are higher than order $\overline{m}+1$, we use a heuristic algorithm to reduce the order of those moment variables to $\overline{m}+1$. Concretely, we first utilise the neighbourhood relation defined in the previous section to estimate the dependence of a population variable with a list of other population variables. More specifically, we let

$$(x_i, [x_{j_1}, \ldots, x_{j_n}]) \in \mathcal{R}^{(N)} \iff (x_i, x_{j_k}) \in \mathcal{R}^{(d_k)}, \quad 1 \leq k \leq n \wedge \sum_{k=1}^{n} d_k = N.$$

Then, for a moment variable $\mathbb{E}[\mathbf{x}^{\mathbf{m}}] = \mathbb{E}[x_1^{m_1} \cdots x_n^{m_n}]$ with $o(\mathbf{m}) > \overline{m}+1$, we reduce its order, by letting

$$\mathbb{E}[\mathbf{x}^{\mathbf{m}}] = \mathbb{E}[x_1^{m_1} \cdots x_n^{m_n}] \approx \mathbb{E}[x_i^{m_i}]\mathbb{E}[\prod_{j=1,\ldots,n \wedge j \neq i} x_j^{m_j}] \tag{4.3}$$

where x_i is the least correlated population variable in $(x_1, \ldots, x_n)$, such that

$$N_i = \max(N_1, \ldots, N_n),$$

where

$$(x_i, [x_1, \ldots, x_n]) \in \mathcal{R}^{(N_i)} \quad \forall (1 \leq i \leq n).$$

We apply the above algorithm until the order of the moment variable is $\overline{m}+1$, after which, the lognormal moment-closure can be applied. The whole flow of our automatic moment closure approximation method for an arbitrary PALOMA model is given in Algorithm 4.1.

Algorithm 4.1 Automatic moment closure approximation for a PALOMA model

Require: $\mathcal{P}$ {the PCTMC after transition combination}, $\overline{m}$, d

Derive moment ODEs up to order $\overline{m}$ for $\mathcal{P}$ according to Equation 4.1
Put all distinct moment variables in the derived moment ODEs into Set $\mathbb{S}$.
for all moment variables $\mathbb{E}[\mathbf{x}^{\mathbf{m}}]$ in $\mathbb{S}$ **do**
  Construct the correlation graph $\mathcal{G}$ for $\mathbb{E}[\mathbf{x}^{\mathbf{m}}]$ with reduction threshold d
  if $\mathcal{G}$ consists of more than one correlation island **then**
    if $o(\mathbf{m}) \leq \overline{m}$ **then**
      Eliminate the ODE describing $\mathbb{E}[\mathbf{x}^{\mathbf{m}}]$
      Replace $\mathbb{E}[\mathbf{x}^{\mathbf{m}}]$ in all other ODEs according to Equation 4.2
    else if $o(\mathbf{m}) > \overline{m}$ **then**
      Replace $\mathbb{E}[\mathbf{x}^{\mathbf{m}}]$ in all ODEs according to Equation 4.2
      Put any new moment variables on the right hand side of Equation 4.2 into $\mathbb{S}$
    end if
  else if $\mathcal{G}$ consists of a single correlation island **then**
    if $o(\mathbf{m}) > \overline{m}+1$ **then**
      Reduce the order of $\mathbb{E}[\mathbf{x}^{\mathbf{m}}]$ to $\overline{m}+1$ according to Equation 4.3
    else if $o(\mathbf{m}) = \overline{m}+1$ **then**
      Apply lognormal moment-closure approximation to reduce the order of $\mathbb{E}[\mathbf{x}^{\mathbf{m}}]$ to $\overline{m}$
    end if
  end if
end for
Numerically solve the reduced set of moment ODEs

4.4 Case Studies

In this section, we apply our automatic moment closure approximation algorithm to three CAS from different areas, and compare the results with population-level stochastic simulation using the standard SSA. Without loss of generality, we set $\overline{m} = 2$ in the experiments of the first two cases. In the third case, we set $\overline{m} = 3$. The scripts of the three models can be found on the PALOMA Eclipse plugin wiki page `https://github.com/cfeng783/paloma/wiki#the-paloma-eclipse-plug-in`.

4.4.1 An Epidemiological SIS Model

We first consider a classical epidemiological SIS model of individuals partitioned into communities, where individuals move between communities but infections only take place within communities. Each individual is considered to be susceptible (S) or infected (I) with respect to the disease. More specifically, consider a total fixed population of N individuals partitioned into m communities in a ring topology, each of which contains n individuals ($N = n \times m$). A continuous-time SIS epidemiological model is then applied to this population as follows: each individual, regardless of its susceptible or infected status, can move to his/her neighbour communities with rate r. Each infected individual contacts and attempts to infect others in the same community at rate λ. Each contact is with a randomly chosen individual. When an infected individual contacts a susceptible individual, the latter becomes infected as well. Finally, infected individuals independently recover to the susceptible state at rate μ.

The individuals in susceptible and infected states can be represented in PALOMA by the following agents:

$$\begin{aligned} S(\ell_i) &= ??(contact,1)@\mathbf{Wt}\{1\}.I(\ell_i) + \sum_{j\in \text{nearby}(i)} (move_{ij},r).S(\ell_j) \\ I(\ell_i) &= !!(contact,\lambda)@\mathbf{IR}\{local\}.I(\ell_i) + ??(contact,1)@\mathbf{Wt}\{1\}.I(\ell_i) \\ &\quad + (recover,\mu).S(\ell_i) + \sum_{j\in \text{nearby}(i)} (move_{ij},r).I(\ell_j) \end{aligned}$$

where $S(\ell_i)$ and $I(\ell_i)$ denote an individual in the susceptible and infected state currently in community i respectively, $\text{nearby}(i) = \{(i+1) \mod m, (i-1+m) \mod m\}$ is the index of nearby communities of community i.

The initial population of agents are given in the following definition:

$$S(\ell_1)[n-I_1] \parallel I(\ell_1)[I_1] \parallel \ldots \parallel S(\ell_i)[n-I_i] \parallel I(\ell_i)[I_i] \parallel \ldots \parallel S(\ell_m)[n-I_m] \parallel I(\ell_i)[I_m]$$

where I_i denotes the number of initially infected individuals in community i.

In the simulation, we randomly choose 5 out of 50 communities as the source of the epidemic. There are 5 individuals in the 5 chosen source communities who are infected initially. All the other individuals in the model are in the susceptible state initially. Table 4.1 gives the simulation configuration of the SIS model. The values of the parameters in the model are chosen to make the model close to realistic scenarios. For simplicity, the number of simulation runs is chosen to make the first moment and the second moment observable (an alternative choice would be to let the simulation result achieve certain level of statistical significance). The stop time of a simulation run is chosen to let the first moment and the second moment converge. The same standard is also applied in the next two examples.

m	50
n	50
r	1
λ	2
μ	1
I_i (ℓ_i is a source community)	5
I_i (ℓ_i is not a source community)	0
Stop time of a simulation run	20
Number of simulation runs	10,000

Table 4.1: The SIS model simulation configuration

The analysis of interest in the SIS model is the number of infected individuals over all the communities. We apply moment-closure analysis on the SIS model with different reduction thresholds based on the neighbourhood relation of population variables, and then compare the results with the stochastic simulation. Figure 4.2 and 4.3 show the trajectories of the first and second moments of the infectious population in the SIS model in the 10,000 runs of stochastic simulation as well as moment analysis, respectively. It can be seen that with a stricter reduction standard (the larger value of d), the result of moment analysis is closer to the stochastic simulation. Nevertheless, the result of moment analysis when $d = 3$ is almost the same as that with $d = 4$, which means that the second-order joint moments between population variables when their neighbourhood relation is larger than three hops gives little extra information than their first moments. Therefore, we can ignore the correlation between those population variables

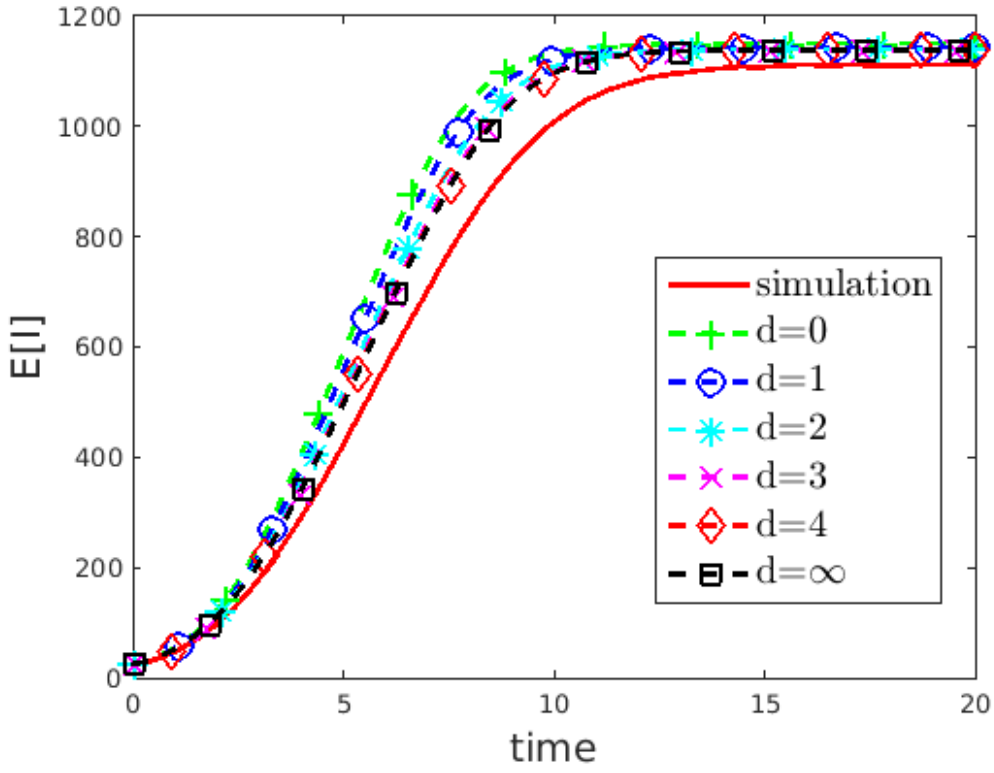

Figure 4.2: The first moment of infected population

without loss of accuracy but with the gain of reduced solution time for the moment ODEs. More evidence is shown in Table 4.2, where the error ratio is calculated by averaging the difference between the stochastic simulation and the moment analysis over 200 data points evenly selected in the trajectories along the simulation time. We can see that the number of ODEs and solution time for moment analysis can be significantly reduced by our reduction method with only limited loss of accuracy compared with the full moment analysis ($d = \infty$) as long as the optimal value of d ($d = 3$ in this case) is chosen.

Furthermore, it is clear that moment analysis can enormously reduce the computational cost of analysing a PALOMA model compared with stochastic simulation (in this thesis, we do not consider situations in which stochastic simulations are run in parallel on multiple machines such as cloud clusters. Moment-closure approximation is less likely to be parallelised).

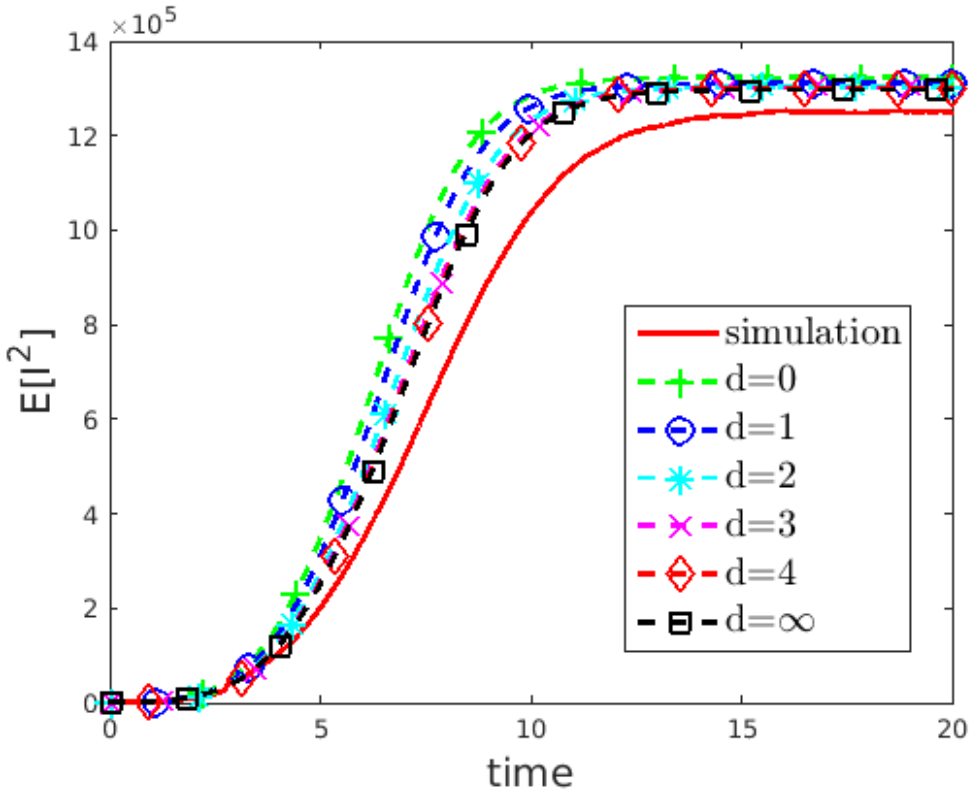

Figure 4.3: The second moment of infected population

4.4.2 A Wireless Sensor Network Model

Here, we discuss a spatial model that represents the spread of pheromone in a multi-hop Wireless Sensor Network (WSN). In nature, pheromone is a hormone laid down by colony-based insects, to indicate popular routes to food sources or new nest sites. In a similar manner pheromone gradients have been adapted in the WSN literature as an abstract means of studying the evolution of routes from source to sink nodes. Several models have been built to investigate the spread of pheromone in such networks [Bruneo et al., 2012, Guenther et al., 2013]. We show how to capture those models in PALOMA. Figure 4.4 visualises the topology of the WSN model, where there is a sink node in cell 13 and there is a sensor deployed in each cell.

21	22	23	24	25
16	17	18	19	20
11	12	13	14	15
6	7	8	9	10
1	2	3	4	5

Figure 4.4: The topology of the WSN model

SIS model	ODE number	Solution time	Error ratio	
			1st moment	2nd moment
Simulation (10,000 runs)	N/A	10.39 hrs	N/A	N/A
Moment analysis with $d = 0$	200	0.31 secs	14.01%	22.78%
Moment analysis with $d = 1$	350	0.83 secs	11.27%	18.37%
Moment analysis with $d = 2$	550	1.1 secs	8.61%	14.16%
Moment analysis with $d = 3$	750	1.34 secs	7.42%	12.30%
Moment analysis with $d = 4$	950	1.82 secs	7.41%	12.21%
Moment analysis with $d = \infty$	5150	31.99 secs	7.16%	11.34%

Table 4.2: Simulation vs. moment analysis of the SIS model

Assuming we are only interested in the spread of pheromone, then the sink node which is also the source of the pheromone spread broadcasts a message containing the maximum pheromone level to all sensor nodes in the network at rate λ_{sink}. Thus, the sink node is represented as follows:

$$Sink(\ell_{13}) \quad = \quad !(ph_{\max}, \lambda_{sink})@\mathbf{IR}\{all\}.Sink(\ell_{13})$$

The pheromone level in a sensor node is denoted by an integer in the range 0 to max. A sensor node can update its pheromone level to max once it receives a broadcast message from the sink node. However, the probability of a message from the sink node being received by a sensor node depends on the physical distance between the sink node and sensor node. The sensor nodes can also exchange pheromone information with their neighbourhood sensor nodes using a Manhattan style communication pattern. The pheromone level in each sensor node is assumed to decrease exponentially at rate μ. Moreover, sensor nodes can also enter an off state at rate r_{off}. Sensor nodes in the off state can do nothing but sleep for a while and return to the on state with rate r_{on}. When a sensor node re-enters the on state, its pheromone level is set to 0.

Thus, the sensor nodes can be represented as follows:

$$\begin{aligned}
Sensor_{ph_k}(\ell_i) &= \sum_{k<j\leq \max} ?(ph_j,1)@\mathbf{Pr}\{\frac{1}{1+dist(\ell_s,\ell_i)}\}.Sensor_{ph_j}(\ell_i) \\
&\quad + !(ph_k,\lambda_{sensor})@\mathbf{IR}\{range(1)\}.Sensor_{ph_k}(\ell_i) \\
&\quad + (evaporate,\mu).Sensor_{ph_{k-1}}(\ell_i) \quad (k>1) \\
&\quad + (\mathrm{off},r_{\mathrm{off}}).Sensor_{\mathrm{off}}(\ell_i) \\
Sensor_{\mathrm{off}}(\ell_i) &= (\mathrm{on},r_{\mathrm{on}}).Sensor_{ph_0}(\ell_i)
\end{aligned}$$

where $Sensor_{ph_k}(\ell_i)$ denotes a sensor node in cell i currently with pheromone level k. The sensor node can receive a message containing a higher level pheromone and then update its pheromone level. $dist(\ell_s,\ell_i)$ is the distance between ℓ_i and the location of the message sender (can be either a sink node or a sensor node). It can also broadcast a message containing its current pheromone level to its neighbourhood at rate λ_{sensor}. The other two actions capture the evaporation of pheromone and the sleep of nodes.

Table 4.3 summarises the simulation configuration of the model, where we set the maximum pheromone level to 5. The analysis of interest is the spread of pheromone: the number of sensor nodes with different pheromone levels and the expected pheromone level in each node. Figures 4.5 and 4.6 show the trajectories of the first moment and second moment of the number of sensor nodes with different pheromone levels (note that total number of sensor nodes is constant in the model, thus some numbers are overestimated whereas some are underestimated). Figure 4.7 shows the expected pheromone level of the sensor nodes in each cell at time 100 which is the stop time of a simulation run. In both cases, we can see that moment analysis with $d = 1$ gives much closer results to stochastic simulation than moment analysis with $d = 0$. Table 4.4 compares the moment analysis with different reduction thresholds with stochastic simulation. In this case, moment analysis cannot be applied without our reduction method since the number of ODEs is too large for Matlab to solve when $d > 1$. Moreover, again we can see that moment analysis with $d = 1$ gives much better results than moment analysis with $d = 0$. This is because fluid limit analysis neglects all the correlations between population variables and can only give a good result at the first moment when the population is large [Tribastone et al., 2012]. However, in this case, the population is too small for this kind of analysis.

max	5
λ_{sink}	0.35
λ_{sensor}	0.35
μ	0.5
r_{off}	0.05
r_{on}	0.25
Stop time of a simulation run	100
Number of simulation runs	10,000

Table 4.3: The WSN model simulation configuration

sensor network model	ODE number	Solution time	Error ratio	
			1st moment	2nd moment
Simulation (10,000 runs)	N/A	15.55 mins	N/A	N/A
Moment analysis with $d = 0$	352	0.21 secs	33.34%	45.42%
Moment analysis with $d = 1$	2427	25.94 secs	8.72%	12.97%
Moment analysis with $d = 2$	12332	out of memory	N/A	N/A

Table 4.4: Simulation vs. moment analysis of the WSN model

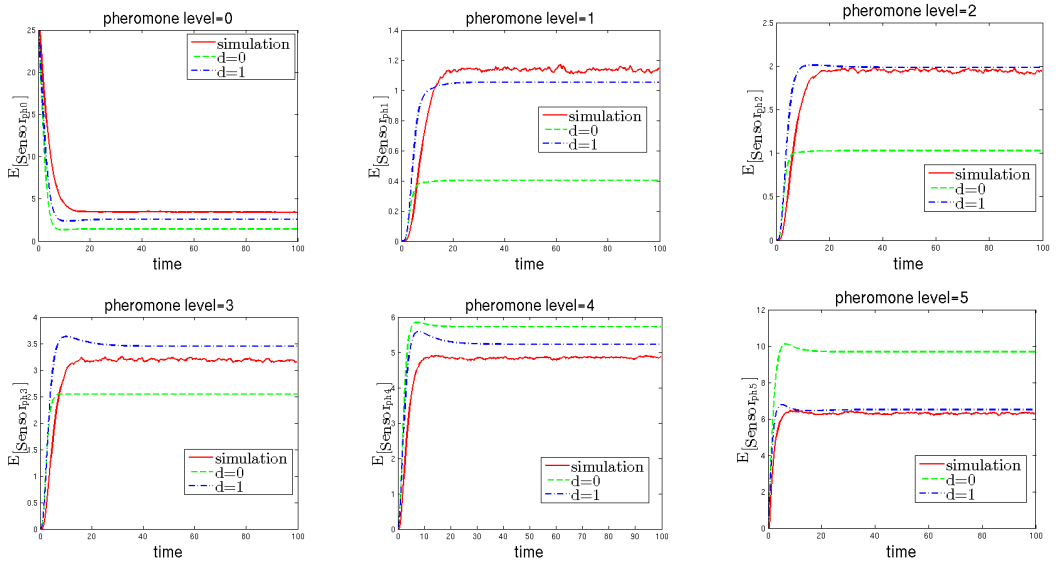

Figure 4.5: The first moment of number of sensor nodes with different pheromone level

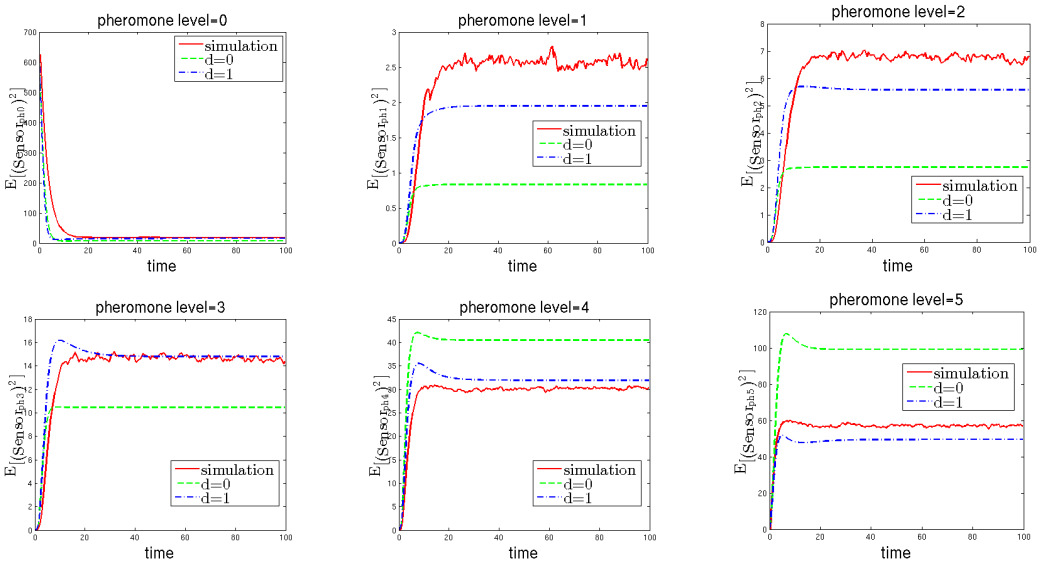

Figure 4.6: The second moment of number of sensor nodes with different pheromone level

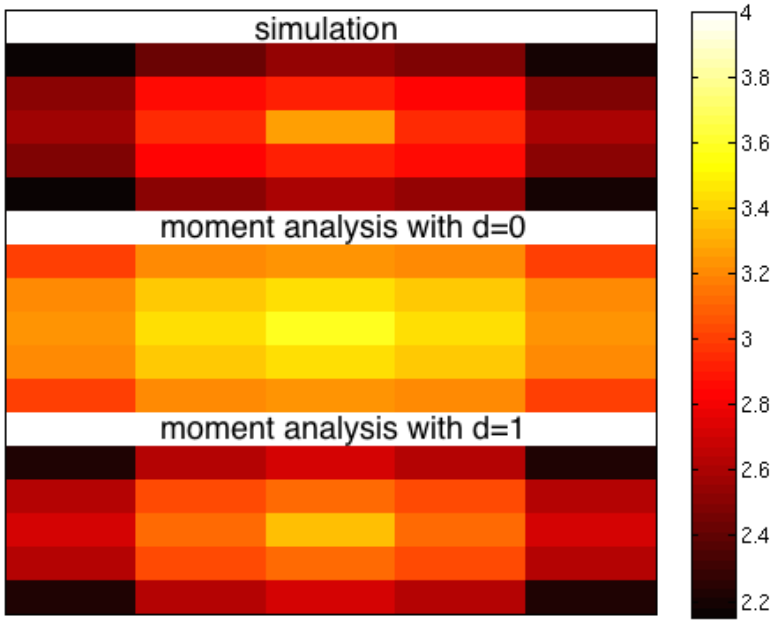

Figure 4.7: The expected pheromone level in each cell at time 100.

m	16
d	1
θ_0	0
θ_1	0.5
θ_2	0.5
N_{pd_i}	25
N_{br_i}	5
N_{s_i}	5
N_{b_i}	10
Stop time of a simulation run	150
Number of simulation runs	10,000

Table 4.5: The bike-sharing model simulation configuration

4.4.3 The City Bike-sharing Model

The last example we discuss is the city bike-sharing example from the previous chapter. Here, we consider the city is divided into 16 zones in a 4×4 grid. Table 4.5 gives the simulation configuration of the bike-sharing model (only the values of key parameters are listed). Here, we set $\overline{m} = 3$, thus the analysis of interest is the first three moments of the number of available bikes in each station over time. Table 4.6 gives the detailed comparison between moment closure with different reduction thresholds and stochastic simulation, where the error ratio takes into account of the trajectories of available number of bikes in all the 16 stations. For illustration, we give the trajectories of the first three moments of the number of available bikes in the central station in Figure 4.8. Here, we can observe again that moment analysis with $d > 1$ is infeasible due to the extremely large number of moment ODEs. Neglecting all correlations between population variables results in very poor accuracy ($d = 0$). Nevertheless, moment analysis with $d = 1$ can achieve good accuracy since it can capture the most important correlations between population variables in the PCTMC.

4.5 Summary

In this chapter, we proposed a moment-closure approximation method that can be automatically applied to an arbitrary PALOMA model. Moment-closure techniques have been studied for many years in different scientific areas. The goal is to achieve a closed

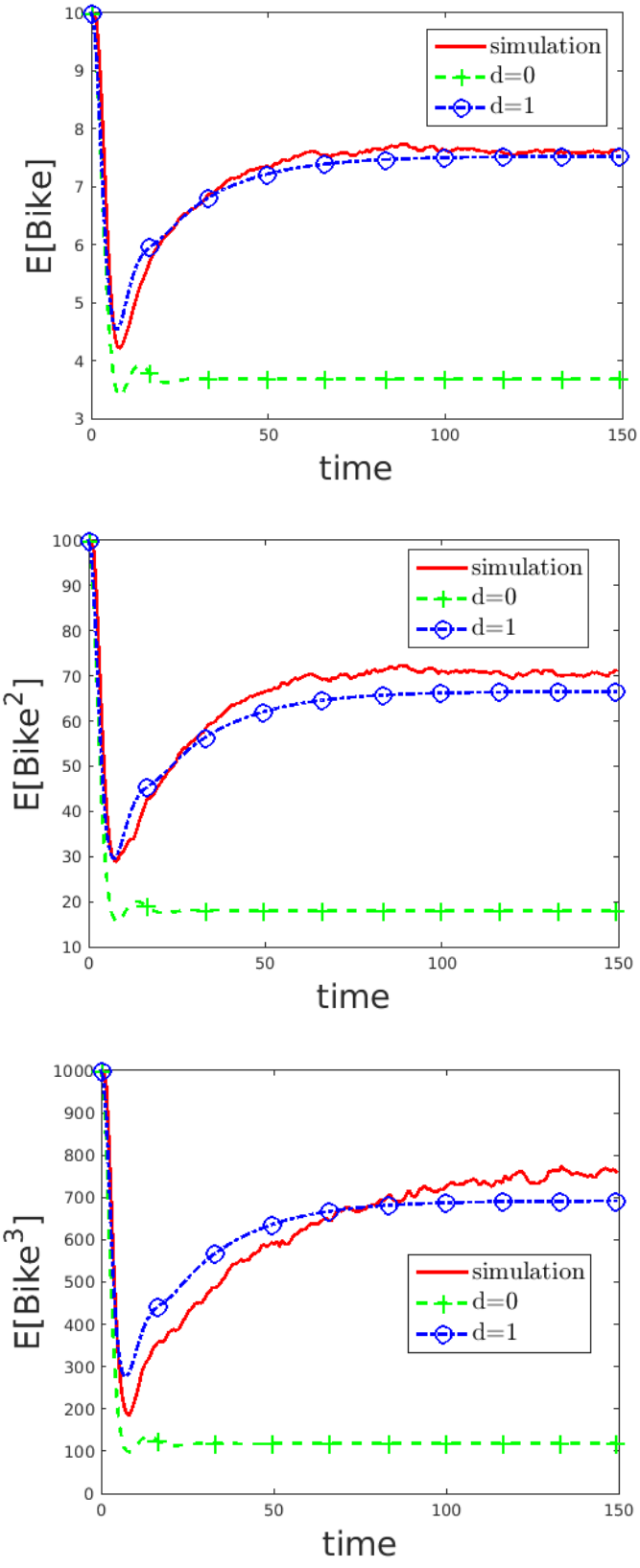

Figure 4.8: The first three moments of number of available bikes in the central station

bike-sharing model model	ODE number	Solution time	Error ratio		
			1st moment	2nd moment	3rd moment
Simulation (10,000 runs)	N/A	19.96 hrs	N/A	N/A	N/A
Moment analysis with $d = 0$	912	4.14 secs	42.51%	64.92%	76.91%
Moment analysis with $d = 1$	19360	9.7 mins	2.74%	6.39%	9.92%
Moment analysis with $d = 2$	271266	out of memory	N/A	N/A	N/A

Table 4.6: Simulation vs. moment analysis of the bike-sharing model

form of an infinite set of coupled differential equations by expressing higher-order moments in terms of lower-order moments. Many different closure techniques have been introduced (cf. Section 2.2.2), each is suitable for solving a particular set of models. Our moment-closure method for PALOMA models firstly utilises the neighbourhood relation between population variables to reduce the order of moment variables which consists of population variables whose correlation can be ignored without causing significant deviation on results. Consequently, a large set of ODEs for joint-moments can be removed since they can be approximated by moment variables with lower orders. The lognormal closure is applied afterwards on moment variables whose orders cannot be reduced using the neighbourhood relation. Specifically, the lognormal closure is chosen because its closure operation can be easily automated and the lognormal distribution assumption only assigns probabilities to positive values for population variables. In our experiments, we showed that the results of the our method can significantly improve the scalability of moment-closure approximation for PALOMA models and still have acceptably low levels of error. Lastly, since both our moment-closure and ODE reduction methods are defined on the PCTMC level, we think the same approach can be applied to other high-level modelling formalisms as long as their underlying mathematical models are also PCTMCs, especially for those with large population sizes.

Chapter 5

The Speed-up of Stochastic Simulation of PCTMCs

In the previous chapter, we showed that moment-closure approximation is a much more efficient option for analysing large-scale PCTMCs compared than stochastic simulation. However, this does not mean moment-closure approximation can always replace stochastic simulation. First of all, although moment-closure approximations usually work well in practice, rigorous justification of them is rather difficult and there is no straightforward way to predict their accuracy [Kuehn, 2016, Schnoerr et al., 2015]. Therefore, in practice, even if moment-closure approximation is used, a certain number of stochastic simulation runs are still needed in order to check whether the approximation works well. More importantly, although stochastic simulation is computationally expensive, it is also the most informative computational technique for analysing PCTMCs. For instance, an important goal of representing dynamic systems such as CAS by PCTMCs is to undertake stochastic model checking where certain properties of the underlying systems can be verified by investigating the associated mathematical model, and such properties are often expressed using probabilities [Kwiatkowska et al., 2007]. For example, in a SIS model, we might be interested in whether the probability that the number of infected individuals exceeds N at any time point between t_0 and t_1 is less than 5%. Thus, in these cases, only knowing the moments of populations is not sufficient to do the verification. Although moments can be used to reconstruct the probability distribution of populations by the maximum entropy approach [Andreychenko et al., 2015], the maximum entropy approach requires knowledge of higher moments in order to achieve a faithful reconstruction. This tends to increase the risk of the underlying moment ODEs becoming intractable for CAS.

Thus, in some circumstances, statistical model checking [Legay et al., 2010] in which probabilities are obtained through sampling from multiple stochastic simulation runs, is the only feasible approach for verifying properties expressed using probabilities for CAS.

Fortunately, although the scale and spatially distributed nature of CAS increase the complexity of the underlying PCTMCs, on the other hand, it also offers possibilities for model reduction. For instance, two agents which are located far away from each other are generally less likely to influence each other than another two agents located in close proximity. Therefore, if we are interested in only a few populations in the PCTMC (which is often the case for model checking), we can remove some population variables and transitions which have very limited influence on our target from simulation. As modellers we know that a model is an abstraction of the system in the real world. Thus it inevitably contains some deviation from the real system due to details that are omitted in the abstraction process. Consequently, except for the case of particular safety critical systems, it is generally acceptable to allow some minor noise to be introduced to a model during construction. Taking this perspective a little further, we can consider the transitions and population variables that we removed from the simulation as noise factors which have negligible impact on the evolution of populations of interest. Based on this idea, we propose a novel approach to speed up stochastic simulation of PCTMCs by removing a set of transitions and population variables which will not cause a significant error in the simulation result.

Specifically, in order to identify those removable population variables and transitions, we define a directed coupling graph for an arbitrary PCTMC which quantifies the coupling between population variables and transitions in the PCTMC. The graph can be constructed at a relatively low computational cost compared with the total simulation cost. Using the graph, a reduction proposal which specifies the most likely removable population variables and transitions, with respect to some target populations of interest, can be automatically generated. Moreover, in order to ensure that the generated reduction proposal will not cause unacceptable error in the simulation results of target populations, we utilise a deterministic model to efficiently check the deviation of the dynamics of target populations before and after a reduction proposal. An optimal reduction proposal can be automatically derived based on an acceptable error threshold (for the target populations). Then, the simulation can be safely accelerated according to the optimal reduction proposal. This chapter is based on the work published in EPEW 2015 [Feng and Hillston, 2015].

5.1 Related Works

There has been much work which focuses on improving the speed of stochastic simulation, such as the tau-leaping algorithm and many other multi-time scale based algorithms (cf. Section 2.2.1). Even though so many algorithms have been proposed, stochastic simulation of PCTMCs is still very expensive due to the large number of required simulation runs and the increasing size of systems under study. Especially for CAS, just like the three case-study models in the previous chapter, it is likely that there is no opportunity for time-scale separation even when faced with very large-scale systems. The tau-leaping algorithm is also not efficient enough to deal with large models since it still tries to simulate every transition event in the system. Therefore, in this chapter, we seek a different approach to speed up stochastic simulation of PCTMCs through model reduction which is particularly suitable for CAS due to their highly distributed nature. This is achieved by automatically generating reduction proposals which specify a set of removable transitions and population variables from simulation.

Similar to our work is the directed relation graph (DRG)-based methods for skeletal mechanism reduction for the simulation of hydrocarbon oxidation, where a graph-based model reduction approach is also used for removing unimportant species and reactions whose contribution to species of interest is negligible [Lu and Law, 2005]. The approach has since been improved by researchers in the combustion research domain such as DRG with error propagation [Pepiot-Desjardins and Pitsch, 2008], DRG with sensitivity analysis [Niemeyer et al., 2010], etc. Our work is inspired by the DRG-based methods, however, there are some key differences. First, the DRG-based methods are used to reduce deterministic models whereas our work is applied to stochastic simulations. Second, since our goal is to speed up stochastic simulation, our primary focus is on transition reduction instead of the species (population variables) reduction that is the focus of the DRG-based methods. Lastly, although the DRG-based methods work well in the combustion simulation domain, they are still heuristics since no accuracy can be guaranteed for reductions. Our work has an error control step where the reduced model can be guaranteed to satisfy an error threshold, which makes our work more convincing and easier to apply. The whole process of our automatic reduction method is fast, and has low computational cost compared to the total simulation cost. In the following section, the details of how to generate a reduction proposal are described.

5.2 Reduction Proposal Generation

The cost of the standard stochastic simulation algorithm (SSA) depends on the number of transition events in the system (see Algorithm 2.1), thus, to speed up stochastic simulation, given an arbitrary PCTMC $\mathcal{P} = (\mathbf{X}, \mathcal{T}, \mathbf{X}_0)$, our primary goal is to specify a largest set of transitions $\mathcal{T}_{rm}$ that can be removed from simulation without causing an unacceptable impact on the target populations. A group of population variables $\mathbf{X}_{rm}$ which are not involved in any transitions after transition elimination can also be identified. A reduction proposal which represents a reduced version of $\mathcal{P}$, denoted as $\hat{\mathcal{P}} = (\hat{\mathbf{X}}, \hat{\mathcal{T}}, \hat{\mathbf{X}}_0)$, where $\hat{\mathbf{X}} = \mathbf{X} - \mathbf{X}_{rm}$, $\hat{\mathcal{T}} = \mathcal{T} - \mathcal{T}_{rm}$, $\hat{\mathbf{X}}_0$ is the associated initial condition for $\hat{\mathbf{X}}$, is then generated. This is achieved by defining appropriate coupling coefficients as a measure of the influence of the transitions on the target populations. Since removing a transition which has very small influence on the target populations is less likely to induce a significant error in them, we can treat transitions whose coupling coefficients to the target populations are lower than a threshold as removable. In the remaining part of this section, the definition of coupling coefficients will be described.

5.2.1 Direct Coupling Coefficient

Transitions and populations can be coupled through direct and indirect influence on each other. The direct coupling coefficients are defined as a measure of the direct influence of a transition on the dynamics of a population variable, or the other way around. The direct influence of a transition on a population variable is measured differently to the direct influence of a population variable on a transition. Thus, their definitions are also given separately.

The direct coupling coefficient of a transition τ_j to a population variable x_i is defined as:

$$c_{x_i,\tau_j} = \frac{|d^i_{\tau_j} N_{\tau_j}|}{\sum_{\tau \in \mathcal{T}} |d^i_\tau N_\tau|} \tag{5.1}$$

where d^i_τ is the update of x_i caused by the firing of transition τ, N_τ is the firing count of transition τ during a simulation run. Intuitively, c_{x_i,τ_j} measures the proportional contribution of the transition τ_j to the evolution of population variable x_i. With smaller values of c_{x_i,τ_j}, the removal of transition τ_j from simulation will be less likely to immediately induce a significant error in the evolution of population variable x_i.

The direct coupling coefficient of a population variable x_i to a transition τ_j is de-

fined as:

$$c_{\tau_j,x_i} = \begin{cases} 1, & \text{if population variable } x_i \text{ contributes to transition } \tau_j \\ 0, & \text{otherwise} \end{cases} \tag{5.2}$$

where we say x_i contributes to transition τ_j if and only if x_i appears at the reactant side of τ_j (assuming expressing transitions in the chemical reaction style) or the rate of τ_j depends on x_i. Since removing a population variable which contributes to a transition will immediately invalidate the transition, the direct influence of a population variable on a transition is either 100% or 0.

The direct coupling coefficients between two population variables or two transitions are always defined to be zero because we assume they are always not directly coupled:

$$c_{x_i,x_j} = 0, \quad \forall(x_i,x_j) \tag{5.3}$$

$$c_{\tau_i,\tau_j} = 0, \quad \forall(\tau_i,\tau_j) \tag{5.4}$$

5.2.1.1 Evaluate the Firing Count of Transitions

A key point for the computation of direct coupling coefficients is the evaluation of N_τ, the firing count of transitions during a simulation run (all other factors in the definitions of direct coupling coefficients in Equations 5.1 and 5.2 can be directly obtained from the PCTMC description). Thus, in order to achieve a convincing evaluation of N_τ for each transition, we compute the average firing count of the transition over infinite simulation runs. This can be achieved efficiently by moment-closure approximation of a PCTMC with additional dummy population variables representing the counter of the firing of transitions. Specifically, based on a PCTMC $\mathcal{P} = \Big(\mathbf{X} = (x_1,\ldots,x_n), \mathcal{T} = (\tau_1,\ldots,\tau_m), \mathbf{X}_0 = \big(x_1(0),\ldots,x_n(0)\big)\Big)$ for stochastic simulation, we can construct another PCTMC $\mathcal{P}' = (\mathbf{X}', \mathcal{T}', \mathbf{X}'_0)$, in which:

- $\mathbf{X}' = (x_1,\ldots,x_n,x_{n+1},\ldots,x_{n+m})$, where x_{n+i} $(1 \le i \le m)$ is a dummy population variable representing the counter of the firing of transition τ_i.
- $\mathcal{T}' = (\tau'_1,\ldots,\tau'_m)$, where for $1 \le i \le m$, $\tau'_i = (r_{\tau'_i}(\mathbf{X}'), \mathbf{d}_{\tau'_i})$ such that $r_{\tau'_i}(\mathbf{X}') = r_{\tau_i}(\mathbf{X})$, $\mathbf{d}_{\tau'_i} = (d^1_{\tau'_i},\ldots,d^n_{\tau'_i},d^{n+1}_{\tau'_i},\ldots,,d^{n+m}_{\tau'_i})$ in which

$$d^j_{\tau'_i} = \begin{cases} d^j_{\tau_i} & \text{if } 1 \le j \le n \\ 0 & \text{if } n+1 \le j \le n+m \wedge j \ne n+i \\ 1 & \text{if } j = n+i \end{cases}$$

- $\mathbf{X}'_0 = \big(x'_1(0),\ldots,x'_n(0),x'_{n+1}(0),\ldots,x'_{n+m}(0)\big)$ where

$$x'_i(0) = \begin{cases} x_i(0) & \text{if } 1 \le i \le n \\ 0 & \text{if } n+1 \le i \le n+m \end{cases}$$

Intuitively, the above PCTMC will increase the counter for a transition by one whenever the transition is fired. Then, by computing the first moments of the population variables using Algorithm 4.1 for the PCTMC $\mathcal{P}'$, we can evaluate $N_{\tau_i} = \mathbb{E}[x_{n+i}](t_e)$ for $1 \le i \le m$, where t_e is the end time of the simulation.

5.2.2 Directed Coupling Graph

With the evaluation of direct coupling coefficients, we can construct a directed coupling graph for an arbitrary PCTMC. The definition of the directed coupling graph is given as follows:

Definition 3. *The directed coupling graph for a PCTMC with n population variables and m transitions is a graph consisting of $m+n$ nodes, in which each node represents a population variable or a transition in the PCTMC, and there exists a weighted directed edge from node i to node j if the direct coupling coefficient $c_{i,j} > 0$. In this case, $c_{i,j} > 0$ is the weight for the edge.*

For example, for a PCTMC consisting of four population variables (A,B,C,D), and four transitions $(\tau_1,\tau_2,\tau_3,\tau_4)$ as follows:

$$\begin{aligned} A &\longrightarrow_{\tau_1} D \quad \text{at rate } c_1 \times A \\ A+B &\longrightarrow_{\tau_2} 2A \quad \text{at rate } c_2 \times AB \\ B &\longrightarrow_{\tau_3} C \quad \text{at rate } c_3 \times B \\ C &\longrightarrow_{\tau_4} D \quad \text{at rate } c_4 \times C \end{aligned}$$

Assuming $N_{\tau_1} = 70, N_{\tau_2} = 30, N_{\tau_3} = 10, N_{\tau_4} = 30$, the corresponding directed coupling graph is given in Figure 5.1. As can be seen from the graph, there is an edge from node A to node τ_1 since $c_{A,\tau_1} = N_{\tau_1}/(N_{\tau_1}+N_{\tau_2}) = 0.7$, an edge from node τ_1 to node A since A appears in the reactant side and the rate function of transition τ_1. There is no edge from node τ_1 to node D since D only appears in the product side of transition τ_1, thus makes no direct contribution to it.

In the above example, if A is a target population, then removing transition τ_3 and τ_4 will not induce an immediate error on the evolution of A, since $c_{A,\tau_3} = c_{A,\tau_4} = 0$.

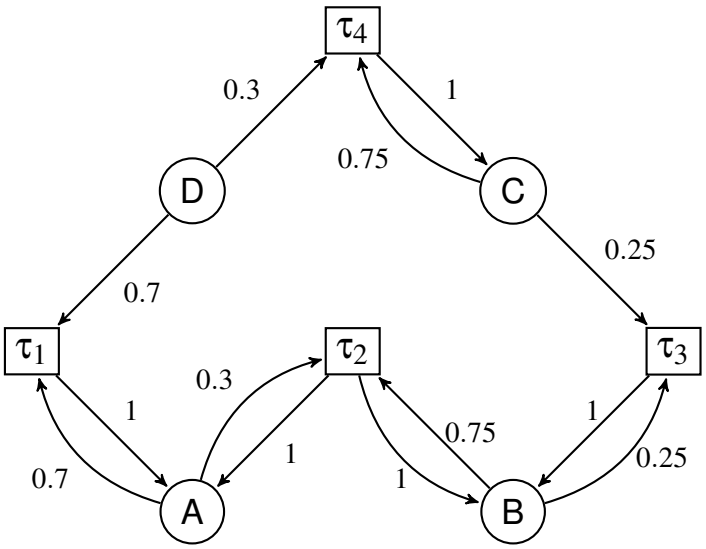

Figure 5.1: The directed coupling graph for the PCTMC with population variables (A,B,C,D), and transitions $(\tau_1,\tau_2,\tau_3,\tau_4)$. Weights on edges are the direct coupling coefficients.

But, from the model definition, we can clearly see that removing τ_4 will make zero impact on A either directly or indirectly, however, removing τ_3 can affect A through an indirect coupling with population variable B. This indirect coupling effect can be captured by a propagation method using the directed coupling graph.

5.2.3 Coupling Propagation

Transitions can influence the evolution of population variables by coupling propagation through intermediate populations and transitions. Specifically, for a target population x_t and a transition τ which are not directly connected in the directed coupling graph, we quantify the indirect coupling coefficient of the transition τ to the target population x_t by a path dependent coefficient $c^{\gamma}_{x_t,\tau}$, which is the product of the direct coupling coefficients along an acyclic path γ from node x_t to node τ in the directed coupling graph:

$$c^{\gamma}_{x_t,\tau} = \prod_{ij\in\gamma} c_{i,j} \tag{5.5}$$

Clearly, indirect coupling becomes weaker with more intermediate nodes.

For the purpose of model reduction, we assume that all coupling with respect to the target populations which are less than a small threshold ε can be ignored without causing significant error to the target populations. Then this is equivalent to charactering

the influence of removing an arbitrary transition τ on the evolution of a target population x_t by a coupling coefficient $C_{x_t,\tau}$, which is the maximum of the path dependent coefficients:

$$C_{x_t,\tau} = \begin{cases} \max\limits_{all\ paths\,\gamma} c^{\gamma}_{x_t,\tau}, & \text{if there exists a path from node } x_t \text{ to node } \tau \\ 0, & \text{otherwise} \end{cases}$$

Consequently, we can specify whether a transition is removable with respect to a target population by simply checking if $C_{x_t,\tau} < \varepsilon$.

In the example in Figure 5.1, if A is a target population, then we can obtain $C_{A,\tau_1} = 0.7$, $C_{A,\tau_2} = 0.3$, $C_{A,\tau_3} = 0.3 \times 1 \times 0.25 = 0.075$, $C_{A,\tau_4} = 0$. Therefore, if we set $\varepsilon = 0.01$, only τ_4 will be identified as removable. However, if we set $\varepsilon = 0.1$, then τ_3 will also be treated as removable. Thus, the reduction threshold ε can be thought of as a value to control the extent of model reduction. With a larger value of ε, more transitions will be treated as removable. As a result, a larger error on the target population will also be induced.

5.2.4 Generating Algorithm for Reduction Proposals

Given an arbitrary PCTMC, and a set of target populations $(x_{t_1}, x_{t_2}, \ldots, x_{t_n})$, a reduction proposal is associated with a reduction threshold ε. Specifically, for each target population x_{t_i}, we regard a transition τ as removable with respect to it if $C_{x_{t_i},\tau} < \varepsilon$. The generated reduction proposal discards transitions which are removable with respect to all target populations, and population variables which are not involved in any transitions except those removable transitions (a population variable is not involved in a transition if it does not appear either in the reactant side, product side or the rate function of the transition). The procedure for generating a reduction proposal is summarised in Algorithm 5.1.

Algorithm 5.1 Reduction Proposal Generation

Require: $\mathcal{P} = (\mathbf{X}, \mathcal{T}, \mathbf{X}_0)$, $(x_{t_1}, x_{t_2}, \ldots, x_{t_n})$, ε

Define $\mathcal{T}_{rm_i} = \emptyset \quad \forall i \in (1, 2, ..., n)$, $\mathbf{X}_{rm} = \emptyset$
for all τ in the transition set $\mathcal{T}$ of $\mathcal{P}$ **do**
  for all x_{t_i} in $(x_{t_1}, x_{t_2}, \ldots, x_{t_n})$ **do**
    if $C_{x_{t_i},\tau} < \varepsilon$ **then**
      add τ to $\mathcal{T}_{rm_i}$
    end if
  end for
end for
Let $\mathcal{T}_{rm} = \mathcal{T}_{rm_1} \cap \mathcal{T}_{rm_2} \cap \ldots \cap \mathcal{T}_{rm_n}$, $\hat{\mathcal{T}} = \mathcal{T} - \mathcal{T}_{rm}$
for all x_i in the population vector $\mathbf{X}$ of $\mathcal{P}$ **do**
  if x_i is not involved in any transition in $\hat{\mathcal{T}}$ **then**
    add x_i to $\mathbf{X}_{rm}$
  end if
end for
Let $\hat{\mathbf{X}} = \mathbf{X} - \mathbf{X}_{rm}$, $\hat{\mathbf{X}}_0$ be the initial state of $\hat{\mathbf{X}}$
return $\hat{\mathcal{P}} = (\hat{\mathbf{X}}, \hat{\mathcal{T}}, \hat{\mathbf{X}}_0)$

5.3 Error Control of Reduction Proposals

A reduction proposal specifies a reduced PCTMC which discards the most likely removable transitions and population variables from a stochastic simulation with respect to the given target populations according to a reduction threshold ε. However, there is no direct estimate of the amount of error which will be caused by the reduction.

Therefore, an error control step is needed to ensure the error caused by a reduction proposal is lower than an acceptable error threshold. Since the dynamics of a population variable in a single simulation run is random, it is more reasonable to evaluate the error caused by a reduction proposal based on the mean dynamics of target populations over infinite simulation runs. Again, this is achieved by computing the expectation of target populations at a random time point during simulations using the moment-closure approximation which is described in the previous chapter. Specifically, to evaluate the error caused by a reduction proposal, we compute the first moments of target populations using Algorithm 4.1 for the original PCTMC $\mathcal{P}$ and the reduced PCTMC $\hat{\mathcal{P}}$. In

both cases, we set $\overline{m} = 2$, $d = 1$ in which second order moments are used as correction terms to make the computed first moments more accurate.

Sampling the first moments of target populations at m time points evenly distributed along the simulation through moment closure approximation, we can evaluate the error caused by a reduction proposal on a target population x_{t_i} at a time point j by:

$$Error_{i,j} = \frac{|\mathbb{E}(x_{t_i}^j) - \mathbb{E}(\hat{x}_{t_i}^j)|}{\mathbb{E}(x_{t_i}^j)} \tag{5.6}$$

where $\mathbb{E}(x_{t_i}^j)$ and $\mathbb{E}(\hat{x}_{t_i}^j)$ are expectations of the target population x_{t_i} at time point j before and after reduction, respectively.

Then, we characterise the error caused by a reduction proposal by the maximum of the error samples for all target populations, which is defined as follows:

$$Error_{\hat{\mathcal{P}}} = \max_{i \in (1,2,\ldots,n), j \in (1,2,\ldots,m)} Error_{i,j} \tag{5.7}$$

Thus, we can treat a reduction proposal $\hat{\mathcal{P}}$ as acceptable if $Error_{\hat{\mathcal{P}}} < \theta$, where θ is the acceptable error threshold.

5.4 Searching for the Optimal Reduction Proposal

The optimal reduction proposal specifies a reduced PCTMC which has the smallest set of transitions that satisfies the acceptable error threshold on target populations. Since the size of the reduced set of transitions is a monotone function of the reduction threshold ε, the optimal reduction proposal can be found by searching the largest value of ε which generates a reduced PCTMC $\hat{\mathcal{P}}$ that satisfies $Error_{\hat{\mathcal{P}}} < \theta$. This is achieved by a modified binary search method illustrated in Algorithm 5.2. Specifically, steps $4-17$ implement the binary search for a reduction proposal which satisfies $\theta - \Delta\theta \leq Error_{\hat{\mathcal{P}}} \leq \theta$, where $[\theta - \Delta\theta, \theta]$ is a convergence interval for which we think the optimal reduction proposal is found. Step 4 defines the initial upper bound and lower bound of the reduction threshold, in which we conservatively set the initial upper bound to 1. However, in practice, it is more efficient to set $\bar{\varepsilon} = \theta$ in order to save some search attempts since the error caused by a reduced PCTMC with reduction threshold ε will almost always be larger than ε. Step 5 defines the termination condition for the binary search in which δ is a small value close to zero. If a reduction proposal whose error falls into the convergence interval can not be found, then the optimal reduction proposal should be the last found PCTMC $\hat{\mathcal{P}}$ which satisfies $Error_{\hat{\mathcal{P}}} < \theta - \Delta\theta$ (Step 18-20).

Otherwise, no reduction proposal can be found to satisfy $Error_{\hat{\mathcal{P}}} < \theta$, thus the original PCTMC is returned (Step 22). Using Algorithm 5.2, the optimal reduction proposal can always be found if it exists. Then, the associated PCTMC $\hat{\mathcal{P}}$ which contains the smallest set of transitions can be used in future simulation runs.

5.5 Evaluation

In this section, we use two examples to evaluate the usefulness of our automatic model reduction algorithm for the speed-up of stochastic simulation. The first one is the PCTMCs derived by the PALOMA bike-sharing model in Section 3.2. The second one is a PCTMC for the smart taxi scenario which is a simplified version of the model described in [Hillston and Loreti, 2015]. In the experiments for both examples, the usefulness of our reduction algorithm is evaluated by the size of the reduced model (the proportion of removed transitions), the decrease of simulation time, and the error caused by the reduction.

5.5.1 Experiments on the Bike-sharing Example

We first consider the PCTMCs derived by the bike-sharing model described in Section 3.2. Specifically, to make our experiments more thorough, we generated 50 bike-sharing models each with 30 locations. There are 50 pedestrians and a bike station which is equipped with 25 available bikes and 5 available slots initially in each location in the simulation. The topology of the locations and the value of other parameters in each model are generated randomly. This means we have generated 50 different PCTMCs for the bike-sharing example for experiments.

To achieve a fair comparison, we first simulate each PCTMC without reduction for 500 runs. Then, we simulate each PCTMC with our reduction algorithm with different acceptable error thresholds each for 500 runs. The end time of each simulation run for both cases is $t_e = 150$. Without loss of generality, we pick the number of available bikes in two random stations as our target populations in each simulation.

We first compare the size of the reduced PCTMC and the total simulation time cost using our reduction method (the overhead cost to run our reduction algorithm is included) with simulation without reduction. Figure 5.2 gives the average proportional reduction of transitions and simulation time with different acceptable error thresholds. It can be seen that our reduction algorithm can significantly reduce the number of

Algorithm 5.2 Searching for the optimal reduction proposal

Require: $\mathcal{P} = (\mathbf{X}, \mathcal{T}, \mathbf{X}_0)$, $(x_{t_1}, x_{t_2}, \ldots, x_{t_n})$, θ, $\Delta\theta$, δ

1: Evaluate the firing count of transitions during a simulation run through moment-closure approximation of a PCTMC $\mathcal{P}'$ with additional dummy population variables as counters of the firing of transitions
2: Construct the directed coupling graph for $\mathcal{P}$
3: Obtain the first moments of $(x_{t_1}, x_{t_2}, \ldots, x_{t_n})$ using the moment-closure approximation result of $\mathcal{P}'$
4: Define $\bar{\varepsilon} = 1$, $\underline{\varepsilon} = 0$
5: **while** $|\bar{\varepsilon} - \underline{\varepsilon}| > \delta$ **do**
6: Let $\varepsilon = (\bar{\varepsilon} + \underline{\varepsilon})/2$
7: Generate a reduced PCTMC $\hat{\mathcal{P}}$ with reduction threshold ε using Algorithm 5.1
8: Compute the first moments of $(\hat{x}_{t_1}, \hat{x}_{t_2}, \ldots, \hat{x}_{t_n})$ through moment-closure approximation of $\hat{\mathcal{P}}$
9: Compute $Error_{\hat{\mathcal{P}}}$
10: **if** $Error_{\hat{\mathcal{P}}} > \theta$ **then**
11: Let $\bar{\varepsilon} = \varepsilon$
12: **else if** $\theta - \Delta\theta \leq Error_{\hat{\mathcal{P}}} \leq \theta$ **then**
13: **return** $\hat{\mathcal{P}}$
14: **else if** $Error_{\hat{\mathcal{P}}} < \theta - \Delta\theta$ **then**
15: Let $\underline{\varepsilon} = \varepsilon$
16: **end if**
17: **end while**
18: **if** $\underline{\varepsilon} > 0$ **then**
19: Generate a reduced PCTMC $\hat{\mathcal{P}}$ with reduction threshold $\underline{\varepsilon}$ using Algorithm 5.1
20: **return** $\hat{\mathcal{P}}$
21: **else**
22: **return** $\mathcal{P}$
23: **end if**

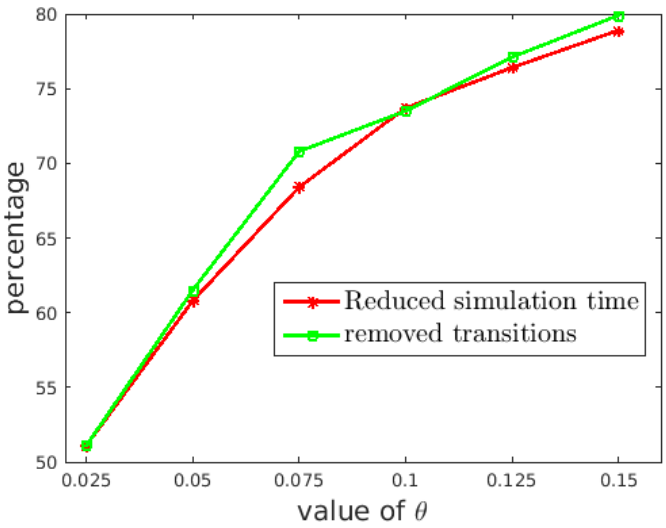

Figure 5.2: The proportional reduction of simulation time, number of transitions with different acceptable error thresholds in the experiments on the bike-sharing example

transitions as well as the simulation time even with a small error threshold. With larger error threshold, more transitions are removed and more simulation time is reduced. The overhead costs of our reduction algorithm in all the experiments are between 2.68 to 3.85 minutes, which are approximately the time cost of 9 to 14 simulation runs of a full PCTMC for the bike-sharing model. This means the overhead costs of our reduction algorithm is insignificant if a large number of simulation runs is required.

Furthermore, in order to measure the error caused by reduction, we evenly sample the mean value of target populations at 200 time points along each simulation. The error on a target population x_t at a time point i can be quantified by:

$$Error_{t,i} = \frac{|\,\overline{x^f_{t,i}} - \overline{x^r_{t,i}}\,|}{\overline{x^f_{t,i}}}$$

where $\overline{x^r_{t,i}}$ and $\overline{x^f_{t,i}}$ are the average value of target population x_t at time point i in the 500 simulation runs with and without reduction. If we treat each $Error_{t,i}$ where $t \in \{t_1, t_2\}$, $i \in (1, 2, \ldots, 200)$ as an error sample, then the average error caused by the reduction algorithm in our experiments can be measured by:

$$\overline{Error_{t_1,\ldots,t_n}} = \frac{\sum_{i=1}^{200}(Error_{t_1,i} + \ldots + Error_{t_n,i})}{n \times 200} \tag{5.8}$$

where $n = 2$ and x_{t_1}, x_{t_2} are the number of available bikes in the two chosen stations in this case. Table 5.1 gives the average error with 99% confidence interval caused by

Value of θ %	2.5	5	7.5	10	12.5	15
Mean Error %	0.66 ± 0.14	0.89 ± 0.17	1.02 ± 0.18	3.4 ± 0.66	8.7 ± 1.69	10.9 ± 2.44

Table 5.1: The average error (with 99% confidence interval) caused by reduction with different acceptable error thresholds in the experiments on the bike-sharing example.

reduction with different acceptable error thresholds. It can be seen the error caused by our reduction algorithm is well controlled since 99% of the error samples caused by our reduction algorithm are below the acceptable error threshold which we assign to the target populations.

5.5.2 Experiments on the Smart Taxi Example

The next example we consider is a PCTMC for the smart taxi scenario. Specifically, we model a system which consists of a number of taxis and users in a city that is divided to N regions in a grid topology. The users arrive in different regions at different rates. After arrival, a user makes a call for a taxi and then waits in that region until they are successfully taken by a taxi and move to another randomly chosen region. Unengaged taxis move about the city voluntarily or influenced by the calls made by users. For readability, the transitions of the PCTMC are given in the chemical reaction style as follows:

$$\emptyset \longrightarrow User\{\ell : i, dest : j, state : call\} \qquad \text{at } \lambda_i \cdot p^i_j \quad \forall i, j \in N \tag{5.9}$$

$$\begin{aligned} &User\{\ell : i, dest : j, state : call\} + Taxi\{\ell : k, state : unengaged\} \longrightarrow \\ &\qquad User\{\ell : i, dest : j, state : wait\} + Taxi\{\ell : k, pickup : i, state : engaged\} \\ &\text{at } \alpha^k_i \cdot \min\left(\#(User\{\ell : i, dest : j, state : call\}), \#(Taxi\{\ell : k, state : unengaged\})\right) \\ &\qquad \forall i, j, k \in N \wedge dist(i,k) < r \end{aligned} \tag{5.10}$$

$$\begin{aligned} &User\{\ell : i, dest : j, state : wait\} + Taxi\{\ell : k, pickup : i, state : engaged\} \longrightarrow \\ &\qquad Taxi\{\ell : i, dest : j, state : occupied\} \\ &\text{at } \beta^k_i \cdot \#(Taxi\{\ell : k, pickup : i, state : engaged\}) \quad \forall i, j, k \in N \end{aligned} \tag{5.11}$$

$$Taxi\{\ell : i, dest : j, state : occupied\} \longrightarrow Taxi\{\ell : j, state : unengaged\}$$
$$\text{at } \gamma^i_j \cdot \#(Taxi\{\ell : i, dest : j, state : occupied\}) \quad \forall i, j \in N \quad (5.12)$$
$$Taxi\{\ell : i, state : unengaged\} \longrightarrow Taxi\{\ell : j, state : unengaged\}$$
$$\text{at } \mu^i_j \cdot \#(Taxi\{\ell : i, state : unengaged\}) \quad \forall i, j \in N \wedge j \in nearby(i) \quad (5.13)$$

in which (5.9) represents a user arriving at Region i and calling for a taxi to Region j, where λ_i is the arrival rate of users in Region i, p^i_j is the probability that the destination is Region j if a user calls for a taxi in Region i; (5.10) represents an unengaged taxi in Region k accepting a call from region i (the distance between Region k and Region i must be less than r), where $\alpha^k_i \cdot \min\big(\#(User\{\ell : i, dest : j, state : call\}), \#(Taxi\{\ell : k, state : unengaged\})\big)$ is the rate at which a taxi in Region k responds to a call in region i if there are currently $\#(User\{\ell : i, dest : j, state : call\})$ users calling for a taxi from Region i to Region j, and $\#(Taxi\{\ell : k, state : unengaged\})$ unengaged taxis in Region k; (5.11) represents a user being picked up by a taxi, where $1/\beta^k_i$ is the expected time to pick up a user in Region i starting from Region k; (5.12) represents the taxi finishing its service from Region i to Region j, where $1/\gamma^i_j$ is the expected time for a journey from Region i to Region j; (5.13) represents an unengaged taxi moving to a nearby region voluntarily.

In the experiments, we set $N = 25$, and the city is divided into a 5×5 grid. Furthermore, we set $r = 1$ which means only taxis currently in the same region or adjacent regions can accept user calls from that region. 1000 taxis are randomly distributed across the city initially in the simulation. The arrival rates of users in all the regions are set to a value between 10 to 40. All other parameters are set to a value between 0 to 1. Moreover, we choose all the users in the *call* state in one region as our target populations in each experiment. In each experiment, we simulate the model with and without our reduction algorithm, both for 1000 runs. The end time of each simulation run is $t_e = 100$. Figure 5.3 shows the average proportional reduction of simulation time and transitions with different acceptable error thresholds in the experiments. Table 5.2 gives the associated mean error with 99% confidence interval caused by reduction with different acceptable error thresholds according to the same evaluation standard as was used in the bike-sharing example. The overhead costs of our reduction algorithm in this case are between 4.12 to 5.85 minutes, which are approximately the time cost of 10 to 15 simulation runs of the full PCTMC.

We can see that our algorithm even achieves a larger reduction on transitions and simulation time in this case, and the error caused by our reduction is still well con-

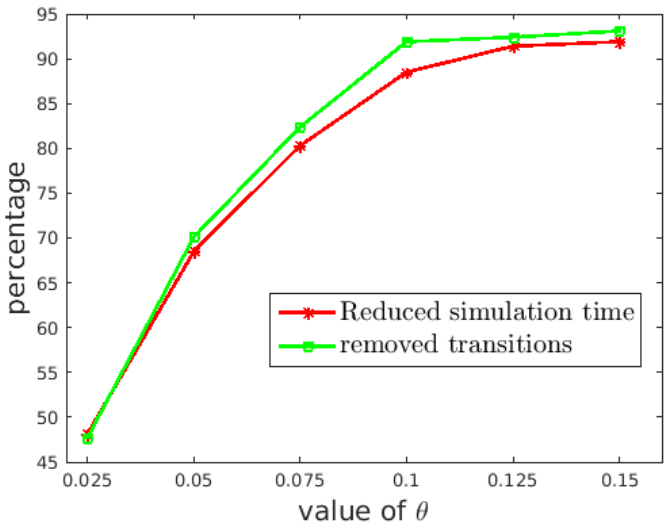

Figure 5.3: The proportional reduction of simulation time, number of transitions with different acceptable error thresholds in the experiments on the smart taxi example.

Value of θ %	2.5	5	7.5	10	12.5	15
Mean Error %	0.38 ± 0.1	3.82 ± 1.0	5.24 ± 1.4	5.70 ± 1.7	6.31 ± 1.8	6.55 ± 1.8

Table 5.2: The average error (with 99% confidence interval) caused by reduction with different acceptable error thresholds in the experiments on the smart taxi example.

trolled. Moreover, in both cases, with the increase of acceptance error threshold θ, we can observe that the proportional increase of reduced simulation time almost overlaps with the increase of reduced transitions, this also reflects that the cost of our reduction algorithm is almost negligible compared with the total simulation cost.

5.6 Summary

In this chapter, we have proposed an automatic model reduction algorithm which can significantly accelerate stochastic simulation of PCTMCs assuming that only the dynamics of a few target populations are required to be checked. The algorithm involves the following key steps:

- generating model reduction proposals by constructing the directed coupling graph for the PCTMC, and computing the coupling coefficients between the transitions and the target populations,

- efficiently evaluating the error caused by reduction proposals by moment-closure approximation of the PCTMC before and after reduction,
- obtaining the optimal reduction proposal which satisfies the acceptable error threshold by a binary search algorithm.

We demonstrated the usefulness of our algorithm by applying it to the stochastic simulation of two PCTMC models in the smart transport area. The result shows that the algorithm can achieve significant acceleration of stochastic simulation even with a small acceptable error threshold on the target populations. Lastly, although our algorithm is particularly useful for PCTMCs for CAS, we expect other general PCTMCs can also benefit from our algorithm as long as there are loosely coupled populations and transitions in them.

Chapter 6

Moment-based Availability Prediction for Bike-sharing Systems

In the previous chapters, an abstract bike-sharing model has been used as an example to illustrate our techniques for the modelling and analysis of CAS. By assuming the perfect knowledge of parameters, the model can provide us the chance to investigate all aspects of the behaviour of the system before it is actually put into operation. In this chapter, we will present a more realistic model to study bike-sharing systems from a data-driven prospective. Specifically, we will study the problem of future bike availability prediction of bike stations using a time-inhomogeneous PCTMC whose parameters are fitted using historical data. For the purpose of real time prediction, instead of computing the probability distribution of the number of available bikes in a station through stochastic simulation which is computationally expensive, we will derive the moments of the number of available bikes at a future time point by deterministic moment analysis. Then, the underlying probability distribution of the available number of bikes is reconstructed through the maximum entropy approach based on the derived moments. As a case study chapter, several model reduction techniques introduced in the earlier chapters are successfully applied to speed up the prediction process. Our model is parametrized using historical data from Santander Cycles[1], the bike-sharing system in London. In our experiments, we show that our model outperforms the time-inhomogeneous Markov queueing model on several performance metrics for bike availability prediction. This chapter is an extended version of the work published in a paper in QEST 2016 [Feng et al., 2016b].

[1] https://tfl.gov.uk/info-for/open-data-users/our-feeds?intcmp=3671#on-this-page-4

6.1 Introduction

In recent years, we have seen significant growth of bike-sharing programs all over the world [Fishman, 2016]. Public bike-sharing systems have been launched in many major cities such as London, Paris, and Vienna. Indeed, they have become an important part of urban transportation which provides improved connectivity to other modes of public transit. The concept of bike-sharing systems is rather simple: the system consists of a number of bike stations distributed over a geographic area (city). Each station is equipped with a limited number of bike slots in which public bikes can be parked. When users arrive at a station, they pick up a bike, use it for a while, and then return it to another station of their choice. With the increasing popularity of the smart transport theme, there has been great interest from the research community in the intelligent management of bike-sharing systems. Topics include, but are not limited to, policy design [Lin and Yang, 2011, Pfrommer et al., 2014], intelligent bike redistribution [Nair and Miller-Hooks, 2011, Contardo et al., 2012, Schuijbroek et al., 2013], and user journey planning [Yoon et al., 2012, Gast et al., 2015]. The focus of this chapter is on the probabilistic prediction of the number of available bikes in stations. Having a predictive model is of vital interest to both the user and the system administrator. The user can use it to identify likely origin/destination stations for which a trip can be successfully made. System administrators can use the model to undertake service level agreement checking, and plan bike redistribution for stations which are likely to break the service level requirement.

Specifically, in this chapter we present a novel moment-based prediction model that can provide probabilistic forecasts for the number of available bikes in a bike station. Since the modelling scenario for this case study is rather straightforward, we will directly use PCTMC as our modelling tool instead of PALOMA. Specifically, by representing the bike-sharing system as a PCTMC with time-dependent rates, our model is explanatory as the dynamics of the system are explicitly given. Gast *et al.* [Gast et al., 2015] show the benefits of predicting (*forecasting*) the entire probability distributions of possible bike availabilities in a station, compared with previous models that were only able to produce point estimates, often using time-series-based techniques [Froehlich et al., 2009, Kaltenbrunner et al., 2010, Yoon et al., 2012]. However, unlike [Gast et al., 2015], in which all the considered forecasting methods worked at the level of isolated stations, our model also captures the journey dynamics between stations. Guenther and Bradley [Guenther and Bradley, 2013] also provide a PCTMC

model with time-dependent rates for bike availability prediction, however there are several key differences between that model and ours. Firstly, our model provides the full probability distribution of the number of available bikes in a station which is much more informative from bike users' perspective than their model which only provides a point estimate (e.g., it is more likely that the users want to know the probability of a station being full or empty in a future time point instead of the expected number of bikes or slots in that station). Secondly, we use a model reduction method to prune our PCTMC such that the significant journey dynamics with respect to the target station are guaranteed to be preserved. However, their model aggregates stations which are spatially close, assuming that they have similar journey durations to the target station, which causes the information about the emptiness and fullness of stations to be lost.

We summarize the contribution of this chapter as follows. Firstly, as a real case study for the modelling of CAS, a novel PCTMC model with time-dependent rates is presented to successfully capture the bike-sharing scenario from a data-driven prospective. Secondly, we show that several model reduction methods introduced in earlier chapters can be successfully applied to speed up the deterministic moment analysis of the PCTMC. Finally, we reconstruct the underlying probability distribution of the number of available bikes in the target station using the maximum entropy principle based on a few moments generated from deterministic moment analysis of the PCTMC, and show that the model has better performance on a set of metrics for bike availability prediction compared with the Markov single-station queueing model.

The rest of this chapter is structured as follows. We briefly introduce the concepts of PCTMC with time-dependent rates in the next section. Section 6.3 gives the introduction of the Markov queueing model for bike availability prediction. In Section 6.4, we present our PCTMC model for the bike-sharing scenario. In the next section we show how to reconstruct the probability distribution of the number of available bikes using the maximum entropy approach. Section 6.6 presents the experimental results of our model on the London bike-sharing system compared with the Markov queueing model. Finally, Section 6.7 draws conclusions and discusses possible extensions of our model.

6.2 PCTMC with Time-dependent Rates

While PCTMCs can be used to model many CAS, it would be rather inaccurate to describe the bike-sharing system using a PCTMC exactly as introduced in Section 3.4.1,

since many parameters, such as pickup rates and destinations of bikes, vary with time. Hence, we present an extension to the PCTMC, in which we allow deterministic rate changes that occur at specific time points. This implies that any transition rate $r_\tau(\mathbf{X})$ is now time dependent, i.e. $r_\tau(\mathbf{X},t)$, where:

$$r_\tau(\mathbf{X},t) = \begin{cases} r_\tau(\mathbf{X},t_1) & \text{if } X_i \geq \underline{N_i} \forall i = 1,2,\ldots,n \wedge t < t_1 \\ r_\tau(\mathbf{X},t_2) & \text{if } X_i \geq \underline{N_i} \forall i = 1,2,\ldots,n \wedge t_1 \leq t < t_2 \\ \ldots & \\ r_\tau(\mathbf{X},t_m) & \text{if } X_i \geq \underline{N_i} \forall i = 1,2,\ldots,n \wedge t_{m-1} \leq t < t_m \\ 0 & \text{otherwise.} \end{cases} \tag{6.1}$$

in which $t_1,t_2,...,t_m$ are deterministic time points at which transition rate changes occur. Furthermore, with time-dependent rates, the evolution of the moments of the underlying population-level stochastic process of a PCTMC becomes a hybrid model, in which discrete jumps of rates at some specific points of the numerical simulation of the moment ODEs are also allowed:

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbb{E}[M(\mathbf{X}(t))] = \sum_{\tau \in \mathcal{T}} \mathbb{E}[(M(\mathbf{X}(t)+\mathbf{d}_\tau) - M(\mathbf{X}(t)))r_\tau(\mathbf{X},t)] \tag{6.2}$$

Numerical simulation of the derived moment ODEs might be slower for PCTMCs with time-dependent rates since discrete jumps of rates may increase the stiffness of the ODEs. However, analysing the model by solving the ODEs should be still much more efficient than stochastic simulation. The above equations will be used to derive moment ODEs for the PCTMCs for bike availability prediction in this chapter. Moreover, since there is no non-linear transition rates for the PCTMCs in this chapter, the derived moment ODEs can be directly solved without moment-closure approximation.

6.3 Markov Queueing Model

Before introducing our PCTMC model, we first give the Markov queueing model for bike stations which is going to serve as our comparator.

The most straightforward way to evaluate the behaviour of a station is to analyse it in isolation. In this case, a station can be modelled as a time-inhomogeneous Markov queue $M/M/1/k_i$, illustrated in Figure 6.1.

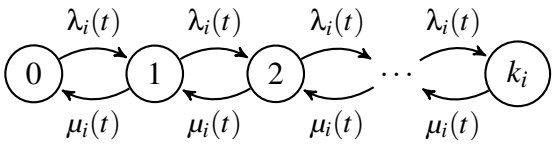

Figure 6.1: The time-inhomogeneous Markov queue for station i

Specifically, k_i denotes the capacity of a station i, $\lambda_i(t)$ and $\mu_i(t)$ are the time-dependent bike arrival and pickup rates of station i at time t of a day. Usually, the time of a day is split into n even slots, $[t_0,t_1),[t_1,t_2),\ldots,[t_{n-1},t_n)$. Then, both $\lambda_i(t)$ and $\mu_i(t)$ can be estimated based on $|D|$ days of observation (all days in D should be either weekdays or weekends since bike usage patterns are rather different during those days (cf. Figure 6.2)), for $t_{j-1} < t < t_j$:

$$\lambda_i(t) = \frac{\sum_{d\in D} \text{No. of bike arrivals at station } i \text{ in } (t_{j-1},t_j) \text{ on day } d}{\sum_{d\in D} \text{time length in } (t_{j-1},t_j) \text{ on day } d \text{ during which station } i \text{ is not full}}$$

$$\mu_i(t) = \frac{\sum_{d\in D} \text{No. of bike pickups at station } i \text{ in } (t_{j-1},t_j) \text{ on day } d}{\sum_{d\in D} \text{time length in } (t_{j-1},t_j) \text{ on day } d \text{ during which station } i \text{ is not empty}}$$

Furthermore, using the transition rate matrix for station i: $Q^i(t)$, where

$$Q^i(t) = \begin{pmatrix} -\mu_i(t) & \mu_i(t) & & & & \\ \lambda_i(t) & -\big(\mu_i(t)+\lambda_i(t)\big) & \mu_i(t) & & & \\ & \ddots & \ddots & \ddots & & \\ & & \lambda_i(t) & -\big(\mu_i(t)+\lambda_i(t)\big) & \mu_i(t) & \\ & & & \lambda_i(t) & -\lambda_i(t) & \end{pmatrix},$$

we can predict the probability that there are y bikes in station i at time $t+h$ given the station has x bikes at time t, by the following equation:

$$\Pr(y \mid x,t,h) = \exp\left(\int_0^h Q^i(t+s)\mathrm{d}s\right)_{x,y}$$

where $\exp(M)_{x,y}$ is the element at row x and column y of the matrix exponential of M. Such a model has been used to make bike availability or station inventory level predictions in several papers in the literature (e.g. [Schuijbroek et al., 2013, Raviv and Kolka, 2013, Gast et al., 2015]).

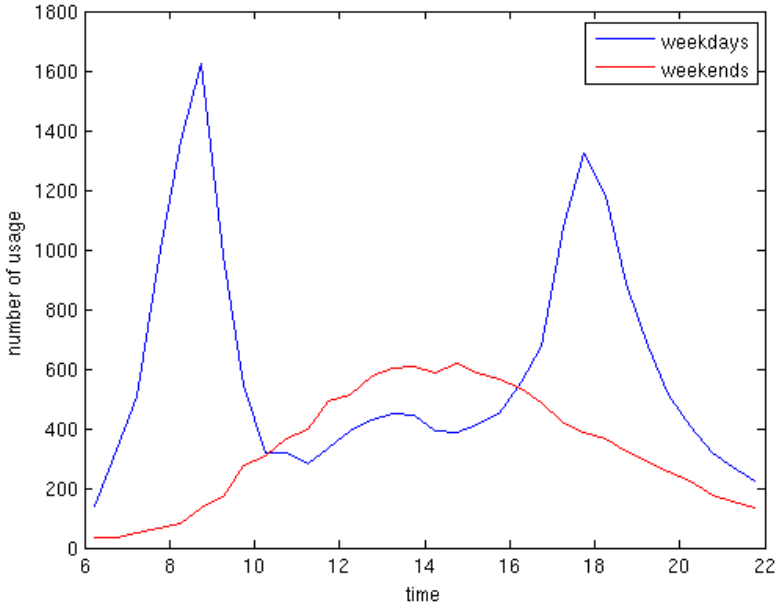

Figure 6.2: The number of bikes in use in 20 minute slots from 06:00 to 22:00 in Santander Cycles, London during weekdays and weekends.

Two assumptions are made in this model. First, the bike arrivals and pickups at stations form Poisson processes. Second, the state of a particular station does not depend on the state of the others. The first assumption is successfully validated for busy stations in [Gast et al., 2015], using historical data from the Velib bike-sharing system in Paris. However, we conjecture that the second assumption is generally not true in practice. For example, when a station is empty, no bikes can depart from it, therefore the arrival rate at other stations should be reduced. Hence, we seek a more realistic model, which captures the journey dynamics between stations.

6.4 PCTMC of Bike-sharing Model

6.4.1 A Naive PCTMC Model

To faithfully represent the journey dynamics between bike stations in a bike-sharing system with N stations, we first propose a naive PCTMC model which contains the

following transitions:

$$
\begin{array}{lll}
Bike_i \longrightarrow Slot_i + Journey^i_j@P_1 & \text{at } \mu_i(t)p^i_j(t) & \forall i,j \in (1,N) \\
Journey^i_j@P_l \longrightarrow Journey^i_j@P_{l+1} & \text{at } (P^i_j/d^i_j)\,\#(Journey^i_j@P_l) & \\
 & \qquad l \geq 1 \wedge l < P^i_j, & \forall i,j \in (1,N) \\
Journey^i_j@P_{P^i_j} + Slot_j \longrightarrow Bike_j & \text{at } (P^i_j/d^i_j)\,\#(Journey^i_j@P_{P^i_j}) & \forall i,j \in (1,N)
\end{array}
$$

where $Bike_i$, $Slot_i$ represent a bike and a slot agent in station i respectively; $Journey^i_j@P_l$ represents a bike agent which is currently on a journey from station i to station j at phase l. Note that since journey durations are generally not exponentially distributed, we fit the journey duration from station i to station j as a phase-type distribution with P^i_j identical phases each with rate P^i_j/d^i_j, where d^i_j is the mean journey duration. $\mu_i(t)$ is the fitted bike pickup rate governed by an exponential distribution in station i at time t, p^i_j is the probability that a journey will end at station j given that it started from station i at time t. $\#(S)$ denotes the population of an agent type S.

Obviously, the above model is not scalable. Since the total number of bike stations N is usually very large (for example there are around 750 bike stations in London), it is computationally infeasible to analyse a model which captures the full set of bike stations. Fortunately, since we are only interested in the prediction of bike availability of a single target station at a time, we only need to model stations which have a significant contribution to the journey flows to the target station (knowing the state of a station which has a very small contribution to the journey flows to the target station will have negligible impact on the accuracy of bike availability prediction for the target station). Thus, a directed contribution graph together with a contribution propagation method which is similar to the reduction method introduced in the previous chapter is proposed to automatically identify the set of stations which need to be modelled with respect to a given target station for bike availability prediction.

6.4.2 Directed Contribution Graph with Contribution Propagation

Here, we show how to derive a set of bike stations $\Theta(v)$ in which all stations have a significant contribution to the journey flows to a given target station $v \in (1,2,\ldots,N)$ for bike availability prediction. Concretely, we first need a way to quantify the contribution of one station to the journey flows to another station. Specifically, we let C_{ij} denote the *contribution coefficient* of station j to station i which quantifies the contribution of station j to the journey flows to station i.

One station can contribute to the journey flows to another station both directly and indirectly. The definition of a *direct contribution coefficient* at time t is given by the following simple formula:

$$c_{ij}(t) = \lambda_i^j(t)/\lambda_i(t)$$

in which $\lambda_i^j(t)$ represents the bike arrival rate from station j to station i at time t and $\lambda_i(t) = \sum_j \lambda_i^j(t)$. Then, it is clear that $c_{ij}(t) \in [0,1]$, $0 \leq \sum_{j \neq i} c_{ij}(t) \leq 1$.

With the definition of directed contribution coefficient, we can construct a directed contribution graph for the bike-sharing system at each time slot of a day. The definition of the directed contribution graph is given as follows (for convenience, we abbreviate $c_{ij}(t)$ to c_{ij}):

Definition 4. *For an arbitrary time t, the directed contribution graph for a bike-sharing system at time t is a graph in which nodes represent the stations in the system, and there is a weighted directed edge from node i to node j if $c_{ij} > 0$, and in this case the weight of the edge is c_{ij}. Thus, the direction of edges is the inverse of contribution flows.*

Figure 6.3 shows a sample directed contribution graph which consists of six bike stations only for illustration purpose (in a real case, the graph will be more connected).

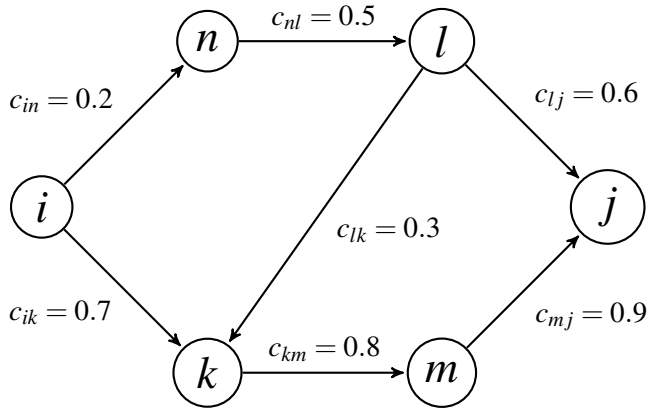

Figure 6.3: An example directed contribution graph with six stations

For those stations which are not directly connected in the directed relation graph, by using a contribution propagation method, we can evaluate the *indirect contribution coefficient* of one station on the journey flows to another station. Specifically, the indirect contribution coefficient is quantified by a path dependent coefficient $c_{ij,\gamma}$, which

is the product of the direct contribution coefficients along an acyclic path γ from node i to node j. Then, the contribution coefficient of station j to station i is characterized by the maximum of the path dependent coefficients:

$$c_{ij,\gamma} = \prod_{kl \in \gamma} c_{kl}$$

$$C_{ij} = \begin{cases} \max\limits_{all\ paths\ \gamma} c_{ij,\gamma} & \text{if there exists a path from node } i \text{ to node } j \\ 0, & \text{otherwise} \end{cases}$$

For example, according to Figure 6.3, the contribution coefficient of station j to station i is $C_{ij} = c_{ik} \times c_{km} \times c_{mj} = 0.504$, since $c_{ik} \times c_{km} \times c_{mj} > c_{in} \times c_{nl} \times c_{lj} > c_{in} \times c_{nl} \times c_{lk} \times c_{km} \times c_{mj}$.

With the contribution coefficient, given a target station v, then for $i \in (1,2,\ldots,N)$, we can infer:

$$i \in \Theta(v) \qquad \text{if } C_{vi} > \theta$$

$$i \notin \Theta(v) \qquad \text{if } C_{vi} \leq \theta$$

where $\theta \in (0,1)$ is threshold value which can be used to control the extent of model reduction. A point to note is that we choose to characterize contribution coefficients by the maximum instead of the sum of path dependent coefficients because we only want to model stations which have at least a significant (direct or indirect) journey flow to the target station. To model stations which have many small journey flows to the target station is costly but the impact is rather unpredictable. Moreover, the maximum of path dependent coefficients has another nice property that if $i \in \Theta(v)$ and $C_{vi} = c_{vi,\gamma}$, then for a station j which is on the path γ, it is certain that $C_{vj} > \theta$, thus $j \in \Theta(v)$. As a result, for all stations which have a significant journey flow to the target station, that journey flow will certainly be captured in the resulting reduced PCTMC. However, this property will not be preserved if we use the sum of path dependent coefficients. For example in Figure 6.3, if we set $\theta = 0.55$ and use the sum instead of the product of path dependent coefficients to characterise contribution coefficients, we get $C_{ij} = \sum_\gamma c_{ij,\gamma} > 0.55$, thus station j is included in the reduced PCTMC. However, since $\sum_\gamma c_{il,\gamma} < 0.55$, station l will not be included in the reduced PCTMC. As a result, $\sum_\gamma c_{ij,\gamma} > 0.55$ will not actually be satisfied in the reduced PCTMC after station l is excluded.

As an illustration of the extent of model reduction, Figure 6.4 shows the empirical cumulative distribution function of contribution coefficients during all time slots between any two bike stations in Santander Cycles (which is computed by historical

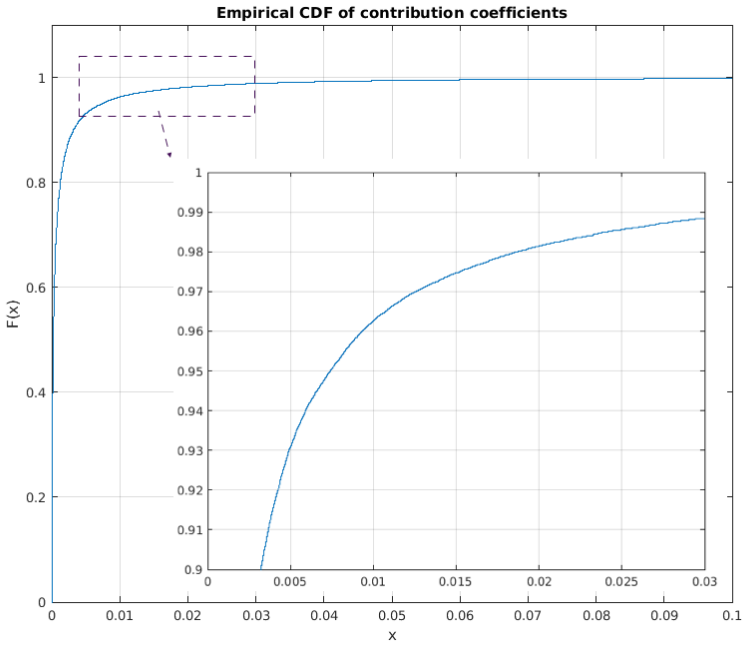

Figure 6.4: The empirical cumulative distribution function of contribution coefficients (x is the value of contribution coefficients)

journey data with 20 minutes slot duration). It can be seen that more than 96% of the computed contribution coefficients are smaller than 0.01. This means that on average more than 96% stations can be excluded even if θ is set to a small value 0.01 for the PCTMC of a random station in Santander Cycles.

6.4.3 The Reduced PCTMC Model

Given a target station v and current time t, suppose we are interested in the number of bikes at the station at time $t+h$, then let $s = (s_1, s_2, \ldots, s_n)$ be the minimal set of time slots which cover $[t, t+h]$, we obtain $\Theta(v) = \Theta(v, s_1) \cup \Theta(v, s_2) \cup \ldots \cup \Theta(v, s_n) \cup v$, where $\Theta(v, s_i)$ is the set of bike stations which have significant contribution to the journey flows to the target station within time slot s_i.

Therefore, the PCTMC for the prediction of bike availability at station v at time

$t+h$ can be represented as follows:

$$Bike_i \longrightarrow Slot_i \qquad \text{at } \mu_i(t)\Big(1-\sum_{j\notin\Theta(v)\vee c_{ji}\leq\theta} p^i_j(t)\Big) \qquad \forall i\in\Theta(v) \tag{6.3}$$

$$Slot_i \longrightarrow Bike_i \qquad \text{at } \sum_{j\notin\Theta(v)\vee c_{ij}\leq\theta} \lambda^j_i(t) \qquad \forall i\in\Theta(v) \tag{6.4}$$

$$Bike_i \longrightarrow Slot_i + Journey^i_j@P_1 \qquad \text{at } \mu_i(t)p^i_j(t) \qquad \forall i,j\in\Theta(v)\wedge c_{ji}>\theta \tag{6.5}$$

$$Journey^i_j@P_l \longrightarrow Journey^i_j@P_{l+1} \qquad \text{at } (P^i_j/d^i_j)\,\#(Journey^i_j@P_l) \quad l\geq 1\wedge l<P^i_j, \forall i,j\in\Theta(v)\wedge c_{ji}>\theta \tag{6.6}$$

$$Slot_j + Journey^i_j@P_{P^i_j} \longrightarrow Bike_j \qquad \text{at } (P^i_j/d^i_j)\,\#(Journey^i_j@P_{P^i_j}) \quad \forall i,j\in\Theta(v)\wedge c_{ji}>\theta \tag{6.7}$$

$$Journey^i_j@P_{P^i_j} \longrightarrow \varnothing \qquad \text{at } \mathbf{1}\big(Slot_j(t)=0\big)(P^i_j/d^i_j)\,\#(Journey^i_j@P_{P^i_j}) \quad \forall i,j\in\Theta(v)\wedge c_{ji}>\theta \tag{6.8}$$

where (6.3) represents a bike in station i is picked for a journey to a station outside $\Theta(v)$ or a station to which the journey flow is negligible (the direct contribution coefficient $c_{ji}\leq\theta$ indicates that journey flow from i to j must not be a significant journey flow); (6.4) represents a bike is returned to station i from a station outside $\Theta(v)$ or a station from which the journey flow is negligible; (6.5) represents a bike in station i is picked for a journey to a station j inside $\Theta(v)$ and the journey flow is significant; (6.6), (6.7) represent progress and completion of the journey, respectively; (6.8) assumes a bike in transit from station i to station j will be returned to another station outside $\Theta(v)$ when there is no empty slot in station j, where $\mathbf{1}(Slot_j(t)=0)$ is an indicator function which returns 1 when the number of empty slots at station j at time t is zero, otherwise returns 0.

6.4.3.1 Dealing with Indicator Function

Since we are going to numerically solve the PCTMC using moment ODEs as illustrated in Equation 6.2, we can only access the moments of the number of empty slots at a station i at time t, denoted as u^m_i, during numerical simulation (here we let u^m_i denote $\mathbb{E}[\big(Slot_i(t)\big)^m]$, where m is the order of the moment), whereas the number of empty slots at station i at time t is a random variable. Thus, we propose a method to approximate the indicator function $\mathbf{1}(Slot_i(t)=0)$ by a function of the moments of the number of empty slots and the capacity of the station: $\mathbf{1}(Slot_i(t)=0)\sim f(u^1_i,u^2_i,\ldots,u^m_i,k_i)$. Concretely, given the first m moments of the random variable $Slot_i(t)$, and the value domain $Slot_i(t)\in[0,1,\ldots,k_i]$, we can approximate the probability distribution of $Slot_i(t)$

by a discrete distribution with finite support k_i. For example, if we only know the first moment of $Slot_i(t)$ (which is u_i^1), we can fit a binomial distribution $Slot_i(t) \sim Binomial(k_i, u_i^1/k_i)$ to the probability distribution of $Slot_i(t)$. In this case, we get $\mathrm{P}(Slot_i(t)=0) = (1-u_i^1/k_i)^{k_i}$. Furthermore, if we know the first two moments (u_i^1, u_i^2), then we can fit a beta-binomial distribution $Slot_i(t) \sim BetaBinomial(k_i, \alpha, \beta)$, where

$$\alpha = \frac{u_i^1 u_i^2 - k_i (u_i^1)^2}{k_i (u_i^1)^2 + k_i u_i^1 - k_i u_i^2 - (u_i^1)^2} \qquad \beta = \frac{(k_i - u_i^1)(k_i u_i^1 - u_i^2)}{k_i (u_i^1)^2 + k_i u_i^1 - k_i u_i^2 - (u_i^1)^2}$$

Thus, we get

$$\mathrm{P}(Slot_i(t)=0) = \frac{B(\alpha, k_i+\beta)}{B(\alpha,\beta)}$$

where $B(a,b)$ is a beta function. Theoretically, with knowledge of more moments of $Slot_i(t)$, the estimation of $\mathrm{P}(Slot_i(t)=0)$ will be more accurate. Finally, we let

$$\mathbf{1}(Slot_i(t)=0) = \begin{cases} 1 & \text{if } \mathrm{P}(Slot_i(t)=0) > p \\ 0 & \text{if } \mathrm{P}(Slot_i(t)=0) \le p \end{cases}$$

where $\mathrm{P}(Slot_i(t)=0) = f(u_i^1, u_i^2, \ldots, u_i^m, k_i)$, p is a threshold value above which we believe the number of available slots in station i is zero. In general, p should be set to a value close to 1 in order to make sure the station is only treated as full when there is no available slot with a high confidence. In our later experiments, we explicitly set $p = 0.9$.

6.4.3.2 Specifying the initial state

Given a snapshot of the bike-sharing system at a time instant t which contains the following information[2]:

$$Bike_i(t), \ldots, Slot_i(t), \ldots, Journey^i(t, \Delta t), \ldots$$

where $Bike_i(t)$ and $Slot_i(t)$ are the current number of available bikes and empty slots at a station i; $Journey^i(t, \Delta t)$ represents there is a bike currently en route from station i, and the journey started at time $t - \Delta t$. Then, for each $Journey^i(t, \Delta t)$, we use a random number to determine the destination of the journey, and the time Δt to determine the appropriate phase of the journey time. Thus we generate a random number α uniformly distributed in $(0,1)$, and let $p_k^i(t-\Delta t), \forall k$ be the probability that the journey will end at station k given that the journey started from station i at time $t - \Delta t$. Then

$$Journey^i(t, \Delta t) = Journey_j^i(t, \Delta t) \quad \text{if } \alpha \ge \sum_{k=0}^{j-1} p_k^i(t-\Delta t) \wedge \alpha < \sum_{k=0}^{j} p_k^i(t-\Delta t).$$

[2]This information is actually recorded for the London bike-sharing system

Furthermore, we let

$$Journey^i_j(t,\Delta t) = Journey^i_j@P_l \quad \text{if } \Delta t \geq (l-1)d^i_j/P^i_j \wedge \Delta t < l \times d^i_j/P^i_j,$$

where $l \leq P^i_j$. Otherwise, if $l > P^i_j$, we let $Journey^i_j(t,\Delta t) = Journey^i_j@P_{P^i_j}$.

6.4.3.3 Solving the moment ODEs

We derive the moment ODEs following Equation 6.2 for the above PCTMC for the first m order of moments. Furthermore, using the moment ODE reduction method in Section 4.2, we can make a further reduction to the size of the moment ODEs by utilizing the neighbourhood relation between population variables in the above PCTMC. Specifically, we set reduction threshold $d = 1$ which means all population variables which are not directly involved in any transition will be treated as independent of each other. The derived moment ODEs can be solved by numerical simulation using standard methods.

6.5 Reconstructing the Probability Distribution Using the Maximum Entropy Approach

From the moment analysis of the PCTMC for the bike-sharing model, we gain the first m moments of the number of available bikes in the target station at the prediction time $t+h$, i.e. $\Big(\big(Bike_v(t+h)\big)^1, \big(Bike_v(t+h)\big)^2, \ldots, \big(Bike_v(t+h)\big)^m\Big)$, which we denote as $(u^1, u^2, \ldots, u^m)$ in the following. Our goal is to predict the probability that the station has a specific number of bikes at time $t+h$. This means the problem is to reveal $\mathrm{P}\big(Bike_v(t+h) = i \mid u^1, u^2, \ldots, u^m, k_v\big)$, where $i \in (1,2,\ldots,k_v)$. Therefore, we need to reconstruct the entire probability distribution of the random variable $Bike_v(t+h)$ based on its first m moments. The corresponding distribution is generally not uniquely determined. Hence, to select a particular distribution, we apply the maximum entropy principle to minimize the amount of bias in the reconstruction process. In this way, we assume the least amount of prior information about the true distribution. Note that the maximum entropy approach has been successfully applied to reconstruct distributions based on moments in many areas, e.g. physics [Mead and Papanicolaou, 1984], stochastic chemical kinetics [Andreychenko et al., 2015], and performance analysis [Tari et al., 2005].

6.5.1 Reconstruction Algorithm

Let X_v denote $Bike_v(t+h)$ for convenience, $\mathcal{G}$ be the set of all possible probability distributions for X_v. Then, based on the maximum entropy principle, the goal is to select a distribution g to maximize the entropy $H(g)$ over all distributions in $\mathcal{G}$. The problem can be denoted as follows:

$$\arg\max_{g\in\mathcal{G}} H(g) = \arg\max_{g\in\mathcal{G}} \big(-\sum_{x=0}^{k_v} g(x)\ln g(x)\big)$$

Furthermore, given $(u^1, u^2, \ldots, u^m)$, we know the following constraints should be satisfied:

$$\sum_{x=0}^{k_v} x^n g(x) = u^n, \quad n = 0, 1, \ldots, m$$

where $u^0 = 1$ to ensure that g is a probability distribution. Now, the problem becomes a constrained optimization program. Thus to perform the constrained maximization of the entropy, we introduce one Lagrange multiplier λ_n per moment constraint. We thus seek extrema of the Lagrangian functional:

$$L(g,\lambda) = -\sum_{x=0}^{k_v} g(x)\ln g(x) - \sum_{n=0}^{m} \lambda_n \big(\sum_{x=0}^{k_v} x^n g(x) - u^n \big)$$

Functional variation with respect to the unknown distribution function $g(x)$ yields:

$$\frac{\partial L}{\partial g(x)} = 0 \implies g(x) = \exp\left(-1 - \lambda_0 - \sum_{n=1}^{m} \lambda_n x^n \right)$$

Since $u^0 = 1$, we get

$$\sum_{x=0}^{k_v} \exp\left(-1 - \lambda_0 - \sum_{n=1}^{m} \lambda_n x^n \right) = 1.$$

Thus we can express λ_0 in terms of the remaining Lagrange multipliers

$$e^{1+\lambda_0} = \sum_{x=0}^{k_v} \exp\left(-\sum_{n=1}^{m} \lambda_n x^n \right) \equiv Z$$

Then, the general form of $g(x)$ can be given as follows:

$$g(x) = \frac{1}{Z} \exp\left(-\sum_{n=1}^{m} \lambda_n x^n \right)$$

Inserting the preceding equation into the Lagrangian, we can then transform the problem into an unconstrained minimization problem of the following function with respect to variables $\lambda_1, \lambda_2, \ldots, \lambda_n$:

$$\Gamma(\lambda_1, \lambda_2, \ldots, \lambda_n) = \ln Z + \sum_{n=1}^{m} \lambda_n u^n$$

The convexity of the function Γ is proved in [Mead and Papanicolaou, 1984], which guarantees the existence of a unique solution. Thus, a close approximation $(\lambda_1^*, \lambda_2^*, \ldots, \lambda_n^*)$ of the true solution can be obtained by the classic gradient descent approach [Snyman, 2005]. After finding $(\lambda_1^*, \lambda_2^*, \ldots, \lambda_n^*)$ through gradient descent, we can finally predict

$$\mathrm{P}\big(X_v = x\big) = \frac{\exp\left(-\sum_{n=1}^{m} \lambda_n^* x^n\right)}{\sum_{i=0}^{k_v} \exp\left(-\sum_{n=1}^{m} \lambda_n^* i^n\right)}, \quad \forall x \in (1, 2, \ldots, k_v)$$

6.6 Experiments

In this section, we test the time cost and accuracy of our prediction model in different cases and compare the accuracy of our model with the Markov queueing model. We use the historic journey data and bike availability data from January 2015 to March 2015 from the London Santander Cycles Hire scheme to train our PCTMC model as well as the Markov queueing model, and the data in April 2015 to test their prediction accuracy. As in [Guenther and Bradley, 2013], we fit the number of journey phases between stations using the HyperStar tool [Reinecke et al., 2012] command line interface. Specifically, we set the maximum value of P_j^i to 20 to make our model compact and also to avoid overfitting. Moreover, for parameter estimation, we split a day into slots of 20 minute duration. In our experiments, given the bike availability in a station at time t, we predict the probability distribution of the number of available bikes in that station at time $t+h$, where h is set to 10 minutes for short range prediction and 40 minutes for long range prediction.

The evaluation of our model is twofold. The first is accuracy, the second is efficiency. These two aspects are both influenced by the value of two important parameters, namely m, the highest order of moments being derived, and θ, the coefficient threshold for the identification of bike stations which have significant contribution to the journey flow to the target station. For higher values of m, the solution cost of our model becomes larger since more moment ODEs are derived, however the model should become more accurate due to more constraints in the probability distribution reconstruction based on the maximum entropy principle. For higher values of θ, more

	10min	40min	
Markov queueing model	1.52	3.03	
PCTMC with $\theta = 0.03$	1.49	2.81	$m = 1,2,3$
PCTMC with $\theta = 0.02$	1.49	2.81	$m = 1,2,3$
PCTMC with $\theta = 0.01$	1.48	2.79	$m = 1,2,3$

Table 6.1: The calculated RMSE on the prediction of the number of available bikes

stations are excluded in the reduced PCTMC for a target station whereas the model accuracy can be potentially reduced. Thus, to observe the effects on these two parameters, we do experiments with values $m = 1,2,3$, $\theta = 0.01, 0.02, 0.03$.

6.6.1 Root Mean Square Error

For prediction accuracy, we first consider the classic criterion based on root mean square error (RMSE), a commonly used metric for evaluating point predictions (i.e., predictions that only state the expected number of bikes). Specifically, given a vector $\mathbf{x}$ of predictions and $\mathbf{y}$ of observations, with A the set of prediction/observation pairs, the RMSE is defined as:

$$\sqrt{\frac{1}{|A|}\sum_{i \in A}(x_i - y_i)^2}$$

Table 6.1 compares the RMSE of the prediction results of our PCTMC model with the Markov queueing model. As can be seen, the PCTMC model outperforms the Markov queueing model in both prediction ranges. Especially in the long range, a considerable improvement is observed. For the PCTMC models, smaller values of θ only reduce the RMSE slightly. This means capturing less significant journey flows will have little impact on the prediction accuracy. Moreover, we find that the derived highest moments have almost no impact on the RMSE. This is obvious since the expected number of available bikes is only decided by the first moment.

6.6.2 Probability of Making a Correct Recommendation

Predicting the expected number of available bikes is important for system administrators when they want to decide how to redistribute bikes in the system. However, a user is interested in whether there is a bike in the target station when she wants to pick up

a bike from there, or whether there is a free slot in the target station when she wants to return a bike to that station. We are specifically interested in being able to make correct recommendations for the queries "Will there be a bike?" and "Will there be a slot?"[3] to measure the accuracy of our model. Specifically, for the "Will there be a bike?" query, we respond "Yes" if the predicted probability of that station having more than one bike is greater than 0.8, and respond "No" if the predicted probability of that station having more than one bike is less than 0.8. As is argued in [Gast et al., 2015], the root mean square error is not an appropriate evaluation metric in this setting. After all, we need a prediction of the probability of the recommendation being correct rather than just a point estimate of the number of available bikes/slots. Instead, a suitable evaluation scheme is proposed in [Gast et al., 2015] that ensures that the best prediction algorithm can always be expected to obtain the highest score. Such a scheme is called a *proper scoring rule*. Specifically, for the "Will there be a bike?" query, the following scoring rule is proper:

$$\text{Score} = \begin{cases} 1 & \text{if } \mathrm{P}(X_v > 0) > 0.8 \wedge x_v > 0 \\ -4 & \text{if } \mathrm{P}(X_v > 0) > 0.8 \wedge x_v = 0 \\ 1 & \text{if } \mathrm{P}(X_v > 0) < 0.8 \wedge x_v = 0 \\ -\frac{1}{4} & \text{if } \mathrm{P}(X_v > 0) < 0.8 \wedge x_v > 0 \end{cases}$$

Note that incorrect predictions need to be penalised by a negative score for the rule to be proper. The evaluation of recommendations to the "Will there be a slot?" query follows a similar pattern. Table 6.2 and 6.3 show the experimental results for different models and parameters. Note that the PCTMC model with $m = 1$ is excluded since at least two moments are needed to make a meaningful reconstruction of the probability distribution. As can be seen from the tables, the PCTMC model clearly has a better performance in making such recommendations. Moreover, we also observe that with higher values of m, the average score increases. This is because, with higher values of m, the reconstructed probability distribution is closer to the true distribution.

6.6.3 Time Cost

The time cost of making a prediction is also important. Table 6.4 shows the time cost for making a prediction using our PCTMC model with different parameters (we do not show the time costs for the Markov queueing model since they are negligible due to

[3] These queries can be readily extended to "Will there be n bikes?" and "Will there be n slots?"

	10min	40min	
Markov queueing model	0.90 ± 0.05	0.87 ± 0.06	
PCTMC with $\theta = 0.03$	0.91 ± 0.04	0.89 ± 0.05	$m = 2$
	0.92 ± 0.04	0.91 ± 0.04	$m = 3$
PCTMC with $\theta = 0.02$	0.91 ± 0.04	0.89 ± 0.05	$m = 2$
	0.92 ± 0.04	0.91 ± 0.04	$m = 3$
PCTMC with $\theta = 0.01$	0.92 ± 0.04	0.89 ± 0.05	$m = 2$
	0.93 ± 0.04	0.91 ± 0.04	$m = 3$

Table 6.2: Average score of making a recommendation to the "Will there be a bike?" query with 95% confidence interval

its small state space because of independence assumption). For real time application, we assume that the time cost of making a prediction must be less than one second. Thus, for point prediction, we recommend to set $\theta = 0.01, m = 1$ for both prediction ranges. For probability distribution prediction, we recommend to set $\theta = 0.02, m = 2$ for short range prediction, $\theta = 0.03, m = 2$ for long range prediction. Note that we used an Intel CORE i7 laptop with 8GB RAM to run our experiments, the time cost could be considerably reduced if a more powerful machine, e.g. a server, were used.

6.7 Conclusions

We have presented a moment-based approach to make predictions of availability in bike-sharing systems. The moments of the number of available bikes are automatically derived via a PCTMC with time-inhomogeneous rates, fitted from historical data. The entire probability distribution is reconstructed using a maximum entropy approach. Our model is easy to understand since it explicitly captures the dynamics of the bike-sharing system. We demonstrated that it outperforms the Markov queueing model in several performance metrics for prediction accuracy. Moreover we have also shown that by using the directed contribution graph and the moment ODE reduction method, the model size can be significantly reduced to such an extent that it is suitable for real time application.

In future work we plan to explore the impact of neighbouring stations, and extend

	10min	40min	
Markov queueing model	0.91 ± 0.04	0.88 ± 0.05	
PCTMC with $\theta = 0.03$	0.91 ± 0.04	0.90 ± 0.05	$m = 2$
	0.92 ± 0.04	0.91 ± 0.04	$m = 3$
PCTMC with $\theta = 0.02$	0.91 ± 0.04	0.90 ± 0.05	$m = 2$
	0.92 ± 0.04	0.91 ± 0.04	$m = 3$
PCTMC with $\theta = 0.01$	0.92 ± 0.04	0.91 ± 0.05	$m = 2$
	0.93 ± 0.04	0.92 ± 0.04	$m = 3$

Table 6.3: Average score of making a recommendation to the "Will there be a slot?" query with 95% confidence interval

our model to capture their effects. For example, if a station is empty, then the user is likely to pick up a bike from a neighbouring station, thus increasing the pickup rate at the neighbouring station. Conversely, if a station is full, then the user is likely to return a bike to a neighbouring station, increasing the bike arrival rate there. We think another merit of our PCTMC model is that it can be easily extended to capture such impact by using the indicator function to check whether a neighbouring station is empty or full in order to alter the bike arrival and pickup rate of a station. Unfortunately we do not currently have data to capture the impact of neighbouring stations.

	10min	40min	
PCTMC with $\theta = 0.03$	1.76 ± 0.2ms	6.98 ± 0.77ms	$m = 1$
	103 ± 13.7ms	328 ± 43ms	$m = 2$
	2.2 ± 0.2sec	8.9 ± 0.83sec	$m = 3$
PCTMC with $\theta = 0.02$	4.25 ± 0.4ms	15.72 ± 1.42ms	$m = 1$
	251 ± 25.5ms	1.1 ± 0.1sec	$m = 2$
	8.9 ± 1.2sec	37 ± 3.5sec	$m = 3$
PCTMC with $\theta = 0.01$	13.5 ± 0.9ms	49.1 ± 3.92ms	$m = 1$
	8.8 ± 1.1sec	30.1 ± 0.31sec	$m = 2$
	33.9 ± 5.4sec	157 ± 17.8sec	$m = 3$

Table 6.4: Time cost to make a prediction with 95% confidence interval

Chapter 7

Conclusions

7.1 Summary

This thesis has explored methods for quantitative modelling and scalable analysis of CAS. A novel stochastic process algebra, PALOMA, was presented to allowing for intuitively capturing complex dynamic behaviours in CAS using its rather simple grammar. In comparison with other process algebras like PEPA [Hillston, 1996] and Bio-PEPA [Ciocchetta and Hillston, 2009], PALOMA has the advantages that the space is explicitly captured, and a richer set of interaction patterns are supported by functional unicast and broadcast communication between entities, which makes PALOMA a language specifically tailored for modelling CAS. Based on its formal semantics, the underlying mathematical model, PCTMC, can be automatically derived for an arbitrary PALOMA model. The generated PCTMC can be used for quantitative analysis of the modelled system using both discrete-event stochastic simulation and Ordinary Differential Equation (ODE)-based fluid approximation.

Fluid approximation, or more specifically moment-closure approximation, which approximates the evolution of the moments (mean, variance, covariance, skewness, etc.) of population variables in a PCTMC using a set of coupled ODEs is generally much more efficient than stochastic simulation for analysing PCTMCs. However, due to the large system scale and highly heterogeneous nature of CAS, the number of derived ODEs from PALOMA models can easily become too large, which makes them infeasible to solve using contemporary ODE solvers. In Chapter 4, we proposed a novel moment-closure approximation method based on the combination of the neighbourhood relation between population variables and the lognormal closure method, that can be automatically applied to an arbitrary PALOMA model. Several experiments

showed that our method can improve the scalability of moment-closure approximation by significantly reducing the required number of ODEs to describe the evolution of the moments, but still achieves reasonable accuracy.

Stochastic simulation is costly, but it is also the most informative analysis technique for PCTMCs. In practice, the inefficiency of stochastic simulation often becomes an obstacle for many PCMTCs for large-scale systems such as CAS. In Chapter 5, we proposed a novel model reduction algorithm that can significantly speed up stochastic simulation of PCTMCs by removing a set of unimportant transitions and population variables with respect to some target populations of interest. The removable transitions and population variables can be efficiently identified by constructing a directed coupling graph for the PCTMC. The error caused by our reduction algorithm can be effectively controlled by an acceptable error threshold set by the modeller. Our algorithm is particularly useful for PCTMCs for CAS since entities in CAS are usually highly distributed which increases their chance of decoupling. Experiments on two example CAS models illustrated the usefulness of our algorithm.

In Chapter 6, we adapted and applied our scalable analysis techniques to the prediction of bike availability in Santander Cycles, the public bike-sharing system in London. Specifically, the moments of the number of available bikes in a station are derived by fluid approximation of a pruned PCTMC in which only significant journey flows with respect to the station are explicitly captured. The method for pruning the PCTMC is adapted from the reduction algorithm in Chapter 5. The number of ODEs for the PCTMC is further reduced based on the neighbourhood relation between population variables that is proposed in Chapter 4. The entire probability distribution is reconstructed to maximise the entropy which minimises the bias introduced by the reconstruction process. The experiments showed that our prediction model can achieve a higher accuracy than the Markov queueing model. Moreover, by using the scalable analysis techniques, the model size can be significantly reduced to such an extent that it is suitable for real time application.

7.2 Further Works

Having summarised our achievements, we conclude this thesis by outlining possible future research directions to enhance the works in this thesis.

7.2.1 Enhancing Expressiveness of PALOMA

There are still several things that can be done to improve the expressiveness of PALOMA in capturing CAS. For example, agents in PALOMA are only parametrised by their location attributes. This can be extended by allowing other user defined attributes in order to support more general attribute-based communication between agents. Moreover, the creation and destruction of agents can also be introduced to make the language more expressive. Taking the smart taxi model in Section 5.5.2 as an example, the model can be easily specified in PALOMA if the above two features are added. The inclusion of more features can improve the expressiveness of PALOMA, but it can also increase the risk of the underlying PCTMC becoming intractable more easily. For example, introducing a new attribute for agents means adding a new dimension to the state space of agents. This is likely to result in a drastic increase of the number of population variables in the PCTMC. Consequently, fluid approximation in which the number of generated ODEs depends on the number of population variables can easily become intractable. Therefore, some other analysis techniques such as population aggregation based on behaviour or spatial equivalence [Piho and Hillston, 2016] and other coarser representation of populations are worth exploring. More importantly, as a modelling language, its user experience is of vital importance. Thus, in order to show, and possibly improve the benefits of using PALOMA for the modelling and quantitative analysis of CAS, it is also highly useful to collect feedbacks from PALOMA users, especially on the compactness and intuitiveness of PALOMA for the modelling of CAS compared with other modelling languages.

7.2.2 Defining Useful Performance Measures for CAS

Traditionally, performance measures derived from probability distributions can be broadly divided into three categories as was discussed by us in [Feng et al., 2015]:

State-based: an expectation over the states of the system. In its simplest form this is the probability that a certain property holds (Boolean values attributed to states). Utilisation is an example of this type. But such measures can also be based on more meaningful values for states, such as queue length where the value for each state is the number of customers in a queue. When the probability distribution is the steady state distribution the derived values will the average values, where at other times they will be transient, based on the transient probability.

Rate-based: an expectation over the rates of the system. Typical examples are throughput, loss probabilities, collision probability etc. Essentially these are also calculated as expectations over the states but the rewards associated with the states are now the rate at which events occur within the given state. Again either the transient or the steady state probability distribution may be used in the calculation of the expectation.

Time-based: an average time, or a probability distribution with respect to time with respect to some behaviour. The classic example is perhaps response time which, via Little's Law can be expressed in terms of throughput (a rate-based measure) and average number (a state-based measure). For non steady state measures, a passage time calculation will usually be required.

For CAS in which there are both temporal and spatial aspects of behaviour, we think it is reasonable to also define space-based and spatial-temporal performance measures. For instance, when spatial information is also represented in the system, the states of interest may be those in which certain spatial conditions are satisfied. Thus we might think of a form of spatial utilisation, the percentage of time that a particular location or set of locations are occupied. There has already been some recent work on the definition and evaluation of spatial-temporal properties for CAS [Nenzi et al., 2015, Bortolussi and Tschaikowski, 2016]. A preliminary study of the types of useful spatial-temporal performance measures for CAS has been presented by us in [Feng et al., 2015]. In future work we would like to investigate our identified measures further to see how well they match to the user and operator performance requirements for CAS.

7.2.3 Statistical Model Checking of CAS

Having shown that our reduction algorithm in Chapter 5 can significantly reduce the computational cost of stochastic simulations for CAS models, and the the error caused by the reduction can be well controlled by the acceptable error threshold set by the modeller, we anticipate that the benefit to be gained from our approach could be particularly valuable in statistical model checking since it usually requires thousands of simulation runs in order to check whether a hypothesis holds. For example, for the bike-sharing system, suppose we want to check whether the following hypothesis holds: $\Pr(\mathbf{G}_{[0,100]} 0 < x_b < C) \geq 95\%$ where x_b is the number of bike agents in a station, and C is the capacity of that station. This means we require that in the first 100 time points,

the probability of the station being empty or full should be less than 5%. Thus, if we set the bike agents in that station as our target population, the simulation speed can be significantly boosted by using our reduction algorithm. We plan to explore and exploit this promising application of our approach in future work.

7.2.4 Learning Model Parameters From Data

Finally, an important problem when constructing PCTMC models for CAS is the inference of parameters from real data. Parameter inference is frequently carried out by Approximate Bayesian Computing approaches [Beaumont et al., 2002, Toni et al., 2009]. These methods which rely on exhaustive stochastic simulation runs and accept parameter values if the differences between simulation and data is sufficiently small will be extremely costly for CAS. Inference using Finite State Projection (FSP) [Munsky and Khammash, 2006] is usually more efficient, however FSP is still very limited since directly evaluating the probability densities for populations will end up with an prohibitive number of ODEs for large scale systems like CAS. Recently, moment-closure approximation has been regarded as a promising tool for parameter inference for biochemical reaction networks [Bogomolov et al., 2015, Fröhlich et al., 2016]. For CAS, traditional moment-closure approximation without ODE reduction may still be over-expensive for many realistic models, thus using our moment-closure approximation method for parameter inference in the construction of PCTMCs for realistic CAS should be another promising direction to explore.

Bibliography

[Ale et al., 2013] Ale, A., Kirk, P., and Stumpf, M. P. (2013). A general moment expansion method for stochastic kinetic models. *The Journal of Chemical Physics*, 138(17):174101.

[Anderson, 2012] Anderson, W. J. (2012). *Continuous-time Markov chains: An applications-oriented approach*. Springer Science & Business Media.

[Andreychenko et al., 2015] Andreychenko, A., Mikeev, L., and Wolf, V. (2015). Model reconstruction for moment-based stochastic chemical kinetics. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 25(2):12.

[Antonaki and Philippou, 2012] Antonaki, M. and Philippou, A. (2012). A process calculus for spatially-explicit ecological models. *EPTCS*, 100:14–28.

[Balbo, 2001] Balbo, G. (2001). Introduction to stochastic Petri nets. In *Lectures on Formal Methods and Performance Analysis*, volume 2090 of *LNCS*, pages 84–155. Springer.

[Beaumont et al., 2002] Beaumont, M. A., Zhang, W., and Balding, D. J. (2002). Approximate Bayesian computation in population genetics. *Genetics*, 162(4):2025–2035.

[Bernardo and Gorrieri, 1998] Bernardo, M. and Gorrieri, R. (1998). A tutorial on EMPA: A theory of concurrent processes with nondeterminism, priorities, probabilities and time. *Theoretical Computer Science*, 202(1):1–54.

[Bogomolov et al., 2015] Bogomolov, S., Henzinger, T. A., Podelski, A., Ruess, J., and Schilling, C. (2015). Adaptive moment closure for parameter inference of biochemical reaction networks. In *Computational Methods in Systems Biology*, volume 9308 of *LNCS*, pages 77–89. Springer.

[Bortolussi, 2006] Bortolussi, L. (2006). Stochastic concurrent constraint programming. *Electronic Notes in Theoretical Computer Science*, 164(3):65–80.

[Bortolussi et al., 2015a] Bortolussi, L., De Nicola, R., Galpin, V., Gilmore, S., Hillston, J., Latella, D., Loreti, M., and Massink, M. (2015a). CARMA: collective adaptive resource-sharing Markovian agents. In *Proceedings Thirteenth Workshop on Quantitative Aspects of Programming Languages and Systems, QAPL 2015, London, UK, 11th-12th April 2015.*, pages 16–31.

[Bortolussi et al., 2013] Bortolussi, L., Hillston, J., Latella, D., and Massink, M. (2013). Continuous approximation of collective system behaviour: A tutorial. *Performance Evaluation*, 70(5):317–349.

[Bortolussi et al., 2015b] Bortolussi, L., Milios, D., and Sanguinetti, G. (2015b). Efficient stochastic simulation of systems with multiple time scales via statistical abstraction. In *Computational Methods in Systems Biology*, volume 9308 of *LNCS*, pages 40–51. Springer.

[Bortolussi and Paškauskas, 2014] Bortolussi, L. and Paškauskas, R. (2014). Mean-field approximation and quasi-equilibrium reduction of Markov population models. In *Quantitative Evaluation of Systems*, volume 8657 of *LNCS*, pages 106–121. Springer.

[Bortolussi and Tschaikowski, 2016] Bortolussi, L. and Tschaikowski, M. (2016). Fluid analysis of spatio-temporal properties of agents in a population model. In *Analytical and Stochastic Modeling Techniques and Applications*, volume 9845 of *LNCS*, pages 92–106. Springer.

[Brodo et al., 2007] Brodo, L., Degano, P., and Priami, C. (2007). A stochastic semantics for BioAmbients. In *Parallel Computing Technologies*, volume 4671 of *LNCS*, pages 22–34. Springer.

[Bruneo et al., 2012] Bruneo, D., Scarpa, M., Bobbio, A., Cerotti, D., and Gribaudo, M. (2012). Markovian agent modeling swarm intelligence algorithms in wireless sensor networks. *Performance Evaluation*, 69(3):135–149.

[Buchholz, 1994] Buchholz, P. (1994). Exact and ordinary lumpability in finite Markov chains. *Journal of Applied Probability*, 31(1):59–75.

[Cao et al., 2005] Cao, Y., Gillespie, D. T., and Petzold, L. R. (2005). The slow-scale stochastic simulation algorithm. *The Journal of Chemical Physics*, 122(1):014116.

[Cao et al., 2006] Cao, Y., Gillespie, D. T., and Petzold, L. R. (2006). Efficient step size selection for the tau-leaping simulation method. *The Journal of Chemical Physics*, 124(4):044109.

[Cao et al., 2004] Cao, Y., Li, H., and Petzold, L. (2004). Efficient formulation of the stochastic simulation algorithm for chemically reacting systems. *The Journal of Chemical Physics*, 121(9):4059–4067.

[Cao and Petzold, 2008] Cao, Y. and Petzold, L. (2008). Slow-scale tau-leaping method. *Computer Methods in Applied Mechanics and Engineering*, 197(43):3472–3479.

[Cardelli, 2008a] Cardelli, L. (2008a). From processes to ODEs by chemistry. In *Fifth IFIP International Conference On Theoretical Computer Science–TCS 2008*, pages 261–281. Springer.

[Cardelli, 2008b] Cardelli, L. (2008b). On process rate semantics. *Theoretical Computer Science*, 391(3):190–215.

[Cerotti et al., 2008] Cerotti, D., Gribaudo, M., and Bobbio, A. (2008). Disaster propagation in inhomogeneous media via Markovian agents. volume 5508 of *LNCS*, pages 328–335. Springer.

[Cerotti et al., 2010] Cerotti, D., Gribaudo, M., Bobbio, A., Calafate, C. T., and Manzoni, P. (2010). A Markovian agent model for fire propagation in outdoor environments. In *Computer Performance Engineering*, volume 6342 of *LNCS*, pages 131–146. Springer.

[Ciardo and Miner, 1999] Ciardo, G. and Miner, A. S. (1999). A data structure for the efficient Kronecker solution of GSPNs. In *Petri Nets and Performance Models, 1999. Proceedings. The 8th International Workshop on*, pages 22–31. IEEE.

[Ciardo and Siminiceanu, 2002] Ciardo, G. and Siminiceanu, R. (2002). Using edge-valued decision diagrams for symbolic generation of shortest paths. In *International Conference on Formal Methods in Computer-Aided Design*, volume 2517 of *LNCS*, pages 256–273. Springer.

[Ciocchetta and Hillston, 2009] Ciocchetta, F. and Hillston, J. (2009). Bio-PEPA: A framework for the modelling and analysis of biological systems. *Theoretical Computer Science*, 410(33):3065–3084.

[Clark et al., 2007] Clark, A., Gilmore, S., Hillston, J., and Tribastone, M. (2007). Stochastic process algebras. In *International School on Formal Methods for the Design of Computer, Communication and Software Systems*, volume 4486 of *LNCS*, pages 132–179. Springer.

[Contardo et al., 2012] Contardo, C., Morency, C., and Rousseau, L.-M. (2012). Balancing a dynamic public bike-sharing system. Technical Report CIRRELT-2012-09.

[Deavours and Sanders, 1998] Deavours, D. D. and Sanders, W. H. (1998). On-the-fly solution techniques for stochastic Petri nets and extensions. *IEEE Transactions on Software Engineering*, 24(10):889–902.

[Engblom, 2006] Engblom, S. (2006). Computing the moments of high dimensional solutions of the master equation. *Applied Mathematics and Computation*, 180(2):498–515.

[Feng et al., 2015] Feng, C., Gribaudo, M., and Hillston, J. (2015). Performance analysis of collective adaptive behaviour in time and space. *Electronic Notes in Theoretical Computer Science*, 318:53–68.

[Feng and Hillston, 2014] Feng, C. and Hillston, J. (2014). PALOMA: A process algebra for located Markovian agents. In *Quantitative Evaluation of Systems*, volume 8657 of *LNCS*, pages 265–280. Springer.

[Feng and Hillston, 2015] Feng, C. and Hillston, J. (2015). Speed-up of stochastic simulation of PCTMC models by statistical model reduction. In *Computer Performance Engineering*, volume 9272 of *LNCS*, pages 291–305. Springer.

[Feng et al., 2016a] Feng, C., Hillston, J., and Galpin, V. (2016a). Automatic moment-closure approximation of spatially distributed collective adaptive systems. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 26(4):26.

[Feng et al., 2016b] Feng, C., Hillston, J., and Reijsbergen, D. (2016b). Moment-based probabilistic prediction of bike availability for bike-sharing systems. In *Quantitative Evaluation of Systems*, volume 9826 of *LNCS*, pages 139–155. Springer.

[Fishman, 2016] Fishman, E. (2016). Bikeshare: A review of recent literature. *Transport Reviews*, 36(1):92–113.

[Froehlich et al., 2009] Froehlich, J., Neumann, J., and Oliver, N. (2009). Sensing and predicting the pulse of the city through shared bicycling. In *IJCAI*, volume 9, pages 1420–1426.

[Fröhlich et al., 2016] Fröhlich, F., Thomas, P., Kazeroonian, A., Theis, F. J., Grima, R., and Hasenauer, J. (2016). Inference for stochastic chemical kinetics using moment equations and system size expansion. *PLoS Computational Biology*, 12(7):e1005030.

[Fujita et al., 1997] Fujita, M., McGeer, P. C., and Yang, J.-Y. (1997). Multi-terminal binary decision diagrams: An efficient data structure for matrix representation. *Formal Methods in System Design*, 10(2-3):149–169.

[Gast et al., 2015] Gast, N., Massonnet, G., Reijsbergen, D., and Tribastone, M. (2015). Probabilistic forecasts of bike-sharing systems for journey planning. In *The 24th ACM International Conference on Information and Knowledge Management*, CIKM '15, pages 703–712. ACM.

[Gibson and Bruck, 2000] Gibson, M. A. and Bruck, J. (2000). Efficient exact stochastic simulation of chemical systems with many species and many channels. *The Journal of Physical Chemistry A*, 104(9):1876–1889.

[Gillespie, 1977] Gillespie, D. T. (1977). Exact stochastic simulation of coupled chemical reactions. *The Journal of Physical Chemistry*, 81(25):2340–2361.

[Gillespie, 2001] Gillespie, D. T. (2001). Approximate accelerated stochastic simulation of chemically reacting systems. *The Journal of Chemical Physics*, 115(4):1716–1733.

[Gilmore et al., 2001] Gilmore, S., Hillston, J., and Ribaudo, M. (2001). An efficient algorithm for aggregating PEPA models. *IEEE Transactions on Software Engineering*, 27(5):449–464.

[Gotz et al., 1992] Gotz, N., Herzog, U., and Rettelbach, M. (1992). TIPP – a language for timed processes and performance evaluation. Technical Report Technical Report 4/92, IMMD VII, University of Erlangen-Nurnberg.

[Grassmann, 1977] Grassmann, W. K. (1977). Transient solutions in Markovian queueing systems. *Computers & Operations Research*, 4(1):47–53.

[Gribaudo et al., 2008] Gribaudo, M., Cerotti, D., and Bobbie, A. (2008). Analysis of on-off policies in sensor networks using interacting Markovian agents. In *6th Annual IEEE International Conference on Pervasive Computing and Communications*, pages 300–305. IEEE.

[Guenther and Bradley, 2013] Guenther, M. C. and Bradley, J. T. (2013). Journey data based arrival forecasting for bicycle hire schemes. In *Analytical and Stochastic Modeling Techniques and Applications*, volume 7984 of *LNCS*, pages 214–231. Springer.

[Guenther et al., 2012] Guenther, M. C., Stefanek, A., and Bradley, J. T. (2012). Moment closures for performance models with highly non-linear rates. In *European Workshop on Performance Engineering*, pages 32–47. Springer.

[Guenther et al., 2013] Guenther, M. C., Stefanek, A., and Bradley, J. T. (2013). Moment closures for performance models with highly non-linear rates. In *Computer Performance Engineering*, volume 7587 of *LNCS*, pages 32–47. Springer.

[Hayden and Bradley, 2010] Hayden, R. A. and Bradley, J. T. (2010). A fluid analysis framework for a Markovian process algebra. *Theoretical Computer Science*, 411(22):2260–2297.

[Helms et al., 2015] Helms, T., Ewald, R., Rybacki, S., and Uhrmacher, A. M. (2015). Automatic runtime adaptation for component-based simulation algorithms. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, 26(1):7.

[Helms et al., 2014] Helms, T., Maus, C., Haack, F., and Uhrmacher, A. M. (2014). Multi-level modeling and simulation of cell biological systems with ML-Rules—a tutorial. In *Proceedings of the Winter Simulation Conference 2014*, pages 177–191. IEEE.

[Hiebeler, 2006] Hiebeler, D. (2006). Moment equations and dynamics of a household SIS epidemiological model. *Bulletin of Mathematical Biology*, 68(6):1315–1333.

[Hillston, 1996] Hillston, J. (1996). *A Compositional Approach to Performance Modelling*. Cambridge University Press.

[Hillston, 2005] Hillston, J. (2005). Fluid flow approximation of PEPA models. In *2nd International Conference on the Quantitative Evaluation of Systems*, pages 33–42. IEEE.

[Hillston, 2013] Hillston, J. (2013). Challenges for quantitative analysis of collective adaptive systems. In *International Symposium on Trustworthy Global Computing*, volume 8358 of *LNCS*, pages 14–21. Springer.

[Hillston and Loreti, 2015] Hillston, J. and Loreti, M. (2015). Specification and analysis of open-ended systems with CARMA. In *Agent Environments for Multi-Agent Systems IV*, volume 9068 of *LNCS*, pages 95–116. Springer.

[Himmelspach and Uhrmacher, 2007] Himmelspach, J. and Uhrmacher, A. M. (2007). Plug'n simulate. In *40th Annual Simulation Symposium (ANSS'07)*, pages 137–143. IEEE.

[Hoare, 1985] Hoare, C. A. R. (1985). *Communicating Sequential Processes*. Prentice-Hall Englewood Cliffs.

[Isserlis, 1918] Isserlis, L. (1918). On a formula for the product-moment coefficient of any order of a normal frequency distribution in any number of variables. *Biometrika*, 12(1/2):134–139.

[John et al., 2008a] John, M., Ewald, R., and Uhrmacher, A. M. (2008a). A spatial extension to the π calculus. *Electronic Notes in Theoretical Computer Science*, 194(3):133–148.

[John et al., 2008b] John, M., Lhoussaine, C., Niehren, J., and Uhrmacher, A. M. (2008b). The attributed pi calculus. In *Computational Methods in Systems Biology*, volume 5307 of *LNCS*, pages 83–102. Springer.

[Kaltenbrunner et al., 2010] Kaltenbrunner, A., Meza, R., Grivolla, J., Codina, J., and Banchs, R. (2010). Urban cycles and mobility patterns: Exploring and predicting trends in a bicycle-based public transport system. *Pervasive and Mobile Computing*, 6(4):455–466.

[Keeling, 1999] Keeling, M. J. (1999). The effects of local spatial structure on epidemiological invasions. *Proceedings of the Royal Society of London B: Biological Sciences*, 266(1421):859–867.

[Keeling, 2000] Keeling, M. J. (2000). Multiplicative moments and measures of persistence in ecology. *Journal of Theoretical Biology*, 205(2):269–281.

[Knottenbelt and Harrison, 1999] Knottenbelt, W. J. and Harrison, P. G. (1999). Distributed disk-based solution techniques for large Markov models. In *Proceedings of the 3rd International Meeting on the Numerical Solution of Markov Chains NSMC*, volume 99, pages 58–75.

[Krishnarajah et al., 2005] Krishnarajah, I., Cook, A., Marion, G., and Gibson, G. (2005). Novel moment closure approximations in stochastic epidemics. *Bulletin of Mathematical Biology*, 67(4):855–873.

[Krivine et al., 2008] Krivine, J., Milner, R., and Troina, A. (2008). Stochastic bigraphs. *Electronic Notes in Theoretical Computer Science*, 218:73–96.

[Kuehn, 2016] Kuehn, C. (2016). Moment closure-a brief review. In *Control of Self-Organizing Nonlinear Systems*, Understanding Complex Systems, pages 253–271. Springer.

[Kurtz, 1981] Kurtz, T. G. (1981). *Approximation of population processes*. SIAM.

[Kwiatkowska et al., 2007] Kwiatkowska, M., Norman, G., and Parker, D. (2007). Stochastic model checking. In *International School on Formal Methods for the Design of Computer, Communication and Software Systems*, volume 4486 of *LNCS*, pages 220–270. Springer.

[Legay et al., 2010] Legay, A., Delahaye, B., and Bensalem, S. (2010). Statistical model checking: An overview. In *Runtime Verification*, volume 6418 of *LNCS*, pages 122–135. Springer.

[Lin and Yang, 2011] Lin, J.-R. and Yang, T.-H. (2011). Strategic design of public bicycle sharing systems with service level constraints. *Transportation Research Part E: Logistics and Transportation Review*, 47(2):284–294.

[Lu and Law, 2005] Lu, T. and Law, C. K. (2005). A directed relation graph method for mechanism reduction. *Proceedings of the Combustion Institute*, 30(1):1333–1341.

[Mastny et al., 2007] Mastny, E. A., Haseltine, E. L., and Rawlings, J. B. (2007). Two classes of quasi-steady-state model reductions for stochastic kinetics. *The Journal of Chemical Physics*, 127(9):094106.

[Mead and Papanicolaou, 1984] Mead, L. R. and Papanicolaou, N. (1984). Maximum entropy in the problem of moments. *Journal of Mathematical Physics*, 25(8):2404–2417.

[Milner, 1989] Milner, R. (1989). *Communication and Concurrency*. Prentice-Hall, Inc.

[Munsky and Khammash, 2006] Munsky, B. and Khammash, M. (2006). The finite state projection algorithm for the solution of the chemical master equation. *The Journal of Chemical Physics*, 124(4):044104.

[Nair and Miller-Hooks, 2011] Nair, R. and Miller-Hooks, E. (2011). Fleet management for vehicle sharing operations. *Transportation Science*, 45(4):524–540.

[Nardini et al., 2008] Nardini, C., Kozma, B., and Barrat, A. (2008). Who's talking first? consensus or lack thereof in coevolving opinion formation models. *Physical Review Letters*, 100(15):158701.

[Nåsell, 2003] Nåsell, I. (2003). An extension of the moment closure method. *Theoretical Population Biology*, 64(2):233–239.

[Nenzi et al., 2015] Nenzi, L., Bortolussi, L., Ciancia, V., Loreti, M., and Massink, M. (2015). Qualitative and quantitative monitoring of spatio-temporal properties. In *Runtime Verification*, volume 9333 of *LNCS*, pages 21–37. Springer.

[Nicola et al., 2014] Nicola, R. D., Loreti, M., Pugliese, R., and Tiezzi, F. (2014). A formal approach to autonomic systems programming: The SCEL language. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, 9(2):7.

[Niemeyer et al., 2010] Niemeyer, K. E., Sung, C.-J., and Raju, M. P. (2010). Skeletal mechanism generation for surrogate fuels using directed relation graph with error propagation and sensitivity analysis. *Combustion and flame*, 157(9):1760–1770.

[Norris, 1998] Norris, J. R. (1998). *Markov chains*. Cambridge University Press.

[Pepiot-Desjardins and Pitsch, 2008] Pepiot-Desjardins, P. and Pitsch, H. (2008). An efficient error-propagation-based reduction method for large chemical kinetic mechanisms. *Combustion and Flame*, 154(1):67–81.

[Pfrommer et al., 2014] Pfrommer, J., Warrington, J., Schildbach, G., and Morari, M. (2014). Dynamic vehicle redistribution and online price incentives in shared mobility systems. *IEEE Transactions on Intelligent Transportation Systems*, 15(4):1567–1578.

[Piho and Hillston, 2016] Piho, P. and Hillston, J. (2016). Stochastic and spatial equivalences for PALOMA. In *Proceedings of the Workshop on FORmal methods for the quantitative Evaluation of Collective Adaptive SysTems, FORECAST@STAF 2016, Vienna, Austria, 8 July 2016*, volume 217 of *EPTCS*, pages 69–80.

[Plateau and Atif, 1991] Plateau, B. and Atif, K. (1991). Stochastic automata network of modeling parallel systems. *IEEE transactions on Software Engineering*, 17(10):1093–1108.

[Plotkin, 2004] Plotkin, G. D. (2004). A structural approach to operational semantics. *J. Log. Algebr. Program.*, 60(61):17–139.

[Priami, 1995] Priami, C. (1995). Stochastic π-calculus. *The Computer Journal*, 38(7):578–589.

[Pu et al., 2011] Pu, Y., Watson, L. T., and Cao, Y. (2011). Stiffness detection and reduction in discrete stochastic simulation of biochemical systems. *The Journal of Chemical Physics*, 134(5):054105.

[Rangan and Cai, 2006] Rangan, A. V. and Cai, D. (2006). Maximum-entropy closures for kinetic theories of neuronal network dynamics. *Physical Review Letters*, 96(17):178101.

[Rao and Arkin, 2003] Rao, C. V. and Arkin, A. P. (2003). Stochastic chemical kinetics and the quasi-steady-state assumption: application to the Gillespie algorithm. *The Journal of Chemical Physics*, 118(11):4999–5010.

[Raviv and Kolka, 2013] Raviv, T. and Kolka, O. (2013). Optimal inventory management of a bike-sharing station. *IIE Transactions*, 45(10):1077–1093.

[Reinecke et al., 2012] Reinecke, P., Krauss, T., and Wolter, K. (2012). Hyperstar: Phase-type fitting made easy. In *9th International Conference on Quantitative Evaluation of Systems*, pages 201–202. IEEE.

[Schnoerr et al., 2015] Schnoerr, D., Sanguinetti, G., and Grima, R. (2015). Comparison of different moment-closure approximations for stochastic chemical kinetics. *The Journal of Chemical Physics*, 143(18):185101.

[Schuijbroek et al., 2013] Schuijbroek, J., Hampshire, R., and van Hoeve, W.-J. (2013). Inventory rebalancing and vehicle routing in bike sharing systems. Technical Report 2013-E1, Tepper School of Business, Carnegie Mellon University.

[Singer, 2004] Singer, A. (2004). Maximum entropy formulation of the Kirkwood superposition approximation. *The Journal of Chemical Physics*, 121(8):3657–3666.

[Singh and Hespanha, 2006] Singh, A. and Hespanha, J. P. (2006). Lognormal moment closures for biochemical reactions. In *45th IEEE Conference on Decision and Control*, pages 2063–2068. IEEE.

[Slepoy et al., 2008] Slepoy, A., Thompson, A. P., and Plimpton, S. J. (2008). A constant-time kinetic Monte Carlo algorithm for simulation of large biochemical reaction networks. *The Journal of Chemical Physics*, 128(20):205101.

[Snyman, 2005] Snyman, J. (2005). *Practical mathematical optimization: an introduction to basic optimization theory and classical and new gradient-based algorithms*. Springer Science & Business Media.

[Stewart, 2009] Stewart, W. J. (2009). Probability, Markov chains, queues, and simulation: the mathematical basis of performance modeling. Princeton University Press.

[Tari et al., 2005] Tari, Á., Telek, M., and Buchholz, P. (2005). A unified approach to the moments based distribution estimation–unbounded support. In *Formal Techniques for Computer Systems and Business Processes*, volume 3670 of *LNCS*, pages 79–93. Springer.

[ter Beek et al., 2014] ter Beek, M., Bortolussi, L., Ciancia, V., Gnesi, S., Hillston, J., Latella, D., and Massink, M. (2014). A quantitative approach to the design and analysis of collective adaptive systems for smart cities. *ERCIM News*, 2014(98).

[Toni et al., 2009] Toni, T., Welch, D., Strelkowa, N., Ipsen, A., and Stumpf, M. P. (2009). Approximate Bayesian computation scheme for parameter inference and model selection in dynamical systems. *Journal of the Royal Society Interface*, 6(31):187–202.

[Tribastone et al., 2012] Tribastone, M., Gilmore, S., and Hillston, J. (2012). Scalable differential analysis of process algebra models. *IEEE Transactions on Software Engineering*, 38(1):205–219.

[Wu et al., 2012] Wu, S., Fu, J., Li, H., and Petzold, L. (2012). Automatic identification of model reductions for discrete stochastic simulation. *The Journal of Chemical Physics*, 137(3):034106.

[Yoon et al., 2012] Yoon, J. W., Pinelli, F., and Calabrese, F. (2012). Cityride: a predictive bike sharing journey advisor. In *13th IEEE International Conference on Mobile Data Management (MDM)*, pages 306–311. IEEE.