Formal modelling and approximation-based analysis for mode-switching population dynamics

Paul Piho

Doctor of Philosophy
Laboratory for Foundations of Computer Science
School of Informatics
University of Edinburgh
2020

Abstract

This thesis explores aspects of model specification and analysis for population dynamics which arise when modelling complex interactions and communication structures in agent or component collectives. The motivating examples come from the design of man-made systems where the optimal parametrisations for the behaviours of agents or components are not known a priori. In particular, we introduce a formal modelling framework to support the specification of control problems for collective dynamics in a high-level process algebraic language. A natural choice for the underlying semantics is to consider continuous time Markov decision processes due to their close relation to continuous time Markov chains that have traditionally been used as the mathematical model in numerous high-level modelling languages for stochastic dynamics.

Although the theory of the resulting decision processes has a long history, the practical considerations, like computation time, present challenges due to the problem of state space explosion when considering large systems with complex behaviours. State space explosion problems are especially apparent in formal modelling paradigms where the specification of models usually happens at a component or an agent level in terms of a discrete set of states with defined rules for composing the specified behaviours into the dynamics of a system. Such specifications often give rise to very large models which are costly to analyse in full detail. However, when analysing models of collectives we are usually interested in the resulting macro-scale dynamics in terms of some aggregate measures. With that in mind, the second aspect of analysing collective dynamics that is considered in this thesis relates to fluid, linear noise and moment closure-based approximation methods which aim to give a good representation of the macro-scale dynamics of the models while being computationally less costly to analyse.

We address a class of models where the population structure results from the assumption that components or agents can only be distinguished from each other based on the state they are in and focus on the particular cases where the population dynamics can be separated into a discrete set of modes. Our study of these models is motivated by considering information propagation via broadcast communication where the behaviour of components can change drastically when new information is received from the rest of the population. We consider existing approximation methods for resulting stochastic processes and propose a novel approach for applying these methods to models incorporating broadcast communication where each level of information available to the collective corresponds to a discrete dynamic mode. The resulting approximations combine continuous dynamics with discrete stochastic jumps and are not immediately simple to treat numerically. To that end we propose further approximations that allow for a computationally efficient analysis. Finally, we demonstrate how the formal modelling framework in conjunction with the developed approximation methods can be used for an example in policy synthesis.

Lay Summary

Mathematical models are an important tool in the study of real-world systems and phenomena. Typically such models are highly simplified or abstracted representations of their real-world counterparts. The value of the constructed models lies in their predictive power. A good model allows us to make predictions about the real-world behaviours of the systems and reason about their properties. In the case of engineered systems a good model helps us to design the system so that it behaves as desired in practise. We concentrate on collectives, like robot swarms, and study a certain kind of mathematical models, called continuous time Markov decision processes, and use them to describe the dynamics of the collectives together with the choices we can make in designing or controlling the behaviour of the individual components making up the collective.

The models of collective dynamics described above are often complex to specify directly and thus it is useful to express them in specialised languages which allow the construction of the underlying mathematical model to be automated. To that end we propose an extension to an existing language that allows for a concise description of collective dynamics along with the possible control or design choices. Secondly, once we have constructed a model we have to find a way to analyse it. As more components are considered the size of the model grows, making the computational analysis of the model behaviour slow. In order to alleviate that problem there exist approximation methods that derive a simplified model which acts as a good proxy for the properties of interest but is faster to analyse. We study a certain class of models where components in the collective are allowed to broadcast knowledge about their environment to the rest of the collective changing its dynamics. As a contribution we show a novel application of existing approximation methods to this class of models and propose further approximations that help us make computationally efficient analyses of the constructed models.

Acknowledgements

First, I would like to express how exceedingly thankful I am to my supervisor Jane Hillston for her patience, support and honesty in guiding me through the past years. Her insight and experience have been invaluable in shaping this thesis and my development as a researcher.

My time as a student in Informatics Forum has been made thoroughly enjoyable by a number of great friends and colleagues. Especially I would like to mention Dan, Philip, Caoimhín, Daniel, Amna, Reese and Vanya form my PPar cohort who have been part of this journey from the very beginning till the end and have been a fantastic company. I would like to thank Ludovica, Michalis, Anastasis and Maria for being welcoming and for stimulating discussions and help.

I would like to thank my parents and my sisters for their support and understanding. Last but definitely not least I thank Julie for her kind unwavering encouragement and keeping me reasonably sane.

This work was supported by the Engineering and Physical Sciences Research Council (grant EP/L01503X/1), EPSRC Centre for Doctoral Training in Pervasive Parallelism at the University of Edinburgh, School of Informatics.

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 materials presented in Chapters 3 and 6 has previously appeared in [104]. Parts of Chapters 4 and 5 are under review for publication [103].

(*Paul Piho*)

Table of Contents

1 Introduction **1**

1.1 Contributions . . . 2

1.2 Structure . . . 3

2 Background **5**

2.1 Stochastic population dynamics . . . 5

2.1.1 Definitions . . . 5

2.1.2 Random time change representation . . . 8

2.1.3 Transient analysis . . . 8

2.2 Hybrid stochastic systems . . . 10

2.3 Model reduction techniques . . . 11

2.4 Continuous approximation methods . . . 11

2.4.1 Fluid approximation . . . 12

2.4.2 Linear noise approximation . . . 13

2.4.3 Moment closure approximations . . . 15

2.5 Formal modelling . . . 16

2.5.1 Collective adaptive resource-sharing Markovian agents . . . 17

2.5.2 Model checking . . . 21

2.6 Stochastic decision processes . . . 22

3 Model specification **25**

3.1 Semantics . . . 26

3.1.1 Store . . . 27

3.1.2 Environment . . . 30

3.1.3 Resolving non-determinism . . . 31

3.1.4 Interleaving semantics . . . 32

3.1.5 Population model . . . 43

3.2 Policy synthesis . . . 44

4 Approximations for mode-switching dynamics **47**

4.1 Mode-switching population models . . . 48

4.2 Transition-based partitioning . . . 49

4.3 Approximations . . . 53

4.3.1 Fluid approximation methods . . . 54

4.3.2 Linear noise approximation . . . 59

4.3.3 Moment based approximations . . . 60

4.4 Results . . . 66

4.5 Conclusion . . . 68

5 Heuristic marginalisation based computational strategies **69**

5.1 Marginal dynamics . . . 70

5.2 Time-inhomogeneous Markov process . . . 73

5.2.1 Filtering heuristics . . . 73

5.3 Fluid approximation . . . 75

5.3.1 Direct coupling . . . 77

5.3.2 Iterative method . . . 79

5.4 Linear noise approximation . . . 83

5.5 Moment closure approximation . . . 84

5.6 Results . . . 86

5.7 Conclusions . . . 91

6 Synthesis example **97**

6.1 Carma-C model . . . 97

6.2 pCTMDP model . . . 99

6.3 Policy synthesis . . . 101

6.4 Discussion . . . 103

7 Conclusions **107**

Bibliography **111**

A Hybrid linear noise theorem **121**

B Conditional moments **123**

B.1 Proof of Lemma 1 . . . 123

B.2 Conditional moments for the running example . . . 124

C Filtering distribution **125**

D Carma-C model of policy synthesis example **127**

Chapter 1

Introduction

The overarching story of this thesis is motivated by the notion of collective systems which encompasses examples of robot swarms, wireless sensor networks and insect swarms. When analysing the dynamics of such systems, we run into the problem that even when considering significantly abstracted or simplified behaviours and communication rules, the emergent behaviour of the systems is often hard to predict or verify. It is harder still to know a priori how to design the behaviour and capabilities of individual components or agents in such collectives in order to achieve a system level objective.

Computational modelling and simulation methods provide a useful way to study and predict behaviours of such complex systems. The approach taken here starts with formal methods, in particular stochastic process algebras with continuous time Markov chain (CTMC) semantics [14, 71, 88], as a useful framework for modelling collective systems by allowing compositional definitions of complex models and feature formal semantics that allow for automation of model creation. The thesis deals with problems in the following two directions:

- **Model specification:** How to formally specify models for policy synthesis problems for collective systems based on capabilities of individual components and the desired system behaviour.
- **Model analysis:** How to perform computationally efficient analysis of the resulting models.

In general, continuous time Markov decision processes (CTMDPs) are a well-studied mathematical model in control theory and operations research providing a natural formalisation of policy synthesis problems. In the context of formal modelling and as an extension to CTMCs they are a good fit for the similar compositional semantics that are available for CTMCs. Note however that, like all discrete state modelling paradigms, CTMDPs suffer from the state space explosion problem. This makes policy synthesis for CTMDP models a computationally challenging problem.

In many cases such computational challenges can be addressed by appealing to specially structured models. A large body of work exists linked to CTMCs where the structure of the model corresponds to that of the interacting populations (population CTMCs, pCTMCs). The developed continuous state approximation methods, like fluid [83], linear noise [122] and moment closure approximations, can also help with analysis and policy synthesis problems for population processes stated in terms of the CTMDP framework [46]. The optimisation problems arising from applications of fluid approximations are known in the control theory literature as mean field control problems [13] with swarm robotics emerging as a recent application field of such continuous state approximation methods as a means for controller design [49].

The first aim of this thesis is to construct a framework that bridges the gap in the formal high-level modelling and policy synthesis problems for collectives, such as robot swarms, resulting in a high-level compositional approach for specifying CTMDP models for collective dynamics. Secondly we consider systems composed of a homogeneous collective of agents or components and study the limitations of continuous state approximations. Notably, it is not trivial to apply the existing approximation methods, for example, when modelling the effects of communication and information available to the agents within a population on the macro-level behaviour of the population. As a particular example we consider models where agents are equipped with knowledge, they are able to learn about their environment through experience and share this information with the population through broadcast communication. A consequence of this is that an action of a single agent (a broadcast) can change the macro-level dynamics of the whole population. For example, the objectives of agents may change as more information is acquired, leading to a change in the overall behaviour of the population. In this thesis we focus our attention on cases of models which incorporate such information cascades and consequent shifts in behaviour.

1.1 Contributions

In this thesis we present the following extensions to the existing body of work.

- We give alternative operational semantics for the CARMA process algebra [88] for formally specifying policy synthesis problems in the CTMDP framework. This is an extended version of the ideas published in [104] which allows for non-determinism in the CARMA component descriptions. The non-determinism is interpreted as possible control actions or decisions that can be taken.

- We concentrate on CTMC models of populations which capture the idea of broadcast communication between the components in the system and demonstrate the applicability of existing hybrid-continuous approximation methods which combine continuous state approximations with discrete stochastic jumps. Preliminary work on utilising fluid approximation-based results was published in [104] and elaborated on in the paper [103].

- The arising hybrid-continuous approximations are non-trivial to simulate or analyse numerically. Thus we propose additional approximate constructions aimed at describing the population dynamics in a computationally efficient manner. The proposed constructions form a part of [103].
- We highlight the progress made towards a cohesive formal framework for policy synthesis problems of collective dynamics through a simple swarm robotics-inspired example. The example given in this thesis is based on the one published as a part of [104].

1.2 Structure

The structure of the thesis is given as follows. The fundamental definitions and overview of relevant literature for the discussions in this thesis are presented in Chapter 2. In Chapter 3 we set up the formal modelling framework for the study of collective dynamics and the related parameter and policy synthesis problem. In particular, we introduce the CTMDP semantics for the existing Carma process algebra. In Chapter 4 we shift our focus to population models. More specifically we introduce mode-switching population dynamics and present how existing hybrid-continuous approximations to continuous time Markov chains with a population structure can be adapted for the analysis of such models. Chapter 5 addresses the problem of simulation of the approximations constructed in the previous chapter. Chapter 6 gives a worked example on policy synthesis for a simple model of a collective system. The intention of this chapter is mainly to present how the ideas discussed in the main content chapters of this thesis fit together into a cohesive framework for policy and parameter synthesis for collective dynamics. We end the thesis with concluding observations and a discussion on future research directions in Chapter 7.

Chapter 2

Background

This chapter is used to establish the background and relevant literature references. The work in this thesis is based on the continuous time Markov chain (CTMC) view of dynamical behaviour. Thus, the first part of this chapter gives a definition of such stochastic processes and discusses the relevant numerical analysis methods. We also present the particular class of such models, namely population CTMCs (pCTMCs), which are going to be considered throughout this thesis as an underlying mathematical model for collective dynamics. This view, that represents the behaviour of a system through transitions between a discrete set of states, stems from the application of tools from formal methods in computer science to modelling dynamics of systems. Secondly, we give an overview of model reduction techniques and approximation methods that have been developed to ease the computational burden when analysing pCTMC models. The methods of fluid approximation, linear noise approximation and moment closure are introduced in more detail as they feature heavily in Chapters 4 and 5. We follow this with a discussion of formal modelling languages for high level specification of CTMC models. In particular, we introduce the Carma [88] process algebra – a formal modelling paradigm for stochastic collective dynamics – which has served as a starting point for much of the work presented in this thesis. As the overall motivation comes from control problems for large collectives we finally introduce the decision processes closely related to CTMCs.

2.1 Stochastic population dynamics

2.1.1 Definitions

The mathematical underpinnings of the models under consideration in this thesis are well established and the basic definitions related to stochastic processes and CTMCs (the terminology "Markov jump process" is also common) given below are standard and can be found in texts like [99].

A Markov process is a time-indexed family of random variables $\{X(t), t \in [0,\infty)\}$ such that the future behaviour of the process is not dependent on its past. In the rest of the thesis we are going to employ the notation X to denote a Markov process. The process X is a Markov process if for all collections of times $0 \leq t_1 \leq t_2 \leq \ldots \leq t_n$ and states $i_1, i_2, \ldots, i_n$ we have

$$\mathbb{P}(X(t_n) = i_n \mid X(t_{n-1}) = i_{n-1}, \ldots, X(t_0) = i_0) = \mathbb{P}(X(t_n) = i_n \mid X(t_{n-1}) = i_{n-1})$$

A particular example of a Markov process that is going to be discussed throughout this thesis is the continuous time Markov chain (CTMC). A CTMC is defined via its infinitesimal generator matrix describing the rates or intensities with which the CTMC moves between its states. For any $t \in [0,\infty)$ let Q be a matrix with (i,j)-th entry defined by $q(i,j)$ such that the following properties hold.

1. $0 \leq -q(i,i) < \infty$
2. $0 \leq q(i,j)$ for $i \neq j$
3. $\sum_j q(i,j) = 0$

With that in mind we can define a CTMC in terms its Q matrix in the following way [99].

Definition 1 (Continuous time Markov chain)**.** *Let X be a Markov process with values in a countable set E. Write $Q = (q(i,j) \mid i,j \in E)$ for the associated generator matrix. For all $n \geq 0$, all times $0 \leq t_0 \leq t_1 \leq \cdots \leq t_{n+1}$ and all states $i_0, \cdots, i_{n+1}$*

$$\mathbb{P}(X(t_{n+1}) = i_{n+1} \mid X(t_0) = i_0, \cdots, X(t_n) = i_n) = p(i_{n+1}; t_{n+1} \mid i_n; t_n)$$

satisfying the following Kolmogorov forward equation

$$\frac{\partial}{\partial t} p(j;t \mid i;s) = \sum_k p(k;t \mid i;s) q(k,j), \qquad \text{on } (s,\infty) \text{ with } p(j;s \mid i;s) = \delta_{ij} \tag{2.1}$$

where δ_{ij} is the Kronecker delta taking the value 1 *if i and j are equal and taking the value* 0 *otherwise. Equation 2.1 is often referred to in biochemical modelling literature as the chemical master equation. For CTMCs and stochastic processes in general the function $p(j;t \mid i;s)$ defining the probability that the state of the process at time t is j given at time s the state was s, is called the* transition density*.*

When the generator matrices are constant and not dependent on time, as in the definition above, the CTMC is said to be *time-homogeneous*. Otherwise the CTMC is said to be *time-inhomogeneous*.

Example 1. *We can consider the Poisson process, which is usually interpreted as a counting process for arrival events, as a simple example of a time-homogeneous CTMC. Suppose the CTMC X takes values in the set $\{0,1,2,\dots\}$ with the generator matrix Q defined by*

$$q(i,j) = \begin{cases} \lambda & \textit{for } j = i+1 \\ -\lambda & \textit{for } j = i \\ 0 & \textit{otherwise} \end{cases} \tag{2.2}$$

The defined process corresponds exactly to the Poisson process with arrival rate λ.

An important special instance of a CTMC is given by the population CTMCs which consider a more compact representation of the state space when modelling systems of indistinguishable agents or components. The population CTMCs are going to be the main mathematical framework in this thesis. The corresponding definition below constructs a model where the state of the system is represented by a counting vector identifying the number of agents or components in each of the possible states the components can be in. In biochemical literature these counts are usually referred to as species populations.

Definition 2 (Population continuous time Markov chain (pCTMC))**.** *A pCTMC is defined by a tuple $\mathcal{P} = (\mathbf{X}, \mathcal{T}, \mathbf{X}_0)$ where*

- $\mathbf{X} = (X_1, \dots, X_K) \in \mathbb{Z}^K_{\geq 0}$ *is a variable with the i-th entry representing the current count of agents or components in state $i \in S$.*
- $\mathcal{T}$ *is a set of transitions of the form $\tau = (r_\tau, \mathbf{v}_\tau)$ where*
  - $r_\tau : \mathbb{Z}^K_{\geq 0} \to \mathbb{R}_{\geq 0}$ *is the rate function associating transition τ with the rate of an exponential distribution depending on the state of the model.*
  - $\mathbf{v}_\tau : \mathbb{Z}^K_{\geq 0} \to \mathbb{Z}^K$ *is a function which gives the net change for each population variable in $\mathbf{X}$ caused by transition τ.*
- $\mathbf{X}_0$ *is the initial state of the model.*

In many cases one is interested in the systems where the total number of components or agents remains fixed throughout the evolution of the system. While this is a rather strict condition it is nevertheless often satisfied when modelling man-made or engineered systems.

Definition 3. *A pCTMC $\mathcal{P} = (\mathbf{X}, \mathcal{T}, \mathbf{X}_0)$ is called* conservative *if $\sum_{j=1}^K X_j^N = N$ for all times $t \geq 0$.*

2.1.2 Random time change representation

Poisson processes and CTMCs are connected through the exponential distribution. In particular, the inter-arrival times of the Poisson process and the waiting times in each of the CTMC states are both exponentially distributed. This allows us to construct a representation of (time-inhomogeneous) CTMCs in terms of time-changed Poisson processes. Results on fluid approximation in Section 2.4.1 and the discussion in Chapter 4 make use of this convenient representation. In addition, it is the basis for many simulation algorithms for CTMCs.

Note that for any countable state CTMC we can always make a choice of coordinates that would translate the CTMC into a $\mathbb{R}^d$-valued jump process. In the case of population processes [41] occupancy measures interpreted as counts or proportions of individuals in each of the possible states are often natural choices for such coordinates. The idea is then to express the Markov process in terms of a set of Poisson processes, with the rate parameters dependent on the state of the process, counting the number of times a given transition has happened. A simple example of this would be a birth-death process which can be expressed in terms of two Poisson processes – one counting the number of births and the other the number of deaths that have occurred. Knowing the initial population size and the count of births and deaths that have occurred up to a given time is then sufficient to know the population size at that time. Similarly, in pCTMC models from the previous section each transition is associated with a Poisson process counting the number of times the transition has occurred.

In slightly more detail, suppose $\mathbf{X}$ is a CTMC over states in $\mathbb{R}^d$ with an infinitesimal generator matrix defining the transition intensities of the form $q(k, k+l) = \beta_l(k)$. That is, we write $q(k, k+l) = \beta_l(k)$ for transition intensities of the CTMC inducing a change $l \in \mathbb{R}^d$ in the state of the CTMC. It can then be shown that $\mathbf{X}$ satisfies

$$\mathbf{X}(t) = \mathbf{X}(0) + \sum_l l Y_l \left(\int_0^t \beta_l(\mathbf{X}(s)) ds \right) \tag{2.3}$$

where Y_l are independent Poisson processes with rate parameter 1, the initial condition $\mathbf{X}(0)$ is non-random and $l \in \mathbb{R}^d$. The time parameter of the Poisson process is transformed $t \mapsto \int_0^t \beta_l(\mathbf{X}(s))ds$ so that the Poisson process Y_l counts the number of jumps of size l. The birth-death chain example discussed above would be a $\mathbb{R}$-valued jump process with two Poisson processes Y_1 and Y_{-1} counting the births and deaths respectively.

2.1.3 Transient analysis

Most of the literature on CTMCs deal with the time-homogeneous case. Such CTMCs admit a unique solution to the Kolmogorov forward equation given by the matrix exponential as below.

$$P(s,t) = e^{(t-s)Q} = \sum_{k=0}^{\infty} \frac{Q^k (t-s)^k}{k!}$$

It follows that the *transient* state distribution of the CTMC at time t, denoted π_t, is given by

$$\pi_t = \pi_0 \times e^{tQ}$$

with π_0 denoting the initial probability distribution. This gives us a probability distribution over the state space of the CTMC at time t. Numerical calculations of this matrix exponential are non-trivial but a wealth of approximate numerical methods exist. A comprehensive overview of approximate numerical methods based on direct computations of the transient state distribution and, for example, uniformisation [61] methods is given in [117].

For the time-inhomogeneous case the situation is more complicated as it is no longer true that the solution to the Kolmogorov forward equation is given by $e^{\int_s^t Q(\tau)d\tau}$. The relevant analytical treatment of solutions to non-homogeneous Kolmogorov equations can be found in [51] and a presentation of non-homogeneous CTMCs in the context of simulated annealing in [118]. Although more complex, the desirable property of being able to model time-dependent behaviour through time-inhomogeneous CTMCs has led to extensions of numerical approximation techniques for transient analysis from time-homogeneous cases to time-inhomogeneous [120, 121].

In practice, however, the numerical approximation methods for transient analysis for the time-homogeneous cases as well as time-inhomogeneous cases are usually infeasible due to large numbers of states that have to be considered even in simple modelling scenarios. The Stochastic Simulation Algorithm, introduced in the context of chemical kinetics [59], gives a method for simulating exact Monte Carlo realisations of time-homogeneous CTMCs. The algorithm relies on the property of CTMCs that in each state the time to the next jump is exponentially distributed, allowing one to generate realisations of CTMC trajectories by repeatedly sampling from exponential distributions. With enough realisations the estimates for the statistical properties of the process are expected to converge to their true values. For time-inhomogeneous cases related simulation approaches presented in [7, 75, 123] have been proposed. However, in order to obtain accurate estimates for the transient state distributions one needs to gather a large number of such realisations which for complex models becomes computationally expensive. To combat this, one can appeal to model reduction techniques or continuous approximation methods. Such methods are discussed in Sections 2.3 and 2.4, respectively.

2.2 Hybrid stochastic systems

Another class of Markov processes, which is discussed in Chapters 4 and 5 of this thesis, is continuous time Markov processes $Y(t) = (\mathbf{X}(t), Z(t))$ taking values in hybrid state space $\mathcal{E} = \mathbb{R}^d \times \mathcal{Q}$ where $\mathcal{Q}$ is a finite set. Process Y is thought of as a joint process where the continuous part of dynamics is described by the variable $\mathbf{X}$ and the discrete part by Z. Two examples of this are piecewise deterministic Markov processes [43] and jump diffusion processes [94] which both describe a continuous evolution coupled with stochastic jumps. There are different ways in which one can define such stochastic processes. To be consistent with the definition of CTMCs given before we characterise these hybrid processes in terms of their transition density functions according to [12, 43, 68].

The first process we consider is commonly known as the jump diffusion process. The continuous part of the process models a fluid flow subject to random perturbations while the jump process corresponds to discrete events. The corresponding transition density is given by

$$\frac{d}{dt}p(\mathbf{x}', z'; t \mid \mathbf{x}, z; s) = \mathcal{L}^* p(\mathbf{x}', z'; t \mid \mathbf{x}, z; s) + \sum_k p(\mathbf{x}', k; t \mid \mathbf{x}, z; s) q_Z^{\mathbf{x}'}(k, z')$$

where $\mathcal{L}^*$ is the forward generator corresponding to a diffusion process such that

$$\begin{aligned} \mathcal{L}^* p(\mathbf{x}', z'; t \mid \mathbf{x}, z; s) = & -\sum_i \partial_i \left[\mathbf{F}_i(\mathbf{x}) p(\mathbf{x}', z'; t \mid \mathbf{x}, z; s)\right] \\ & + \sum_i \sum_j \partial_i \partial_j \left[\mathbf{G}_{ij}(\mathbf{x}') p(\mathbf{x}', z'; t \mid \mathbf{x}, z; s)\right] \end{aligned} \tag{2.4}$$

The above generator describes the probability density that the state of the process Y at time t is $(\mathbf{x}', z')$ given the state at time s was $(\mathbf{x}, z)$ in terms of elements $\mathbf{F}_i$ of the drift vector $\mathbf{F}$ and $\mathbf{G}_{ij}$ of the diffusion matrix $\mathbf{G}$.

The second process is an example of piecewise deterministic Markov process [43] where the corresponding transition density is derived, for example in [12], as

$$\frac{d}{dt}p(\mathbf{x}', z'; t \mid \mathbf{x}, z; s) = \mathcal{L}_1^* p(\mathbf{x}', z'; t \mid \mathbf{x}, z; s) + \sum_k p(\mathbf{x}', k; t \mid \mathbf{x}, z; s) q_Z^{\mathbf{x}'}(k, z')$$

where $\mathcal{L}_1^*$ is the forward generator corresponding to the deterministic evolution. In fact, setting the diffusion matrix $\mathbf{G}$ to 0 in Equation 2.4, thus eliminating the random perturbations from the fluid flow, gives us the desired generator.

$$\mathcal{L}_1^* p(\mathbf{x}', z'; t \mid \mathbf{x}, z; s) = -\sum_i \partial_i \left[\mathbf{F}_i(\mathbf{x}') p(\mathbf{x}', z'; t \mid \mathbf{x}, z; s)\right] \tag{2.5}$$

There are several examples of works where the hybrid processes arise as limit behaviours of CTMCs. In particular, in Chapter 4 we are going to consider [18] where piecewise deterministic Markov processes are derived as limit behaviours for pCTMCs where some sub-populations of components are present in large quantities while others in small. The limiting results are given by a piecewise deterministic Markov process where the continuous evolution approximates the dynamics of components present in high quantities while the state changes of components in low quantities correspond to discrete jumps.

2.3 Model reduction techniques

As alluded to in Section 2.1.3, in order to alleviate the computational cost of stochastic simulation of complex CTMC models there exist a large number of model reduction techniques. At a high level these techniques work by constructing and analysing a simplified model based on the original one. As one might expect such simplifications result in loss of information about the behaviour. By reasoning about the inherent properties of the original models the aim is to have the simplified model act as a good proxy for studying properties of the CTMC behaviour.

The first category of methods make use of the lumpability property of Markov chains [30, 32, 82]. Such methods partition the state-space of the CTMC into sets of states so that the new process defined by the transitions between the classes of the partition is a CTMC. In practise, such partitions often result from symmetries in the model [60]. A prime example of this comes from considering models of indistinguishable agents where a population CTMC presents an example of a partitioning of the state space to represent the counts of individual agents in each of the possible states. The second class of techniques relies on time-scale separation [23, 35, 110]. In particular, the original process in decomposed into weakly interacting fast sub-processes. These fast sub-processes are assumed to reach their equilibrium distribution independently from each other. The slow sub-process, resulting from the composition of fast sub-processes, is then simulated based on the equilibrium states of the fast sub-processes.

2.4 Continuous approximation methods

In many cases the structure of pCTMCs allows us to make use of the continuous approximation techniques presented in this section. The aim of such methods is to approximate the moments of the transient probability distribution over pCTMC states by relaxing the discrete-state process defined by the original pCTMC to a continuous-state process that in many cases admits more efficient solution methods. Here we give a brief overview of the methods of fluid, linear noise and moment closure approximations in the context of pCTMCs defined in Definition 2. A recent thorough overview of the methods from the perspective of biochemical modelling applications can be found, for example, in [114].

2.4.1 Fluid approximation

Fluid (sometimes called mean field) approximation methods describe the mean behaviour of a stochastic pCTMC model through a deterministic ordinary differential equation (ODE) model. In particular, it turns out that under certain assumptions the sample paths of the pCTMC can be guaranteed to lie, with high probability, close to the solution of a differential equation [41]. These methods have been applied in various contexts – epidemiology, modelling biological and ecological systems [24, 111], performance modelling of computer networks [72, 128] and model checking [15], to name a few. The exposition and theoretical results in this section are based on the seminal work by Kurtz [83] with the full details on the relevant theory of Markov processes being available for the interested reader in references [41] and [50].

The results presented here are asymptotic in the limit of infinite population of conservative pCTMCs. In particular, let $\mathcal{P}^N = \left(\mathbf{X}^N, \mathcal{T}^N, \mathbf{X}_0^N\right)$ with $\mathbf{X}^N \in \mathbb{Z}^K_{\geq 0}$ be a conservative pCTMC over a homogeneous population of size N. The fluid approximation of $\mathcal{P}^N$ arises from considering a change of variable that maps the pCTMC $\mathcal{P}^N$ to a process $\hat{\mathcal{P}}^N = (\hat{\mathbf{X}}, \hat{\mathcal{T}}^N, \hat{\mathbf{X}}_0^N)$ describing the evolution of normalised population variables corresponding to proportions of population in each of the K states. The assumption of having a conservative pCTMC here is used to achieve the appropriate scaling of population variables. While other assumptions can be made to derive fluid approximation results, in the context of this thesis the assumption of having a fixed population of agents is the most natural.

First, we define the rescaled state of the system as $\hat{\mathbf{X}}^N \overset{\text{def}}{=} \frac{1}{N}\mathbf{X}^N$ and analogously the initial condition as $\hat{\mathbf{X}}_0^N \overset{\text{def}}{=} \frac{1}{N}\mathbf{X}_0^N$. Secondly, the transitions in $\hat{\mathcal{T}}^N$ are of the form $\hat{\tau} = (\hat{r}_\tau^N, \frac{1}{N}\mathbf{v}_\tau)$ where $\hat{r}_\tau^N$ is the rate function over normalised state variables. For the asymptotic results on the convergence of the sequence $\{\hat{\mathbf{X}}^N\}_{N\in\mathbb{N}}$ as stated by Kurtz [83] we need the rates to be density dependent. Formally, for all $\tau \in \mathcal{T}^N$ there exists a Lipschitz continuous function $f_\tau^N : \mathbb{R}^K \to \mathbb{R}_{\geq 0}$ such that

$$r_\tau(\mathbf{X}^N) = N f_\tau^N \left(\frac{\mathbf{X}^N}{N}\right)$$

The normalised rate function is then given by

$$\hat{r}^N \left(\frac{\mathbf{X}^N}{N}\right) = f_\tau^N \left(\frac{\mathbf{X}^N}{N}\right)$$

Finally, the constructed normalised pCTMC $\hat{\mathcal{P}}^N$ is defined by a $\mathbb{R}^K$-valued CTMC $\hat{\mathbf{X}}^N$

with the jump intensities given by

$$q^N(\mathbf{i},\mathbf{j}) = \sum_{\tau \in \hat{\mathcal{T}}^N | \mathbf{v}_\tau = \mathbf{j}-\mathbf{i}} \hat{r}_\tau(\mathbf{i})$$

In particular, the rate of moving from state $\mathbf{i}$ to $\mathbf{j}$ is given by adding the rates of transitions which can cause the transition.

Under the assumption that f_τ^N converges uniformly, as $N \to \infty$, to a locally Lipschitz continuous function f_τ we can construct the drift vector

$$F(\hat{\mathbf{x}}) = \sum_{\tau \in \hat{\mathcal{T}}^N} \mathbf{v}_\tau f_\tau(\hat{\mathbf{x}})$$

and state the following theorem.

Theorem 1 (Deterministic approximation theorem [83])**.** *With* $\hat{\mathbf{X}}_0^N$ *we assume there exists a point* $\hat{\mathbf{x}}_0$ *such that* $\lim_{N\to\infty} \|\hat{\mathbf{X}}_0^N - \hat{\mathbf{x}}_0\| = 0$ *almost surely. Then for every* $t \in [0,\infty)$ *and* $\epsilon > 0$ *we have*

$$\mathbb{P}\left(\lim_{N\to\infty} \sup_{u \le t} |\hat{\mathbf{X}}^N(u) - \hat{\mathbf{x}}(u)| > \epsilon\right) = 0$$

where $\hat{\mathbf{x}}$ *is a solution to* $\frac{d\hat{\mathbf{x}}}{dt} = F(\hat{\mathbf{x}})$ *with* $\hat{\mathbf{x}}(0) = \hat{\mathbf{x}}_0$*.*

In particular, the discrete stochastic behaviour of the pCTMC is approximated by a continuous deterministic one which corresponds to the limiting behaviour of the stochastic process as the considered population size N approaches infinity.

Notably, the fluid approximation result relies on the specific structure of pCTMCs. More general structures, with emphasis on models that are “close” to having the population structure, have been considered recently in [95]. Secondly, fluid approximation results guarantee convergence to the asymptotic limit but in order for the approximation to give a good empirical approximation we need that each of the components in the population to be present in large quantities. This can be seen as a rather limiting condition from the modelling perspective. The work in [18] aims to relax that requirement by dealing with populations where one might encounter sub-populations with small or fixed sizes and uses piecewise deterministic Markov processes as the limiting process. We are going to refer to the methods that combine continuous approximations of certain variables with discrete representation of the rest as hybrid-continuous approximations.

2.4.2 Linear noise approximation

Although the fluid approximation provides a computationally very efficient way to study population dynamics by considering the limiting behaviour through a wealth of available ODE solvers, we lose information about the stochastic behaviour. While this might not be a big limitation when considering very large populations, systems with a population in the order of hundreds of individuals exhibit a behaviour that is intrinsically probabilistic. In such cases the linear noise approximation, or central limit approximation, provides an improved estimation over the fluid limit of the stochastic dynamics of the system. In particular, the stochastic fluctuations around the average deterministic behaviour, given by the fluid approximation, are approximated by a Gaussian process. In the following, we give a concise exposition of the relevant result based on [50, 122].

Define a stochastic process $\mathbf{V}^N(t) \overset{\mathsf{def}}{=} N^{\frac{1}{2}}\left(\hat{\mathbf{X}}^N(t) - \hat{\mathbf{x}}(t)\right)$ capturing rescaled fluctuations of the CTMC $\hat{\mathbf{X}}^N$ around the fluid limit $\hat{\mathbf{x}}$. One can prove that the sequence $\mathbf{V}^N$ converges in distribution to the stochastic process $\{\mathbf{V}(t) \in \mathbb{R}^n \mid t \in \mathbb{R}\}$ such that at any time t the distribution of $\mathbf{V}$ is a multivariate Gaussian with mean $\mathbb{E}(t)$ and covariance $\mathbb{C}(t)$. The mean $\mathbb{E}(t)$ and covariance $\mathbb{C}(t)$ are given as solutions to the following ODE systems

$$\begin{cases} \frac{d}{dt}\mathbb{E}(t) = \mathbf{J}_{\mathbf{F}}(\hat{\mathbf{x}}(t))\mathbb{E}(t) \\ \mathbb{E}(0) = 0 \end{cases}$$

$$\begin{cases} \frac{d}{dt}\mathbb{C}(t) = \mathbf{J}_{\mathbf{F}}(\hat{\mathbf{x}}(t))\mathbb{C}(t) + \mathbb{C}(t)\mathbf{J}_{\mathbf{F}}^T(\hat{\mathbf{x}}(t)) + \mathbf{G}(\hat{\mathbf{x}}(t)) \\ \mathbb{C}(0) = 0 \end{cases}$$

where $\mathbf{J}_{\mathbf{F}}(\hat{\mathbf{x}}(t))$ denotes the Jacobian of the limit drift $\mathbf{F}$ calculated along $\hat{\mathbf{x}}(t)$ and

$$\mathbf{G}(\hat{\mathbf{x}}) = \sum_{\tau \in \hat{\mathcal{T}}^N} \mathbf{v}_\tau \mathbf{v}_\tau^T f_\tau(\hat{\mathbf{x}})$$

Theorem 2 (Linear noise approximation theorem)**.** *With the above notation suppose that*

$$\lim_{N\to\infty} \mathbf{V}^N(0) = \mathbf{V}_0$$

Then $\mathbf{V}^N(t)$ converges in distribution to $\mathbf{V}(t)$.

The immediate consequence of the above theorem is that $\hat{\mathbf{X}}^N(t)$ converges in distribution to $\tilde{\mathbf{X}}(t) = N^{-\frac{1}{2}}\mathbf{V}(t) + \hat{\mathbf{x}}(t)$. In particular, the linear noise approximation takes the deterministic approximation and perturbs it with Gaussian noise. An equivalent characterisation of the limiting process is stated by the following stochastic integral equation

$$\tilde{\mathbf{X}}(t) = \tilde{\mathbf{X}}_0 + \int_0^t F(\tilde{\mathbf{X}}(s))ds + N^{-\frac{1}{2}} \int_0^t \mathbf{G}(\tilde{\mathbf{X}}(t))d\mathbf{W}_s \tag{2.6}$$

where $\mathbf{W}_s$ is the n-dimensional Wiener process. Note that the linear noise approximation is again stated in terms of limiting behaviour as the population size grows. However, the results, by incorporating a noise model, aim to give a better understanding of the system's behaviour for moderately sized populations than the fluid approximation. As with fluid approximation a large number of works have applied such results. For example, in recent years there have been a number of works applying the results in the model checking context [19, 22, 36].

2.4.3 Moment closure approximations

The last class of continuous approximation techniques we are going to discuss are the moment closure methods. Such methods have been popular in the study of stochastic effects in chemical reaction networks modelled by the chemical master equation. Note that the master equation is equivalent to the Kolmogorov forward equation for CTMCs in Definition 1 and is simply the terminology used in biochemical and biological modelling literature. Using the notation introduced for pCTMCs before we can formulate the so-called master equation for a population CTMC $\mathcal{P} = (\mathbf{X}, \mathcal{T}, \mathbf{X}_0)$ in the following way

$$\frac{\partial}{\partial t} p(\mathbf{X}(t)) = \sum_{\tau \in \mathcal{T}} r_\tau(\mathbf{X}(t) - \mathbf{v}_\tau) p(\mathbf{X}(t) - \mathbf{v}_\tau) - \sum_{\tau \in \mathcal{T}} r_\tau(\mathbf{X}(t)) p(\mathbf{X}(t)) \tag{2.7}$$

where the $\mathbf{v}_\tau$ are the changes induced in the state vector $\mathbf{X}$ by some action τ and r_τ corresponds to the rate at which the action τ is expected to happen.

Instead of attempting to solve the Equation 2.7 directly, which is usually not possible, one can attempt to give a description of the resulting time-evolution of the probability distribution $p(\mathbf{X}(t))$ in terms of its moments [6]. In particular,

$$\frac{d}{dt} \mathbb{E}[h(\mathbf{X}(t))] = \sum_{\tau \in \mathcal{T}} \mathbb{E}[(h(\mathbf{X}(t) + \mathbf{v}_\tau) - h(\mathbf{X}(t)) r_\tau(\mathbf{X}(t)))]$$

where h is a chosen polynomial function. The above equation for moments follows from the application of Dynkin formula [101] to the pCTMC. The exact ODEs for the moments of $\mathbf{X}$ can then be found by choosing an appropriate polynomial function h. For example, choosing $h : \mathbf{X}(t) \mapsto \mathbf{X}(t)$ gives us the mean of the process $\mathbf{X}(t)$

$$\frac{d}{dt} \mathbb{E}[\mathbf{X}(t)] = \sum_{\tau \in \mathcal{T}} \mathbf{v}_\tau r_\tau(\mathbf{X}(t))$$

Similarly, we can recover the evolution of covariance $\mathbb{C}[\mathbf{X}(t)]$ by choosing $h : \mathbf{X}(t) \mapsto \mathbf{X}^2(t)$ and noting that by definition

$$\mathbb{C}[\mathbf{X}(t)] = \mathbb{E}\left[\mathbf{X}^2(t)\right] - \mathbb{E}[\mathbf{X}(t)]^2$$

The difficulties arise if the transition functions r_τ are anything other than linearly dependent on the state variables. The moment expressions above then depend on higher order moments resulting in an infinite system of coupled differential equations. Moment closure methods like normal [125], lognormal [116] and Poisson [100] closure put different assumptions on the properties of the transient distribution to derive a closed set of moment equations. For example, normal moment closure assumes a multivariate normal distribution for the population variables. The general downside of moment closure methods, in common with the other continuous approximation methods, is that whether or not the assumptions are well founded depends on the problem at hand and is generally difficult assess a priori. A comparison of different moment closure methods in the context of stochastic chemical kinetics can found in [114]. In the context of stochastic process algebra models moment closure methods have, for example, been applied in [53, 63]. The hybrid-continuous version of moment closure approximation is presented in [65] and is going to be discussed in more detail in Chapter 4.

2.5 Formal modelling

Direct construction of Markov chain models for complex systems is often cumbersome and error-prone. To deal with that problem there exist a number of formal modelling languages that allow for automated construction of CTMC models from a high-level specification via well-defined semantic rules.

One large class of examples consists of stochastic process algebras like PEPA [71], sCCP [16], stochastic π calculus [107] and Carma [88], which are based on classical process algebras CCS [96] and CSP [74]. These languages give a high-level tool for the description of interactions, communications and synchronisations between collections of agents or processes. While languages like CCS and CSP provide a qualitative description of the system — possible configurations of the system (system states) and actions denoting transitions between the configurations — the stochastic process algebras aim to incorporate quantitative information. In particular, the actions are associated with delays. The above examples, notably PEPA which was one of the first process algebras to incorporate quantitative timing properties, assume that the duration of actions follows the exponential distribution making them suitable for describing complex CTMCs. When modelling delays, other assumptions could be made, which would give rise to non-Markovian processes. A large amount of research has been concerned with extending process algebras to model different aspects like location dependent behaviour (PALOMA [52], MELA [89]) and attribute based communication (Carma [88], attributed π [78], SCEL [45]) to create more expressive languages.

Notable alternative approaches to stochastic process algebras are stochastic Petri nets [10] which provides a straightforward graphical notation for the description of step-wise processes, chemical reaction networks [76, 37] and rule-based modelling paradigms like Kappa [27] and ML-rules [92].

In the above we have concentrated on quantitative modelling formalisms where the underlying mathematical model is a CTMC. There exist many formal modelling languages that use alternative semantics. The most relevant ones to this thesis model continuous time processes. From the process algebra side there are, for example, HYPE [55] and HyPA [40] which aim to capture hybrid systems. In such systems the dynamics of the components exhibit both discrete and continuous behaviour. In addition, Petri net-based formalisms, like hybrid Petri nets [42, 62] and fluid stochastic Petri nets [77], have been proposed over the years and combine deterministic continuous evolution with discrete stochastic transitions.

For model specification, the work in this thesis concentrates on stochastic process algebras and, in particular, Carma. The language features a set of communication primitives that, in conjunction with attribute-based filtering of communication partners, are capable of capturing a versatile set of communication behaviours. The set of communication primitives in Carma correspond to broadcast and unicast making it particularly suitable for open collectives, like robot swarms, where the participants of the communications cannot be known ahead of time.

2.5.1 Collective adaptive resource-sharing Markovian agents

In the following, we introduce a particular stochastic process algebra named Carma [88], which was developed as part of the QUANTICOL project [3]. Carma will serve as the basis for constructing models of collective dynamics in this thesis. Carma is an expressive high-level language that was designed for modelling and reasoning about Collective Adaptive Systems (CAS) where heterogeneous populations of autonomous agents cooperate towards common goals. The distinctive features of the Carma language are a rich set of communication primitives enabling attribute-based broadcast and unicast communication, a way to specify the behaviour agents depending on the locally available information and explicit representation of the environment for testing under different scenarios. The language, along with the developed analysis tools (available as an Eclipse Plug-In [1]) have been used, for example, in modelling pedestrian movement [56], urban transportation services [129] and availability of cloud services [90].

Syntax

Here we follow the most recent description of the syntax of the language given in [88] where the language has the structure as depicted in Figure 2.1. In particular, a *system* S in Carma consists of a *collective* N operating in an *environment* $\mathcal{E}$ denoted

$$S \stackrel{\mathsf{def}}{=} N \ \mathbf{in} \ \mathcal{E}$$

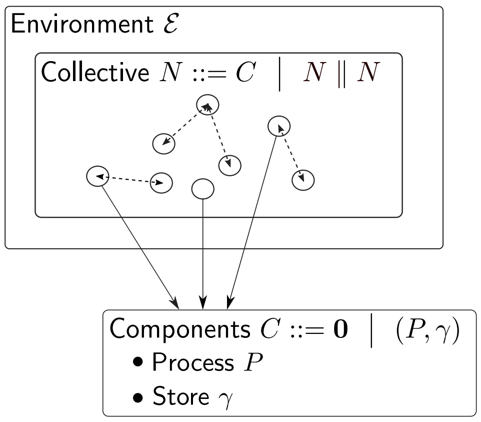

Environment $\mathcal{E}$
Collective $N ::= C \mid N \parallel N$
Components $C ::= \mathbf{0} \mid (P, \gamma)$
• Process P
• Store γ

Figure 2.1: Structure of Carma models.

The *collective* N consists of a set of interacting *components* C, or *agents*, modelling the behaviour of the system.

$$N \overset{\mathsf{def}}{=} C \mid N \parallel N$$

The description of the *components* consists of two parts. First, the behaviour of the component depends on the *knowledge* available to it, captured in its local *store* γ. The behaviour of the *component* is defined by the *process* P. In particular,

$$P \overset{\mathsf{def}}{=} \mathbf{0} \mid (P, \gamma)$$

The precise meaning of a store is a mapping from a set of attribute names to a set of values

$$\gamma \overset{\mathsf{def}}{=} \{a_1 \mapsto v_1, \ldots, a_n \mapsto v_n\}$$

For example, in many scenarios the spatial distribution of agents can be taken into consideration by defining a location attribute for the agents. In the semantics, the attributes are used to provide support for attribute-based communication and behaviour. For example, if two agents are too far from each other their ability to communicate with each other might be limited. The processes, defining the possible behaviours of individual agents, are given by the following grammar

$$
\begin{aligned}
P,Q &\overset{\text{def}}{=} \mathbf{nil} \\
&\mid act.P \\
&\mid P+Q \\
&\mid P \parallel Q \\
&\mid [\pi]P \\
&\mid A \qquad (A \overset{\triangle}{=} P)
\end{aligned}
\qquad
\begin{aligned}
act &\overset{\text{def}}{=} \alpha^{*}[\pi_s]\langle\vec{e}\rangle\sigma \\
&\mid \alpha[\pi_r]\langle\vec{e}\rangle\sigma \\
&\mid \alpha^{*}[\pi_s](\vec{x})\sigma \\
&\mid \alpha[\pi_s](\vec{x})\sigma \\
e &\overset{\text{def}}{=} a \mid \mathsf{my}.a \mid x \mid v \\
\pi_s,\pi_r,\pi &\overset{\text{def}}{=} \top \mid \bot \mid \pi \bowtie \pi \mid \neg\pi \mid \pi \wedge \pi \mid \cdots
\end{aligned}
$$

In particular, processes are defined using standard constructs from process algebras literature — action prefix (.), choice (+), and parallel composition (‖). In addition, the definition of the inactive process **nil**, process **kill** that removes the component from the system and guards on processes are allowed. The action primitives are defined for broadcast and unicast output in the forms $\alpha^{*}[\pi_s]\langle\vec{e}\rangle\sigma$ and $\alpha[\pi_s]\langle\vec{e}\rangle\sigma$, respectively, and for broadcast and unicast input in the forms $\alpha^{*}[\pi_r](\vec{x})\sigma$ and $\alpha[\pi_r](\vec{x})\sigma$, respectively. The following notation is used:

- α denotes an action type which is used to distinguish between different actions.
- π_s, π_r, π denote boolean predicates that have to be satisfied before the action can be executed. As mentioned previously, the communication in Carma is attribute-based and thus such guards are used to filter out communication partners based on attributes such as location or communication range.
- e is an expression built using appropriate combinations of values, attributes and variables. In the semantics, the expressions are evaluated over the sending component's local store and passed on to the receiving component.
- x is a variable which takes on the values that were communicated to the receiving process by the sender.
- σ is a function from $\Gamma \to Dist(\Gamma)$ where $Dist(\Gamma)$ is the set of probability distributions over set of possible stores Γ. The function σ thus denotes a store *update* and defines how the given store is changed as a result of an action.
- $\vec{\cdot}$ notation is used to indicate a sequence of elements.

To give a small example for the notation we consider the following broadcast output and input actions.

$$
\begin{gathered}
send^{*}[\mathsf{loc} == \mathsf{my.loc}]\langle 1\rangle\{\circ\} \\
send^{*}[\circ](x)\{\mathsf{message} \mapsto x\}
\end{gathered}
$$

The action type *send** is used to distinguish the action from other broadcast actions in the system. The filter of the output action makes sure that the message 1 can only be received by a component with the same location attribute loc as the sender. Finally, the update defined in the input action writes the received message to the receiver's local store.

In relation to the components forming a collective the *environment* models all aspects that are intrinsic to the context in which the components operate and mediates the interactions between the *components*. For example, the *environment* deals with the idea that where the *component* is will have an effect on what it can do. This is achieved by letting the environment determine the parameters defining the duration and the probability of success of actions.

In detail, an environment is defined by a *global store* γ_g that models the overall state of the system and an *evolution rule* ρ. The evolution rule gives, depending on the *current time*, the global store and the current state of the collective, a tuple of functions $\varepsilon = \langle \mu_p, \mu_w, \mu_r, \mu_u \rangle$ called the *evaluation context*. The functions composing the evaluation context regulate the system behaviour by setting rates of actions, probabilities of receiving broadcast and unicast messages and how the environment changes after an action is performed.

- $\mu_p(\gamma_s, \gamma_r, \alpha)$ gives the probability that a component with local store γ_r can receive a broadcast message from a component with local store γ_s when α is executed.

- $\mu_w(\gamma_s, \gamma_r, \alpha)$ defines the weight, which is used to compute the probability that a component with store γ_r can receive a unicast message from a component with store γ_s when α is executed. A unicast message can only be received by a single component in the system. However, a priori there may be more than one eligible receiver for the message. The probability of a given receiver being chosen as the communication partner results from normalising the weight over the sum of weights of all possible receivers.

- $\mu_r(\gamma_s, \alpha)$ computes the execution rate of the action α executed at a component with store γ_s.

- $\mu_u(\gamma_s, \alpha)$ defines the updates on the environment (global store and collective) induced by the action α of the component with store γ_s.

Semantics

Formal semantics of stochastic process algebras are usually defined via *Structured Operational Semantics (SOS)* in the style proposed by Plotkin [105]. Such semantics consist of a collection of rules in the form

$$\frac{Premise_1, \cdots, Premise_n}{Conclusion}$$

giving rise to *Labelled Transition Systems* defining a set of states, set of actions and the corresponding set of transition relations. In the context of stochastic process algebras this structure can usually then be easily translated to a numerical CTMC model or simulated directly via the Stochastic Simulation Algorithm.

The semantics of the Carma language is defined in the FuTS (state-to-function transition systems) framework [44] which offers a more compact representation of operational semantics compared to small step SOS. The formal rules are used to give precise meaning or semantics to the syntax in the previous section in terms of time-inhomogeneous CTMCs. In general, the transition rules in FuTS are given as triplets $s \overset{\lambda}{\rightarrowtail} f$ where s denotes a source state, λ the label of the transition and f the continuation function associating each state s' with some value. In the case of Carma, and other stochastic process algebras, the continuation functions would give, for example, the transition rate from the state s to any other state of the system. The formal semantics of Carma, as presented in [88], are not going to be replicated here but serve as a template for the semantics presented in Chapter 3. Another application of FuTS can be seen later in Chapter 3 where we extend the Carma semantics to cover cases of non-determinism for uses in the context of control problems for collective dynamics.

2.5.2 Model checking

One of the important problems considered in the field of formal methods is model checking. We can think of a model as representing our understanding of the system under study. Model checking techniques aim to check whether or not this model satisfies a specification given in terms of logic formulae. A classical example of such a specification from concurrency theory is freedom from deadlocks which ensures that in a set of processes accessing a shared resource there is a process that will eventually make progress. The specifications are formulated in one of the many precisely defined logical frameworks like Continuous Stochastic Logic [8], Probabilistic Computation Tree Logic [64] or Linear Temporal Logic [106] with tools like PRISM [84] and STORM [47] providing an interface between model checking techniques and model specifications in formal languages. While much of the work on model checking has concentrated on temporal properties, features beyond temporal ones, like space, have also been subject to study [54, 97].

Note that exact model checking algorithms for CTMCs rely on transient and steady state probability analysis. As already mentioned, this presents scalability issues when considering complex models of large systems. An alternative is provided by statistical model checking where finitely many sample trajectories of a stochastic system are used to infer whether the samples provide statistical evidence for the satisfaction of the logical specification [5, 86].

Finally, much of the interest in the formal methods community into stochastic approximation methods like fluid, central limit and moment closure approximation have been motivated by conducting efficient model checking on population models [19, 21, 22, 36].

2.6 Stochastic decision processes

The exposition so far has concentrated on construction of CTMC models for which the dynamics are fully specified. However, in many applications we have to consider sources of non-determinism in the behaviour of the systems. This has led to the development of extensions to Markov chains that aim to incorporate non-determinism into the model description. Notable examples like Interval Markov Chains (IMC) [79] and Constraint Markov Chains (CMC) [34] are based on the discrete time version of Markov chains. The non-determinism in those models results from the idea that the transition parameters are constant but we might be uncertain about the exact values. A similar idea in the context of CTMCs was considered in [58] to incorporate data and uncertainty into process algebra models in a meaningful way.

In this thesis we opt for another view on non-determinism with a long history, where the non-determinism is defined by the possible behaviours of a controller or a decision maker. In particular, continuous time Markov decision processes and their discrete time counterparts [108] have seen applications in the control of queueing systems [98, 124], epidemic and population processes [46, 85] making them a natural choice for an underlying mathematical model to bridge the gap between formal languages and control problems in collective dynamics.

Definition 4. *A continuous-time Markov decision process (CTMDP) is defined by the tuple*

$$\{\mathcal{S}, \mathcal{A}, q(i,j \mid a)\}$$

where $\mathcal{S}$ is the countable set of states, $\mathcal{A}$ is the set of actions and $q(i,j \mid a)$ gives the transition rates $i \to j$ given the control action a.

Throughout this thesis we are going to denote the feasible set of actions from state $i \in \mathcal{S}$ as $\mathcal{A}(i)$ and assume that $\mathcal{A} = \bigcup_{i \in \mathcal{S}} \mathcal{A}(i)$. We make use of this in Chapter 3 where the action space of a CTMDP is constructed by specifying the feasible actions for each state of the model. In terms of the action space we are not restricting our view to discrete action spaces but rather allow any set of actions over which a probability distribution can be defined.

The evolution of CTMDPs is described by the following: after the process reaches some state and an action is chosen, the process performs a transition to the next state depending only on the current state and the chosen action. The time it takes for state transitions to happen is governed by an exponential distribution with a rate given by the function q in Definition 4. The actions at every such step are chosen according to some policy as defined below.

Definition 5. *A policy is a measurable function $\pi : \mathbb{R}_{\geq 0} \times \mathcal{S} \times \mathcal{A} \to [0,1]$ which for every time $t \in \mathbb{R}_{\geq 0}$, state $s \in \mathcal{S}$ and action $a \in \mathcal{A}(s)$ assigns a probability $\pi(t,s,a)$ that the action a is chosen in s at time t. In other words policy π defines a distribution over actions in any state of the CTMDP at time t. We call a policy where for every $t \in \mathbb{R}_{\geq 0}$ and $s \in S$ we have that $\pi(t,s,a) \in \{0,1\}$ a deterministic policy. A policy π independent of t is a stationary policy.*

Analogously to CTMCs one can define the population version of the CTMDPs.

Definition 6. *A population CTMDP (pCTMDP) is a tuple $(\mathbf{X}, \mathcal{T}, \mathcal{A}, \beta)$ defined by:*

- $\mathbf{X} = (X_1, \cdots, X_n) \in \mathcal{S} = \mathbb{Z}^n_{\geq 0}$ *where each X_i takes values in a finite domain $\mathcal{D}_i \subset \mathbb{Z}_{\geq 0}$.*
- *β is a function such that $\beta(a,\mathbf{X})$ returns a boolean value indicating whether action $a \in \mathcal{A}$ is available from state $\mathbf{X}$.*
- *$\mathcal{T}$ is a set of transitions of the form $\tau = (a, \mathbf{v}_\tau, r_\tau(\mathbf{X}))$ such that $\beta(a,\mathbf{X}) = 1$, $\mathbf{v}_\tau$ is an update vector specifying that the state after execution of transition τ is $\mathbf{X} + \mathbf{v}_\tau$ and $r_\tau(\mathbf{X})$ is a rate function.*

In order to give semantics to the above definition of a population CTMDP we associate it with the equivalent CTMDP in the following way:

- the state and action space of the corresponding CTMDP is the same as for the population CTMDP.
- the set of feasible actions for state $\mathbf{i} \in \mathcal{S}$, denoted $\mathcal{A}(\mathbf{i})$, is defined by

$$\mathcal{A}(\mathbf{i}) = \{a \in \mathcal{A} \mid \beta(a,\mathbf{i}) = 1\}$$

- the rate function q is defined as

$$q(\mathbf{i},\mathbf{j} \mid a) = \sum_{\tau \in \mathcal{T}, \tau = (a, \mathbf{v}_\tau, r_\tau(\mathbf{j})), \mathbf{i} = \mathbf{j} + \mathbf{v}_\tau} r_\tau(\mathbf{i})$$

Note that both in the case of CTMDPs and pCTMDPs fixing a policy π resolves the non-determinism in the model with the resulting dynamics corresponding to a CTMC or a pCTMC respectively. The final piece missing for defining an optimisation problem is a reward or a cost function which maps a chosen policy to a real value. There are many ways in which a cost or reward can be defined. A common approach for defining a reward function, for example, is as a function of the expected behaviour of the resulting CTMC or pCTMC. As an example, suppose a pCTMC resulting from policy π consists of two counting variables, X_1^π, X_2^π, and it is desirable to have $X_1^\pi > X_2^\pi$. A simple instance of a reward function in terms of the expected behaviour can be given by associating policies π such that $\mathbb{E}(X_1^\pi) > \mathbb{E}(X_2^\pi)$ with reward 1 and 0 otherwise.

The two related problems considered in the context of CTMDPs are model checking and optimisation. Model checking is considered in the formal methods community and often relates to the computation of time-bounded reachability properties [9, 31, 33]. In particular, what is the probability of reaching a given state within some time bound. Optimisation on the other hand has a long history, for example in operations research, and deals with optimal policies given a reward or a cost function on the CTMDP trajectories [29, 112].

Chapter 3

Model specification

One of the main motivations for the construction and study of stochastic models of engineered systems is the desire to learn how to better optimise aspects of the system's behaviour. For example, in the case of performance modelling one concentrates on the quality of service (QoS) measures like service availability, throughput and latency or resource usage. As discussed in Section 2.6, CTMDP models are widely used in various stochastic optimisation problems in reinforcement learning and operations research. We argue that, in the formal modelling context, the CTMDP semantics are a natural extension of the more common CTMC semantics for high-level compositional descriptions of optimisation problems. Thus, the aim of this chapter is to provide a view of process algebraic model specification where optimisation problems are constructed directly from the syntactic description of the model.

In particular, we concentrate our efforts on the CTMDP dynamics of population processes motivated by the study of collective adaptive systems and propose a novel semantics for the Carma process algebra that introduces non-determinism into the derived transition system. The introduced non-determinism will have two intuitive interpretations:

- The non-determinism arises from the possible control actions that can be taken by components at the local level or by a global policy maker at the system level.
- The non-determinism arises from the parameters governing the actions being unknown.

While expressed in the same CTMDP framework the differences arise from how the two types of non-determinism are treated in the analysis of the resulting models. We are going to mostly concentrate on the first interpretation where the interest lies in finding optimal or approximately optimal policies given some well-defined optimality criterion. However, we are also going to make a note that uncertainty in the parameters can easily be considered in the context of the same framework both independently or combined with the interpretation of non-determinism as possible control actions. For example, the following are possible when considering uncertainty in parameters.

- Observation data from a real system is used in the cost function for optimisation in order to parametrise a model in a way that gives a good representation of the observed system's behaviour. Such parameter synthesis problems are common, for example, in modelling of biological systems. That is, the possible ways to resolve non-determinism can be differentiated based on how well they fit the data available from the real system we are modelling.

- A combination of parameters can be considered — those corresponding to control actions and those corresponding to unknown parameters in the system. Interesting optimisation problems would then arise from synthesising optimal policies given some aspects of the system's behaviour remain uncertain.

Further study of these cases will be left for further work and, instead, in this thesis we will mostly give examples for the interpretation of non-determinism in the constructed models as possible control actions.

In this chapter we are going to consider the existing Carma process algebra as a starting point and endow it with alternative semantics based on CTMDP models. We adopt the Carma syntax given in Section 2.5.1 without changes and extend the FuTs style semantics presented in [88] to the CTMDP based framework. The high-level idea for the semantics is that the control actions are encoded in terms of attributes in knowledge stores where the values of such attributes are left partially specified — instead of particular values we define the value domains for the attributes. The idea for the semantics appears in [104].

3.1 Semantics

In order to distinguish between the CTMC and CTMDP semantics of Carma we refer to Carma-C when discussing the language equipped with CTMDP semantics. To begin we are going to follow [88] and introduce some additional notation referring to the sets of syntactic structures of interest. Let Sys be the set of systems S, and Col be the set of collectives N, where a collective is either a component C in the set of components Comp or the parallel composition of collectives.

In order to relate syntactic descriptions of Carma-C models to CTMDPs we are going to define the set of admissible controls via the stores. Instead of fully specifying store variables, as done in Carma, we allow them to take values in a general set defined in the model. The set of feasible actions in the underlying CTMDP (as in Definition 4) then corresponds to possible refinements of store variables to particular values. Throughout this chapter we are going to use the following example to illustrate the presented semantics. Before presenting the example let us make a comment about notation. We are going to use $\circ$ to denote trivial contents and expressions. In the case of message contents $\circ$ would denote an empty message and in the case of filters and guards it denotes trivial expressions always evaluating to true.

Example 2. *Let us consider a simple foraging-inspired scenario where robots need to locate a target by exploring and sensing their local environment and then move to the location of the target. Components in the system correspond to the robots with the following behaviour: robots can move on a grid, take measurements from their location and broadcast this information to the rest of the swarm. The model we use to describe the behaviour of the individual robots is illustrated in Figure 3.1. The exploring behaviour is modelled through the robots performing a random walk over a finite set of locations with a specified graph topology. This behaviour is specified through the action random*. Once the target is known to the robots the movement towards it is specified by the directed walk modelled by the action directed*. The action sense* models the robot taking measurements of its locale and broadcasting the result to the swarm – the current location is broadcast if the location corresponds to the target and an empty message is sent otherwise. The addition of the broadcast input action for sense* makes each robot capable of sending and receiving the information about the target location. The action fail*, resulting in the robot not performing further actions, models failure.*

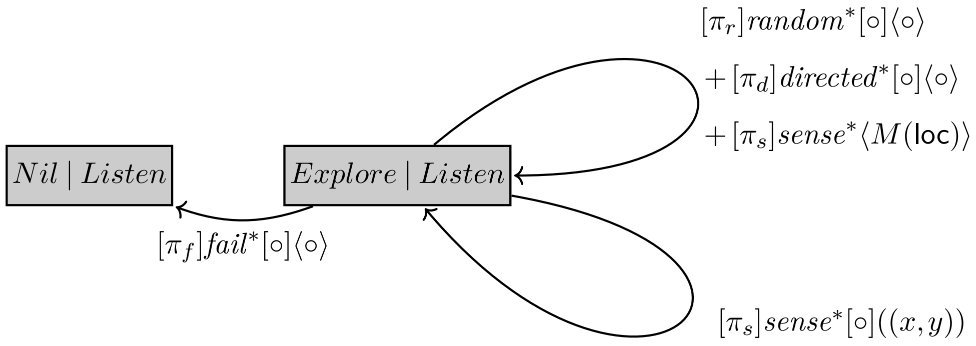

$$\begin{tikzcd}
Nil \mid Listen & Explore \mid Listen \arrow[l, "{[\pi_f] fail^*[\circ]\langle\circ\rangle}"] \arrow[loop above, "{[\pi_r] random^*[\circ]\langle\circ\rangle + [\pi_d] directed^*[\circ]\langle\circ\rangle + [\pi_s] sense^*\langle M(\mathsf{loc})\rangle}"] \arrow[loop below, "{[\pi_s] sense^*[\circ]((x,y))}"]
\end{tikzcd}$$

Figure 3.1: Behaviour of individual *Robot* components.

3.1.1 Store

The store in CARMA is a function that maps attribute names to particular values which are then used in the semantics for the transition rate calculations. In CARMA-C, we instead define the store as a function that maps attribute names to permitted value domains. Thus, a store γ maps a set of attribute names $a_0,\dots,a_n$ in its domain to the value domains of the attributes. This introduces non-determinism in the choice of particular store values. In order to make this idea precise we first concentrate on the definition of a store. In particular, let

- $\textsc{Attr}$ be the set of *attribute names* $a_0, a_1, \dots$;
- $\textsc{ValDom}$ be the set of *value domains* $V_0, V_1, \dots$ which define the sets of allowable values for the store variables.
- For a subset $A \subseteq \textsc{Attr}$ we define the store $\gamma_A : A \to \textsc{ValDom}$ as a function mapping the attributes to their respective value domains. We denote the space of all possible stores by Γ. We are going to use the following notation: if $a_0, \cdots, a_n \in A \subseteq \textsc{Attr}$ then the corresponding store γ_A is given by

  $$\{a_0 \mapsto V_0, \cdots, a_n \mapsto V_n\}$$

  for some value domains $V_0, \cdots, V_n$.

The idea above encompasses the original Carma semantics as fully determined store attributes can be modelled by value domains defined by singleton sets.

Example 3. *For the running example we define the local store of each robot consisting of attributes* loc *giving the location of the robot, and* target *holding the set of locations identified as targets. Figure 3.2 illustrates the effects of actions on the local stores. We define actions random* and directed* so that they change the location of the robot. The former picks, with uniform probability, a target location reachable from location (x,y), such that the corresponding update is denoted by $R(x,y)$. The latter picks a location that takes the robot closer to the closest target in L, where the update is denoted by $D((x,y),L)$. If the robot's location corresponds to a target location the action sense* sends the coordinates of the location otherwise it broadcasts an empty message. Denote by $M((x,y))$ the function that evaluates to empty set if the location (x,y) is not a target and to $\{(x,y)\}$ if it is. In addition the message updates the sender's own store of known targets while the corresponding input action adds the received message to the list of known targets.*

We note here that all updates act element-wise on all the store values. In order to make sure that for example the location of the robot is always known exactly we simply need to take care that the value domain corresponding to location always gets updated to another singleton value domain.

We use guards to activate and deactivate processes depending on the state of the store. Although it is natural to describe the guards here we have to keep in mind that in

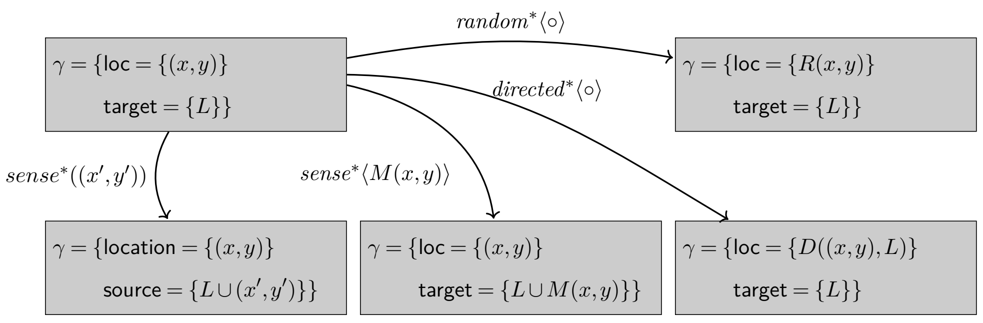

$random^*\langle\circ\rangle$
$directed^*\langle\circ\rangle$
$sense^*((x',y'))$
$sense^*\langle M(x,y)\rangle$
$\gamma = \{\mathsf{loc} = \{(x,y)\}$ $\mathsf{target} = \{L\}\}$
$\gamma = \{\mathsf{loc} = \{R(x,y)\}$ $\mathsf{target} = \{L\}\}$
$\gamma = \{\mathsf{location} = \{(x,y)\}$ $\mathsf{source} = \{L \cup (x',y')\}\}$
$\gamma = \{\mathsf{loc} = \{(x,y)\}$ $\mathsf{target} = \{L \cup M(x,y)\}\}$
$\gamma = \{\mathsf{loc} = \{D((x,y),L)\}$ $\mathsf{target} = \{L\}\}$

Figure 3.2: Local component store changes induced by actions.

the semantics the evaluation of them happens after fixing a control action. Specifically, the guard π_r for random* is true when the attribute **target** *corresponds to the empty set – target location has not yet been found. Conversely, the guard for directed* is true when the attribute* **target** *defines a non-empty set of locations. The guard π_f for fail* evaluates to true everywhere except when the robot is at the target location.*

With the above set-up the processes Explore and Listen in Figure 3.1 are precisely given by the following syntactic expressions:

$$\begin{aligned}
Explore \overset{\mathsf{def}}{=} & [\pi_r]random^*[\circ]\langle\circ\rangle\{\mathsf{loc} \mapsto R(\mathsf{loc})\}.Explore + \\
& + [\pi_d]directed^*[\circ]\langle\circ\rangle\{\mathsf{loc} \mapsto D(\mathsf{loc},\mathsf{target})\}.Explore \\
& + sense^*[\circ]\langle M(\mathsf{loc})\rangle\{\mathsf{target} \mapsto \mathsf{target} \cup M(\mathsf{loc})\}.Explore \\
& + fail^*[\circ]\langle\circ\rangle.\mathbf{nil} \\
Listen \overset{\mathsf{def}}{=} & sense^*[\circ](x)\{\mathsf{target} \mapsto \mathsf{target} \cup x\}.Listen
\end{aligned}$$

The component Robot is given by the parallel composition of the two processes and a store defining the value domains for the attributes loc *and* target*.*

$$Robot \overset{\mathsf{def}}{=} (Listen \parallel Explore, \{\mathsf{loc} \mapsto \{(x,y)\}, \mathsf{target} \mapsto \{\emptyset\}\})$$

Note that in this case both of the value domains are defined to contain a single value — location (x,y) in the case of attribute loc *and the empty set in the case of attribute* target*. The store updates above describe the possible evolutions of the store*

$$\{\mathsf{loc} \mapsto \{(x,y)\}, \mathsf{target} \mapsto \{\emptyset\}\}$$

For example, in the case of the action random* we apply the random function R to each location in the value domain of loc *(in this case containing just the location (x,y)). This gives us the distribution over the stores of the form*

$$\{\mathsf{loc} \mapsto \{R(x,y)\}, \mathsf{target} \mapsto \{\emptyset\}\}$$

that can result from the action random*.

3.1.2 Environment

As in Carma, the environment in Carma-C models aspects intrinsic to the context, where the agents under consideration are operating. It sets the rates of actions and mediates the interactions between components. For a system $S \in \textsc{Sys}$ we say that the environment $\mathcal{E}$ is defined by the global store γ_g and an evolution rule ρ. The evolution rule ρ is defined as a function which, given any refinement or resolution of the store values in the system definition, the process state of the collective and the current time, returns a tuple of $\varepsilon = \langle \mu_p, \mu_w, \mu_r, \mu_u \rangle$ called the evaluation context. The functions given by the evaluation context are interpreted as follows: μ_p expresses the probability of receiving a broadcast; μ_w associates each unicast interaction with a weight that is used to calculate the probability that a given component receives the unicast message; μ_r specifies the execution rates of actions; μ_u determines the updates on the environment store and the collective. For example, these updates can be used to introduce arrivals of new components into the collective. In order to precisely define the evaluation context, however, we need to define what exactly is meant by the instantiation or resolutions of the stores with non-deterministic values. Before doing that let us continue the running example by defining the environment for it.

Example 4. *For the running example we define the following environment: the global store γ_g defines store attributes* $\mathsf{failr} \in [0, \infty)$ *and* $\mathsf{senser} \in [0, \infty)$ *corresponding to the failure rate of individual robots and the rate of broadcast output action sense*; the probability of receiving the broadcast for the sense* action is set to 1; the constant rate of spontaneous actions random* and directed* is given by r_m; we set μ_u such that the global store remains unchanged through the evolution and no new components are introduced. The non-determinism in the behaviour of the system arises as only the viable ranges of rates for actions sense* and fail* are given rather than a particular value or a distribution over the values that would allow probabilistic choice to be made. Note that the example considers both types of non-determinism mentioned in the introduction to this chapter. The failure rate of robots is expected to be not known while the rate of sensing is a decision about the behaviour of the robots we should expect to make, and optimise for, when designing the behaviour. As mentioned in the introduction of this chapter an appropriate resolution to non-determinism could also relate to observed dynamics of a real system. For example, we might have experimental data for the failure rate of robots.*

3.1.3 Resolving non-determinism

In order to make sense of the evaluation context we first discuss how the non-determinism in the model is resolved. In the context of policy synthesis problems there are going to be better and worse ways to resolve non-determinism based on the defined optimality criteria. This aspect will be discussed in Section 3.2. At this stage we are only going to consider how possible ways to resolve the non-determinism in a model are treated in the context of the semantics of CARMA-C. To that end let us consider a system S defined by

$$S \overset{\mathsf{def}}{=} (P_1, \gamma_1) \mid \cdots \mid (P_n, \gamma_n) \textbf{ in } (\gamma_g, \rho)$$

The set of control actions $\mathcal{A}(S)$ available from S is defined by the following: let $\mathcal{A}(S)$ be a set of functions such that for all $\gamma \in \{\gamma_1, \cdots, \gamma_n, \gamma_g\}$, $f \in \mathcal{A}(S)$ maps all attributes a in the domain of γ to particular value in $\gamma(a)$. The following notation

$$f(\gamma)(a) = x$$

denotes the value of attribute $a \in \textsc{Attr}$ being $x \in \gamma(a)$ given the control action f. That is, a set of feasible control actions from a system S is the set of possible functions that fix the store values. Later on we will consider control policies which specify how a f control action is going to be chosen for a given state of the system and time t. In the formal semantics we introduce a refinement step labelled by a chosen control action f that transforms S into

$$S_f \overset{\mathsf{def}}{=} (P_1, f(\gamma_1), \gamma_1) \mid \cdots \mid (P_n, f(\gamma_n), \gamma_n) \textbf{ in } (f(\gamma_g), \gamma_g, \rho)$$

The corresponding rule is given by the following in the FuTS style [44]

$$\textsf{\textbf{A-Choice}} \qquad \frac{S \overset{\mathsf{def}}{=} ((P_1, \gamma_1) \mid \cdots \mid (P_n, \gamma_n)) \textbf{ in } (\gamma_g, \rho) \in \textsc{Sys} \qquad f \in \mathcal{A}(S)}{S \to [(P_1, f(\gamma_1), \gamma_1) \mid \cdots \mid (P_n, f(\gamma_n), \gamma_n) \textbf{ in } (f(\gamma_g), \gamma_g, \rho) \mapsto 1]}$$

Namely, given a system S and control action f, the rule A-Choice gives us the corresponding resolved system S_f. More precisely, the continuation S_f is assigned value 1 while all other continuations get value 0 thus separating resolved components reachable via control action f from unreachable. Note that the resolution keeps track of the unresolved version store. This allows us to define the store updates on the unresolved store depending on the chosen control action. For example, the space of control actions for a system can thus change during the evolution. Let us define the sets $\textsc{Sys}^f$, $\textsc{Col}^f$ and $\textsc{Comp}^f$ as sets of systems, collectives and components after application of f. We assume that elements of $\textsc{Sys}^f$, $\textsc{Col}^f$ and $\textsc{Comp}^f$ are derived only from elements in $\textsc{Sys}$, $\textsc{Col}$ and $\textsc{Comp}$ for which f is sufficient to fully resolve the non-determinism in the behaviour. We call such sets *resolved systems, collectives* and *components*, respectively. Finally, we complete our construction of the evaluation context ρ by defining it as a function on resolved system S.

Example 5. *In the case of the running example the resolution of non-determinism corresponds to making explicit the values for global store attributes* failr *and* senser*. For example, suppose f is defined such that $f(\gamma)(\mathsf{failr}) = r_f$ and $f(\gamma)(\mathsf{senser}) = r_s$. The local store for each Robot component is resolved trivially as the value domains are defined as singleton sets. Note that this description of the store values does not mean the rates cannot change in time, as our choice of the control action f can change in time and resolve the store attributes to different values. This is discussed later in this chapter.*

3.1.4 Interleaving semantics

The second stage of the semantics determines the rates at which a system changes state given a control action. This is achieved through construction of functions $\mathbb{C}_f$, $\mathbb{N}_{\varepsilon,f}$ and $\mathbb{S}_{t,f}$ parametrised by a chosen control action f. The function $\mathbb{C}_f$ takes a resolved component in $\textsc{Comp}^f$ and an action label and returns a probability distribution over components in $\textsc{Comp}$. Components assigned a non-zero probability are reachable from the resolved component. The function, $\mathbb{N}_{\varepsilon,f}$ builds on $\mathbb{C}_f$ to describe the behaviour of collectives. Based on a resolved collective in $\textsc{Col}^f$ and an action label it returns a probability distribution over $\textsc{Col}$. As before non-zero probabilities are assigned to reachable collectives. Finally, function $\mathbb{S}_{t,f}$ takes a resolved system in $\textsc{Sys}^f$ and an action label and returns a function over systems $\textsc{Sys}$ that specifies the rate at which the transitions happen. Note that in general this can also depend on time. Thus the semantics describe the evolution of an unresolved system to a resolved system via a control action and after which it becomes, again, an unresolved system. This provides a mechanism for changing the available control actions based on the state of the system. The semantic rules for the second step resulting in the transition rates between systems closely match the semantics given for Carma in [88]. The main difference is that in the case of Carma-C semantics we need to decide how the evolution of stores is treated. As mentioned, we opt for the construction where store updates are performed on the unresolved stores where the exact update to be performed can depend on the chosen control action.

Semantics of components

We start the definition of CTMDP semantics for Carma by considering single components and their evolution. The rules in this section are simple extensions of the

Nil $$\frac{}{\mathbb{C}_f[(\mathbf{nil}, f(\gamma), \gamma), \ell] = \emptyset}$$ **Zero** $$\frac{}{\mathbb{C}_f[(\mathbf{0}, f(\gamma), \gamma), \ell] = \emptyset}$$

Plus $$\frac{\mathbb{C}_f[(P, f(\gamma), \gamma), \ell] = \mathscr{C}_1 \quad \mathbb{C}_f[(Q, f(\gamma), \gamma), \ell] = \mathscr{C}_2}{\mathbb{C}_f[(P + Q, f(\gamma), \gamma), \ell] = \mathscr{C}_1 \oplus \mathscr{C}_2}$$

Guard $$\frac{f(\gamma) \models \pi \quad \mathbb{C}_f[(P, f(\gamma), \gamma), \ell] = \mathscr{C}}{\mathbb{C}_f[([\pi]P, f(\gamma), \gamma), \ell] = \mathscr{C}}$$ **Guard-F** $$\frac{f(\gamma) \not\models \pi}{\mathbb{C}_f[([\pi]P, f(\gamma), \gamma), \ell] = \emptyset}$$

Par $$\frac{\mathbb{C}_f[(P, f(\gamma), \gamma), \ell] = \mathscr{C}_1 \quad \mathbb{C}[(P, f(\gamma), \gamma), \ell] = \mathscr{C}_2}{\mathbb{C}_f[(P \mid Q, f(\gamma), \gamma), \ell] = \mathscr{C}_1 \mid Q \oplus P \mid \mathscr{C}_2}$$

Rec $$\frac{A \overset{\triangle}{=} P \quad \mathbb{C}_f[(P, f(\gamma), \gamma), \ell] = \mathscr{C}}{\mathbb{C}_f[(A, f(\gamma), \gamma), \ell] = \mathscr{C}}$$

Figure 3.3: Component semantics — base rules.

corresponding rules given for Carma in [88]. The only notable difference is that we assume that some control action f has been chosen which means that the rules operate on a set of resolved components Comp^f. The next state, however, is going to be an unresolved component in Comp. In particular, we aim to define the function

$$\mathbb{C}_f : \text{Comp}^f \times \text{Lab}^f \to [\text{Comp} \to \mathbb{R}_{\geq 0}]$$

The construction from resolved into unresolved components is going to be made possible by the fact that the components in Comp^f carry with them the information about the unresolved store definitions.

At the component level we are going to work with the set of labels Lab^f associated with the defined action types and generated by the following grammar:

$$\begin{aligned}
\ell := &\alpha^*[\pi]\langle \mathbf{v} \rangle, f(\gamma) \\
&\mid \alpha^*[\pi]\,(\mathbf{v})\,, f(\gamma) \\
&\mid \alpha[\pi]\langle \mathbf{v} \rangle, f(\gamma) \\
&\mid \alpha[\pi]\,(\mathbf{v})\,, f(\gamma)
\end{aligned}$$

These labels contain the reference to the action which is performed, the values transmitted with the given action and the resolved store. The function $\mathbb{C}_f$ is defined so that for any component C resolved by action f and transition label ℓ, $\mathbb{C}[C, \ell]$ gives the weights of possible next states of C resulting from transition ℓ. If $\mathbb{C}_f[C, \ell] = \mathscr{C}$ and $\mathscr{C}(C') = w$ then $C \in \text{Comp}^f$ evolves to $C' \in \text{Comp}$ with a weight w when ℓ is executed. Function $\mathbb{C}_f$ is formally defined in Figure 3.3 dealing with composition rules for the processes and Figures 3.4 and 3.5, which define the rules for broadcast and unicast actions respectively. In the following we give a description of the defined rules.

B-Out $$\frac{[\![\pi_s]\!]_{f(\gamma)} = \pi_s' \quad [\![\mathbf{e}]\!]_{f(\gamma)} = \mathbf{v} \quad \mathbf{p} = \sigma(f(\gamma),\gamma)}{\mathbb{C}_f[(\alpha^*[\pi_s]\langle\mathbf{e}\rangle\sigma.P, f(\gamma),\gamma), \alpha^*[\pi_s']\langle\mathbf{v}\rangle, f(\gamma)] = (P,\mathbf{p})}$$

B-Out-F $$\frac{[\![\pi_s]\!]_{f(\gamma)} = \pi_s' \quad [\![\mathbf{e}]\!]_{f(\gamma)} = \mathbf{v} \quad \ell \neq \alpha^*[\pi_s']\langle\mathbf{v}\rangle, f(\gamma)}{\mathbb{C}_f[(\alpha^*[\pi_s]\langle\mathbf{v}\rangle\sigma.P, f(\gamma),\gamma), \ell] = \emptyset}$$

B-In $$\frac{f(\gamma_r) \models \pi_s \quad f(\gamma_s) \models \pi_r[\mathbf{v}/\mathbf{x}] \quad \mathbf{p} = \sigma[\mathbf{v}/\mathbf{x}](f(\gamma))}{\mathbb{C}_f[(\alpha^*[\pi_r](\mathbf{x})\sigma.P, f(\gamma_r),\gamma_r), \alpha^*[\pi_s](\mathbf{v}), f(\gamma_s)] = (P[\mathbf{v}/\mathbf{x}],\mathbf{p})}$$

B-In-F $$\frac{\ell \neq \alpha^*[\pi_s](\mathbf{v}), f(\gamma_s)}{\mathbb{C}_f[(\alpha^*[\pi_r](\mathbf{v})\sigma.P, f(\gamma_r),\gamma_r), \ell] = \emptyset}$$

B-In-F2 $$\frac{f(\gamma_r) \not\models \pi_s \vee f(\gamma_s) \not\models \pi_r[\mathbf{v}/\mathbf{x}]}{\mathbb{C}_f[(\alpha^*[\pi_r](\mathbf{x})\sigma.P, f(\gamma_r),\gamma_r), \alpha^*[\pi_s](\mathbf{v}), f(\gamma_s)] = \emptyset}$$

Figure 3.4: Component semantics — broadcast rules.

Nil, Zero The rule **Nil** states that the inactive process cannot perform any action. The function $\mathbb{C}_f$ gives the 0 constant function for all labels. Similarly, **Zero** defines function $\mathbb{C}_f$ to be 0 constant function when evaluated for the component given by the **0** process.

Guard and Guard-F These rules deal with guards on processes. If the guard π is satisfied, the component $[\pi](P, f(\gamma),\gamma)$ behaves as $(P, f(\gamma),\gamma)$. Otherwise no component is reachable from $[\pi](P, f(\gamma),\gamma)$. The guard π is satisfied if $\pi(f(\gamma))$ returns true. If this is true we say $f(\gamma) \models \pi$.

Plus, Par and Rec These rules define the standard operations for process algebra which deal with process composition at the component level: choice, parallel composition and recursion. For choice, the expression $\mathscr{C}_1 \oplus \mathscr{C}_2$ is used to denote the function that maps each component term $C \in \textsc{Comp}$ to $\mathscr{C}_1(C) + \mathscr{C}_2(C)$. That is, C is reachable if it is reachable from either P or Q. In other words, if either $\mathscr{C}_1(C)$ or $\mathscr{C}_2(C)$ are non-zero.

For parallel composition we define how the processes $P \mid Q$ interleave. Firstly, we define for each component C and process Q

$$C \mid Q = \begin{cases} \mathbf{0} & C \equiv \mathbf{0} \\ (P \mid Q, \gamma) & C \equiv (P,\gamma) \end{cases}$$

where the notation $Q \mid C$ is defined symmetrically.

Now for each $\mathscr{C} : \textsc{Comp} \to \mathbb{R}_{\geq 0}$ and process Q, $\mathscr{C}_1 \mid Q$ (respectively $P \mid \mathscr{C}_2$) denotes the function that maps each term of the form $C \mid Q$ (respectively $P \mid C$) to $\mathscr{C}_1(C)$ (respectively $\mathscr{C}_2(C)$) while the others are mapped to 0. Thus, $C \mid Q$ or $P \mid C$ are reachable if C is reachable from P (corresponding to $\mathscr{C}_1(C)$ non-zero) or C is reachable from Q (corresponding to $\mathscr{C}_2(C)$ non-zero).

Finally, for completeness we have the standard rule Rec which allows recursive definitions of processes in the language.

Broadcast output To start, the rule **B-Out** states that a broadcast output $\alpha^*[\pi_s]\langle \mathbf{v}\rangle\sigma$ sends a message $[\![\mathbf{e}]\!]_{f(\gamma)}$ to all components where $[\![\mathbf{e}]\!]_{f(\gamma)}$ denotes the evaluation of the expression $\mathbf{e}$ with respect to the resolved store $f(\gamma)$. Note that the evaluation of the message is dependent on the resolved store. This allows, for example, the definition of a set of control actions corresponding to a set of messages that the component can decide to send.

Similarly, $[\![\pi_s]\!]_{f(\gamma)}$ denotes the predicate expression π_s evaluated with respect to the resolved store $f(\gamma)$. The local stores resulting from the execution of an action are determined by the update σ which takes the resolved store $f(\gamma)$, the unresolved store $f(\gamma)$ and returns a probability distribution $\mathbf{p} = \sigma(f(\gamma),\gamma) \in \mathit{Dist}(\Gamma)$ over the unresolved stores.

The outcome is given by the probability distribution $(P,\mathbf{p})$ defined by

$$(P,\mathbf{p})(C) = \begin{cases} 1 & P \equiv Q|\mathbf{kill} \ \wedge \ C \equiv \mathbf{0} \\ \mathbf{p}(\gamma) & C \equiv (P,\gamma) \\ 0 & \text{otherwise} \end{cases}$$

That is, the process evolution to P is given syntactically and the probability of evolving to a component C can only be non-zero if C is defined by the process P. Moreover, the above defines that if **kill** is specified in the process P then the only possible outcome is a component defined by the $\mathbf{0}$ process. If **kill** is not specified then the probability of the component evolving to C is given by the defined probability distribution on the set of unresolved stores Γ.

The rule **B-Out-F** states that the broadcast output can only be involved in labels of the form $\alpha^*[\pi_s]\langle \mathbf{v}\rangle, f(\gamma)$. That is, the output action can only happen over the label which matches the syntactic definition of the action.

Broadcast input Similarly to the output action the rule **B-In** states that the component $\alpha^*[\pi_r](\mathbf{x})\sigma.P, f(\gamma_r)$ can perform the action labelled $\alpha^*[\pi_r](\mathbf{v}), f(\gamma_r)$, with the next states given by probability distribution $(P[\mathbf{v}/\mathbf{x}],\mathbf{p})$. The evolution of the component is found when the input message $\mathbf{v}$ is substituted into the expressions given in terms of $\mathbf{x}$.

U-Out $$\frac{[\![\pi_s]\!]_{f(\gamma)} = \pi_s' \quad [\![\mathbf{e}]\!]_{f(\gamma)} = \mathbf{v} \quad \mathbf{p} = \sigma(f(\gamma),\gamma)}{\mathbb{C}_f[(\alpha[\pi_s]\langle\mathbf{v}\rangle\sigma.P, f(\gamma),\gamma), \alpha[\pi_s]\langle\mathbf{v}\rangle, f(\gamma)] = (P,\mathbf{p})}$$

U-Out-F $$\frac{[\![\pi_s]\!]_{f(\gamma)} = \pi_s' \quad [\![\mathbf{e}]\!]_{f(\gamma)} = \mathbf{v} \quad \ell \neq \alpha[\pi_s]\langle\mathbf{v}\rangle, f(\gamma)}{\mathbb{C}_f[(\alpha[\pi_s]\langle\mathbf{v}\rangle\sigma.P, f(\gamma),\gamma), \ell] = \emptyset}$$

U-In $$\frac{f(\gamma_r) \models \pi_s \quad f(\gamma_s) \models \pi_r[\mathbf{v}/\mathbf{x}] \quad \mathbf{p} = \sigma[\mathbf{v}/\mathbf{x}](f(\gamma_r),\gamma_r)}{\mathbb{C}_f[(\alpha[\pi_r](\mathbf{x})\sigma.P, f(\gamma_r),\gamma_r), \alpha[\pi_s](\mathbf{v}), f(\gamma_s)] = (P[\mathbf{v}/\mathbf{x}],\mathbf{p})}$$

U-In-F $$\frac{\ell \neq \alpha[\pi_s](\mathbf{v}), f(\gamma_s)}{\mathbb{C}_f[(\alpha[\pi_r](\mathbf{v})\sigma.P, f(\gamma),\gamma), \ell] = \emptyset}$$

U-In-F2 $$\frac{f(\gamma_r) \not\models \pi_s \vee f(\gamma_s) \not\models \pi_r[\mathbf{v}/\mathbf{x}]}{\mathbb{C}_f[(\alpha[\pi_r](\mathbf{v})\sigma.P, f(\gamma),\gamma), \alpha[\pi_s](\mathbf{v})] = \emptyset}$$

Figure 3.5: Component semantics — unicast rules.

The rule **B-In-F** is identical to **B-Out-F** and states that the broadcast input is only defined for the labels of the form $\alpha^*[\pi_s]\langle\mathbf{v}\rangle, f(\gamma)$.

The rule **B-In-F2** models the fact that if the receiving component's store does not satisfy the filter set by the sender of the broadcast message or if the received values do not satisfy the receiver's requirements then the component does not receive the broadcast message.

Unicast The semantic rules for the unicast communication at the component level mirror the broadcast rules precisely.

Example 6. *We continue the running example to briefly demonstrate the application of the semantic rules at the component level. Here we are going to analyse the resolved component*

$$\mathit{Robot} \overset{\mathsf{def}}{=} \Big(\mathit{Explore} \parallel \mathit{Listen}, \{\mathsf{loc} \mapsto (x,y), \mathsf{target} \mapsto \emptyset\}, \\ \{\mathsf{loc} \mapsto \{(x,y)\}, \mathsf{target} \mapsto \{\emptyset\}\}\Big)$$

resulting from the parallel composition of processes for Explore and Listen. Note that in this case the unresolved store, corresponding to loc *and* target *attributes, has value domains defined as singleton sets and thus any choice of control action would give the same behaviour at the component level. To get an idea of how the semantics rules are used we consider, as an example, the application of **B-Out** and **B-Out-F** to the following sub-process in the definition of process Explore.*

$$\mathit{sense}^*[\circ]\langle M(\mathsf{loc})\rangle\{\mathsf{target} = \mathsf{target} \cup M(\mathsf{loc})\}.\mathit{Explore}$$

The application of the said rules would give us

$$\begin{aligned}
&\mathbb{C}_f\Big[\Big(\mathit{sense}^*[\circ]\langle M(\mathsf{loc})\rangle\{\mathsf{target}\mapsto\mathsf{target}\cup M(\mathsf{loc})\}.\mathit{Explore}, && \textit{(Process description)}\\
&\qquad\{\mathsf{loc}\mapsto(x,y),\mathsf{target}\mapsto\emptyset\}, && \textit{(Resolved store)}\\
&\qquad\{\mathsf{loc}\mapsto\{(x,y)\},\mathsf{target}\mapsto\{\emptyset\}\}\Big), && \textit{(Unresolved store)}\\
&\quad \mathit{sense}^*[\circ]\langle M(x,y)\rangle,\{\mathsf{loc}\mapsto(x,y),\mathsf{target}\mapsto\emptyset\}\Big] && \textit{(Transition label)}\\
&\quad=(\mathit{Explore},\mathbf{p})
\end{aligned}$$

The resulting expression $(\mathit{Explore},\mathbf{p})$ defines the following function over the unresolved components Comp

$$(\mathit{Explore},\mathbf{p})(C)=\begin{cases}1 & \text{for } C\equiv(\mathit{Explore},\{\mathsf{loc}\mapsto\{(x,y)\},\mathsf{target}\mapsto\{\{M(x,y)\}\}\})\\ 0 & \text{otherwise}\end{cases}$$

Thus, when it comes to the component Robot performing the broadcast action sense* there is just one possible outcome for the said component. Namely, that the process describing the robot evolves to Explore $\|$ Listen and all the sets in the value domain (in this case just $\emptyset$) corresponding to the attribute for the identified targets target *gets updated with the $M(x,y)$. If the location (x,y) is a target then we get* $\mathsf{target}\mapsto\{\{(x,y)\}\}$ *and* $\mathsf{target}\mapsto\{\emptyset\}$ *otherwise. Similarly, applying rules* ***B-In***, ***B-In-F*** *and* ***B-In-F2*** *to the sub-process Listen gives*

$$\begin{aligned}
&\mathbb{C}_f\Big[\Big(\mathit{sense}^*[\circ](x)\{\mathsf{target}\mapsto\mathsf{target}\cup x\}.\mathit{Listen}, && \textit{(Process description)}\\
&\qquad\{\mathsf{loc}\mapsto(x,y),\mathsf{target}\mapsto\emptyset\} && \textit{(Resolved store)}\\
&\qquad\{\mathsf{loc}\mapsto\{(x,y)\},\mathsf{target}\mapsto\{\emptyset\}\}), && \textit{(Unresolved store)}\\
&\quad \mathit{sense}^*[\circ](M(x,y)),\{\mathsf{loc}\mapsto(x,y),\mathsf{target}\mapsto\emptyset\}\Big] && \textit{(Transition label)}\\
&\quad=(\mathit{Listen}[M(x,y)/\mathbf{x}],\mathbf{p})
\end{aligned}$$

Again, the resulting expression $(\mathit{Listen}[M(x,y)/\mathbf{x}],\mathbf{p})$ is defined over components in Comp. *In particular, we get*

$$(\mathit{Listen}[M(x,y)/\mathbf{x}],\mathbf{p})(C)=\begin{cases}1 & \text{for } C\equiv(\mathit{Listen},\{\mathsf{loc}\mapsto\{(x,y)\},\\ & \qquad\qquad \mathsf{target}\mapsto\{\{M(x,y)\}\}\})\\ 0 & \text{otherwise}\end{cases}$$

where, again, the precise form of the component C for which the probability is 1 depends on whether the location (x,y) is a target or not. The rules are similarly applied to all of the defined processes and allows us to derive the full set of unresolved components that are reachable in a single step from the resolved component.

Semantics of collectives

Following the framework in [88] the next step is to consider the transitions at the collective level. We define the operational semantics at the collective level via the function

$$\mathbb{N}_{\varepsilon,f} : \textsc{Col}^f \times \textsc{Lab}_I^f \to [\textsc{Col} \to \mathbb{R}_{\geq 0}]$$

The function $\mathbb{N}_{\varepsilon,f}$ is defined in Figure 3.6 and describes how a collective reacts when a message is received. Thus, only input labels corresponding to broadcast and unicast are considered with the label set $\textsc{Lab}_I$ composed of labels in the following form:

$$\ell \overset{\mathsf{def}}{=} \alpha^*[\pi](\mathbf{v}), f(\gamma)$$
$$|\alpha[\pi](\mathbf{v}), f(\gamma)$$

The function $\mathbb{N}_{\epsilon,f}$ is parametrised with respect to the evaluation function obtained from the environment where the collective operates as well as the chosen control action that resolves the non-determinism. Again, apart from having to deal with choice of control action and the transition from a resolved store to an unresolved store, the definitions given in Figure 3.6 mirror the relevant definitions from [88]. The explanations of the rules are summarised below.

Zero States that the inactive component is unaffected by the broadcast messages.

Broadcast The rule **Comp-B-In** states that if a resolved component $(P, f(\gamma), \gamma)$ can receive a broadcast message over the name α $(\mathbb{C}[(P, f(\gamma), \gamma), \alpha^*[\pi_r](\mathbf{v}), f(\gamma)] = \mathscr{N} \neq \emptyset)$ then the component receives the message with probability $\mu_p(f(\gamma), \alpha^*)$ while the message is not received with probability $1 - \mu_p(f(\gamma), \alpha^*)$. The resulting probability is renormalised to indicate that each element in P receives the message with the same probability. We use $\oplus\mathscr{N}$ to denote $\sum_{N \in \textsc{Col}} \mathscr{N}(N)$.

Comp-B-In-F gives that if a component is not able to receive a broadcast message $(\mathbb{C}[(P, f(\gamma), \gamma), \alpha^*[\pi_r](\mathbf{v}), f(\gamma)] = \emptyset)$ then the component remains unchanged with probability 1.

Finally, **B-In-Sync** models the fact that when two collectives operate in parallel then potentially both of them can receive a broadcast message.

Unicast As with the component level rules the collective rules for unicast mirror the broadcast ones.

Example 7. *We return to the running example to demonstrate the application of the semantics rules in this section. Let us consider a collective given by only two resolved components.*

$$\mathit{Collective} \overset{\mathsf{def}}{=} \mathit{Robot} \parallel \mathit{Robot}$$

Zero $$\frac{}{\mathbb{N}_{\varepsilon,f}[\mathbf{0},\ell] = \emptyset}$$

Comp-B-In $$\frac{\mathbb{C}_f[(P,f(\gamma),\gamma),\alpha^*[\pi_s](\mathbf{v}),f(\gamma)] = \mathcal{N} \quad \mathcal{N} \neq \emptyset \quad \varepsilon = \langle \mu_p,\mu_r,\mu_u,\mu_w \rangle}{\begin{array}{l}\mathbb{N}_{\varepsilon,f}[(P,f(\gamma),\gamma),\alpha^*[\pi_s](\mathbf{v}),f(\gamma)) \\ \quad = \mu_p(f(\gamma),\alpha^*)\frac{\mathcal{N}}{\oplus\mathcal{N}} + [(P,\gamma) \mapsto (1-\mu_p(f(\gamma),\alpha^*))]\end{array}}$$

Comp-B-In-F $$\frac{\mathbb{C}_f[(P,f(\gamma),\gamma),\alpha^*[\pi_s](\mathbf{v}),f(\gamma)] = \emptyset}{\mathbb{N}_{\varepsilon,f}[(P,f(\gamma),\gamma),\alpha^*[\pi_s](\mathbf{v}),f(\gamma)] = [(P,\gamma) \mapsto 1]}$$

Comp-U-In $$\frac{\mathbb{C}_f[(P,f(\gamma_1),\gamma_1),\alpha[\pi_s](\mathbf{v}),f(\gamma_2)] = \mathcal{N} \quad \mathcal{N} \neq \emptyset \quad \varepsilon = \langle \mu_p,\mu_r,\mu_u,\mu_w \rangle}{\mathbb{N}_{\varepsilon,f}[(P,f(\gamma_1),\gamma_1),\alpha[\pi_s](\mathbf{v}),f(\gamma_2)] = \mu_w(f(\gamma_1),f(\gamma_2),\alpha)\frac{\mathcal{N}}{\oplus\mathcal{N}}}$$

Comp-U-In-F $$\frac{\mathbb{C}_f[(P,f(\gamma_2),\gamma_2),\alpha[\pi_s](\mathbf{v}),f(\gamma_1)] = \emptyset}{\mathbb{N}_{\varepsilon,f}[(P,f(\gamma_2),\gamma_2),\alpha[\pi_s](\mathbf{v}),f(\gamma_1)] = \emptyset}$$

B-In-Sync $$\frac{\mathbb{N}_{\varepsilon,f}[N_1,\alpha^*[\pi_s](\mathbf{v}),f(\gamma)] = \mathcal{N}_1 \quad \mathbb{N}_{\varepsilon,f}[N_2,\alpha^*[\pi_s](\mathbf{v}),f(\gamma)] = \mathcal{N}_2}{\mathbb{N}_{\varepsilon,f}[N_1 \parallel N_2,\alpha^*[\pi_s](\mathbf{v}),f(\gamma)] = \mathcal{N}_1 \parallel \mathcal{N}_2}$$

U-In-Sync $$\frac{\mathbb{N}_{\varepsilon,f}[N_1,\alpha[\pi_s](\mathbf{v}),f(\gamma)] = \mathcal{N}_1 \quad \mathbb{N}_{\varepsilon,f}[N_2,\alpha[\pi_s](\mathbf{v}),f(\gamma)] = \mathcal{N}_2}{\mathbb{N}_{\varepsilon,f}[N_1 \parallel N_2,\alpha[\pi_s](\mathbf{v}),f(\gamma)] = \mathcal{N}_1 \parallel N_2 \oplus N_1 \parallel \mathcal{N}_2}$$

Figure 3.6: Collective semantics.

In addition, suppose that the probability of successfully receiving the broadcast is 1 *for all resolved sender* (γ_s) *and receiver* (γ_r) *component stores:*

$$\mu_p(f(\gamma_s),f(\gamma_r),sense^*) = 1$$

The application of rules ***Comp-B-In*** *and* ***Comp-B-In-F*** *to either of the Robot components then gives us the following:*

$$\begin{aligned}
&\mathbb{N}_{f,\varepsilon}\Big[\Big(Robot,\{\mathsf{loc} \mapsto (x,y),\mathsf{target} \mapsto \emptyset\} && \textit{(Component, Resolved store)}\\
&\qquad\qquad \{\mathsf{loc} \mapsto \{(x,y)\},\mathsf{target} \mapsto \{\emptyset\}\}\Big), && \textit{(Unresolved store)}\\
&\quad sense^*[\circ](M(x,y)),\{\mathsf{loc} \mapsto (x,y),\mathsf{target} \mapsto \emptyset\}\Big] && \textit{(Transition label)}\\
&\quad = (Explore \parallel Listen[M(x,y)/\mathbf{x}],\mathbf{p})\\
&\qquad\quad + [(Robot,\{\mathsf{loc} \mapsto \{(x,y)\},\mathsf{target} \mapsto \{\emptyset\}\}) \mapsto 0]
\end{aligned}$$

with

$$(Explore \parallel Listen[M(x,y)/\mathbf{x}],\mathbf{p})(C) = \begin{cases} 1 & C \equiv (Robot,\{\mathsf{loc} \mapsto \{(x,y)\}, \\ & \qquad\qquad \mathsf{target} \mapsto \{\{M(x,y)\}\}\}) \\ 0 & otherwise \end{cases}$$

In this case, by definition whenever a broadcast input action happens the Robot is guaranteed to accept it and change its store accordingly. If we had more Robot components *in the collective the rule **B-In-Sync** would combine the effects of the transitions due to a given broadcast action into a single transition on the collective.*

Semantics of systems

At the system level, given we have resolved the non-determinism in the components, we are interested in calculating the transition rates. Let us consider the set of labels $\textsc{Lab}^f_S$ given by the following grammar:

$$\ell \overset{\mathsf{def}}{=} \alpha^*[\pi_s]\langle \mathbf{v}\rangle, f(\gamma)$$
$$\mid \tau[\alpha[\pi_s]\langle \mathbf{v}\rangle, f(\gamma)]$$

Thus, the actions are labelled by the firing broadcast or unicast output action and the resolved store of the component performing the action. Our aim is to give a rule which, given a system $S \in \textsc{Sys}^f$ resulting from resolving the non-determinism through applying some control action f, calculates the rate of transition from S to $S' \in \textsc{Sys}$ for any action label ℓ. Analogously to CARMA the operational semantics of CARMA-C systems are defined through the function

$$\mathbb{S}_{t,f} : \textsc{Sys}^f \times \textsc{Lab}^f_S \to [\textsc{Sys} \to \mathbb{R}_{\geq 0}]$$

This function, indexed by the chosen control action, calculates the rates of transition for any system at a given time t. In detail, for any resolved system S and any label $\ell \in \textsc{Lab}^f_S$, if $\mathbb{S}_{t,f}[S,\ell] = \mathscr{S}$ then $\mathscr{S}(S')$ is the rate of the transition from S to S' over label ℓ. If $\mathscr{S}(S') = 0$ then S' is not reachable from S via ℓ. The rules defined in this section are again modified from [88] to make explicit the dependence on a chosen control action.

Firstly, we combine the outcomes of broadcast output performed by the resolved component C, given by $\mathscr{C}$, and the complementary input performed by N, given $\mathscr{N}$. The result is multiplied by the rate of action induced by the environment:

$$\mu_r(f(\gamma), \alpha^*[\pi_s]\langle \mathbf{v}\rangle)$$

This is done in the rule, named **Sync**, given below.

$$\textbf{Sync}\quad \frac{\begin{array}{l}\varepsilon = \langle \mu_p, \mu_r, \mu_u, \mu_w\rangle \\ \mathbb{C}_f[C, \alpha^*[\pi_s]\langle \mathbf{v}\rangle, f(\gamma)] = \mathscr{C} \\ \mathbb{N}_{\epsilon,f}[N, \alpha^*[\pi_r](\mathbf{v}), f(\gamma)] = \mathscr{N}\end{array}}{\mathsf{bSync}_{\varepsilon,f}(C, N, \alpha^*[\pi_s]\langle \mathbf{v}\rangle, f(\gamma)) = \mu_r(f(\gamma), \alpha^*[\pi_s]\langle \mathbf{v}\rangle) \cdot \mathscr{C} \parallel \mathscr{N}}$$

The function $\mathsf{bSync}_\varepsilon$ gives the rate at which a given new state is reached via the broadcast synchronisation between component C and collective N over label $\alpha^*[\pi_s]\langle \mathbf{v}\rangle, f(\gamma)$. Similarly, the rules below regulate the unicast communication.

Sync-U
$$\frac{\begin{array}{l}\varepsilon = \langle \mu_p, \mu_r, \mu_u, \mu_w \rangle \\ \mathbb{C}_f[C, \alpha[\pi_s]\langle \mathbf{v} \rangle, f(\gamma)] = \mathscr{C} \\ \mathbb{N}_{\epsilon,f}[N, \alpha[\pi_s](\mathbf{v}), f(\gamma)] = \mathscr{N} \neq \emptyset\end{array}}{\mathsf{uSync}_{\varepsilon,f}(C, N, \alpha[\pi_s]\langle \mathbf{v} \rangle, f(\gamma)) = \mu_r(f(\gamma), \alpha[\pi_s]\langle \mathbf{v} \rangle) \cdot \mathscr{C} \parallel \frac{\mathscr{N}}{\oplus \mathscr{N}}}$$

Sync-U-F
$$\frac{\mathbb{N}_{\epsilon,f}[N, \alpha[\pi_s](\mathbf{v}), f(\gamma)] = \emptyset}{\mathsf{uSync}_{\varepsilon,f}(C, N, \alpha[\pi_s]\langle \mathbf{v} \rangle, f(\gamma)) = \emptyset}$$

Note that the additional rule **Sync-U-F** implements the blocking nature of unicast communication. If there are no available receivers in the system the unicast output action will not be performed.

The final step is to combine the definitions given up to this point into a top level semantic rule for our language. To that end we start by considering any system S at time t resolved by a control action f which results in a resolved system of the form $\mathbb{A}_t(S, f) = N$ **in** $(f(\gamma_g), \gamma_g, \rho)$. We suppose that the evaluation context is of the form $\rho(t, f(\gamma_g), \gamma_g, N) = \varepsilon = \langle \mu_p, \mu_w, \mu_r, \mu_u \rangle$ and that the update on the global store is given by $\mu_u(f(\gamma_g), \gamma_g, \alpha^*) = \sigma$. Next we consider what happens if several components in a collective are capable of inducing a broadcast synchronisation over the same label. We combine these rates to get the total rate of synchronisation by summing the outcome of rule **Sync** over all possible senders $C \in N$ and multiply it by the multiplicity of $C \in N$, denoted $N(C)$. We say that, given these premises, the result is a function mapping the resolved system with the given label to a function $\textsc{Sys} \to \mathbb{R}_{\geq 0}$.

In more detail, the rules **Sys-B** and **Sys-U** given below say the following. For each resolved collective N, $\mathscr{N} : \textsc{Col} \to \mathbb{R}_{\geq 0}$, $\mathscr{S} : \textsc{Sys} \to \mathbb{R}_{\geq 0}$ and $\mathbf{p} \in \mathit{Dist}(\Gamma)$, denoting the distributions over unresolved stores, we let $\mathscr{N}$ **in** $(\mathbf{p}, \rho)$ denote the function mapping each system N' **in** (γ, ρ) to $\mathscr{N}(N') \cdot \mathbf{p}(\gamma)$. That is, every resolved system is assigned a function that maps any given unresolved system definition to a rate parameter.

Sys-B
$$\frac{\begin{array}{l}\mathbb{A}_t(S, f) = N \textbf{ in } (f(\gamma_g), \gamma_g, \rho) \\ \rho(t, f(\gamma_g), \gamma_g, N) = \varepsilon = \langle \mu_p, \mu_r, \mu_u, \mu_w \rangle \quad \mu_u(f(\gamma_g), \gamma_g, \alpha^*) = \sigma \\ \sum_{C \in N} N(C) \mathsf{bSync}_{\varepsilon,f}(C, N - C, \alpha^*[\pi_s]\langle \mathbf{v} \rangle, f(\gamma)) = \mathscr{N}\end{array}}{\mathbb{S}_{t,f}[N \textbf{ in } (f(\gamma_g), \gamma_g, \rho), \alpha^*[\pi_s]\langle \mathbf{v} \rangle, f(\gamma)] = \mathscr{N} \textbf{ in } (\sigma(f(\gamma_g), \gamma_g, \rho)}$$

Sys-U
$$\frac{\begin{array}{l}\mathbb{A}_t(S, f) = N \textbf{ in } (f(\gamma_g), \gamma_g, \rho) \\ \rho(t, f(\gamma_g), \gamma_g, N) = \varepsilon = \langle \mu_p, \mu_r, \mu_u, \mu_w \rangle \quad \mu_u(f(\gamma_g), \gamma_g, \alpha^*) = \sigma \\ \sum_{C \in N} N(C) \mathsf{uSync}_{\varepsilon,f}(C, N - C, \tau[\alpha^*[\pi_s]\langle \mathbf{v} \rangle, f(\gamma)]) = \mathscr{N}\end{array}}{\mathbb{S}_{t,f}[N \textbf{ in } (f(\gamma_g), \gamma_g, \rho), \tau[\alpha^*[\pi_s]\langle \mathbf{v} \rangle, f(\gamma)]] = \mathscr{N} \textbf{ in } (\sigma(f(\gamma_g), \gamma_g, \rho)}$$

Example 8. *For this final step of the semantics we need to define the rates of actions* *prescribed by the environment. For example, suppose that after resolving the store by f the rate of sense* becomes*

$$\mu_r(f(\{\mathsf{failr} \mapsto [0,\infty), \mathsf{senser} \mapsto [0,\infty)\}), \mathit{sense}^*) = \\ \mu_r(\{\mathsf{failr} \mapsto r_f, \mathsf{senser} \mapsto r_s\}, \mathit{sense}^*) = r_s$$

*As before in this chapter we briefly illustrate the steps taken to construct the system level transitions. We start by constructing the function $\mathsf{bSync}_{\varepsilon,f}$ using the rule **Sync**.*

$$\begin{aligned} \mathsf{bSync}_{\varepsilon,f}\Big[&\mathit{Robot}, && \text{(Component)} \\ &\mathit{Robot} \parallel \mathit{Robot}, && \text{(Collective)} \\ &\mathit{sense}^*\langle M(x,y)\rangle, \{\mathsf{loc} \mapsto (x,y), \mathsf{target} \mapsto \emptyset\}\}\Big] && \text{(Transition label)} \\ &= r_s \cdot (\mathit{Robot}, \mathbf{p}_1) \parallel (\mathit{Robot}, \mathbf{p}_2) \end{aligned}$$

where

$$(\mathit{Robot}, \mathbf{p}_1) \parallel (\mathit{Robot}, \mathbf{p}_2)(N) = \begin{cases} 1 & N \equiv (\mathit{Listen} \parallel \mathit{Explore}, \{\mathsf{loc} \mapsto \{(x,y)\}, \\ & \qquad \mathsf{target} \mapsto \{\{M(x,y)\}\}\}) \\ & \quad \parallel (\mathit{Listen} \parallel \mathit{Explore}, \{\mathsf{loc} \mapsto \{(x,y)\}, \\ & \qquad \mathsf{target} \mapsto \{\{M(x,y)\}\}\}) \\ 0 & \mathit{otherwise} \end{cases}$$

Thus, the synchronisation over the broadcast action sense* has the effect of changing the local store values for identified targets target *to include the location (x,y) depending on whether it is specified as a target by function M or not. Finally applying **Sys-B** gives the transition rates in the following way:*

$$\begin{aligned} \mathbb{S}_{t,f}\Big[&\mathit{Robot} \parallel \mathit{Robot}\ \mathbf{in}\ (\{\mathsf{failr} \mapsto r_f, \mathsf{senser} \mapsto r_s\}, \\ &\quad \{\mathsf{failr} \mapsto [0,\infty), \mathsf{senser} \mapsto [0,\infty)\}, \rho), && \text{(System)} \\ &\quad \mathit{sense}^*[\pi_s]\langle M(x,y)\rangle, \{\mathsf{loc} \mapsto (x,y), \mathsf{target} \mapsto \emptyset\}\Big] && \text{(Transition label)} \\ = (&\mathit{Robot}, \mathbf{p}_1) \parallel (\mathit{Robot}, \mathbf{p}_2)\ \mathbf{in}\ \Big(\sigma(\{\mathsf{failr} \mapsto r_f, \mathsf{senser} \mapsto r_s\}, \\ &\qquad \{\mathsf{failr} \mapsto [0,\infty), \mathsf{senser} \mapsto [0,\infty)\}, \rho)\Big) \end{aligned}$$

which evaluates to the rate $2r_s$ for system descriptions consisting of collective

$$N \equiv (\mathit{Listen} \parallel \mathit{Explore}, \{\mathsf{loc} \mapsto \{(x,y)\}, \mathsf{target} \mapsto \{\{M(x,y)\}\}\}) \\ \parallel (\mathit{Listen} \parallel \mathit{Explore}, \{\mathsf{loc} \mapsto \{(x,y)\}, \mathsf{target} \mapsto \{\{M(x,y)\}\}\})$$

and the global store

$$\gamma_g = \{\mathsf{failr} \mapsto [0,\infty), \mathsf{target} \mapsto [0,\infty)\}$$

For other system descriptions the rate corresponding to broadcast action sense is given by* 0.

3.1.5 Population model

In this section we describe how the semantics defined in this chapter can be seen to give rise to a CTMDP for a given syntactic description of a model. Here we are interested in population CTMDP models as defined in Section 2.6 motivated by the study of collective behaviour.

The population CTMDP model $\mathcal{M}^N = (\mathbf{X}, \mathcal{T}, \mathcal{A}, \beta)$ for a system $S \in \textsc{Sys}$ can be derived iteratively based on the assumption that components with the same configuration (same process state and store) are indistinguishable. We start with S consisting of a collective $C_1 \parallel \cdots \parallel C_N$ operating in an environment $\mathcal{E}$. The function $\mathbb{C}_f$ can be used to determine all possible future configurations of each of the components C_i. If the union of all possible component configurations is finite we can define the finite state space $\mathcal{S}$ of $\mathcal{M}$ as the space of counting vectors specifying all possible future configurations of the system S.

For each state $\mathbf{s} \in \mathcal{S}$ we have a set $Sys_{\mathbf{s}} \subset \textsc{Sys}$ of corresponding unresolved Carma-C systems with a set of feasible actions $\mathcal{A}(\mathbf{s})$. For the derivation of the population model we need to add a restriction that any control action acts in the same way on the set of indistinguishable components — that is, the components that are indistinguishable before the application of the control action remain indistinguishable after. The rates corresponding to chosen actions and the reachable states are found using the function $\mathbb{S}_{t,f}$. Given a control action f denote the system S resolved by f by S_f. The rate of transition from $\mathbf{s} \in \mathcal{S}$ to $\mathbf{s}' \in \mathcal{S}$ at time t given control action f is then given by

$$\sum_{S \in Sys_{\mathbf{s}}} \sum_{S' \in Sys_{\mathbf{s}'}} \sum_{\ell \in \textsc{Lab}_S} \mathbb{S}_{t,f}[S_f, \ell](S')$$

Example 9. *When considering the running example suppose there is a unique target location in the system at* $(1,1)$. *Let us denote the component given by the process Listen* $\parallel$ *Explore at location* (x,y) *(that is* $\mathsf{loc} = \{(x,y)\}$*) by*

$$\begin{cases} \mathit{Robot}((x,y),0) & \textit{if}\ \mathsf{target} = \{\emptyset\} \\ \mathit{Robot}((x,y),1) & \textit{otherwise} \end{cases}$$

Suppose the system is given by the collective

$$\mathit{Collective} \overset{\mathsf{def}}{=} \mathit{Robot}((x,y),0)[100]$$

denoting a parallel composition of 100 *replicas of the* $\mathit{Robot}((x,y),0)$ *component. Fixing the number of available locations gives us a finite set of possible future configurations* *of the system. The population variables would simply count the numbers of Robot components in each location that share the same knowledge about the target. The choice of actions for each configuration of the system involves specifying the attributes* failr *and* senser*.*

3.2 Policy synthesis

Even with the constructed CTMDP model that relates to the Carma-C specification we are only partway towards establishing an optimisation problem. In order to do that we need to establish some optimality criteria by defining a cost or a reward function for the model. Consider a population model $\mathcal{M}^N = (\mathbf{X}^N, \mathcal{T}, \mathcal{A}, \beta)$ derived from a Carma-C model with N components. Consider the functional $Q^N : \Pi \to \mathbb{R}$, where Π is the space of possible policies. As an example, deterministic and stationary policies could be considered. The optimisation problem is thus defined as maximising some defined functional Q^N, i.e., finding a policy π^* that satisfies

$$Q^N[\pi^*] = \sup_{\pi \in \Pi} Q^N[\pi]$$

Suppose we fix a policy π and consider the resulting pCTMC model denoted $\mathcal{P}^N_\pi$. As before, let $\mathbf{X}^N_\pi(t)$ denote the stochastic process describing the time-evolution of the pCTMC. Let $V : \mathscr{D}_\mathcal{S} \to \mathbb{R}$ be a reward function on the space of trajectories of the stochastic process $\mathbf{X}^N_\pi(t)$. The corresponding reward functional on the space of policies is then given by

$$Q^N[\pi] \overset{\text{def}}{=} V(\mathbf{X}^N_\pi(t))$$

Example 10. *Take a state reward function $r : \mathcal{S} \to \mathbb{R}_{\geq 0}$ mapping the states of the CTMDP to positive real values. Define a value function corresponding to reward function r and policy π as the expected finite time-horizon ($0 \leq t \leq T$) cumulated reward:*

$$Q^N[\pi] \overset{\text{def}}{=} V(\mathbf{X}^N_\pi(t)) \overset{\text{def}}{=} \mathbb{E} \int_0^T r(\mathbf{X}^N_\pi(t))dt$$

Example 11. *A simple example of an optimality criterion would involve associating a non-zero reward with trajectories where a certain proportion (say 80%) of the components reaches the target location by time T. The control policies for the CTMDP arising from the running example come from the non-determinism in the definitions of attributes* failr *and* senser*. The conceptual differences in the two sources of non-determinism lead to the following considerations:*

- *Suppose that the failure rate as well as sensing rate is something we can control. In the case of failure rate this would then correspond to a robustness characteristic of* *the components and we might want to associate a cost for making the components more robust.*

- *Suppose that the failure rate is not something we can control. In that case we can decide to construct a space of policies where a policy is deterministic with respect to the attribute* senser *but assigns a fixed probabilistic policy for the attribute* failr.

In the context of the modelling framework described in this chapter we think of choosing a particular optimality criterion as part of the modelling process. For example, in the context of logical specification of desired behaviours of the system, the construction of the formal specification based on informal requirements is seen as part of the process of constructing or choosing an optimality criterion. Similarly, when incorporating data from real-life systems there are again decisions to be made about the precise way in which the available data is used. For example, choosing an appropriate metric for optimisation. Finally, in the context of multi-objective optimisation where one might deal with conflicting goals, there are choices to be made with respect to relative importance of the goals.

There are various aspects that can make the constructed policy synthesis problems difficult to solve. In formal modelling the optimality criteria for policy synthesis are commonly specified in a chosen logic. This makes it straightforward to define the value function in terms of satisfaction of the logical specification. However, model-checking the constructed models against a logical specification is highly non-trivial. Moreover, in the area of multi-objective optimisation it is often the case that the objectives are in conflict with each other making the optimisation problem difficult. In the rest of the thesis we are going to concentrate on the problem where, after defining a suitable value function applying exact solution methods quickly becomes infeasible with increasing model size. This is similar to the analysis and model-checking of pCTMCs.

Fixing a policy and using stochastic simulation methods is an option for gathering Monte-Carlo estimates for the values of the chosen function for different policies in the chosen policy space. An application of this idea was, for example, considered in [11] for time-dependent policies that maximise a reachability criterion. It can be easily seen that stochastic approximation methods like fluid, linear noise and moment closure approximation can, in many cases, be used to estimate the defined value function for given policies with potential for computationally efficient approximate solutions. An example of this idea in the discrete time setting can, for example, be seen in [57]. However, the class of models arising from Carma-C presents non-trivial challenges in applying the said approximation methods. This is due to the use of broadcast communication. For example, in order to apply fluid approximation, informally, we rely on the effect of actions being bounded as more components are introduced to the system.

In the following chapters we are going to consider this problem and propose a class of population systems for which the problems arising from broadcast communication can be mitigated.

Chapter 4

Approximations for mode-switching dynamics

As pointed out in Chapter 3, exact solutions to the arising optimisation problems are generally out of the question apart from in very trivial cases. In this chapter, motivated by Section 3.2, we concentrate on the CTMC dynamics of the population models arising from fixing a control policy to a CARMA-C model and study the applicability of existing approximation methods. When trying to apply methods like fluid, linear noise and moment closure approximations to the resulting population models we quickly notice the issues arising from application of broadcast communication. Namely, the existing methods make one of the following assumptions:

- The effects of transitions in the system are density dependent. Intuitively this means that the effects of individual transitions becomes in some sense "small" as larger and larger population sizes are considered allowing us to leverage fluid and linear noise approximation results.
- The state space can be partitioned according to variables corresponding to components that exist in the system in high-copy number and those that exist in low-copy numbers, allowing one to apply various hybrid approximation methods where part of the system description is kept discrete while the rest is given a continuous approximation.

Neither of these assumptions are satisfied in general for population models utilising broadcast. If no additional constraints on the number of broadcast receivers is imposed, then the update corresponding to broadcast communication cannot be normalised as done in standard fluid and linear noise approximation. On the other hand, we are not dealing with hybrid behaviour in the sense of low and high copy numbers of components which would allow for immediate application of the corresponding hybrid methods [18, 65]. In this chapter we show how a class of systems with broadcast communication can be framed so that existing fluid, linear noise and moment closure approximation methods are applicable. To that end we are going to introduce a way of "partitioning" the state space based on the magnitude of certain transitions which in our case correspond to information propagation through broadcast communication. Each class within such a partition is going to give rise to one of the discrete dynamic modes. In addition, we are going to extend the hybrid fluid limit results from [18] to the case of linear noise approximation and demonstrate the usage of existing approximation methods to the population models that feature broadcast communication.

4.1 Mode-switching population models

We start by defining an interesting class of stochastic population models. In particular, we consider models where the dynamics of a population can be separated into modes of behaviour. In [104] we argued that such models arise naturally from broadcast communication in collective systems with modes corresponding to the information that the collective has about its operating environment when broadcast is used to share the information amongst components. The stochastic model we are going to consider throughout the chapter is given by the following definition:

Definition 7. *We define a mode-switching population system as a joint Markov process* $\mathbf{Y}(t) = (\mathbf{X}(t), Z(t)) \in \mathbb{Z}^n \times \mathbb{Z}$ *where we make the assumption that each transition only changes the state of* $\mathbf{X}(t)$ *or* $Z(t)$ *but not both. Moreover, suppose that*

- *conditional on* $Z = z$*, the process* $(\mathbf{X}(t))_{t\geq 0}$ *is given by a population process* $\mathcal{P}_z = (\mathbf{X}, \mathcal{T}_z, \mathbf{X}_0)$ *with jump intensities* $q^z_{\mathbf{X}}(\mathbf{x}, \mathbf{x}')$*.*
- *conditional on* $\mathbf{X} = \mathbf{x}$*, the process* $(Z(t))_{t\geq 0}$ *is a continuous time Markov chain with intensities* $q^{\mathbf{x}}_Z(z, z')$*.*

The transition intensities for the joint process $\mathbf{Y}(t) = (\mathbf{X}(t), Z(t))$ *are thus defined as*

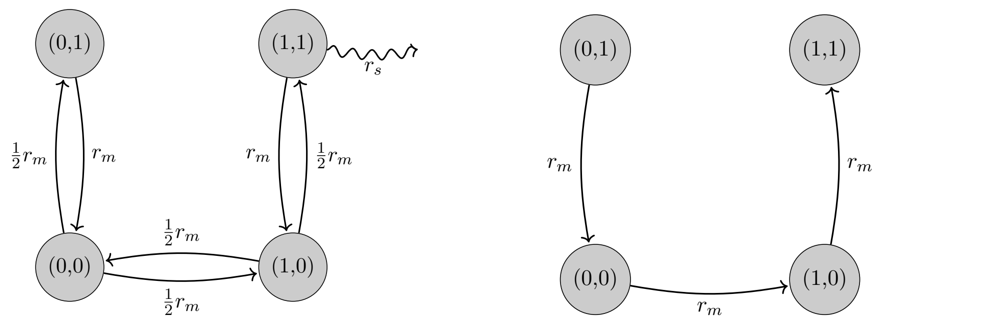
$$q(\mathbf{x}, \mathbf{x}', z, z') = \begin{cases} q^z_{\mathbf{X}}(\mathbf{x}, \mathbf{x}') & \text{for } z = z' \\ q^{\mathbf{x}}_Z(z, z') & \text{for } \mathbf{x} = \mathbf{x}' \\ 0 & \text{otherwise} \end{cases}$$

This defines a fairly general class of stochastic processes encompassing, for example, many stochastic gene expression models. In the context of the examples involving changes in population dynamics due to broadcast communication the process $\mathbf{X}$ corresponds to the current state of the population and Z corresponds to the dynamic mode being followed by the population. Similar models appear in works that consider biological processes coupled with their environment [126, 127] and where the behaviour of the environment is assumed to not depend on the behaviour of the population. As discussed in the introduction to this thesis we are going to consider situations where broadcast actions of a single component can bring about changes in the dynamics of the population. In such case we can note that the population process $\mathbf{X}$ is going to affect the mode-switching process Z and vice versa. In addition, there has been work on models of biochemical systems where the model is partitioned into low and high-copy number species with the two partitions treated differently [65, 66] with the low-copy number species similar to the mode-switching process in our case. Finally, Definition 7 has conceptual similarities to hybrid automata [67] and stochastic hybrid systems [69] in that models with different properties are coupled to describe the system. In the case of hybrid automata and stochastic hybrid systems the continuous behaviour is coupled with switching behaviour that depends on and also influences the continuous dynamics of the system. In the case of Definition 7 we similarly consider population dynamics coupled with a stochastic switching behaviour. The following example returns to a simple swarm robotics-inspired example to motivate and exemplify the construction and methods that follow.

Example 12. *A basic example of a mode-switching process as described in Definition 7 is given by the same swarm foraging scenario as considered in Chapter 3 where the robots are looking for a designated target area and gather to it. Unlike in Chapter 3, however, we are not going to consider the failure of robots. The model is constructed so that an exploration phase is followed by aggregating to the target area. The switch from the exploration to the aggregation phase happens when an individual in the swarm detects the target area and broadcasts this information to the rest of the swarm. When the space in which the swarm exists has a graph structure we can build an example mode-switching model by considering two population processes — one modelling a random walk on the graph and the second, a directed walk towards to the target node. A simple instance of such a model over a 2-by-2 grid is given by the transition diagrams for individuals in the swarm depicted in Figure 4.1.*

4.2 Transition-based partitioning

Note that in the context of this thesis the modelling workflow does not start with a direct specification of the mode-switching population model as defined in the previous section. Rather, in Chapter 3 we saw how CARMA-C models under a chosen control policy give rise to a pCTMC model. In this section we are going to give a construction



(a) Before broadcast – random walk (b) After broadcast – directed walk

Figure 4.1: Behaviour of individuals in the swarm model with 4 locations. The wavey line denotes the broadcast message being sent out from location (1,1) with rate r_s. The parameters r_m, r_s give the movement rate between locations, sensing rate and probability of detecting the location $(1,1)$ as the target, respectively.

under which such a population model, derived from a Carma-C specification, can be studied as a mode-switching population in Definition 7. The additional structure of mode-switching systems allows us to apply existing fluid, linear noise and moment closure-based approximations.

In order to characterise pCTMCs that fall into the subclass of mode-switching population dynamics we are going to propose the following transition-based partitioning method. We start with a pCTMC $\mathcal{P} = (\mathbf{X}, \mathcal{T}, \mathbf{X}_0)$ with $\mathbf{X} \in \mathbb{Z}^{kn}$ where k corresponds to the number of modes and n is the number of component states. Up to reordering of the state-space we consider the projections proj_i such that for any element

$$\mathbf{x} = (x_1, \cdots, x_n, x_{n+1}, \cdots, x_{2n}, x_{2n+1}, \cdots, x_{kn}) \in \mathbb{Z}^{kn}$$

the projection proj_i maps $\mathbf{x}$ to the value

$$\mathrm{proj}_i(\mathbf{x}) = (x_{in+1}, \cdots, x_{in+n}) \in \mathbb{Z}^n$$

Using this we define two types of transitions, called *mode switches* and *population transitions* of the population process $\mathcal{P}$ based on their effect under the projections above.

Definition 8. *We then say that a transition* $\tau = (\mathbf{v}_\tau(\mathbf{X}), r_\tau(\mathbf{X})) \in \mathcal{T}$ *is a mode switch if there exists exactly one pair of projections* proj_i *and* proj_j, $i \neq j$, *defined above such that for all* $\mathbf{x}$ *for which* $\mathrm{proj}_i(\mathbf{x}) \neq 0$

- $\mathrm{proj}_i(\mathbf{v}_\tau(\mathbf{x})) \neq 0$ *and* $\mathrm{proj}_j(\mathbf{v}_\tau(\mathbf{x}))) \neq 0$.
- $\mathrm{proj}_i(\mathbf{v}_\tau(\mathbf{x})) = -\mathrm{proj}_j(\mathbf{v}_\tau(\mathbf{x}))$.

- $\mathrm{proj}_i(\mathbf{x}+\mathbf{v}_\tau(\mathbf{x})) = 0$.

The first condition of the definition above defines a mode switch as a transition that causes a non-zero change in the state of the pCTMC under exactly two different projections. The transition is such that the change under the two projections is of equal magnitude but opposite in direction. The third condition states that as a result of a mode switch one of the two projections becomes zero. Note that if the state vector is non-zero under a single projection then this is the case also after a mode switch. We are going to use this property to classify population systems where the state remains non-zero under a single, but possibly changing, projection throughout the evolution. This is going to allow us to relate the pCTMCs arising from modelling broadcast communication in Carma-C to a mode-switching population system in Definition 7.

Example 13. *Consider the running example to illustrate the above definition for a mode switch. Up to reordering of the state-space we can consider the first* 4 *copies of* $\mathbb{Z}$ *to correspond to counts of robots at locations* $(0,1),(0,0),(1,0)$ *and* $(1,1)$ *that do not have information about the target and the second* 4 *copies of* $\mathbb{Z}$ *to correspond to the counts where the information about the target location is known. Thus, we have a copy of* $\mathbb{Z}^4$ *for each level of information — in this case two levels. Suppose the state before the broadcast is given by*

$$(49,0,0,1,0,0,0,0)$$

corresponding to 49 *robots in location* $(0,1)$ *and* 1 *robot in location* $(1,1)$ *with no knowledge about the target location. An update corresponding to broadcast would be given by*

$$\mathbf{v}_\tau(\mathbf{x}) = (-\mathrm{proj}_1(\mathbf{x}), \mathrm{proj}_1(\mathbf{x}))$$

and, in particular, a transition from the state above would result in

$$(49,0,0,1,0,0,0,0)+\mathbf{v}_\tau((49,0,0,1,0,0,0,0)) = (0,0,0,0,49,0,0,1)$$

with the locations of robots staying the same but all of them gaining the knowledge about the target location. It can be easily seen that the transition defined in such a way satisfies the conditions for a mode switch in Definition 8.

Definition 9. *We say that a transition* $\tau = (\mathbf{v}_\tau, r_\tau(\mathbf{X})) \in \mathcal{T}$ *is a population transition if the update vector* $\mathbf{v}_\tau \neq 0$ *is such that if*

$$\mathrm{proj}_i(\mathbf{v}_\tau(\mathbf{x})) \neq 0$$

then for all $j \neq i$ *we have*

$$\mathrm{proj}_j(\mathbf{v}_\tau(\mathbf{x})) = 0$$

The idea is that if the transition τ changes the state under the projection proj_i *then it leaves the state invariant under any other projection.*

Example 14. *It is easy to see then that the transitions corresponding to movement on the grid in the running example can be characterised as population transitions.*

These two types of transitions will suffice to characterise population systems that can be separated into modes.

Definition 10. *A pCTMC $\mathcal{P} = (\mathbf{X}, \mathcal{T}, \mathbf{X}_0)$ with $\mathbf{X} \in \mathbb{Z}^{kn}$, $k \geq 2$, is mode separable, if up to reordering of the state space there exist projections* $\mathrm{proj}_1, \cdots, \mathrm{proj}_k$ *such that*

- *if* $\mathrm{proj}_i(\mathbf{X_0}) \neq 0$ *then* $\mathrm{proj}_j(\mathbf{X_0}) = 0$ *for $i \neq j$. Thus, the initial conditions are non-zero under only one projection as defined above.*

- *if $\tau = (\mathbf{v}_\tau(\mathbf{X}), r_\tau(\mathbf{X})) \in \mathcal{T}$ then either it is a mode switch as defined in Definition 8 or a population transition in Definition 9.*

The above describes conditions under which a general pCTMC is mode separable. We are dealing with a mode separable pCTMC whenever the projections can be found such that transitions with respect to these projections satisfy the conditions in Definition 10. The above definition does not, however, specify how these projections should be constructed. Note that Definition 10 defines a rather strict notion of a mode separable system. The running example, for instance, can be modified so that each robot in the swarm receives the broadcast message with a probability $p < 1$. This creates the situation where robots that know of the target coexist in a system with ones that do not. The above definition would not be able to define such cases as mode-switching systems — the state of the pCTMC can only be non-zero under a single projection at any given time. One could consider imperfect broadcast communication a more general instance of mode switching where all possible outcomes of the broadcast message correspond to a different mode. Constructions that allow this could be considered in future work.

The final step is to specify the construction of mode-switching population models from the pCTMCs that are mode separable according to Definition 10. In particular, a mode separable pCTMC $\mathcal{P} = (\mathbf{X}, \mathcal{T}, \mathbf{X}_0)$ gets translated into mode-switching population process $\mathbf{Y} = (\mathbf{X}', Z)$ as follows:

- if $\mathrm{proj}_i(\mathbf{X}(\mathbf{t})) \neq 0$ then the mode process $Z(t) = i$.

- the state of the population $\mathbf{X}'(t)$ is given by $\sum_i \mathrm{proj}_i(\mathbf{X}(\mathbf{t}))$.

- conditional on $Z(t) = i$ the evolution of the population process $\mathbf{X}'$ is given by the population process $\mathcal{P}_i = (\mathbf{X}'_i, \mathcal{T}_i, \mathbf{X}_0)$ where $\mathcal{T}_i$ consists of population transitions $\mathbf{v}_\tau$ for which $\mathrm{proj}_i(\mathbf{v}_\tau(\mathbf{X}(\mathbf{t}))) \neq 0$.

- the mode-switching process $Z(t)$ changes according to mode switch transitions.

With that we have constructed a model from the mode-separable pCTMC for which we can hope to apply existing hybrid methods where the population variables are given a continuous description while the variable describing dynamics mode currently followed remains discrete.

4.3 Approximations

The problem of constructing a useful and computationally efficient approximation to the mode-switching dynamics as introduced above in Section 4.1, is intimately linked to works in the context of modelling biochemical and biological processes. In particular, previous works have considered the partitioning of the modelled species into two groups — one for components occurring in low-copy numbers and the other for high-copy numbers. The hybrid methods would model the high-copy number components continuously (sometimes deterministically) while the components with low-copy numbers are given a stochastic description [48, 66, 93, 113]. Bortolussi [18] gives related convergence results in the context of a stochastic process algebra based on the idea of fluid approximation for systems where certain populations are approximated continuously while others are kept discrete. The method of conditional moment closure [65] gives moment-based description of subsystem corresponding to high-copy number components and uses a stochastic model for the low-copy subsystem. Finally, some analyses have considered dynamics of stochastic reaction networks under fluctuating environments [70, 127] which aim to construct a model of the reaction network dynamics decoupled from its environment.

In general, the following cases arise from the mode-switching models following Definition 7.

- The dynamics of the switching process Z are independent of the marginal population process $\mathbf{X}$. An example would be given by considering Z as the environment for the population process. In such examples, the population is affected by the environmental process but not the other way round.

- The dynamics of the switching process Z depend on the marginal process $\mathbf{X}$ but the behaviour of the population $\mathbf{X}$ is not affected by the switching process. In such cases Z can be thought of as an observation variable modelling information available about the population to some external observer.

- The dynamics of mode-switching process Z depend on the population process $\mathbf{X}$ and vice versa. An example of such scenario is provided, for example, by models involving broadcast communication as in Example 12.

In this thesis, we are mostly interested in the third case where we have the two-way dependence between the mode-switching process and the population. The rest of this chapter is dedicated to approximation methods that can be applied to study the dynamics in such cases. Our main motivation stemming from the study of collectives where broadcast communication plays an important role in the dynamics of the components. In the running example used in this chapter we are going to interpret the states of the mode-switching process as corresponding to the levels of information available to the collective.

4.3.1 Fluid approximation methods

In order to analyse the mode-switching population system introduced in the previous section we are going focus on fluid approximation-based methods motivated by application in mean-field control methods in swarm behaviours [49]. Note that Bortolussi [18] presents a comprehensive set of results on fluid limits of pCTMCs exhibiting hybrid behaviour where the limiting behaviour is given in terms of piecewise deterministic Markov Processes (PDMP) [43]. For example, guarded behaviour, instantaneous transitions and stochastic jumps coupled with a population structure were considered. Here, we are going to leverage these results in the context of mode-switching population systems.

Due to the construction of the mode-switching dynamics the most general treatment of PDMPs is unnecessary for the discussion that follows. An aspect of the constructed models that simplifies the analysis is that the transitions of the mode-switching process do not change the state of the population variables. With this in mind we start by giving a definition of a subclass of PDMPs sufficient here (a similar restriction was also considered in [17]). The intuition is that a PDMP defines a process with continuous deterministic dynamics interrupted by random switching.

Definition 11. *(Simplified) piecewise deterministic Markov process (PDMP) [43] . Let E be a countable set and $M \subset \mathbb{R}^n$ a compact subset giving the domain of the continuous evolution. We assume that*

1. *For all $z \in E$ we have a smooth time-independent vector field $F^z : M \to \mathbb{R}^n$ such that the ODE $\frac{d}{dt}\mathbf{y}(t) = F^z(\mathbf{y}(t))$, for each initial condition $\mathbf{y}(0) \in M$, has a unique solution such that $\mathbf{y}(t) \in M$ for all $t \geq 0$.*

2. *All states $z, z' \in E$ are identified with unit vectors $e_z, e_{z'}$ in $\mathbb{R}^{|E|}$ and for all $\mathbf{x} \in M$ there is a continuous time Markov chain defined via the infinitesimal generator matrix $Q^{\mathbf{x}}$ with entries $q^{\mathbf{x}}(z, z')$.*

We define the corresponding PDMP as a stochastic process $(X(t), Z(t)) \in M \times \mathbb{R}^{|E|}$ *satisfying the equation*

$$\begin{pmatrix} X(t) \\ Z(t) \end{pmatrix} = \begin{pmatrix} X(0) + \int_0^t F(X(s))ds \\ Z(0) + \sum_z \sum_{z' \neq z} (e_{z'} - e_z) N_{zz'} \left(\int_0^t \bar{q}^{X(s)}(Z(s), z, z')ds \right) \end{pmatrix}$$

where

$$F(X(t)) = F^z(X(t)) \qquad \text{if } Z(t) = e_z$$

$$\bar{q}^{X(s)}(Z(s), z, z') = \begin{cases} q^{X(s)}(z, z') & \text{if } Z(s) = z \\ 0 & \text{otherwise} \end{cases}$$

and each $N_{zz'}$ *is a time-changed Poisson process counting the number of transitions from state* e_z *to* $e_{z'}$.

We have defined the process Z in terms of the Poisson or random time change representation of Markov chains [50] (described in Section 2.1.2) and constructed it so that its state is always given by a unit vector in $\mathbb{R}^{|E|}$. As described in Section 2.1.2, the time parameter is transformed based on the state of X and Z so that the process $N_{zz'}$ counts transitions from z to z' only. This gives us a compact and simplified definition of PMDPs. In the rest of the thesis we are going to use a shorthand and describe the state space of Z by integer values. However, the underlying construction as a process taking values in $\mathbb{R}^{|E|}$ remains in place. Details and the definition of PDMPs in a more general setting can be found in [43]. For the limit results in [18] the more general definition was considered, for example, to deal with instantaneous transitions which are not considered here.

In the following we are going to apply the results of [18] to conservative mode-switching population systems as introduced in Definition 7. We denote the joint process corresponding to a mode-switching population process of a fixed number of components by $(\mathbf{X}^N, Z^N)$ to indicate the population size N. For each state z of the mode-switching process Z^N we have a pCTMC $\mathcal{P}_z^N = (\mathbf{X}_z^N, \mathcal{T}_z^N, \mathbf{X}_0^N)$ corresponding to the population dynamics for the given mode z. We are going to consider the following conditions:

1. The pCTMCs $\mathcal{P}_z^N$ scale for all z as described in Section 2.4.1 — the background section for fluid approximations. In particular, for all z consider the scaled pCTMC $\hat{\mathcal{P}}_z^N = (\hat{\mathbf{X}}_z^N, \hat{\mathcal{T}}_z^N, \hat{\mathbf{X}}_0^N)$.

2. For the intensities $q_Z^{\mathbf{x},N}(z, z')$ of the process Z^N conditional on the state of the population $\mathbf{X}^N = \mathbf{x}$ we require that there exists a Lipschitz continuous function $f_{zz'}^N : \mathbb{R}^K \to \mathbb{R}$ such that
$$q_Z^{\mathbf{x},N}(z, z') = f_{zz'}^N \left(\frac{\mathbf{x}}{N} \right)$$

and $f^N_{zz'}$ converges uniformly to a continuous function $f_{zz'}$ as $N \to \infty$. Note that the scaling for the mode-switching system or discrete variables follows different rules and requires the rates to vary with the scaled population variables.

If the above conditions are satisfied then we can state the hybrid fluid limit theorem in the context of the mode-switching population system as follows.

Theorem 3 (Hybrid fluid limit [18])**.** *Let $(\mathbf{X}^N, Z^N)$ be a mode-switching population system satisfying the scaling conditions above and let $(\hat{\mathbf{X}}^N, \hat{Z}^N)$ be the corresponding scaled process. Suppose the initial conditions $\mathbf{X}^N(0) \to \hat{\mathbf{x}}(0)$ and $\mathbf{Z}^N(0) \to \hat{Z}^N(0)$ almost surely. Then the sequence of scaled processes $\{(\hat{\mathbf{X}}^N, \hat{Z}^N)\}$ converges weakly to a piecewise deterministic process $(\hat{\mathbf{x}}, \hat{Z})$*

$$\begin{pmatrix} \hat{\mathbf{x}}(t) \\ \hat{Z}(t) \end{pmatrix} = \begin{pmatrix} \hat{\mathbf{x}}_0 + \int_0^t F(\hat{\mathbf{x}}(s)ds) \\ \hat{Z}(0) + \sum_z \sum_{z' \neq z} (e_{z'} - e_z) N_{z'z} \left(\int_0^t \bar{f}_{zz'}(\hat{Z}(s), \hat{\mathbf{x}}(s)) ds \right) \end{pmatrix}$$

where

$$F(\hat{\mathbf{x}}) = \sum_{\tau \in \hat{\mathcal{T}}_z} f_\tau(\hat{\mathbf{x}}) \qquad \text{if} \quad \hat{Z}(t) = z$$

$$\bar{f}_{zz'}(\hat{Z}(s), \hat{\mathbf{x}}(s)) = \begin{cases} f_{zz'}(\hat{\mathbf{x}}(s)) & \text{if } \hat{Z}(s) = z \\ 0 & \text{otherwise} \end{cases}$$

The rates f_τ for $\tau \in \hat{\mathcal{T}}_z$ are given by the constructions in Section 2.4.1 applied to the pCTMC $\hat{\mathcal{P}}^N_z = (\hat{\mathbf{X}}^N_z, \hat{\mathcal{T}}^N_z, \hat{\mathbf{X}}^N_0)$ defining the process $\hat{\mathbf{X}}^N$ conditional on the state z of the mode-switching process.

Proof. The result is a special case of the result proposed in [18]. The idea of the proof is to consider the limiting behaviour of the process $\hat{\mathbf{X}}^N$ inductively between the stochastic jumps of the process Z^N. The details of the proof can be found in [18]. □

For cases like the running example we are not quite ready to apply the above theorem yet. To see that let us consider the following example.

Example 15. *Suppose each of the robots that reaches the location $(1,1)$ is capable of causing the mode switch at some rate r_s. Let the mode corresponding to exploration be denoted by $Z = 0$ and the mode corresponding to gathering to the target location be denoted by $Z = 1$. The total rate at which the mode switch happens is given by the number of robots at $(1,1)$, denoted X_{11}, multiplied by the rate r_s. That is, we get*

$$q_Z^{\mathbf{X},N}(0,1) = r_s X_{11}$$

which does not satisfy the scaling requirement defined above for the mode switches due to the dependence on the non-scaled population variables. Considering the limiting *behaviour for such transitions would not lead to anything interesting. In particular, taking $N \to \infty$, the limiting behaviour of the corresponding mode-switching process is such that the probability*

$$\mathbb{P}(\textit{broadcast has happened by time } t) \to 1 \qquad \textit{for all} \quad t > 0$$

Thus, the limit behaviour of the mode-switching process corresponding to a broadcast message being sent is expected to immediately reach its absorbing state. While the limit of the switching population constructed in such a way is valid it is not very useful when our aim is to understand the behaviour of the system at a fixed finite population size N.

In the following we construct an approximation dependent on the population size. To that end, in order to leverage Theorem 3 in the case of the running example we start by constructing a special instance of the mode-switching process Z^N that fixes the behaviour of the mode-switching process to a given population size.

Consider the fixed population of size $\tilde{N}$ and suppose for intensities $q_Z^{\mathbf{x},\tilde{N}}(z,z')$ of the mode-switching process $Z^{\tilde{N}}$ given the state of the population process $\mathbf{X}^{\tilde{N}} = \mathbf{x}$ there exists a Lipschitz continuous function $f_{zz'}^{\tilde{N}} : \mathbb{R}^K \to \mathbb{R}$ such that

$$q_Z^{\mathbf{x},\tilde{N}}(z,z') = \tilde{N} f_{zz'}^{\tilde{N}}\left(\frac{\mathbf{x}}{\tilde{N}}\right)$$

We then construct a special instance $\hat{Z}^N$ of Z^N such that the density-dependent scaling inside $f_{zz'}^N$ is done according to the population size N which is then multiplied by the population size $\tilde{N}$ that is kept constant. Effectively this fixes the behaviour of the mode-switching process Z^N to the case where the total population size is assumed to be $\tilde{N}$.

The choice of $\tilde{N}$ is entirely up to the modeller and is chosen based on the modelling problem at hand. For example, if we are interested in the approximate behaviour of the system with 100 homogeneous individual components we would set $\tilde{N} = 100$. In many scenarios the sensitivity to the chosen population size $\tilde{N}$ will also have to be considered. In particular, we might be interested to find out how much the overall dynamics of the population are expected to vary when varying $\tilde{N}$ around 100. Finally, if our modelling interest lies in swarms of around 100 robots it would be inappropriate to fix the population size $\tilde{N}$ for the approximation to a value an order of magnitude lower or higher as the quantitative and qualitative properties of the resulting dynamics will not give a good representation of the dynamics of interest.

The constructed process $\hat{Z}^N$ conditional on the state of the population $\mathbf{X}^N = \mathbf{x}$ is a continuous time Markov chain with the intensities defined by

$$q_{\hat{Z}}^{\mathbf{x},N}(z,z') = \tilde{N} f_{zz'}^{\tilde{N}}\left(\frac{\mathbf{x}}{N}\right)$$

In particular, we fix the rate function of the switching process depending on the scaled population variables to its behaviour at a chosen population level $\tilde{N}$.

Example 16. *Let us consider the running example. If again the population size at the location* $(1,1)$ *is* X_{11} *then the broadcast happens with rate* $r_s X_{11}$*. Similarly, the broadcast for the scaled population process given population density* $\frac{X_{11}}{N}$ *at location* $(1,1)$ *happens with the rate* $N r_s \frac{X_{11}}{N}$*. The special instance of the joint process at population level* 100 *according to the construction above is then defined by saying that the rate of broadcast is given by* $100 r_s \frac{n}{N}$*. This now satisfies the scaling condition for the mode-switching transitions.*

Note that the joint process $(\hat{\mathbf{X}}^N, \hat{Z}^N)$, constructed from $(\mathbf{X}^N, Z^N)$ via taking the special instance of Z^N, satisfies the conditions of Theorem 3 and thus converges weakly, as $N \to \infty$ to the PMDP $(\hat{\mathbf{x}}, \hat{Z})$ with $\hat{Z}$ defined by the intensities $q_{\hat{Z}}^{\mathbf{i},N}$ as $N \to \infty$. As mentioned, the motivation here is that while the behaviour of the population process is taken to its asymptotic limit the behaviour of the mode-switching process is kept fixed. Considering a sequence of such instances of the mode-switching process gives us a better idea about how the hybrid limit in Theorem 3 is reached.

Example 17. *For the running example we can easily see that the deterministic behaviour is given by the following drifts.*

$$\frac{d}{dt}\hat{\mathbf{x}}(t) = \begin{cases} \begin{pmatrix} r_m(-\hat{x}_{01}(t) + \frac{1}{2}\hat{x}_{00}(t)) \\ r_m(-\hat{x}_{00}(t) + \frac{1}{2}\hat{x}_{01}(t) + \frac{1}{2}\hat{x}_{10}(t)) \\ r_m(-\hat{x}_{10}(t) + \frac{1}{2}\hat{x}_{11}(t) + \frac{1}{2}\hat{x}_{00}(t)) \\ r_m(-\hat{x}_{11}(t) + \frac{1}{2}\hat{x}_{10}(t)) \end{pmatrix} & \text{for } \hat{Z}(t) = 0 \\ \begin{pmatrix} -r_m\hat{x}_{01}(t) \\ r_m(-\hat{x}_{00}(t) + \hat{x}_{01}(t)) \\ r_m(-\hat{x}_{10}(t) + \hat{x}_{00}(t)) \\ r_m(\hat{x}_{10}(t)) \end{pmatrix} & \text{for } \hat{Z}(t) = 1 \end{cases} \tag{4.1}$$

with

$$\hat{\mathbf{x}}(t) = \begin{pmatrix} \hat{x}_{01}(t) & \hat{x}_{00}(t) & \hat{x}_{10}(t) & \hat{x}_{11}(t) \end{pmatrix}^T$$

and initial conditions given by

$$\hat{\mathbf{x}}(0) = \begin{pmatrix} 0 & 1 & 0 & 0 \end{pmatrix}^T$$
$$\hat{Z}(0) = 0$$

The difficulty of treating the approximation numerically still remains as $\hat{Z}$ is a random process depending on $\hat{\mathbf{x}}$ while $\hat{\mathbf{x}}$ depends on $\hat{Z}$. Although it is possible to sample realisations of the time-inhomogeneous Poisson processes describing the evolution of the mode-switching process the computational demand of the problem remains high. More details on this will be given later in the results section of this chapter (Section 4.4).

4.3.2 Linear noise approximation

This section is a direct continuation of the discussion on hybrid fluid approximation in the previous section. That is, we consider the perturbation of the deterministic approximation, constructed according to the fluid approximation results, with Gaussian noise. We recall that while the hybrid approximation in the previous section takes into account the stochasticity due to the mode-switching process, it treats the behaviour within the modes in a deterministic manner. Perturbing the deterministic approximation with a Gaussian noise allows us to better capture the stochasticity within the modes that exists due to the behaviour of the individual components. To that end let $(\mathbf{X}^N, Z^N)$ be a mode-switching population system satisfying the scaling conditions in the previous section, let $(\hat{\mathbf{X}}^N, \hat{Z}^N)$ be the corresponding scaled process.

Theorem 4 (Hybrid linear noise approximation). *If the initial conditions $\hat{\mathbf{X}}^N(0) \to \tilde{\mathbf{X}}(0)$ and $\hat{Z}^N(0) \to \tilde{Z}(0)$ almost surely then for all $t < T < \infty$ the sequence of normalised processes $(\hat{\mathbf{X}}^N, \hat{Z}^N)$ converges weakly to a process $(\tilde{\mathbf{X}}, \tilde{Z})$ given by*

$$\begin{pmatrix} \tilde{\mathbf{X}}(t) \\ \tilde{Z}(t) \end{pmatrix} = \begin{pmatrix} \tilde{\mathbf{X}}(0) + \int_0^t \mathbf{F}(\tilde{\mathbf{X}}(s))ds + N^{-\frac{1}{2}} \int_0^t \mathbf{G}(\tilde{\mathbf{X}}(s))d\mathbf{W}_s \\ \tilde{Z}(0) + \sum_{z,z'}(e_{z'} - e_z)N_{zz'}\left(\int_0^t \bar{g}_{zz'}(\tilde{Z}(s), \tilde{\mathbf{X}}(s))ds\right) \end{pmatrix}$$

where $\mathbf{W}_s$ is the n-dimensional Wiener process and $\mathbf{G}(\mathbf{x})$ is the diffusion matrix defined as follows.

$$\mathbf{G}(\mathbf{x}) = \sum_{\tau \in \hat{\mathcal{T}}_z} \mathbf{v}_\tau \mathbf{v}_\tau^T f_\tau(\mathbf{x}) \qquad \textit{if } \tilde{Z}(t) = z$$

$$\bar{g}_{zz'}(\tilde{Z}(s), \tilde{\mathbf{X}}(s)) = \begin{cases} f_{zz'}(\tilde{\mathbf{X}}(s)) & \textit{if } \tilde{Z} = z \\ 0 & \textit{otherwise} \end{cases}$$

As a result the distribution of $\tilde{\mathbf{X}}$ at time t is Gaussian given the full history of the mode-switching system $\tilde{Z}$ up to time t. The vectors $\mathbf{v}_\tau$ are the changes to the state vector induced by a given transition τ of the underlying pCTMC while f_τ are the transition rates as considered in Section 4.3.1.

Proof. The similar inductive reasoning that is used to prove the hybrid fluid limit in [18] extends the convergence results to hybrid linear noise as stated above. Further discussion and an outline of the proof is given in Appendix A. □

The discussion on constructing a special instance of the mode-switching population carries over immediately with no changes. However, as before, the problem of computational realisation of the resulting approximations remains. Notably, the resulting stochastic differential equation has, both a continuous evolution component and a jump component. In comparison to the hybrid fluid approximation any simulation approach for the constructed hybrid linear noise approximation would also have to deal with the problem that the behaviour in between mode switches is stochastic. To illustrate the approximation we refer again to the running example.

Example 18. *The mean of the process was considered in the previous section so the only thing left to characterise are the Gaussian fluctuations around the mean. This relies on characterising the drift matrix for the noise process. In particular, if $\hat{Z}(t) = 0$ then*

$$\mathbf{G}(\mathbf{x}) = \begin{pmatrix} r_m(x_{01}+x_{00}) & -r_m(x_{01}+x_{00}) & 0 & 0 \\ -r_m(x_{01}+x_{00}) & r_m(x_{01}+2x_{00}+x_{10}) & -r_m(x_{00}+x_{10}) & 0 \\ 0 & -r_m(x_{00}+x_{10}) & r_m(x_{00}+2x_{10}+x_{11}) & -r_m(x_{10}+x_{11}) \\ 0 & 0 & -r_m(x_{10}+x_{11}) & r_m(x_{10}+x_{11}) \end{pmatrix} \tag{4.2}$$

On the other hand, if $\hat{Z}(t) = 1$ then

$$\mathbf{G}(\mathbf{x}(t)) = \begin{pmatrix} r_m x_{01} & -r_m x_{01} & 0 & 0 \\ -r_m x_{01} & r_m(x_{00}+x_{10}) & -r_m x_{00} & 0 \\ 0 & -r_m x_{00} & r_m(x_{00}+x_{10}) & -r_m x_{10} \\ 0 & 0 & -r_m x_{10} & r_m x_{10}) \end{pmatrix} \tag{4.3}$$

4.3.3 Moment based approximations

The one last approximation method left to discuss is based on an existing extension to moment closure methods. Mirroring the sections on fluid and linear noise approximation, the aim is to have a moment-based description of the evolution of the population variables while the mode-switching is kept discrete and stochastic. In particular, in the following we discuss the applicability of the method of conditional moments presented in [65] where a differential algebraic system of equations was constructed for the partitioning of a chemical reaction network model. The idea is to express the evolution of the discrete subsystem in terms of conditional moments. A similar approach was also considered in the context of linear noise approximation in [119]. The following lemma, given in [65], provides us with a starting point. We are going to use $\mathbb{E}[h(\mathbf{X}(t)) \mid z; t]$ to denote the expectation of $h(\mathbf{X}(t))$ conditional on the state $Z(t) = z$ of the mode-switching process. The notation $h(\mathbf{X}(t) = \mathbf{x})$ is used to denote the function h applied to the value $\mathbf{x}$ of the random variable $\mathbf{X}(t)$.

Lemma 1. *For any sufficiently smooth test-function $h : \mathbb{R}^n \to \mathbb{R}^n$*

$$\frac{d}{dt}\mathbb{E}\left[h(\mathbf{X}(t)) \mid z;t\right]p(z;t) = \sum_{\mathbf{x}} h(\mathbf{X}(t)=\mathbf{x})\frac{d}{dt}p(\mathbf{x},z;t) + \sum_{\mathbf{x}} p(\mathbf{x},z;t)\frac{d}{dt}h(\mathbf{X}(t)=\mathbf{x}) \tag{4.4}$$

Proof. Proof from [65] recreated in Appendix B.1. □

The equations that describe the time-evolution of the conditional moments as well as the probability density of the mode-switching process $Z(t)$ can then be found by applying the above lemma. First, by setting h in Equation 4.4 to be the constant function returning the value 1 and substituting in the Kolmogorov forward equation for $\frac{d}{dt}p(\mathbf{x},z;t)$ we get

$$\begin{aligned}
&\frac{d}{dt}p(z;t) = \sum_{\mathbf{x}} \frac{d}{dt}p(\mathbf{x},z;t) \\
&= \sum_{\mathbf{x}} \Bigg[\sum_{z', z\neq z'} q_Z^{\mathbf{x}}(z',z)p(\mathbf{x},z';t) + \sum_{\mathbf{x}'\neq\mathbf{x}} q_{\mathbf{X}}^{z}(\mathbf{x}',\mathbf{x})p(\mathbf{x}',z;t) \\
&\qquad\qquad - \left(\sum_{z', z\neq z'} q_Z^{\mathbf{x}}(z,z') + \sum_{\mathbf{x}'\neq\mathbf{x}} q_{\mathbf{X}}^{z}(\mathbf{x},\mathbf{x}') \right) p(\mathbf{x},z;t) \Bigg] \\
&= \sum_{z', z\neq z'} \left[\mathbb{E}\left[q_Z^{\mathbf{X}}(z',z) \mid z';t\right] p(z';t) \right] - p(z;t) \sum_{z', z\neq z'} \mathbb{E}\left[q_Z^{\mathbf{X}}(z,z') \mid z;t\right]
\end{aligned} \tag{4.5}$$

Note that the terms corresponding to the population transitions that do not have a direct effect on the mode variable cancel in the above expression. The conditional expectations $\mathbb{E}\left[q_Z^{\mathbf{X}}(z,z') \mid z;t\right]$ can be considered via Taylor expansion around the conditional mean, $\mathbb{E}[\mathbf{X}(t) \mid z;t]$. Let us denote the said conditional mean by $\mu(z,t)$ and note that it is a vector in $\mathbb{R}^n$ and thus define $\mu_i(z,t)$ as the i-th variable. The expansion up to the second order moments then gives us the following.

$$\begin{aligned}
\mathbb{E}\left[q_Z^{\mathbf{X}}(z,z') \mid z;t\right] &\approx q_Z^{\mu(z,t)}(z,z') + \\
&\frac{1}{2}\sum_{i,j=1}^{n} \frac{\partial^2 q_Z^{\mu(z,t)}(z,z')}{\partial X_i \partial X_j} \mathbb{E}\left[(X_i - \mu_i(z,t))(X_j - \mu_j(z,t)) \mid z;t\right]
\end{aligned}$$

The term $\mathbb{E}\left[(X_i - \mu_i(z,t))(X_j - \mu_j(z,t)) \mid z;t\right]$ is the conditional covariance of the population variables X_i and X_j. We are going to denote this quantity by $c_{ij}(z,t)$. In particular,

$$\mathbb{E}\left[q_Z^{\mathbf{X}}(z,z') \mid z;t\right] = q_Z^{\mu(z,t)}(z,z') + \frac{1}{2}\sum_{i,j=1}^{n} \frac{\partial^2 q_Z^{\mu(z,t)}(z,z')}{\partial X_i \partial X_j} c_{ij}(z,t)$$

It is worth noting that the above expansion is exact if the infinitesimal rates $q_Z^{\mathbf{X}}(z,z')$ are at most quadratic in the population variables. This results from the partial derivatives $\frac{\partial^3}{\partial X_i X_j X_k}$ for any i, j and k vanishing.

Example 19. *The running example considered throughout this chapter is special in the sense that the transition rates for the population variables depend linearly on the population variables leading to the following.*

$$\mathbb{E}\left[q_Z^{\mathbf{X}}(z,z') \mid z;t\right] = q_Z^{\mu(z,t)}(z,z') \tag{4.6}$$

That is, the expected rate at which the mode-switching happens only depends on the expectation of the population state. The second aspect of the example that greatly simplifies the problem is that the mode-switching is unidirectional and thus the transitions into mode $Z=0$ are associated with rate 0. Putting this into Equations 4.5 gives us

$$\frac{d}{dt}p(z=0;t) = -p(z=0;t)\mu_4(0,t)r_s \tag{4.7}$$

The evolution $p(z=1;t)$ is given symmetrically by

$$\frac{d}{dt}p(z=1;t) = p(z=0;t)\mu_4(0,t)r_s$$

In general, the derived equations for the probability distribution of the mode-switching process depend on conditional means. For example, the expressions for $\frac{d}{dt}p(z;t)$ in the running example depend on conditional mean $\mu_4(0,t)$. To acquire an expression for this mean we again leverage Lemma 1 by setting $h:\mathbf{X}\mapsto X_i$. This is a slight abuse of notation where the test-function h is thought to map the state of the population variable $\mathbf{X}(t)=(X_1(t),\cdots,X_n(t))$ to a vector that takes the value of $X_i(t)$ in its i-th component and is 0 otherwise. Throughout the derivations that follow we are going to use another similar shorthand where for any state $\mathbf{x}=(x_1,\cdots,x_n)$ we use x_i to denote the vector that has the value x_i on its i-th component and is 0 otherwise. With that in mind we get the following

$$\begin{aligned}\frac{d}{dt}\mu(z,t)p(z;t) = \sum_{\mathbf{x}} x_i \frac{d}{dt}p(\mathbf{x},z;t) = \sum_{z',z'\neq z} &\mathbb{E}\left[X_i(t)q_Z^{\mathbf{X}}(z',z) \mid z';t\right]p(z';t)\\ &-p(z;t)\sum_{z',z'\neq z}\mathbb{E}\left[X_i(t)q_Z^{\mathbf{X}}(z,z') \mid z;t\right]\\ &+p(z;t)\mathbb{E}\left[\sum_{\mathbf{x}} x_i q_{\mathbf{X}}^{z}(\mathbf{X}(t),\mathbf{x}) \mid z;t\right]\end{aligned}$$

It is then useful to rewrite the last summand in the following way:

$$\begin{aligned}
&\mathbb{E}\left[\sum_{\mathbf{x}} x_i q^z_{\mathbf{X}}(\mathbf{X}(t),\mathbf{x}) \mid Z(t)=z\right] \\
&\qquad = \mathbb{E}\left[\sum_{\mathbf{x}\neq\mathbf{X}(t)} x_i q^z_{\mathbf{X}}(\mathbf{X}(t),\mathbf{x}) + X_i(t) q^z_{\mathbf{X}}(\mathbf{X}(t),\mathbf{X}(t)) \mid z;t\right] \\
&\qquad = \mathbb{E}\left[\sum_{\mathbf{x}\neq\mathbf{X}(t)} x_i q^z_{\mathbf{X}}(\mathbf{X}(t),\mathbf{x}) - \sum_{\mathbf{x}\neq\mathbf{X}(t)} X_i(t) q^z_{\mathbf{X}}(\mathbf{X}(t),\mathbf{x}) \mid z;t\right] \\
&\qquad = \mathbb{E}\left[\sum_{\mathbf{x}\neq\mathbf{X}(t)} (x_i - X_i(t)) q^z_{\mathbf{X}}(\mathbf{X}(t),\mathbf{x}) \mid z;t\right]
\end{aligned}$$

Notice that although the sum above runs over all possible states of the population variable $\mathbf{X}(t)$, in our case only a small number of transitions will have a non-zero rate associated with them, namely the ones reachable via an update vector in the corresponding pCTMC. Thus letting v^i_τ denote the change induced by transition τ in the variable X_i (again v^i_τ is really a vector that is only non-zero in the i-th position) gives us

$$\begin{aligned}
&\mathbb{E}\left[\sum_{\mathbf{x}\neq\mathbf{X}(t)} (x_i - X_i(t)) q^z_{\mathbf{X}}(\mathbf{X}(t),\mathbf{x}) \mid z;t\right] \\
&\qquad = \mathbb{E}\left[\sum_{\tau\in\mathcal{T}_z} (X_i(t) + v^i_\tau - X_i(t)) r_\tau(\mathbf{X}(t)) \mid z;t\right] = \mathbb{E}\left[\sum_{\tau\in\mathcal{T}_z} v^i_\tau r_\tau(\mathbf{X}(t)) \mid z;t\right]
\end{aligned}$$

To illustrate this derivation and make it specific let us consider again the running example.

Example 20. *By the above discussion we can evaluate the following derivative conditional on $Z(t)=0$.*

$$\begin{aligned}
\frac{d}{dt}\mu_i(0,t)p(z=0;t) = p(z=0;t)\Bigg(& -\mathbb{E}\left[X_i(t) q^{\mathbf{X}}_Z(0,1) \mid z=0;t\right] \\
& + \mathbb{E}\left[\sum_{\tau\in\mathcal{T}_0} v^i_\tau r_\tau(\mathbf{X}(t)) \mid z=0;t\right]\Bigg)
\end{aligned}$$

As before we are going to Taylor expand the parts involving expectations around the conditional means and use the observation given by Equation 4.6.

$$\mathbb{E}\left[X_i(t) q^{\mathbf{X}}_Z(0,1) \mid z=0;t\right] = \mu_i(0,t)\mu_4(0,t) r_s + r_s c_{i4}(0,t)$$

Finally, again making use of the fact that the transitions depend linearly on the population variables we can write down the equation corresponding to the evolution of the *conditional mean at location* $(0,1)$ *as*

$$\begin{aligned}\frac{d}{dt}&\mu_1(0,t)p(z=0;t)\\&=p(z=0;t)\left[\mu_1(0,t)\mu_4(0,t)r_s+r_sc_{14}(0,t)+\frac{1}{2}\mu_2(0,t)r_m-\mu_1(0,t)r_m\right]\end{aligned}$$

Finally, this gives

$$p(z=0;t)\frac{d}{dt}\mu_1(0,t)=p(z=0;t)\left[-r_sc_{14}(0,t)+\frac{1}{2}\mu_2(0,t)r_m-\mu_1(0,t)r_m\right]$$

Similar equations can be written down for the rest of the first order conditional moments and can be found in Appendix B.2.

Next let us consider the higher order moments. That is for non-negative integers $i_1,\cdots i_n$ consider $h:\mathbf{X}(t)\mapsto\prod_{k=1}^n(\mu_k(z,t)-X_k(t))^{i_k}$ in Lemma 1 to get the equations for the higher order conditional moments centred around the conditional mean $\mu(z,t)$.

$$\begin{aligned}\frac{d}{dt}&c_{i_1\cdots i_n}(z,t)p(z;t)\\&=\sum_{z',z'\neq z}\mathbb{E}\left[q_Z^{\mathbf{X}}(z',z)\prod_{k=1}^n(\mu_k(z,t)-X_k(t))^{i_k}\mid z';t\right]p(z';t)\\&\quad-p(z;t)\sum_{z',z'\neq z}\mathbb{E}\left[q_Z^{\mathbf{X}}(z,z')\prod_{k=1}^n(\mu_k(z,t)-X_k(t))^{i_k}\mid z;t\right]\\&\quad-p(z;t)\mathbb{E}\Bigg[\sum_{\mathbf{x}\neq\mathbf{X}(t)}q_{\mathbf{X}}^z(\mathbf{X}(t),\mathbf{x})\prod_{k=1}^n(\mu_k(z,t)-x_k)^{i_k}\\&\qquad\qquad-\sum_{\mathbf{x}\neq\mathbf{X}(t)}q_{\mathbf{X}}^z(\mathbf{X}(t),\mathbf{x})\prod_{k=1}^n(\mu_k(z,t)-X_k(t))^{i_k}\mid z;t\Bigg]\end{aligned}$$

As before the sums over values that the population variable $\mathbf{X}(t)$ can take are determined by the transitions $\tau\in\mathcal{T}_z$ in the pCTMC corresponding to the state of the mode variable. In particular,

$$\begin{aligned}&\mathbb{E}\left[\sum_{\mathbf{x}\neq\mathbf{X}}q_{\mathbf{X}}^z(\mathbf{X},\mathbf{x})\prod_{k=1}^n(\mu_k(z,t)-x_k)^{i_k}-\sum_{\mathbf{x}\neq\mathbf{X}}q_{\mathbf{X}}^z(\mathbf{X},\mathbf{x})\prod_{k=1}^n(\mu_k(z,t)-X_k(t))^{i_k}\mid Z(t)=z\right]\\&=\mathbb{E}\left[\sum_{\tau\in\mathcal{T}_z}r_\tau(\mathbf{X})\prod_{k=1}^n(\mu_k(z,t)-X_k(t)+v_\tau^k)^{i_k}-\sum_{\tau\in\mathcal{T}_z}r_\tau(\mathbf{X})\prod_{k=1}^n(\mu_k(z,t)-X_k(t))^{i_k}\mid z;t\right]\end{aligned}$$

Example 21. *Again for the running example let us give an example of deriving the evolution equation for the second order moments given* $Z(t)=0$*. The covariance* $c_{14}(0,t)$

is, for example given by

$$\begin{aligned}\frac{d}{dt}c_{14}(0,t)p(z=0;t) = -p(z=0;t)\mathbb{E}\Big[X_4(t)r_s\big(X_1(t)X_4(t) - X_1(t)\mu_4(0,t)\\ - X_4(t)\mu_1(0,t) + \mu_1(0,t)\mu_4(0,t)\big) \mid z=0;t\Big]\\ -p(z=0;t)\mathbb{E}\Big[r_m\big(-0.5X_1(t)X_3(t) + X_1(t)X_4(t)\\ - X_1(t)\mu_4(0,t) - 0.5X_2(t)X_4(t) + 0.5X_2(t)\mu_4(0,t)\\ + 0.5X_3(t)\mu_1(0,t) - X_4(t)\mu_1(0,t)\big) \mid z=0;t\Big]\end{aligned}$$

Taking the Taylor expansion of the expectations leads to the following expression.

$$\begin{aligned}\frac{d}{dt}c_{14}(0,t)p(z=0;t) &= -p(z=0;t)\left[r_s\mu_4(0,t)c_{14}(0,t) + \frac{1}{2}r_s c_{441}(0,t)\right]\\ &\quad -p(z=0;t)\left[-\frac{1}{2}r_m c_{13}(0,t) + r_m c_{14}(0,t) - \frac{1}{2}r_m c_{24}(0,t)\right]\end{aligned}$$

Finally we can give the evolution equation, as done in [65], in the following form.

$$\begin{aligned}p(z=0;t)\frac{d}{dt}c_{14}(0,t) &= -p(z=0;t)\left[r_s\mu_4(0,t)c_{14}(0,t) + \frac{1}{2}r_s c_{441}(0,t)\right]\\ &\quad -p(z=0;t)\left[-\frac{1}{2}r_m c_{13}(0,t) + r_m c_{14}(0,t) - \frac{1}{2}r_m c_{24}(0,t)\right]\\ &\quad +p(z=0;t)r_s\mu_4(0,t)c_{14}(0,t)\\ &\approx -p(z=0;t)\left[-\frac{1}{2}r_m c_{13}(0,t) + r_m c_{14}(0,t) - \frac{1}{2}r_m c_{24}(0,t)\right]\end{aligned}$$

Notice that the above equation is not closed but depends on the third order moment. A close set of moment equations together with the constraint

$$p(z=0;t) + p(z=1;t) = 1$$

results in a differential algebraic system (DAE). Rather than deriving the equations by hand we rely on the CERENA toolbox for Matlab [80].

However, the method of conditional moments presents us with the difficulty of finding initial conditions for the arising DAE. Even for the example above, this is not trivial [102]. For example, a numerical solver, like Sundial IDAS [73], first requires the problem first to be reduced to an ODE system by repeatedly differentiating the algebraic equations with respect to the free variable. In addition one needs to provide initial conditions:

$$\begin{array}{ll} p(z;0) & \frac{d}{dt}p(z;t)|_{t=0}\\ \mu(z,0) & \frac{d}{dt}\mu(z,t)|_{t=0}\\ c_{ij}(z,0) & \frac{d}{dt}c_{ij}(z,t)|_{t=0}\end{array}$$

where in the case of $p(z;0)=0$ the derivatives for moments are not readily available. A detailed treatment of such structures in the context of biochemical processes is given in [65].

4.4 Results

In this section we present the simulation results for the methods in the case of the running example with the population $N = 100$ in the finite time interval $[0, 10]$. In addition we present the execution times for the tested implementations of the methods. In particular, we are going to consider simulated trajectories of the hybrid fluid and linear noise as well as the method of conditional moments. The hybrid fluid approximation for the running example is given by Equation 4.1. Similarly, the noise processes detailing the fluctuations around the fluid approximation are given by Equations 4.2 and 4.3. The equations for the method of conditional moments are given in Appendix B.2.

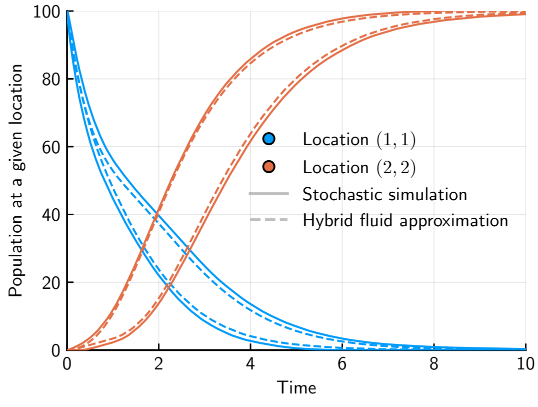
Recall that the hybrid fluid and noise approximations consist of a continuous state process coupled with a discrete jump process. In order to simulate such processes we have used the comprehensive DifferentialEquations.jl package for Julia programming language. The package supports setting up and simulating ODEs and stochastic differential equations (SDEs) coupled with jump transitions with time-varying rates. We have utilised the packaged solver methods for simulations. In particular, the SDEs arising from the hybrid noise approximation were solved using the discrete time-step Euler-Heun method with time-step $dt = 0.01$. Similarly, the packaged stochastic simulation algorithm is used for simulating the continuous time Markov chains. The conditional moment closure example is constructed and simulated using the CERENA toolbox for Matlab [80]. In all cases the models are translated into chemical reaction networks to make use of the existing implementations.

As we can see from Table 4.1 the direct simulations of the resulting hybrid approximations for population size $N = 100$ do not actually give us a reduction in simulation times over simulating the complete pCTMC. As mentioned this is due to the mode switches being defined with continuously varying rates making the simulation inefficient. Note however that increasing the population size would increase the size of the discrete state space and thus the simulation time required by SSA. This is not the case for the hybrid fluid and hybrid noise approximations where the population parameter only affects the dynamics of the system but does not have a direct effect on the complexity of the resulting simulation. The moment closure based method on the other hand offers good speed-up. It has to be noted though that much of the computational burden in the case of the moment-based method is in the symbolic construction of the DAE system. Even for this simple example, with only 4 population variables, the construction of the DAE system took approximately 3 minutes. On the other hand this is a one-off cost and negligible when conducting studies where characterisation of the dynamics under a large number of different parametrisations is needed.

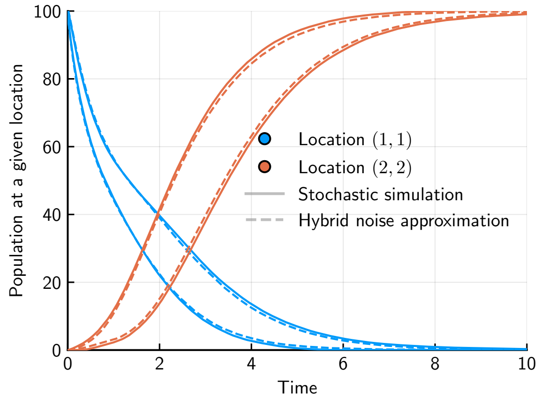
Figure 4.2 shows the visual comparison of the accuracy of the methods for the running example with population size 100. All considered methods result in a good agreement with the stochastic simulation. The high accuracy stems from the random effects from mode-switching dominating the fluctuations in the population behaviour. For example, not much accuracy is lost by moving to the hybrid fluid description where the fluctuations in the population behaviour are not considered.

Table 4.1: Comparison of simulation times of 5000 trajectories for the running example (seconds).

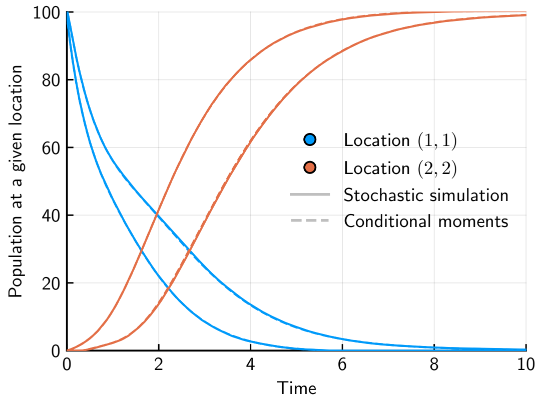
Population size N	Hybrid fluid	Hybrid noise	MCM	SSA
100	492.5	50.2	0.0073	5.3



(a) Hybrid fluid approximation simulation.



(b) Hybrid linear noise approximation simulation.



(c) Method of conditional moments.

Figure 4.2: Comparison of population measures at locations $(1,1)$ and $(2,2)$. Graphs show the intervals for one standard deviation above and below the mean.

4.5 Conclusion

In this chapter we defined mode-switching population models. These models are motivated by incorporating broadcast communication into the models of collectives. We then applied methods that have been developed for the analysis of models where the population variables can be partitioned into two sets, one corresponding to high-copy and the other for low-copy number components. Note however that such partitioning is not in general possible for population model incorporating broadcast. Instead we proposed an alternative method for partitioning of the pCTMC state space which is based on the “magnitudes” of transitions. The idea was to consider pCTMCs for which we can identify disjoint subsets of the state space and two types of transitions.

- Population transitions: the state of the population before and after the transition stays in the same subset of the state space. These transitions were assumed to satisfy scaling conditions for the construction of the continuous part of the hybrid fluid and linear noise approximations.
- Mode switches: the state of the population after the transition occupies a different subset of the state space. Mode-switching process Z was constructed so that each state of Z indicates whether a certain mode switch has happened or not. The process Z corresponded to the discrete process in the constructed approximations.

We concluded the chapter with a brief execution time analysis which underlined the computational challenges when applying the demonstrated methods. In the next chapter we are going to address the problem that although the hybrid fluid and linear noise approximation give excellent accuracy for the transient evolution of the running example they are, for small populations, computationally worse than simulating the original chain.

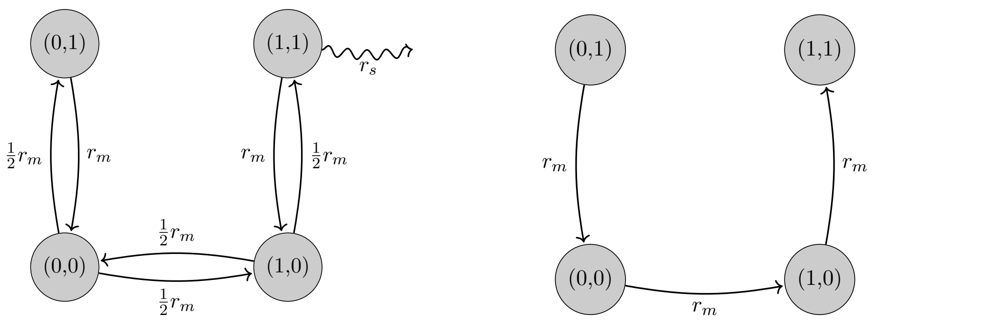
Finally, it is worth mentioning here that computational challenges are not the only hindrances in the application of the presented methods. From a practical perspective, for example, in the case of the method of conditional moments, we also ran into problems with the available software implementation in the package CERENA [80] which had many bugs making the reliable implementation of more complex models than the examples packaged with the tool, infeasible. Examples include ignoring arithmetic operations in propensity definitions of reactions without error notifications and problems with occasional and hard to spot incorrect initialisations of DAE systems.

Chapter 5

Heuristic marginalisation based computational strategies

In Chapter 4 we concentrated on principled approximations to mode-switching population processes. Such processes arise when considering information propagation through broadcast communication. The main idea was to consider stochastic systems where existing hybrid methods based on fluid, linear noise and moment closure approximations can be applied. However, as seen in Section 4.4, these the approximations are not trivial to consider computationally via direct simulation or, in the case of conditional moment closure, solving the underlying differential algebraic system of equations. In this chapter we consider explicit constructions that avoid stochastic simulation of the arising hybrid fluid and linear noise processes. Similarly, in the case of the constructed systems of moment equations we are going to propose an additional approximation that allows us to solve an ODE system rather than the more complex differential algebraic system. Throughout the chapter we are going to refer to the running example of Chapter 4 in order to exemplify the constructions. To begin we are going to present a recap of the model here.

Example 22. *As presented in Example 12 the running example features a simple swarm foraging scenario where the dynamics of the robot population can be split into two modes — a random walk on a graph structure shown with* 4 *nodes, and a directed walk towards the node designated as the target location. The graph structure and the behaviours of individual robots in the swarm are illustrated in Figure 5.1. The switch between those two modes happens via a transition corresponding to a robot at* $(1,1)$ *detecting it as the target location and broadcasting this information to the rest of the swarm. Recall that this results in a single robot in the swarm being able to change the dynamics of the population. In Chapter 4 we constructed the hybrid fluid, linear noise and conditional moment closure approximation for this example. In this chapter the resulting models*



(a) Before broadcast – random walk (b) After broadcast – directed walk

Figure 5.1: Behaviour of individuals in the swarm model with 4 locations. The state labelled B corresponds to the individual having detected a target and broadcast it to the rest of the swarm. The parameters r_m, r_s give the movement rate between locations, sensing rate and probability of detecting the location $(1, 1)$ as the target, respectively.

will be used to illustrate the discussion on computational treatment of the constructed approximations. The model in the example is considered for a fixed parametrisation $r_m = 1.0$ *and* $r_s = 0.2$*. Later, in the results section of the chapter, we are going to consider random parametrisations of* r_m *and* r_s *to gain a better understanding of how the constructed approximations perform.*

5.1 Marginal dynamics

We continue to use the notation $(\mathbf{X}, Z)$, that was set up in Chapter 4, to denote the scaled mode-switching population system. Observe that, in many cases, we are interested in the macro-level behaviour of the marginal process $\mathbf{X}$. This is the case, for instance, if the success or failure of the swarm in Example 22 can be defined through temporal behaviour of the population densities at the four locations. The naive approach to study such behaviour would be to marginalise out the effects of the switching process Z. The difficulties arise from the fact that although the joint process $(\mathbf{X}, Z)$ as well as the conditional processes $\mathbf{X} \mid Z$ and $Z \mid \mathbf{X}$ are Markov processes, this is no longer the case with the marginal processes $\mathbf{X}$ and Z. For example, the transition rates of the marginal process $\mathbf{X}$ at time t will depend on the history of the process $\mathbf{X}$ up to time t. In the following we are going to denote a sample path defining an instance of such history by $\mathbf{x}_{t^-}$ and the state of the history at time t by $\mathbf{x}_t$. We first note that the change in probability density of the marginal process being in state $\mathbf{x}$ at time t

conditional on the process history $\mathbf{x}_{s^-}$ is given by

$$\begin{aligned}\frac{\partial}{\partial t}p(\mathbf{x};t \mid \mathbf{x}_{s^-}) &= \frac{\partial}{\partial t}\sum_{z'} p(\mathbf{x},z';t \mid \mathbf{x}_{s^-}) \\ &= \frac{\partial}{\partial t}\sum_{z}\sum_{z'} p(\mathbf{x},z';t \mid z,\mathbf{x}_{s^-})p(z;s \mid \mathbf{x}_{s^-})\end{aligned}$$

That is, we consider the joint densities

$$p(\mathbf{x},z';t \cap z;s \mid \mathbf{x}_{s^-}) = p(\mathbf{x},z';t \mid z,\mathbf{x}_{s^-})p(z;s \mid \mathbf{x}_{s^-})$$

and marginalise out the mode-switching process Z. As the joint process $(\mathbf{X},Z)$ is Markov by construction the whole history $\mathbf{x}_{s^-}$ in the density $p(\mathbf{x},z';t \mid z,\mathbf{x}_{s^-})$ is redundant and only the state $\mathbf{x}_s$ is needed. From there, substituting in the Kolmogorov forward equation for the underlying CTMC gives us the following expression for the time-evolution of the conditional probability distribution of the marginal process $\mathbf{X}$.

$$\begin{aligned}\frac{\partial}{\partial t}p(\mathbf{x};t \mid \mathbf{x}_{s^-}) &= \frac{\partial}{\partial t}\sum_{z} p(z;s \mid \mathbf{x}_{s^-})\sum_{z'} p(\mathbf{x},z';t \mid \mathbf{x}_s,z;s) \\ &= \sum_{z} p(z;s \mid \mathbf{x}_{s^-})\sum_{z'}\Big[\sum_{\mathbf{x}'} p(\mathbf{x}',z';t \mid \mathbf{x}_s,z;s)q_{\mathbf{X}}^{z'}(\mathbf{x}',\mathbf{x})+ \\ &\qquad \sum_{k} p(\mathbf{x},k;t \mid \mathbf{x}_s,z;s)q_{Z}^{\mathbf{x}}(k,z')\Big]\end{aligned}$$

The term $p(\mathbf{x}',z';t \mid \mathbf{x}_s,z;s)$ in the above expression denotes the probability that the joint process $(\mathbf{X},Z)$ takes the value $(\mathbf{x}',z')$ at time t, given it takes the value $(\mathbf{x}_s,z)$ at time $s \leq t$. Analogous meaning is also used for the marginal probabilities $p(\mathbf{x};t \mid \mathbf{x}_{s^-})$ and $p(z;s \mid \mathbf{x}_{s^-})$. Evaluating the above expression at time $t = s$, we can find the time and history-dependent intensities for the marginal process $\mathbf{X}$. Note that the terms corresponding to jumps in the process Z vanish as they do not change the state of the population process $\mathbf{X}$ directly but only through the effect the mode-switching component Z of the process has on the transition rates of $\mathbf{X}$. Thus we get

$$q_{\mathbf{X}}(\mathbf{x}_s,\mathbf{x}) = \frac{\partial}{\partial t}p(\mathbf{x};t \mid \mathbf{x}_{s^-};s)\Big|_{t=s} = \mathbb{E}_{Z(s)\mid \mathbf{x}_{s^-}}\left[q_{\mathbf{X}}^{Z}(\mathbf{x}_s,\mathbf{x})\right] \tag{5.1}$$

The above characterises the marginal process $\mathbf{X}$ in terms of the conditional distribution of Z up to some sample path $\mathbf{x}_{t^-}$ of $\mathbf{X}$.

$$\frac{d}{dt}p(\mathbf{x};t \mid \mathbf{x}_{t^-};t) = \mathbb{E}_{Z(t)\mid \mathbf{x}_{t^-}}\left[q_{\mathbf{X}}^{Z}(\mathbf{x}_t,\mathbf{x})\right] \tag{5.2}$$

As pointed out at the start of this section, the above equation describing the future behaviour of the marginal process $\mathbf{X}$ at time t, depends on the history of the process up to time t. Thus, in order to understand the marginal process $\mathbf{X}$ we have to reconstruct the stochastic process Z at time t from a sample trajectory $\mathbf{x}_{t-}$ of the population process $\mathbf{X}$. We use the notation

$$\pi_t(z') = p(z';t \mid \mathbf{x}_{t-})$$

to denote the probability of $Z = z'$ given a sample trajectory $\mathbf{x}_{t-}$. Following [28] we can derive the evolution equation for the distribution $\pi_t(z')$ in terms of the marginal process $\mathbf{X}$. We are going to follow the established terminology and call the distribution $\pi_t(z')$ a filtering distribution [101]. Construction of a useful approximate description of the marginal process $\mathbf{X}$ thus relies on the problem of finding a sufficiently simple description of the filtering distribution.

Note that the sample trajectory $\mathbf{x}_{t-}$ is piecewise constant (and discontinuous) and thus the time derivative $p(z';t \mid \mathbf{x}_{t-})$ has discontinuities whenever $\mathbf{x}_{t-}$ jumps. Thus, the filtering distribution $\pi_t(z')$ involves two processes: continuous evolution as long as $\mathbf{x}_{t-}$ is constant and discontinuous jumps whenever $\mathbf{x}_{t-}$ jumps. The following evolution equation is going to describe the continuous change in the filtering distribution at time t given the state of the history at time t is $\mathbf{x}_t = \mathbf{x}$.

$$\frac{d}{dt}\pi_t(z') = \sum_z \pi_t(z) q_Z^{\mathbf{x}}(z,z') - \pi_t(z')\left[q_{\mathbf{X}}^{z'}(\mathbf{x},\mathbf{x}) - \mathbb{E}_{\pi_t}\left[q_{\mathbf{X}}^{Z}(\mathbf{x},\mathbf{x})\right]\right] \tag{5.3}$$

The first part, $\sum_z \pi_t(z) q_Z^{\mathbf{x}}(z,z')$, can be recognised as the Kolmogorov forward equation for the marginal mode-switching process Z consisting of transition that only change the state of Z. The second part of the equation contains the information gained from observing the trajectory $\mathbf{x}_{t-}$. The contributions of this part are more significant for the states z' that are not too unlikely given the trajectory $\mathbf{x}_{t-}$ (that is, $\pi_t(z')$ not too small) and for which the flow out of state $\mathbf{x}$ differs from the expected flow out of $\mathbf{x}$ with respect to the filtering distribution. That is, we gain information about the state of Z from observing the trajectory of $\mathbf{X}$ due to the fact that the dynamics of $\mathbf{X}$ depend on Z. If the process $\mathbf{X}$ jumps at time t then the filtering distribution straight after the jump is given by

$$\pi_{t+}(z') = \frac{\pi_{t-}(z') q_{\mathbf{X}}^{z'}(\mathbf{x}_{t-},\mathbf{x}_{t+})}{\sum_z \pi_{t-}(z) q_{\mathbf{X}}^{z}(\mathbf{x}_{t-},\mathbf{x}_{t+})} \tag{5.4}$$

where $\mathbf{x}_{t-}$ and $\mathbf{x}_{t+}$ denote the state of trajectory $\mathbf{x}_{t-}$ before and after the jump respectively. The trivial example of the filtering distribution happens when the dynamics of the population variable $\mathbf{X}$ do not depend on the mode-switching process Z. In this case the filtering distribution simply corresponds to the time-inhomogeneous CTMC given by $\pi_t(z) = \sum_z \pi_t(z) q_Z^{\mathbf{x}_t}(z,z')$. The above characterisation of the filtering distribution is going to serve as the starting point for the approximations considered in this chapter. The derivation of the above follows [28] exactly and is not recreated. However, the same idea is used to derive a filtering distribution for the hybrid fluid approximation in Appendix C. Note that it would be equally valid to construct a filtering equation for the process $\mathbf{X}$ given a history of Z. The choice made here to construct the filtering equation for Z up to history of $\mathbf{X}$ is done based on Z being a comparatively simpler process.

5.2 Time-inhomogeneous Markov process

The structure of the derived filtering equation is in general too complex to consider directly [28]. In this section we are going to propose heuristic simplifications of the problem that allow us to construct approximations to the marginal population dynamics of process $\mathbf{X}$. We start off by setting up the simplifications from the point of view of the complete mode-switching population process $(\mathbf{X}, Z)$ defined in Section 4.2. The constructions are then carried over to fluid, linear noise and moment closure approximations with minor modifications.

5.2.1 Filtering heuristics

The heuristics for the approximations presented in this chapter are going to rely on observations about the following term in Equation 5.3.

$$\pi_t(z')\left[q_{\mathbf{X}}^{z'}(\mathbf{x},\mathbf{x}) - \mathbb{E}_{\pi_t}\left[q_{\mathbf{X}}^{Z}(\mathbf{x},\mathbf{x})\right]\right]$$

Intuitively, this part accounts for the information gained by observing a sample trajectory of $\mathbf{X}$ up to time t. Note that the above quantity is "small" if the transition rates given the state of the mode-switching process z' deviate little from the expected rates with respect to the filtering distribution — $q_{\mathbf{X}}^{z'}(\mathbf{x},\mathbf{x}) \approx \mathbb{E}_{\pi_t}\left[q_{\mathbf{X}}^{Z}(\mathbf{x},\mathbf{x})\right]$. Intuitively, if the population dynamics in different modes are not very different the observation contains less information about the filtering distribution. For example, if the process $\mathbf{X}$ is statistically independent of the process Z then we have the equality $q_{\mathbf{X}}^{z'}(\mathbf{x},\mathbf{x}) = \mathbb{E}_{\pi_t}\left[q_{\mathbf{X}}^{Z}(\mathbf{x},\mathbf{x})\right]$. Similarly, we can consider the case where the first part of the evolution equation, Equation 5.3, dominates the contribution from the observations:

$$\sum_{z} \pi_t(z) q_Z^{\mathbf{x}}(z,z') \gg \pi_t(z')\left[q_{\mathbf{X}}^{z'}(\mathbf{x},\mathbf{x}) - \mathbb{E}_{\pi_t}\left[q_{\mathbf{X}}^{Z}(\mathbf{x},\mathbf{x})\right]\right] \tag{5.5}$$

In either of these cases an approximation heuristic we can study is to ignore the additional information the history $\mathbf{x}_{t-}$ gives us about the process Z and assume that the Kolmogorov forward equation part of Equation 5.3 tells us everything about the evolution of the distribution of Z at time t given the history $\mathbf{x}_{t-}$. This allows us to treat the continuous evolution of the filtering distribution as a time-inhomogeneous continuous time Markov chain. Thus we get

$$\frac{d}{dt}\pi_t(z') = \sum_z \pi_t(z) q_Z^{\mathbf{x}}(z,z')$$

Similarly, we need to consider the discrete evolution of the filtering distribution. Again, if Equation 5.5 holds then the filtering distribution before the jump in $\mathbf{x}_{t^-}$ is approximately equal to the filtering distribution after the jump. This simplified form of the filtering distribution is going to be used later in conjunction with fluid, linear noise and moment-based approximations to construct an approximation to the mode-switching system.

For the last part of this section we consider the situation where we have constructed the distribution of the marginal process Z at time t. Suppose that this is enough to describe the future behaviour of the marginal process $\mathbf{X}$. In particular, we assume that all memory effects are expressed through the marginal process Z at time t.

$$\frac{d}{dt}p(\mathbf{x}';t \mid \mathbf{x};t) \approx \mathbb{E}_Z\left[q_{\mathbf{X}}^{Z}(\mathbf{x},\mathbf{x}')\right] \tag{5.6}$$

That is, up to knowing the marginal distribution Z at time t, we treat the future evolution of the marginal process $\mathbf{X}$ approximately as a time-inhomogeneous Markov process. At this point it is important to remember that this construction is not valid for the memory-dependent marginal processes considered in this thesis and in particular the running example (Example 12) from Chapter 4. In the following example, however, we are going to consider the above discussion in the case of the running example in order to motivate further study of this simplification.

Example 23. *Consider the approximate marginal evolution of the running example in the time interval* $[0,T]$ *assuming the distribution of* Z *at every point in the interval is known. We can simulate this situation by performing stochastic simulation on the full model* $(\mathbf{X},Z)$ *to find the distribution of* Z *empirically. Given the empirical distribution we can simulate the time-inhomogeneous Markov process given by Equation 5.6. One standard deviation above and below the mean for the population measures corresponding to the locations* $(1,1)$ *and* $(2,2)$ *resulting from* 1000 *simulation runs of the time-inhomogeneous process and the same statistics for the full mode-switching population process simulated for* 5000 *runs are given for reference in Figure 5.2. The smaller set of runs for the inhomogeneous process was chosen due to practical considerations — even though the inhomogeneous model features a smaller state space it is computationally much more inefficient to simulate due to continuously varying rates.*

Expectedly, the constructed time-inhomogeneous chain in Example 23 underestimates the variance. This is due to the loss of information resulting from approximating the

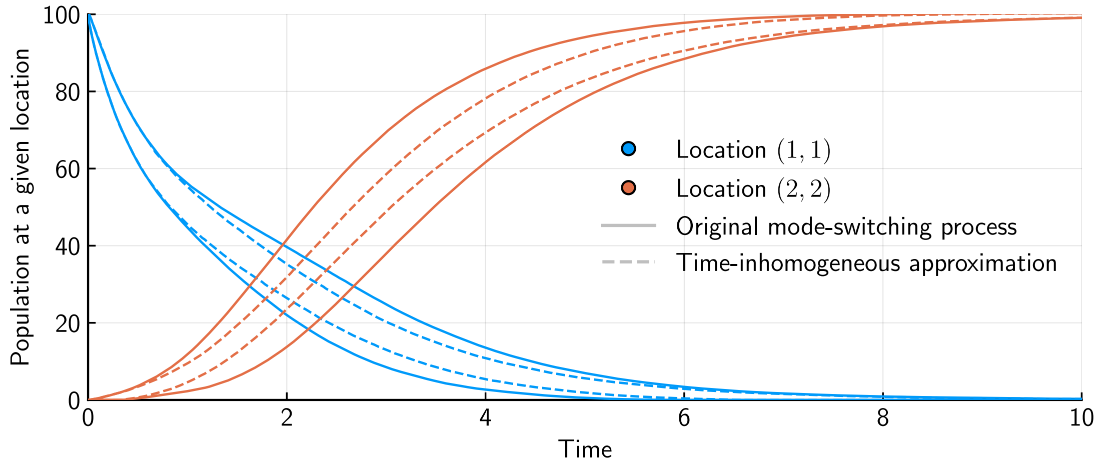


Figure 5.2: Visual comparison of the time-inhomogeneous process constructed given the knowledge of the mode-switching distribution for the running example and the original mode-switching population process. Each maching pair of lines (e.g. dashed orange, solid blue) shows one standard deviation above and below the mean. The mean itself is not shown.

non-Markovian marginal process $\mathbf{X}$ with a process that, although time-inhomogeneous, is Markovian. However, as a lower bound estimate for the population behaviour the approximate time-inhomogeneous process does capture the mode-switching population dynamics in a reasonably good manner and warrants further study. As noted in Example 23 the simulation of the arising time-inhomogeneous processes is very inefficient due to continuously changing rate parameters. The construction, however, does provide some insight on how to take this forward. In particular, in the next section we look at combining the arguments made in this section about the filtering distribution with the results for hybrid fluid, linear noise and moment closure approximations in Chapter 4 in order to achieve a computationally efficient approximation.

5.3 Fluid approximation

In this section we will demonstrate the computational approaches based on the hybrid fluid approximation to the scaled mode-switching process $(\hat{\mathbf{X}}, \hat{Z})$. In particular, let us denote the limit PDMP constructed in Section 4.3.1 by $(\hat{\mathbf{x}}, \hat{Z})$. In the same way as done for the process $(\mathbf{X}, Z)$ previously in Section 5.1 we consider the marginal process $\hat{\mathbf{x}}(t)$ given a sample trajectory $\mathbf{x}_{t-}$. Denoting the limit drift vector of the process $\hat{\mathbf{x}}(t)$ given the state of the mode-switching process is $\hat{Z}(t) = z$ as $\mathbf{F}^z$ and the i-th component of the vector as $\mathbf{F}_i^z$ we get

$$\begin{aligned}\frac{\partial}{\partial t}p(\mathbf{x};t \mid \mathbf{x}_{s-};s) &= \frac{\partial}{\partial t}\sum_z p(z;s \mid \mathbf{x}_{s-};s)\sum_{z'} p(\mathbf{x},z';t \mid \mathbf{x}_s,z;s)\\ &= \sum_z p(z;s \mid \mathbf{x}_{s-};s)\sum_{z'}\Big[-\sum_i \partial_i \mathbf{F}_i^{z'}(\mathbf{x})p(\mathbf{x},z';t \mid \mathbf{x}_s,z;s)\\ &\qquad + \sum_k \left[p(\mathbf{x},k;t \mid \mathbf{x}_s,z;s)q_Z^{\mathbf{x}}(k,z'\right)\Big]\end{aligned}$$

where $p(\mathbf{x},z';t \mid \mathbf{x}_s,z;s)$ denotes the probability that the joint process $(\hat{\mathbf{x}},\hat{Z})$ takes the value $(\mathbf{x},z')$ at time t and the value $(\mathbf{x}_s,z)$ at time $s \leq t$. The above makes use of the characterisation of the Kolmogorov forward equation for the transition density of a stochastic hybrid systems which we briefly introduced in the background chapter, Chapter 2. For full details see [12]. In particular, the expression depends on the continuous evolution given by the limit drifts $\mathbf{F}^z$ and discrete jumps of the mode-switching process. As in the previous section we use the fact that, according to the construction of the mode-switching population systems in Definition 7, the discrete jumps do not have a direct effect on the state of the population variables. Thus, evaluated at $t = s$, we get the following time-evolution of the marginal process where the contribution from the discrete jumps vanishes.

$$\frac{d}{dt}p(\mathbf{x};t \mid \mathbf{x}_{t-}) = \mathbb{E}_{\hat{Z}|\mathbf{x}_{t-}}\Big[-\sum_i \partial_i \mathbf{F}_i^{\hat{Z}}(\mathbf{x})\Big]$$

Again we are going to consider the filtering distribution

$$\pi_t(z') = p(z';t \mid \mathbf{x}_{t-})$$

corresponding to the probability of $\hat{Z} = z'$ given the observed history $\mathbf{x}_{t-}$ of the population process $\hat{\mathbf{x}}$. Note that the sample paths of the population process $\hat{\mathbf{x}}$ are continuous. Thus the filtering distribution can be shown to be

$$\frac{d}{dt}\pi_t(z') = \sum_z \pi_t(z)q_{\hat{Z}}^{\mathbf{x}}(z,z') - \pi_t(z')\Big[\sum_i \partial_i \mathbf{F}_i^{z'}(\mathbf{x}) - \mathbb{E}_{\hat{Z}|\mathbf{x}_{t-}}\Big[\sum_i \partial_i \mathbf{F}_i^{\hat{Z}}(\mathbf{x})\Big]\Big]$$

where the first part of the equation can be recognised as the Kolmogorov forward equation for the marginal process $\hat{Z}$ and the second part corresponds to the contribution from observing the history $\mathbf{x}_{t-}$. The derivation of the filtering distribution follows analogously to the one given for CTMCs in [28] and is given for completeness in Appendix C. As suggested in the previous section we are going to, as the first order heuristic for deriving approximate dynamics, consider the following approximations.

$$\frac{d}{dt}\pi_t(z') \approx \sum_z \pi_t(z)q_{\hat{Z}}^{\mathbf{x}}(z,z') \tag{5.7}$$

$$\frac{d}{dt}p(\mathbf{x};t) \approx \mathbb{E}_{\hat{Z}}\Big[-\sum_i \partial_i \mathbf{F}_i^{\hat{Z}}(\mathbf{x})\Big] \tag{5.8}$$

Recall here that the only source of stochasticity that was left in the PDMP approximation of the mode-switching population process resulted from the jump process describing the mode-switching. However, in general this does not make the marginal process $\hat{\mathbf{x}}$ or even the conditional distribution $\hat{\mathbf{x}}(t)$ given the history $\mathbf{x}_{t^-}$ deterministic. The stochasticity introduced by the mode-switching process $\hat{Z}$ is still present in the marginal process $\hat{\mathbf{x}}$. However, up to knowing the distribution $\hat{Z}$ Equation 5.8 describes a deterministic system with the time-derivative given by

$$\frac{d}{dt}\hat{\mathbf{x}}(t) = \mathbb{E}_{\hat{Z}}\left[\mathbf{F}^{\hat{Z}}(\hat{\mathbf{x}}(t))\right] \tag{5.9}$$

The question then is how to combine Equations 5.7 and 5.8 to construct an approximation to the marginal process $\hat{\mathbf{x}}$.

5.3.1 Direct coupling

The first approach we are going to consider for solving the system given in Equation 5.9 is to directly couple the approximate filtering distribution

$$\frac{d}{dt}\pi_t(z') = \sum_z \pi_t(z) q_{\hat{Z}}^{\mathbf{x}}(z, z')$$

with the approximation of the deterministic drift given by

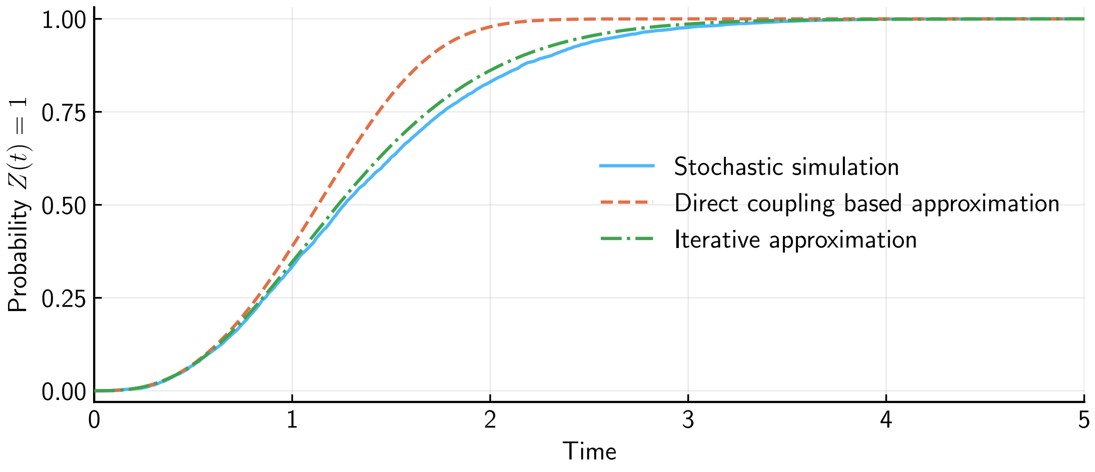
$$\frac{d}{dt}\hat{\mathbf{x}}(t) = \mathbb{E}_{\hat{Z}}\left[\mathbf{F}^{\hat{Z}}(\hat{\mathbf{x}}(t))\right] = \sum_z \pi_t(z)\mathbf{F}^{z}(\hat{\mathbf{x}}(t))$$

This is done by setting $\frac{d}{dt}\pi_t(z') = \sum_z \pi_t(z) q_{\hat{Z}}^{\hat{\mathbf{x}}(t)}(z, z')$. If the number of modes is not too large this can easily be solved as a system of ODEs.

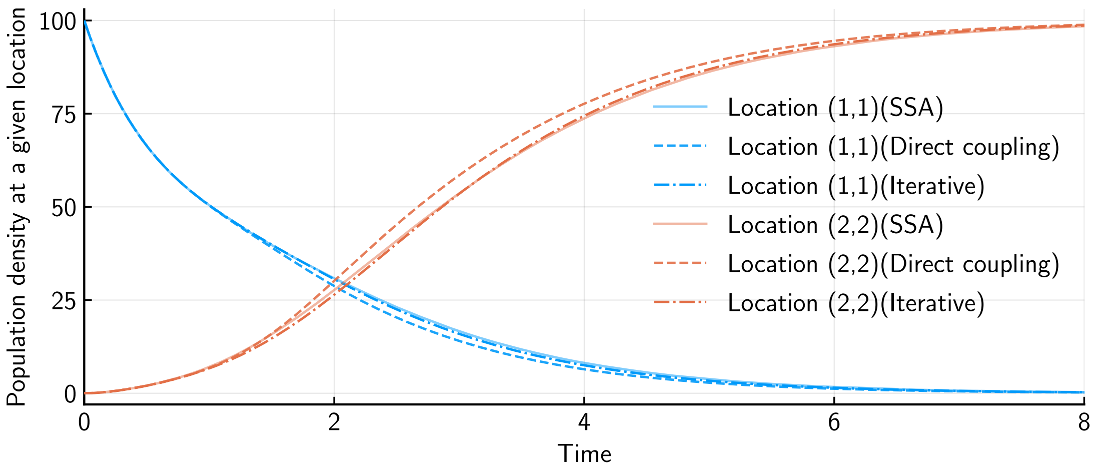
Example 24. *The running example features a single mode switch and thus we can construct the following system of ODEs*

$$\begin{aligned}
\frac{d}{dt}\hat{\mathbf{x}}(t) &= \pi_t(0)\begin{pmatrix} r_m(-\hat{x}_{01}(t) + \frac{1}{2}\hat{x}_{00}(t)) \\ r_m(-\hat{x}_{00}(t) + \frac{1}{2}\hat{x}_{01}(t) + \frac{1}{2}\hat{x}_{10}(t)) \\ r_m(-\hat{x}_{10}(t) + \frac{1}{2}\hat{x}_{11}(t) + \frac{1}{2}\hat{x}_{00}(t)) \\ r_m(-\hat{x}_{11}(t) + \frac{1}{2}\hat{x}_{10}(t)) \end{pmatrix} + \pi_t(1)\begin{pmatrix} -r_m\hat{x}_{01}(t) \\ r_m(-\hat{x}_{00}(t) + \hat{x}_{01}(t)) \\ r_m(-\hat{x}_{10}(t) + \hat{x}_{00}(t)) \\ r_m(\hat{x}_{10}(t)) \end{pmatrix} \\
\frac{d}{dt}\pi_t(0) &= -\pi_t(0) r_s \hat{x}_{11}(t) N \\
\frac{d}{dt}\pi_t(1) &= \pi_t(0) r_s \hat{x}_{11}(t) N
\end{aligned}$$

where N is the chosen population size. According to the constructions presented in Section 4.3.1 the jump rates of the mode-switching process depend on the population size and accordingly we have linked the behaviour of the mode-switching process to a



(a) Comparison of empirical distribution for $Z = 1$ and the same distribution estimated from the fluid approximation-based construction. The iterative method gives an improved estimate.



(b) Empirical means for the locations (0,0) and (1,1) from 5000 stochastic simulation runs and the fluid approximation-based constructions. The results from the iterative construction closely match the means of the stochastic simulation.

Figure 5.3: Fluid approximation directly coupled with the probability density for mode-switching.

population level of interest N. In the following calculations we set $N = 100$. The above system can then be solved using standard ODE solvers for initial conditions

$$\mathbf{x}(0) = (0, 1.0, 0, 0) \qquad \pi_t(0) = 1 \qquad \pi_t(1) = 0$$

and parameters $r_m = 1, r_s = 0.2$. Figure 5.3 gives a visual comparison of the resulting solution with the mean from 5000 runs of the stochastic simulation.

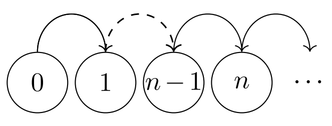


Figure 5.4: Pure birth process.

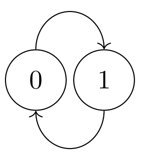


Figure 5.5: Stochastic switch.

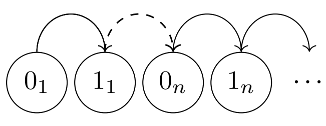


Figure 5.6: Pure birth process from stochastic switch.

5.3.2 Iterative method

The method in the previous section of directly coupling the equations for approximate probability density for the marginal process $\hat{Z}$ arising from the heuristic approximations of the filtering equation gives a reasonably good estimate for the mean of the marginal population process in the case of the running example. However, from Figure 5.3a we can see that even for this simple example the method does not accurately capture the mode-switching dynamics. In this section we present a slightly modified method based on the hybrid fluid approximation that in the case of a certain class of models allow us to more accurately capture the mean dynamics. In particular, we consider mode-switching processes which do not branch.

Non-branching mode-switching processes

We consider non-branching mode-switching processes $\hat{Z}$, where from each state there is at most one transition out. This will allow us to set up an iterative construction presented in this section. Two examples of such processes are pure birth (or death) processes depicted in Figure 5.4 and a stochastic switch given in Figure 5.5. As long as there is no branching the mode-switching processes featuring loops, like the stochastic switch given in Figure 5.5, can be treated equivalently to pure birth processes. In particular, we can unroll the loops by considering the process describing how many times a given state has been visited. Supposing the stochastic switch is initially in state 0 we would then consider the process depicted in Figure 5.6

The key characteristic of such non-branching processes is that we can characterise their behaviour in terms of a sequence of first hitting time problems. To see that, let us give a standard definition of first hitting times.

Definition 12. *Let φ be a continuously differentiable function on the space $\mathbb{R}^{K+1}$ (state space of the process $(\mathbf{X}, Z)$) with $\varphi(\mathbf{X}, Z) > 0$. Let h denote the first hitting time given by*

$$h = \inf\{t \mid \varphi(\mathbf{X}(t), Z(t)) \leq 0\}$$

With that in mind, if h_n denotes the first hitting time corresponding to reaching the state n of the pure birth process in Figure 5.4 then the probability of the process Z being in state n at time t can be given by

$$p(z=n;t)=p(h_n \leq t, h_{n+1} > t)$$

This corresponds to the joint probability that the state n has been reached before time t but the state $n+1$ has not. Similarly, conditioned on $h_{n+1} > t$ we have for the pure birth structure that

$$p(z=n;t \mid h_{n+1} > t) = p(h_n \leq t)$$

In the next section we are going to use these observations to iteratively construct the mean dynamics from the hybrid fluid approximation $(\hat{\mathbf{x}}, \hat{Z})$ under the approximate filtering distribution that discards the history of the population process for the calculation of the future behaviour.

Construction of mean dynamics

We propose an iterative method for constructing the marginal dynamics of $\hat{\mathbf{x}}$ in a finite time interval $[0,T]$. For a non-branching mode-switching process $\hat{Z}$ let us consider a sequence of first hitting times $h_1, h_2, \ldots, h_n, \ldots$ corresponding to mode-switching times into state $1,2,\ldots,n,\ldots$ of $\hat{Z}$. As before we are going to leverage the approximate dynamics given by

$$\frac{d}{dt}\pi_t(z') \approx \sum_z \pi_t(z) q^{\mathbf{x}}_{\hat{Z}}(z,z') \tag{5.10}$$

$$\frac{d}{dt}p(\mathbf{x};t) \approx \mathbb{E}_{\hat{Z}}\Big[-\sum_i \partial_i \mathbf{F}^{\hat{Z}}_i(\mathbf{x})\Big] \tag{5.11}$$

Recall that the sum inside the expectation operator is given over the components of the limit drift vector $\mathbf{F}^{\hat{Z}}$ given the state of $\hat{Z}$. As the structure of the mode-switching process is assumed to be non-branching we can conclude that

$$\frac{d}{dt}\hat{\mathbf{x}}(t) \approx \sum_j p(h_j < t, h_{j+1} \geq t)\mathbf{F}^{\hat{Z}}_j(\hat{\mathbf{x}}(t)) \tag{5.12}$$

Note that the sum above is given over the states of $\hat{Z}$. The iterative construction we propose relies on conditioning the behaviour of the population on a limited set of behaviours of $\hat{Z}$ up to some time t. In that case, we let $\hat{Z}^i_{t-}$ denote the mode-switching process that takes values up to state i within the time interval $[0,t]$ and consider the probability distribution

$$p(z';t \mid \hat{Z}^i_{t-})$$

We can then take the derivative with respect to t and consider the time evolution of the above distribution given that the mode-switching process is contained within the singleton set $\{0\}$ defined by a non-random initial condition.

$$\frac{d}{dt}p(z';t \mid \hat{Z}^0_{t-}) = \frac{d}{dt}\left[\int_{\mathbf{x}_{t-}} p(z';t \mid \mathbf{x}_{t-}, \hat{Z}^0_{t-};t)p(\mathbf{x}_{t-} \mid \hat{Z}^0_{t-})d\mathbf{x}_{t-}\right]$$
$$= \int_{\mathbf{x}_{t-}} \left[p(\mathbf{x}_{t-} \mid \hat{Z}^0_{t-})\frac{d}{dt}p(z';t \mid \mathbf{x}_{t-}, \hat{Z}^0_{t-};t) + p(z';t \mid \mathbf{x}_{t-}, \hat{Z}^0_{t-};t)\frac{d}{dt}p(\mathbf{x}_{t-} \mid \hat{Z}^0_{t-})\right] d\mathbf{x}_{t-}$$

The integral above is taken over all possible trajectories of $\hat{\mathbf{x}}$ up to time t. As we have assumed that the mode-switching process does not leave the singleton set $\{0\}$ the term $p(z';t \mid \mathbf{x}, \hat{Z}^0_{t-};t)$ is only non-zero if $z' = 0$. Furthermore, the term $p(z';t \mid \mathbf{x}_{t-}, \hat{Z}^0_{t-};t)$ corresponds to our heuristic simplification of the filtering equation which discards the history of the population process and makes the additional assumption that the mode-switching process $\hat{Z}$ stays in state 0, denoted $\pi_t(z' \mid \hat{Z}^0_{t-})$.

$$\frac{d}{dt}p(z'=1;t \mid \hat{Z}^0_{t-}) = \int_{\mathbf{x}_{t-}} p(\mathbf{x}_{t-} \mid \hat{Z}^0_{t-})\sum_z \pi_t(z' \mid \hat{Z}^0_{t-})q^{\mathbf{x}}_{\hat{Z}}(z,1)d\mathbf{x}_{t-}$$
$$= \mathbb{E}_{\hat{\mathbf{x}}\mid\hat{Z}^0_{t-}}\left[\pi_t(z' \mid \hat{Z}^0_{t-})q^{\mathbf{x}}_{\hat{Z}}(0,1)\right] = \mathbb{E}_{\hat{\mathbf{x}}\mid\hat{Z}^0_{t-}}\left[q^{\mathbf{x}}_{\hat{Z}}(0,1)\right] \tag{5.13}$$

The above gives the initial step in our iterative construction as under the assumption that the switching process stays in state 0 there is only one possible trajectory of $\hat{\mathbf{x}}$. In order to illustrate it let us briefly consider again the running example.

Example 25. *If the population at location* $(1,1)$, *given the mode-switching process is in state* 0 *at time* t, *is denoted as* $x^0_{11}(t)$ *we get*

$$\mathbb{E}_{\hat{\mathbf{x}}\mid\hat{Z}^0_{t-}}\left[q^{\hat{\mathbf{x}}}_{\hat{Z}}(0,1)\right] = q^{\hat{x}^0_{11}(t)}_{\hat{Z}}(0,1)$$

Conditioning the population dynamics on $\hat{Z}^0_{t-}$ *we can easily find the population variable* $\hat{x}^0_{11}(t)$ *by solving the following system.*

$$\frac{d}{dt}\hat{\mathbf{x}}^0(t) = \mathbf{F}^0(\hat{\mathbf{x}}^0(t)) = \begin{pmatrix} r_m(-\hat{x}_{01}(t) + \frac{1}{2}\hat{x}_{00}(t)) \\ r_m(-\hat{x}_{00}(t) + \frac{1}{2}\hat{x}_{01}(t) + \frac{1}{2}\hat{x}_{10}(t)) \\ r_m(-\hat{x}_{10}(t) + \frac{1}{2}\hat{x}_{11}(t) + \frac{1}{2}\hat{x}_{00}(t)) \\ r_m(-\hat{x}_{11}(t) + \frac{1}{2}\hat{x}_{10}(t)) \end{pmatrix}$$

Let us consider the same non-random initial conditions and parametrisation as previously.

$$\hat{\mathbf{x}}(0) = \begin{pmatrix} 0 & 1 & 0 & 0 \end{pmatrix}^T \qquad \hat{Z}(0) = 0$$

Based on Equation 5.13 we can find the cumulative rate corresponding to the mode switch to state 1 — or equivalently cumulative rate out of state 0. This is given by

$$\Lambda_1(t) = \int_0^t \mathbb{E}_{\hat{\mathbf{x}}\mid\hat{Z}^0_{s-}}\left[q^{\mathbf{x}}_{\hat{Z}}(0,1)\right] ds$$

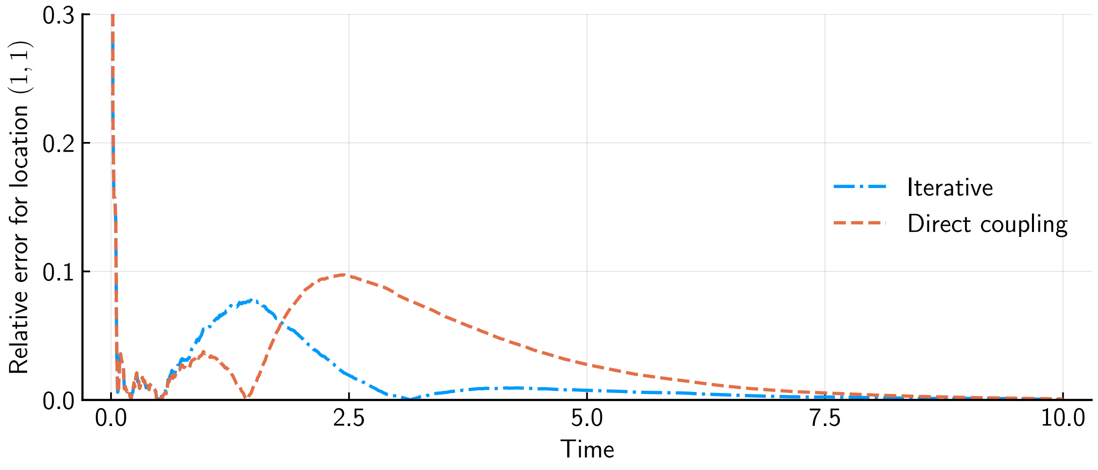


Figure 5.7: Relative error between the empirical mean for the location $(1,1)$ from 5000 stochastic simulation runs and the iterative and direct coupling approximations constructed from the hybrid fluid approximation of the running example.

From that the cumulative distribution function corresponding to the first hitting time of state 1 of $\hat{Z}$ is given by

$$p(h_1 \leq t) = 1 - e^{-\Lambda_1(t)}$$

This is enough to construct an approximation of the marginal behaviour of the population process $\hat{\mathbf{x}}$ conditioned on $\hat{Z}^1_{t^-}$. That is, for the system that does not leave the mode-switching states $\{0,1\}$ we get

$$\begin{aligned}\frac{d}{dt}\hat{\mathbf{x}}(t) &= p(z=0;t)\mathbf{F}^0(\hat{\mathbf{x}}(t)) + p(z=1;t)\mathbf{F}^1(\hat{\mathbf{x}}(t)) \\ &= p(h_1 > t)\mathbf{F}^0(\hat{\mathbf{x}}(t)) + p(h_1 \leq t)\mathbf{F}^1(\hat{\mathbf{x}}(t))\end{aligned}$$

The general construction up to the k-th mode-switch for populations where the mode-switching process has the pure birth structure can then be given by

$$\frac{d}{dt}\hat{\mathbf{x}}(t) = \sum_{i=1}^{k} p(h_i \leq t)\mathbf{F}^i(\hat{\mathbf{x}}(t))$$

where the distribution $p(h_k \leq t)$ is found by constructing the mode-switching process up to the $(k-1)$-th mode giving rise to the iterative construction.

$$\begin{aligned}\Lambda_k(t) &= \int_0^t \mathbb{E}_{\hat{\mathbf{x}}|\hat{Z}^{k-1}_{s^-}}\left[q^{\mathbf{x}}_{\hat{Z}}(k-1,k)\right] ds \\ p(h_k \leq t) &= 1 - e^{-p(h_{k-1}\leq t)\Lambda_k(t)}\end{aligned} \tag{5.14}$$

Example 26. *Based on the discussion in this section and the trajectories from Example 25 we can calculate the distribution $p(h_1 \leq t)$ and solve the system*

$$\frac{d}{dt}\hat{\mathbf{x}}(t) = p(h_1 > t)\mathbf{F}^0(\hat{\mathbf{x}}(t)) + p(h_1 \leq t)\mathbf{F}^1(\hat{\mathbf{x}}(t))$$

where we use $\mathbf{F}^0$ to denote the drift giving the random walk over the grid structure in the running example and $\mathbf{F}^1$ corresponds to the drift towards the target. In Figure 5.7 we give a comparison of the resulting probability distributions for $\hat{Z}(t) = 1$ and the relative errors for the population measure of location $(1,1)$ between the stochastic simulation and the fluid approximation based iterative construction. The distribution for $\hat{Z}(t) = 1$ and relative errors from the direct coupling method in the previous section are given for comparison. Note that for this simple example this approach gives a more accurate representation of the distribution $\hat{Z}(t) = 1$. In addition the relative error, if discarding the values near $t = 0.0$ due to numerical precision when population levels are low, is improved. In particular, the maximum relative error is diminished and the error reaches values near zero faster. Both of the relative error trajectories feature points where the error becomes zero. These points occur when the constructed solutions cross the empirical estimate. In addition, note that the relative error of approximations approaches 0 as time increases. This is due to the existance of the absorbing state of the system where all robots have reached the target location. Similar behaviour can be expected for systems with no absorbing state but a unique attracting equilibrium state.

5.4 Linear noise approximation

The direct coupling and iterative constructions given for the fluid approximation can be extended straightforwardly to the hybrid noise approximation $(\tilde{\mathbf{X}}, \tilde{Z})$ to a mode-switching population system $(\mathbf{X}, Z)$. In the same way as for the hybrid fluid approximation let us first consider the marginal process $\tilde{\mathbf{X}}$ at time t given a sample trajectory $\mathbf{x}_{t-}$. Recalling the forward equation for jump diffusion processes given in Section 2.2 we get

$$
\begin{aligned}
\frac{\partial}{\partial t} p(\mathbf{x}; t \mid \mathbf{x}_{s-}; s) &= \frac{\partial}{\partial t} \sum_{z} p(z; s \mid \mathbf{x}_{s-}; s) \sum_{z'} p(\mathbf{x}, z'; t \mid \mathbf{x}_s, z; s) \\
&= \mathbb{E}_{\tilde{Z} \mid \mathbf{x}_{s-}} \Bigg[\sum_{z'} - \sum_{i} \partial_i \mathbf{F}_i^{z'}(\mathbf{x}) p(\mathbf{x}, z'; t \mid \mathbf{x}_s, z; s) \\
&\quad + \sum_{i} \sum_{j} \partial_i \partial_j \mathbf{G}_{ij}^{N}(\mathbf{x}) p(\mathbf{x}, z'; t \mid \mathbf{x}_s, z; s) + \sum_{k} p(\mathbf{x}, k; t \mid \mathbf{x}_s, z; s) q_{\tilde{Z}}^{\mathbf{x}}(k, z') \Bigg]
\end{aligned}
$$

where $\mathbf{G}^N$ is the diffusion matrix scaled by the term $N^{-\frac{1}{2}}$ resulting from Proposition 4. As before, due to the discrete jumps of $\tilde{Z}$ not changing the state of population variables, we get

$$\frac{d}{dt}\pi_t(z';t) \approx \sum_z \pi_t(z) q_{\tilde{Z}}^{\mathbf{x}}(z,z') \tag{5.15}$$

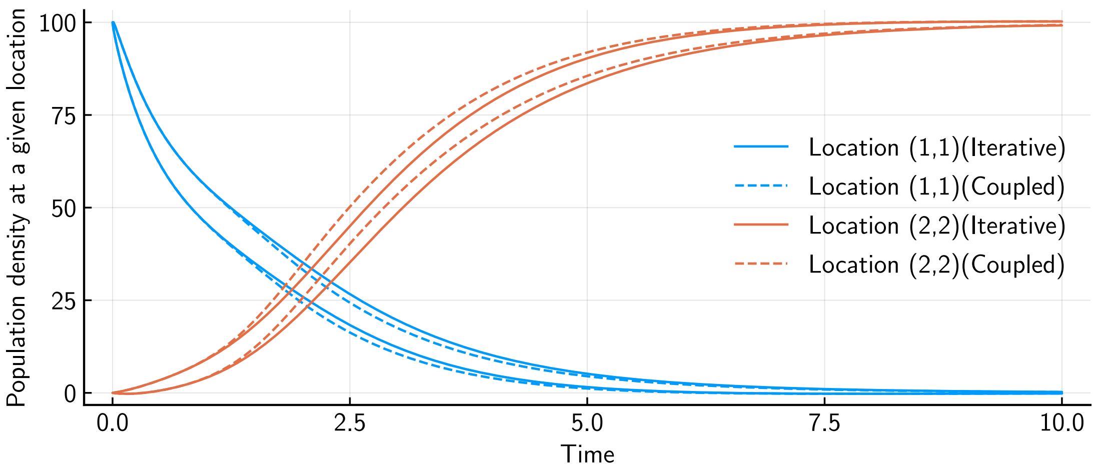
$$\frac{d}{dt}p(\mathbf{x};t) \approx \mathbb{E}_{\tilde{Z}}\left[-\sum_i \partial_i \mathbf{F}_i^{\tilde{Z}}(\mathbf{x}) + \sum_i \sum_j \partial_i \partial_j \mathbf{G}_{ij}^{N}(\mathbf{x})) \right] \tag{5.16}$$

Similarly to the hybrid fluid case, this results in a process that essentially amounts to a diffusion process with time-dependent coefficients. The time-dependence results from the evolution of the mode-switching process. At each time t the drift and diffusion coefficients are given in terms of expectation with respect to the state of the mode-switching process at time t. As is the case with the time-inhomogeneous CTMC approximation to the marginal dynamics considered in Section 5.2 we expect the approximation to lose accuracy around the mode-switching times, giving us a coarse lower bound on the variance of the model. However, the dynamics of the linear noise based approximation can be given in terms of a system of ODEs which can be numerically solved. In addition, the same direct coupling and iterative constructions as before can be easily applied by coupling the filtering distribution and the marginal dynamics of $\tilde{\mathbf{X}}$ through the approximate mean of $\tilde{\mathbf{X}}$.

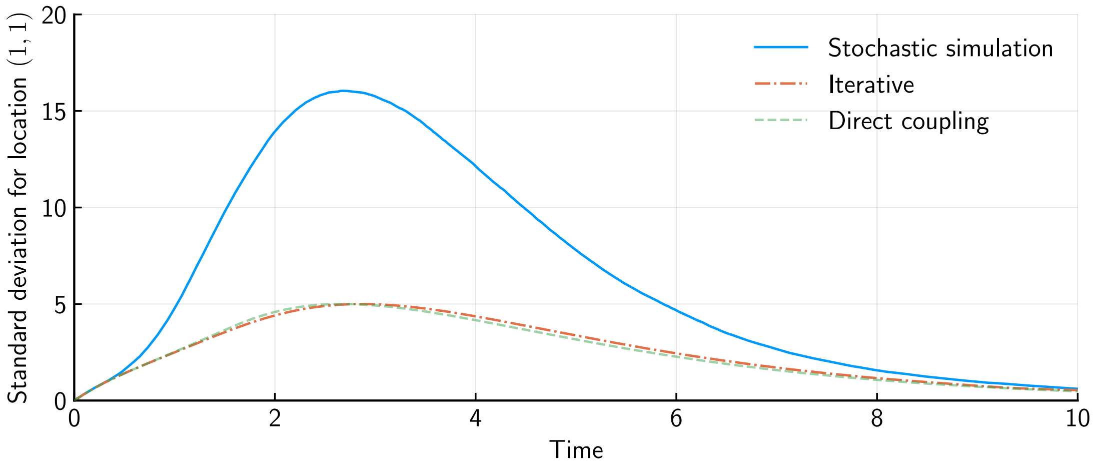
Example 27. *The mean dynamics of the hybrid linear noise-based approximations remain the same as constructed in the previous section for the hybrid fluid case. Figure 5.8 presents the comparison of the direct coupling-based and iterative approximations with respect to the results of* 5000 *independent stochastic simulation runs. In particular, Figure 5.8b presents the variance corresponding to the population densities for location* $(1,1)$ *in the time-interval* $[0.0, 10.0]$*. Expectedly, around the mode-switching time the linear noise based approximation underestimates the standard deviation around the mean. This results from our heuristic simplification of the filtering equation in Section 5.3. The differences in the mean behaviours as constructed by the two methods produce the very small differences in the variance estimates.*

5.5 Moment closure approximation

In Chapter 4 we derived evolution equations for conditional moments according to [65]. While the method has been demonstrated in [65, 81] to be a good approximation to systems exhibiting bistable or multi-modal behaviour is does have problems associated with it. Notably, finding a consistent initialisation of the resulting differential algebraic system is in most cases difficult. In this section we are going to leverage the filtering heuristic and construction from Section 5.2. In particular, recall that we constructed a time-inhomogeneous CTMC approximation to the marginal population process $\mathbf{X}$ by



(a) Comparison of constructed trajectories. Solid and dashed lines correspond to one standard deviation around the mean for the iterative and direct coupling based constructions respectively. The difference between the two sets of results is a consequence of the methods giving a different estimate for the mean behaviour.



(b) Comparison of standard deviation corresponding to the population measure at location $(1,1)$. The LNA-based approximations and the empirical normal distribution from 5000 stochastic simulation runs are shown. Both iterative and direct coupling methods give highly similar estimates for the variance of the process.

Figure 5.8: Approximation of the running example based on hybrid linear noise.

discarding the dependence on the history and assuming that all memory effects are expressed through the state of the marginal process $\hat{Z}$ at time t.

$$\frac{d}{dt}p(\mathbf{x};t \mid \mathbf{x}_{t^-}) \approx \frac{d}{dt}p(\mathbf{x};t \mid \mathbf{x};t) = \mathbb{E}_Z\left[q^Z_{\mathbf{X}}(\mathbf{x},\hat{\mathbf{x}})\right] \tag{5.17}$$

This allows us to consider the standard moment closure approximation, described in Section 2.4.3, for the mode-switching population model as considering time-dependent rates changes little in the derivations. From there we can again use the coupling and iterative methods which were already considered for hybrid fluid and linear noise based approximations.

Example 28. *The rates of population transitions of the running example are all linearly dependent on a single population variable. This means that the moments of each order are going to only depend on the moments up to the said order and not on the higher order moments. In particular, the first two moments, expectation and variance, of the population variables are going to coincide with the appropriately scaled version of the linear noise approximation.*

5.6 Results

In this section we conduct an empirical analysis of presented direct coupling and iterative constructions of mean dynamics. For evaluation the approximations are compared against the empirical mean from 5000 stochastic simulations trajectories. This large number of trajectories was chosen to get a reliable estimate for the mean and standard deviation statistics. To demonstrate the extensibility and scalability of the presented modelling and analysis ideas, we first introduce a larger model inspired by maze navigation.

Example 29. *This example considers a 4-by-4 grid with connections between nodes constructed as shown in Figure 5.9. The mode transitions happen, as before, through instantaneous broadcast communication when a robot navigating the structure reaches* $(2,2)$ *or* $(3,3)$ *and detects them as targets with rate* r_s*. We assume that the robot can give the rest of the swarm enough information to reach its location. This splits the dynamics of the collective into three modes. First we have a random walk on the graph structure. Secondly we have a directed walk towards* $(2,2)$ *with a random exploration of the states* $(1,3),(2,2),(2,3),(3,2)$ *and* $(3,3)$*. Finally, there is the mode corresponding to a directed walk towards location* $(3,3)$*. Analogously to the running example the entire swarm is assumed to start at location* $(0,0)$*.*

The maze navigation example presented above has an absorbing state corresponding to all the robots being in location $(3,3)$. For a contrast, the following provides an example where there is no such absorbing state.

Example 30. *The second example we are going to consider extends the running example, set up in Example 22, with an additional mode switch. Recall that the initial dynamic mode in this example is given by a random walk on the graph structure over a 2-by-2 grid in Figure 5.1. We are going to consider the case where, from the second mode describing a directed walk towards* $(1,1)$*, the population dynamics can revert back to the initial mode. After reverting to the initial random walk dynamics the dynamics*

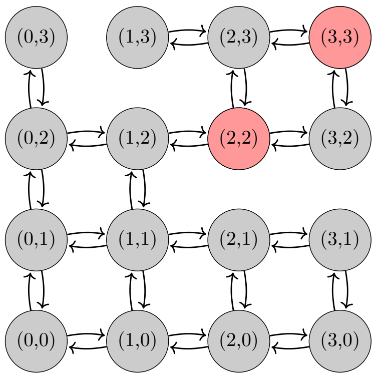


Figure 5.9: Spatial structure of the maze example.

will no longer change. We are going to say that the change from the second mode, defined by the directed walk towards $(1,1)$*, to the third mode, defined by the random walk, is caused again by a broadcast action sent out by a robot at location* $(1,1)$ *happening at rate* $0.1r_s$*.*

Let us consider the direct coupling and iterative methods for the two examples at 5 different population levels $(100, 200, 300, 500, 1000)$. In order to study the behaviour of the approximations under different parametrisations we consider, for each population level, 50 parameter values for r_m and r_s that are randomly sampled from the interval $(0, 1)$. Thus we consider 50 different models operating over the same graph structure, and for each of these we consider five different population levels, giving 250 models in total. The resulting approximations for the maze example are computed for the finite time intervals $[0, 100.0]$. For the 2-by-2 grid example with three modes we use the time interval $[0, 100.0]$ for population levels $(100, 200, 300)$ and the time interval $[0, 10.0]$ for population levels $(500, 1000)$. The solutions were saved at 1000 equally space sample points. The shorter time interval for larger populations was chosen to avoid missing the mode-switching behaviour due to discrete sampling of the trajectories. The results from the approximations then compared to the population measures acquired from 5000 stochastic simulation runs. For the implementation of the moment closure method we are, for the purpose of this chapter, going to set all third and higher order moments to zero [125]. For the two examples considered in this section we can scale the results of the linear noise based approximation up appropriately so that it coincides with the moment closure based approximation. This is due to the rates of all transitions in these models being linearly dependent on a single population variable. These are analogous to unimolecular reactions in biochemical modelling literature. For that reason we are going to consider them together in this section.

Table 5.1 gives an overview of the mean computation times for Example 29 based on the maze navigation example. The experiments were implemented using DifferentialEquations.jl package [109] for the Julia programming language and run in batches corresponding to the population size. In each case the timings of the calculations for first parametrisations were not considered as they include the just-in-time compilation overhead of Julia. The implementations of the examples considered are made available at [2]. The size of the ODE system that was solved is reported as the bottom row of the table. For example, in the case of the approximation resulting from the hybrid fluid approximation introduced in Section 4.3.1 we can see that an ODE for each location plus three filtering equations are needed (one for each state of the mode-switching process). In the case of the iterative method we need to consider two ODEs describing the hitting time distributions for the two mode switches. However, extra computation time results from having to repeat the numerical integration of the ODE systems for each iteration in the iterative constuction. Finally, note that although in the example considered in this section moment closure and linear noise result in the same set of equations our current implementation based on [6] relies on a large number of instances of symbolic differentiaton and substitution, using the SymEngine library [4], which for the maze example led to a relatively large upfront cost in terms of computation time. The current implementation for the three modes of the maze example takes about 45 minutes. This, however, is a one off cost and can hopefully be greatly optimised.

Table 5.1: Comparison of mean computation times for the maze example (seconds). Model dynamics in the finite time interval $[0, 100.0]$ are considered.

	Hybrid fluid		Linear noise/moments		Sampled traj. (5000)
Pop. size	Dir. cpl.	Iter.	Dir. cpl.	Iter.	
100	0.0010	0.15	0.0025	1.30	2.01
200	0.0011	0.14	0.0031	1.30	7.12
300	0.0013	0.15	0.0027	1.37	10.95
500	0.0011	0.15	0.0033	1.44	16.56
1000	0.0011	0.15	0.0033	1.44	27.22
# ODEs	19	18	155	154	

Hybrid fluid approximation

Table 5.2 presents an error analysis for the two examples considered in this section. For the maze example we consider the approximations of the expected population at location (3,3) while for the 2-by-2 example we consider the location (1,1). For each experiment corresponding to a parametrisation of r_m, r_s and choice of population level N we calculate the maximum absolute errors with respect to the empirical mean derived from corresponding stochastic simulation runs. In order to make the absolute errors comparable we have normalised them based on the population size to reflect the error in terms of the proportion of the considered population. For each of the population levels the table displays the mean and standard deviation of these maximum errors as well as the largest of the calculated maximum errors.

Although the iterative method creates some computational overhead in the implementation, we can see from Table 5.2 that in case of the larger maze example the iterative construction offers better accuracy than directly coupling the approximate filtering equations with the fluid drifts. However, in the case of the smaller 2-by-2 example we see that under centrain parametrisations the iterative construction does not result in an improved approximation. In particular, Table 5.2 with $N = 300$ shows the maximum errors 3.3% and 3.1% for the iterative construction and direct coupling respectively. Finally, Figures 5.11 and 5.10 give the corresponding error surfaces for the 4-by-4 maze and the three mode 2-by-2 example respectively at population levels 100 and 200. In particular, maximum errors for each of the parametrisations of r_m and r_s are plotted. These indicate that the iterative construction deals better with the parametrisations where the rate of movement parameter r_m is much lower than the parameter r_s defining the rate of broadcasting.

Table 5.2: Comparison of mean maximum errors resulting from hybrid fluid based approximations. As explained, the moment closure approximation values will, in this case, coincide with the linear noise-based approximation. For the maze example the errors are calculated for location $(3, 3)$. For the 2-by-2 example with three modes the errors are calculated for location $(1, 1)$.

		4-by-4 maze		2-by-2 three modes	
		mean (sd) max.	max.	mean (sd) max.	max.
		$t \in [0, 100.0]$		$t \in [0, 100.0]$	
$N = 100$	Direct coupling	14.5 (7.2)%	39.4%	2.4 (2.9)%	13.5%
	Iterative	7.3 (3.2)%	17.4%	1.7 (1.6)%	6.9%
$N = 200$	Direct coupling	9.9 (6.3)%	33.7%	1.5 (1.8)%	8.5%
	Iterative	4.3 (2.5)%	13.5%	1.2 (1.3)%	5.8%
$N = 300$	Direct coupling	7.8 (5.2)%	28.7%	1.1 (1.5)%	6.9%
	Iterative	3.2 (1.9)%	10.1%	1.0 (1.1)%	5.1%
		$t \in [0, 100.0]$		$t \in [0, 10.0]$	
$N = 500$	Direct coupling	5.9 (4.2)%	21.9%	0.8 (1.0)%	4.7%
	Iterative	2.2 (1.4)%	7.1%	0.7 (1.0)%	4.1%
$N = 1000$	Direct coupling	4.0 (3.0)%	15.7%	0.6 (0.7)%	3.3%
	Iterative	1.4 (1.0)%	4.9%	0.5 (0.8)%	3.4%

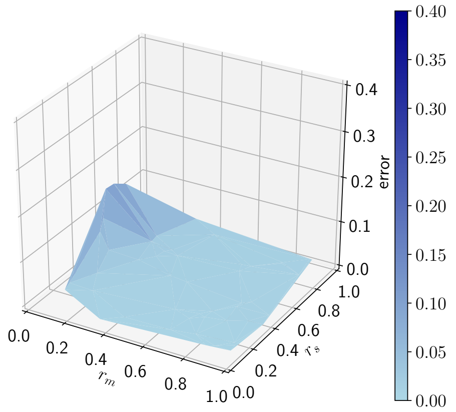
Table 5.3: Comparison of mean maximum errors for the population standard deviation resulting from hybrid linear noise based approximations. For the maze example the errors are calculated for location $(3,3)$. For the 2-by-2 example with three modes the errors are calculated for location $(1,1)$.

		4-by-4 maze		2-by-2 three modes	
		mean (sd) max.	max.	mean (sd) max.	max.
		$t \in [0, 100.0]$		$t \in [0, 100.0]$	
$N = 100$	Direct coupling	18.4 (7.9)%	41.5%	10.6 (7.9)%	35.0%
	Iterative	18.4 (7.9)%	41.1%	10.6 (7.9)%	35.0%
$N = 200$	Direct coupling	13.6 (7.5)%	37.8%	7.9 (6.9)%	30.7%
	Iterative	13.6 (7.5)%	37.8%	7.9 (6.9)%	30.7%
$N = 300$	Direct coupling	11.4 (7.0)%	34.8%	6.6 (6.1)%	27.4%
	Iterative	11.4 (7.0)%	34.8%	6.6 (6.1)%	27.4%
		$t \in [0, 100.0]$		$t \in [0, 10.0]$	
$N = 500$	Direct coupling	9.1 (6.2)%	30.6%	5.2 (5.2)%	23.3%
	Iterative	9.1 (6.2)%	30.6%	5.2 (5.2)%	23.3%
$N = 1000$	Direct coupling	6.6 (4.9)%	24.4%	3.7 (4.0)%	18.2%
	Iterative	6.6 (4.9)%	24.4%	3.7 (4.0)%	18.2%

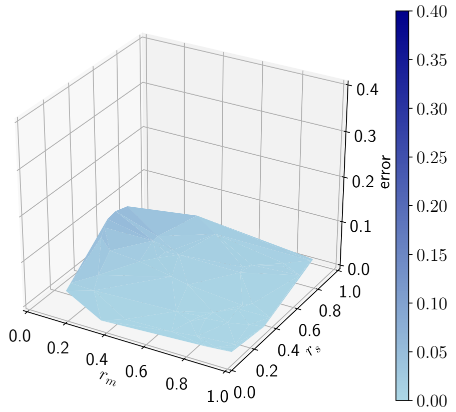
Linear noise and moment closure

For the linear noise and moment closure-based approximations we are interested in how much the approximations underestimate the variance of the population measures. In this work we did not consider higher order moments. Table 5.3 presents the error analysis for the standard deviation of the population measures resulting from the linear noise/moment based approximation. Analogously to the error analysis for the means we calculate the maximum absolute errors with respect to the empirical standard deviation derived from corresponding stochastic simulation runs. For comparison across population levels we normalise them based on the population size to reflect the error in terms of the proportion of the considered population. For each of the population levels the table displays the mean and standard deviation of these maximum errors as well as the largest maximum errors. Note that for these examples the constructed approximations can be used only as a rather coarse lower bound on the measures. Also note that there is no notable difference between the direct coupling and iterative construction when considering variance and standard deviation of the dynamics.

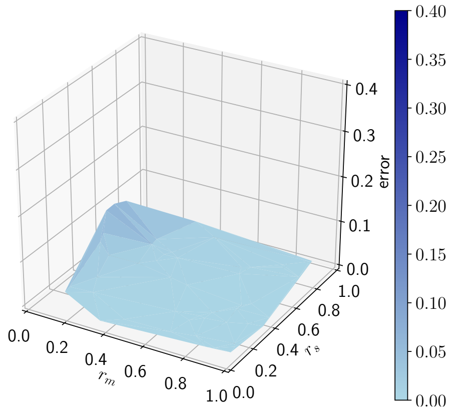
The Figures 5.13 and 5.15 give the error surfaces for the approximate standard deviations for the three mode 2-by-2 grid example and the maze example respectively. The shapes of these can be observed to match the corresponding ones for the means.



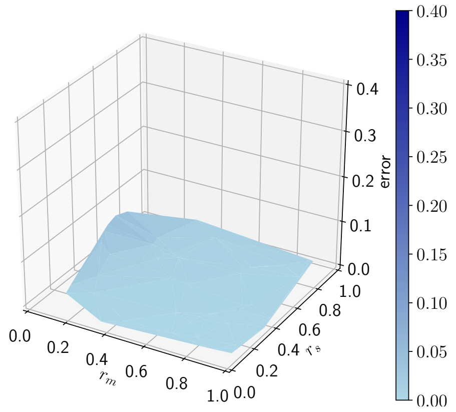
(a) Direct coupling – population 100



(b) Iterative – population 100



(c) Direct coupling – population 200



(d) Iterative – population 200

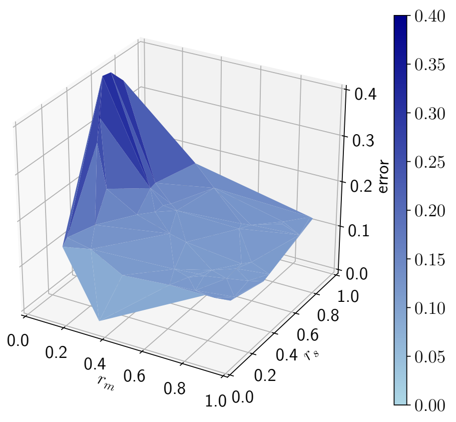
Figure 5.10: Error surfaces for the three mode 2-by-2 grid example. Displays the randomly sampled pairs of r_m and r_s against the corresponding maximum errors for the location $(1, 1)$.

Finally, Figures 5.12 and 5.14 give examples for the time-evolution of standard deviation corresponding to population measures of specified locations of the model.

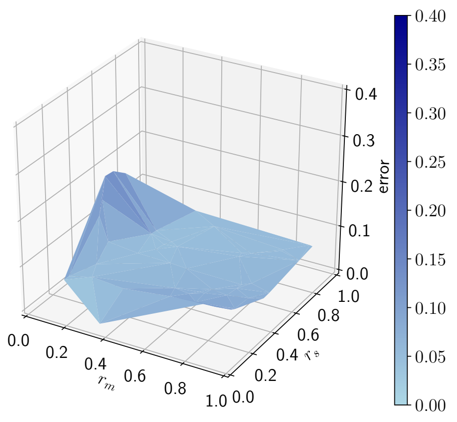
5.7 Conclusions

In this chapter, we presented an additional approximation to the hybrid fluid, linear noise and moment closure-based approximations from Chapter 4. The approximation was derived by considering the history-dependent marginal process corresponding to dynamics of the population and discarding the effects of the history of such a marginal process on its own future behaviour as a simple heuristic approximation. We have made use of this approximation by coupling the resulting Kolmogorov equations for the mode-switching process with the fluid, linear noise and moment closure approximation.

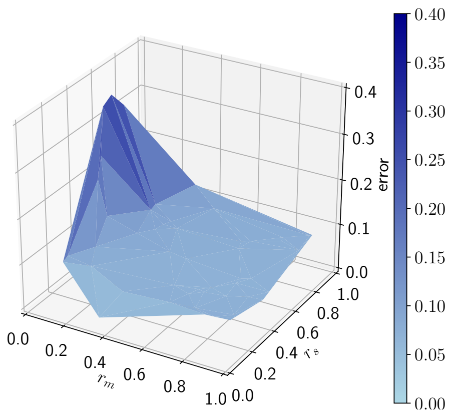
In addition we proposed an iterative construction that can be used to construct the approximation from one mode switch to another and presented a comparison of the



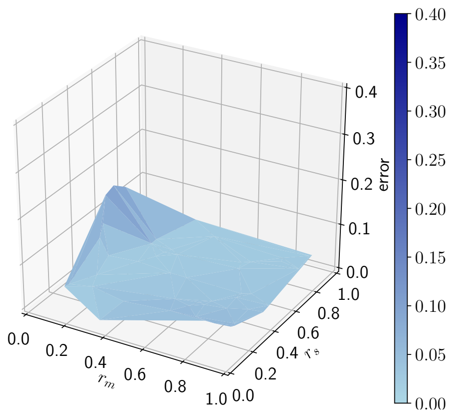
(a) Direct coupling – population 100



(b) Iterative – population 100



(c) Direct coupling – population 200

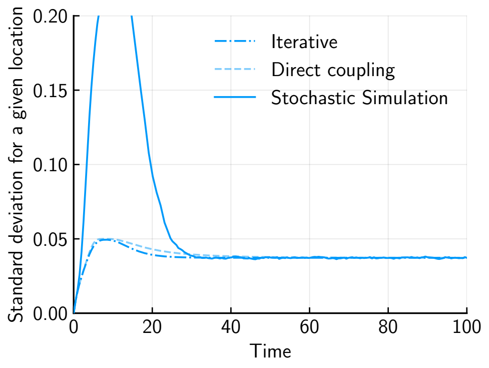


(d) Iterative – population 200

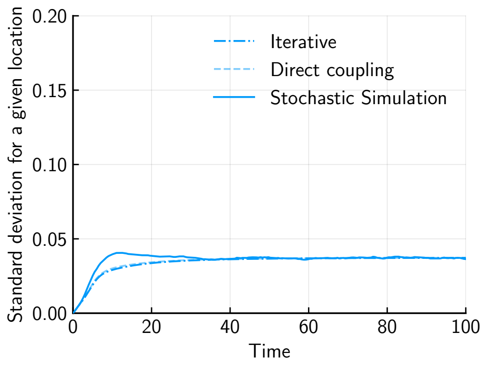
Figure 5.11: Error surfaces for the maze example. Displays the randomly sampled pairs of r_m and $_s$ against the corresponding maximum errors for location $(3,3)$.

computations times and an empirical study of the accuracy of the constructions for two simple examples. Both the direct coupling and iterative constructions are orders of magnitude faster than the direct simulation of the pCTMC. In addition we saw that although our current implementation of the iterative method is slower than the direct coupling, for most of the tested parametrisations of the two examples it gives a better approximation of the mean dynamics. An interesting further question would be under which general conditions does this happen and why. Additionally, it would be interesting to consider higher order approximations for the effects of observation history on the marginal process in order to extract a more refined approximation to the marginal.

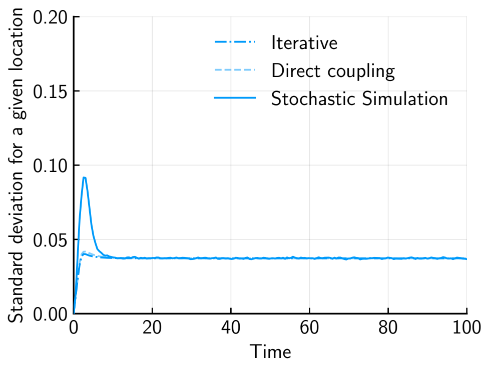
It is important to note that an efficient implementation of the iterative method when there are a large number of mode switches in the time-interval of interest becomes increasingly difficult. Thus in the current state the method is feasible when such switches in the dynamics of the population are rare. Similarly, as pointed out



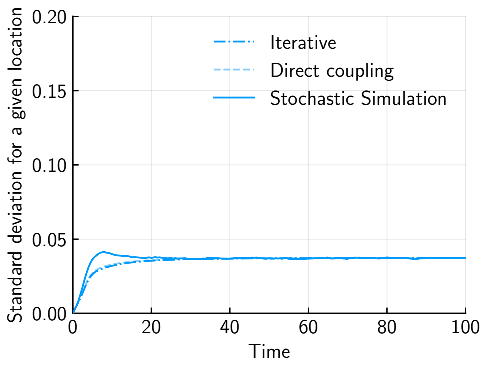
(a) Population 100. $r_m = 0.041$ and $r_s = 0.537$.



(b) Population 100. $r_m = 0.667, r_s = 0.102$.



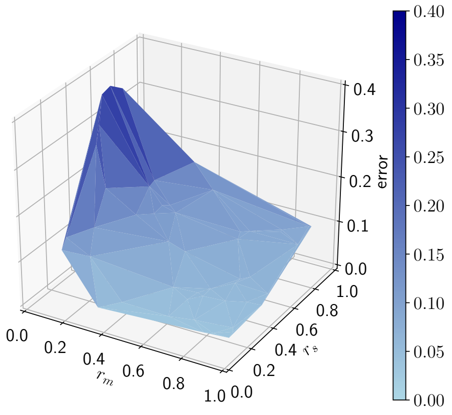
(c) Population 100. $r_m = 0.665, r_s = 0.744$.



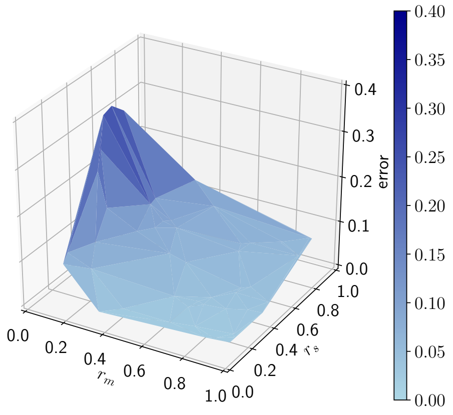
(d) Population 100. $r_m = 0.954, r_s = 0.153$.

Figure 5.12: Standard deviations of the population densities at location (1, 1) the three mode 2-by-2 grid example. As seen previously in the running example, the standard deviations resulting from the iterative and direct coupling methods match closely with each other.

and highlighted by the error surfaces, time-scale separation plays an important role in the experiments. Namely, even if only a few mode switches are considered we need the mode switches to happen at a slower time-scale than the population transitions. Finally, in this chapter we only considered transitions whose rates are linear in the population variables which lead to the situation where, up to appropriate scaling, linear noise and moment closure-based approximation resulted in the same system of ODEs. This last aspect is going to be considered in the next chapter featuring a small case study which links together Chapters 3, 4 and 5.

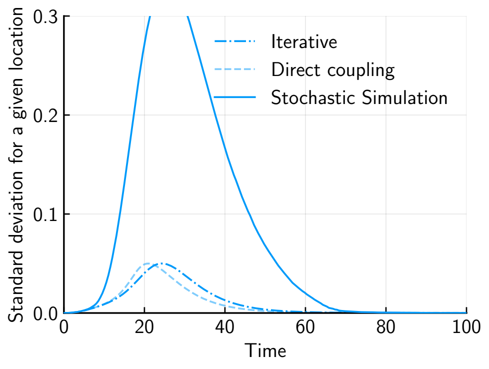


(a) Population 100.

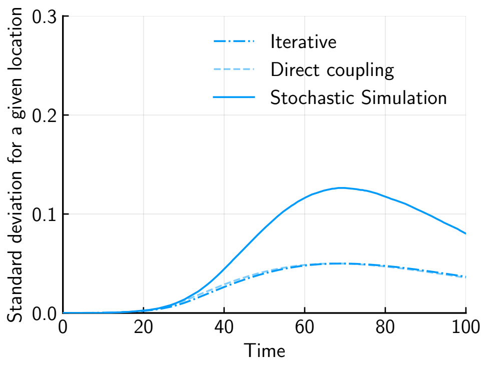


(b) Population 200.

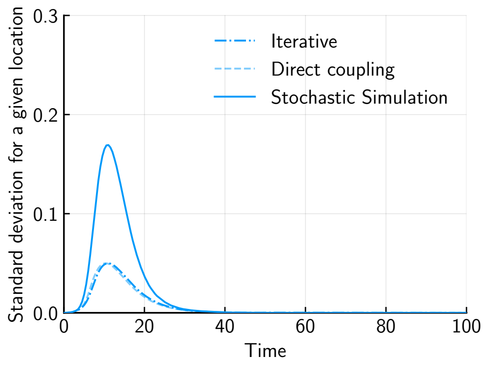
Figure 5.13: Error surfaces for the standard deviation of the three mode 2-by-2 grid example. Displays the randomly sampled pairs of r_m and r_s against the corresponding maximum errors for the location $(1, 1)$.



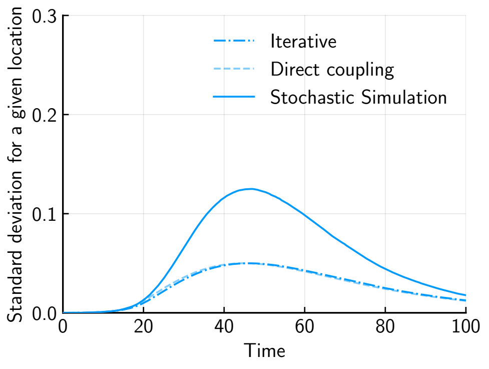
(a) Population 100. $r_m = 0.041$ and $r_s = 0.537$.



(b) Population 100. $r_m = 0.667, r_s = 0.102$.

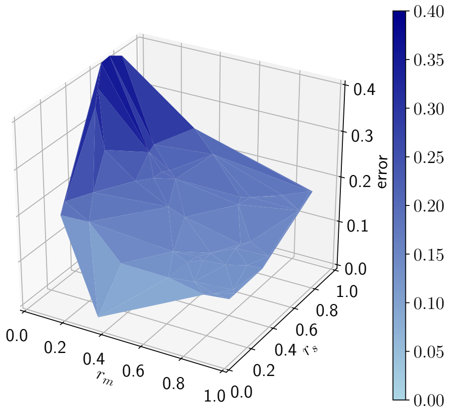


(c) Population 100. $r_m = 0.665, r_s = 0.644$.

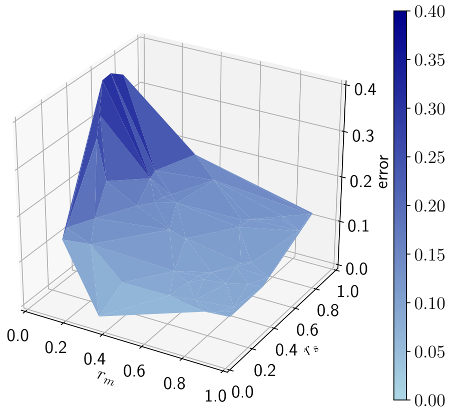


(d) Population 100. $r_m = 0.954, r_s = 0.153$.

Figure 5.14: Standard deviations of the population densities at location $(3, 3)$ of the maze example. As seen previously in the running example, the standard deviations resulting from the iterative and direct coupling methods match closely with each other.



(a) Population 100.



(b) Population 200.

Figure 5.15: Error surfaces for the standard deviation of the maze example. Displays the randomly sampled pairs of r_m and r_s against the corresponding maximum errors for the location $(3, 3)$.

Chapter 6

Synthesis example

In the previous chapters we presented the main technical contributions of this thesis. In particular, let us recall that Chapter 3 equipped the CARMA process algebra with CTMDP semantics for formal high-level specification of parameter and policy synthesis problems. Chapters 4 and 5 dealt with theoretical approximations and computational treatment of the arising population models. The aim of the following is to bring together the ideas from these chapters and present an example on policy synthesis. In particular, based on the running examples in the previous chapters, we construct a CARMA-C model so that the policy of the underlying decision process is given by a single store attribute. We then make use of the approximation in Chapters 4 and 5 to decide for which attribute values the model is expected to satisfy a given objective.

6.1 Carma-C model

The example considered in this chapter is a slightly modified version of the running example presented in Chapters 4 and 5. In particular, we again consider the basic foraging scenario. Let us start by formulating the example in the CARMA-C language which was presented in Chapter 3. Recall that the model considered mode-switching behaviour in the context of a simple robot swarm where the exploration phase was modelled by a random walk on a graph structure given in Figure 6.1.

When the robot detects the location $(1,1)$ as the target it broadcasts a message to the rest of the swarm. After such a message is sent out the dynamics of the swarm change to that of a directed walk towards the location $(1,1)$. In CARMA-C we can model the processes governing the behaviour of the individual robots as illustrated in Figure 6.2. This is essentially the same as considered in Chapter 3, however in this case we are not considering failure of robots. The rest of the model remains unchanged. For the ease of presentation we are going to recall the relevant parts of the model here.

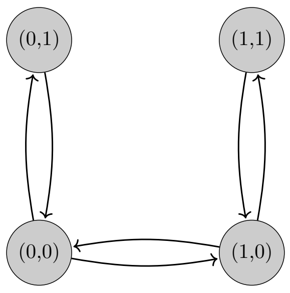


Figure 6.1: Spatial structure for the example.

That is, the broadcast actions *random** and *directed** describe a random walk and a directed walk towards $(1,1)$ respectively. Despite being defined as broadcast actions, neither of these actions have any effect on the other robots in the collective. This is achieved by setting the outgoing message to be empty. The guards π_r and π_d check whether the target location is known or not and make sure only one of the actions *random** and *directed** is enabled at a time.

The *sense** action models the mechanism for mode-switching. In particular, the broadcast output action models the robot detecting and sending the target location to the rest of the swarm. The corresponding broadcast input action models the robot's ability to receive such a message. The resulting CARMA-C expression for the robot process with the store updates omitted is given below.

$$\begin{aligned}
&\mathit{Explore} \stackrel{\mathsf{def}}{=} [\pi_r]\mathit{random}^*[\circ]\langle\circ\rangle + [\pi_d]\mathit{directed}^*[\circ]\langle\circ\rangle + [\pi_r]\mathit{sense}^*[\circ](M(x,y))\\
&\mathit{Listen} \stackrel{\mathsf{def}}{=} [\pi_r]\mathit{sense}^*[\circ]\langle M(\mathsf{loc})\rangle\\
&\mathit{Robot} \stackrel{\mathsf{def}}{=} \mathit{Explore} \parallel \mathit{Listen}
\end{aligned}$$

The local store of each of the robots holds attributes for location, denoted loc, target, denoted target and the robustness parameter, succp, taking values in the interval $[0,1]$. The store updates corresponding to each of the defined actions are given in Figure 6.3. The actions *random** and *directed** change loc attribute of the robot component according to functions R and D respectively. The function R corresponds to the next location being selected uniformly from the set of available next locations defined by the graph structure in Figure 6.1. Similarly, D corresponds to the next location taking the robot closer to the target with some probability p, specified by the robustness attribute succp, and to any of the other directly connected locations with probability $1-p$. This defines a distribution over the possible unresolved local stores the components can evolve to and models the robots being unreliable with respect to navigation. Recall that in order to be consistent with the semantics presented, the locations are defined as singleton sets containing only the current location of the robot. The functions R and D are applied element-wise to all elements in the set defining the value domain of the loc attribute as illustrated in Figure 6.3. The *sense** action updates the set of target locations. In the case of this model this can only be an empty set or a set consisting of $(1,1)$. Function M either returns the singleton set consisting of location $(1,1)$ or an empty set.

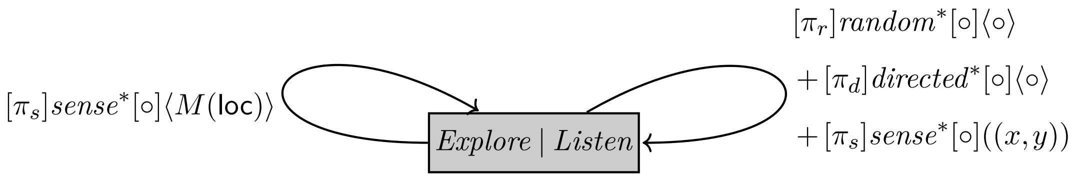

$[\pi_s]sense^*[\circ]\langle M(\mathsf{loc})\rangle$
$Explore \mid Listen$
$[\pi_r]random^*[\circ]\langle\circ\rangle$
$+[\pi_d]directed^*[\circ]\langle\circ\rangle$
$+[\pi_s]sense^*[\circ]((x,y))$

Figure 6.2: Behaviour of individual *Robot* components.

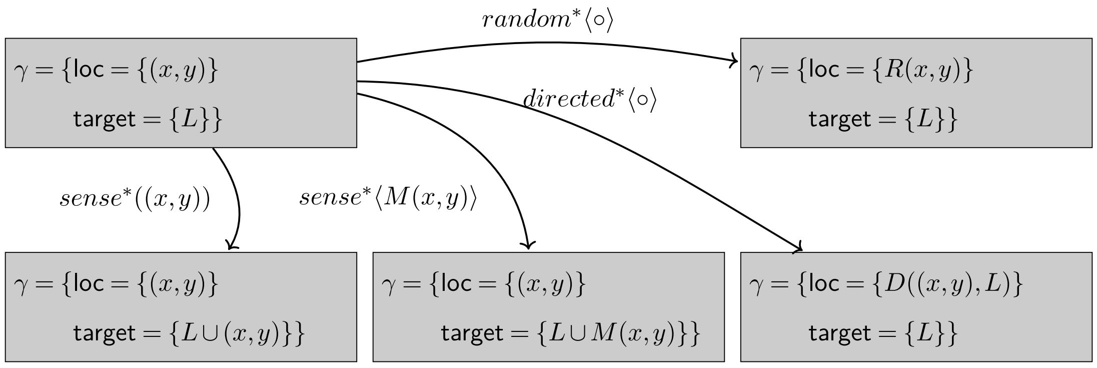

$random^*\langle\circ\rangle$
$directed^*\langle\circ\rangle$
$sense^*((x,y))$
$sense^*\langle M(x,y)\rangle$
$\gamma=\{\mathsf{loc}=\{(x,y)\}$ $\mathsf{target}=\{L\}\}$
$\gamma=\{\mathsf{loc}=\{R(x,y)\}$ $\mathsf{target}=\{L\}\}$
$\gamma=\{\mathsf{loc}=\{(x,y)\}$ $\mathsf{target}=\{L\cup(x,y)\}\}$
$\gamma=\{\mathsf{loc}=\{(x,y)\}$ $\mathsf{target}=\{L\cup M(x,y)\}\}$
$\gamma=\{\mathsf{loc}=\{D((x,y),L)\}$ $\mathsf{target}=\{L\}\}$

Figure 6.3: Local component store changes induced by actions.

Finally we are going to address the global store. In particular, we define two store variables — one corresponding to the rate at which the actions *random** and *directed** happen, denoted mover. We specify the value domain for this variable to be $[0,\infty)$. Similarly, we specify the value domain for the store attribute senser, corresponding to the rate of the action *sense**, to be $[0,\infty)$. We define global updates in a way that keeps the defined value domains unchanged throughout the evolution. The complete CARMA-C model is given in Appendix D.

6.2 pCTMDP model

In this section we are going to describe the resulting pCTMDP model. This model results from the application of semantics described in Chapter 3. The application of the semantics was demonstrated in Chapter 3 and is not replicated here.

Let us denote the state-space of the pCTMDP by the counting variables

$$\mathbf{X}=(X_{01},X_{00},X_{10},X_{11},X'_{01},X'_{00},X'_{10},X'_{11})$$

The robots are distinguished through their state — in this case, the only part of the robot component that changes is the location and whether the target location is known or not.

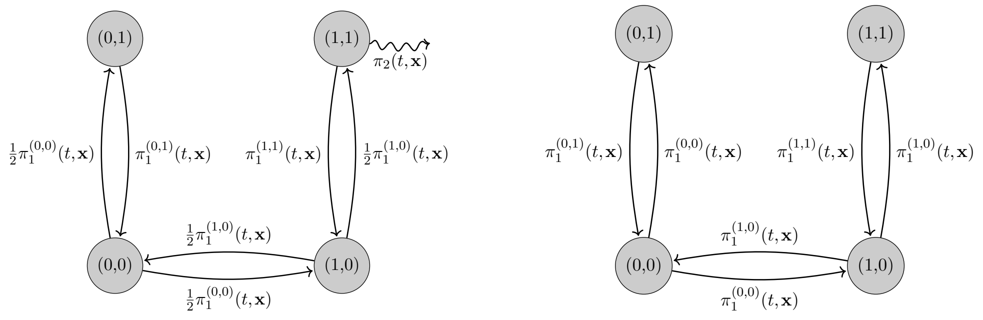
The rates with which the actions are performed are linked to the global store variables mover and senser that are only specified through their value domains. This corresponds to the first part of the action space for the pCTMDP — at each state of the model we need to specify the particular values of mover and senser being used. The second part of the action space corresponds to the local succp attribute. In particular, for each location we have to specify the particular value of succp from the interval $[0,1]$. A policy, according to the definition given in Section 2.6, then is a function

$$\pi : \mathbb{R}_{\geq 0} \times \mathbb{Z}^8 \times \mathbb{R}^2_{\geq 0} \times [0,1]^8 \to [0,1]$$

assigning a probability for each of the possible combinations of attributes mover, senser and succp for each time $t \in \mathbb{R}_{\geq 0}$ and state $\mathbf{x} \in \mathbb{Z}^8$. Remember, that the choice of succp has to be made for each location giving rise to four copies of $[0,1]$ in the signature of the function. It is worth noting that the above corresponds to the non-trivial parts of the policies π. To give a perfectly precise description according to the semantics in Chapter 3 the policy would also have to assign values for each of the loc and target attributes. However, as explained, the value domains for these remain singleton sets throughout the evolutions and thus the choice of policy with respect to those attributes is trivial. Let us denote this resulting space of probability distributions by Π. In the rest of this chapter we are going to consider deterministic policies such that

$$\pi : \mathbb{R}_{\geq 0} \times \mathbb{Z}^8 \times \mathbb{R}^2_{\geq 0} \times [0,1]^8 \to \{0,1\}$$

Application of a policy π to the pCTMDP corresponding to the model gives us behaviours of individual robots as given in Figure 6.4. The same model was considered as a running example in Chapters 4 and 5. Here, we have made explicit that the transition rates depend on a chosen policy π. We have denoted by $\pi_1^{(i,j)}(t,\mathbf{x})$ the rate of robots moving out of location (i,j) at time t given the population state $\mathbf{x}$ under the deterministic policy π. Similarly, $\pi_2(t,\mathbf{x})$ denotes the rate of sensing and broadcasting the message about the target. The mode-switching population model follows for the resulting pCTMC according to the construction in Section 4.2. In particular, as was the case with the running examples of the previous chapter, the before broadcast behaviour is treated as one mode of dynamics and the after broadcast behaviour as the other.


(0,1)
(1,1)
(0,0)
(1,0)
$\pi_2(t,\mathbf{x})$
$\frac{1}{2}\pi_1^{(0,0)}(t,\mathbf{x})$
$\pi_1^{(0,1)}(t,\mathbf{x})$
$\pi_1^{(1,1)}(t,\mathbf{x})$
$\frac{1}{2}\pi_1^{(1,0)}(t,\mathbf{x})$
$\frac{1}{2}\pi_1^{(1,0)}(t,\mathbf{x})$
$\frac{1}{2}\pi_1^{(0,0)}(t,\mathbf{x})$
(0,1)
(1,1)
(0,0)
(1,0)
$\pi_1^{(0,1)}(t,\mathbf{x})$
$\pi_1^{(0,0)}(t,\mathbf{x})$
$\pi_1^{(1,1)}(t,\mathbf{x})$
$\pi_1^{(1,0)}(t,\mathbf{x})$
$\pi_1^{(1,0)}(t,\mathbf{x})$
$\pi_1^{(0,0)}(t,\mathbf{x})$

(a) Before broadcast – random walk (b) After broadcast – directed walk

Figure 6.4: Behaviour of individuals in the swarm model with 4 locations under some deterministic policy π.

6.3 Policy synthesis

A time-dependent policy for the described pCTMDP would lead to a time-inhomogeneous pCTMC. In this section we are going to restrict the space of policies Π to those that are stationary, or in other words, not dependent on time. A simple policy synthesis problem would arise from considering policies that keep the movement rate constant for all states of the system. We are going to make the situation slightly more complex and consider policies where the transition rates are no longer linearly dependent on the population variables. In particular, we consider the effects of congestion, whereby we model the situation where the movement rate of the robots decreases as the density in a given location increases. Congestion or interference is a common problem in swarm robotics that usually leads to degraded performance [87, 91, 115]. This happens especially in the cases where robots are moving towards a common target region and have to compete for available space. For this example we are considering one possible way to capture such effects on the swarm behaviour.

In order to model the congestion effects we are going to construct the policy π so that some maximum movement rate r_m, given by the global store attribute mover, of robots is multiplied by the exponential $e^{-a\times x}$, where x denotes the population density at the given location. In particular, the rate of movement out of location (i,j) under policy π becomes

$$\pi_1^{(i,j)}(t,\mathbf{x}) = r_m e^{-a\times x_{ij}}$$

where x_{ij} is the population density at location (i,j). Such exponential degradation of the performance of individual robots in a swarm was reported, for example, in [87]. The choice of the constant a here is arbitrary to give an example. The higher values of a correspond to more severe effects of congestion while lower values would have lesser effects on the dynamics. For example $a = 0.5$ would give that if the population density at a given location is $x = 1.0$ then the rate of movement is given by $0.6065r_m$ (rounded to 4 decimal places), or roughly halved while $a = 0.9$ would give the rate of movement as $0.4066r_m$. Finally, note that the effects of congestion could equivalently be taken into account directly in the rate definitions for *directed** and *random**.

For the stochastic simulation and moment closure-based approaches we will need a function that gives the equivalent behaviour in the case of non-scaled population variables. In particular, for a population of N robots consider $e^{-a\times\frac{x}{N}}$. The meaning of this model would be that if the entire swarm is in the same location the congestion has the effect of approximately halving the rate of movement. In the context of the running example we consider the synthesis of the succp parameter as a special case of policy synthesis. In particular, how robust the behaviour of the robots should be for the collective to satisfy its goal. We consider two objectives:

- The first considers only the mean behaviour of the models and requires 80% of the swarm to reach the target location $(1,1)$ in the finite time interval $[0,10]$. We are going to refer to this as Obj_1.
- The second one requires the probability of 80% of the swarm to reach the target location $(1,1)$ in the finite time interval $[0,10]$ to be greater than 0.9. We are going to refer to this as Obj_2.

For the first case we are going to use the means from fluid approximation and moment closure-based constructions in the calculations. For the second we assume that the population variables are normally distributed and use the mean and variance from the linear noise and moment closure approximations-based constructions in the objective evaluations. We are going to treat the policy synthesis as a simple logistic regression problem where we aim to separate the values succp based on whether the objectives would be satisfied or not. This is akin to works on parameter synthesis which aim to find the regions of the parameter space where a given specification is satisfied [26, 38, 25].

The set-up for this is standard: consider a linear function $y = w_0 + w_1 p$ of single explanatory variable (in this case value of succp, denoted p) and a logistic function $\sigma(r) = 1/(1+e^{-w_0-w_1 p})$ where $\sigma(p)$ is interpreted as the probability of success given succp value p. We are going to expect the goal to be satisfied if $\sigma(p) > 0.5$. The weights for the regression model are going to be fitted based on trajectories sampled using stochastic simulation and the constructed approximations for 200 random succp values. All calculations were done by fixing the values corresponding to store attributes mover and senser to 1 and 0.1 respectively. In particular, we have considered the policy space where these two attributes are kept constant while the value for succp is allowed to change.

For stochastic simulation experiments we ran 5000 replications for each of the considered failr parameters. The mean and standard deviation summary statistics of the population variables were considered at 100 equally spaced sampling points in the time interval $[0, 10]$. The low sampling rate was chosen to keep the computation time for constructing the summaries low. Tables 6.1 and 6.2 give the comparison of decision boundaries, where $\sigma(p) = 0.5$, obtained for the two objectives for population levels 100 and 200, respectively. For both of the population levels, we sampled 200 succp values and considered the approximations introduced in Chapter 5 to estimate whether the objectives are or are not satisfied. The value of the congestion parameter for experiments shown in Tables 6.1 and 6.2 was set to 0.7. In order to get an idea of the sensitivity of the performed analysis to the congestion parameter a, we have repeated the stochastic simulations of the model for values $a = 0.5$ and $a = 0.9$. The results, summarised in Table 6.3, give an indication of how the results of the parameter synthesis are affected by our choice of parameter a. Expectedly, for higher values of a the behaviour of robots needs to be more robust in order to satisfy the defined goals.

In the case of the smaller population size, $N = 100$, we see that for the mean-based objective all methods give a good estimate for the decision boundary for succp. As was the case with the experimental results in the previous chapter the iterative construction, proposed in Chapter 5, gives an improvement over the direct coupling-based methods. The experiments also reveal that in the case of this example the difference between using a linear noise and moment-based construction is negligible, while the impact of choosing between coupling and iterative constructions is more substantial. This can largely be attributed to the fact that the iterative constructions seem to, at least in most cases as seen in Chapter 5, do better at approximating the mean dynamics.

Table 6.1: Calculated decision boundaries (p bdy.) for the logistic regression problem. Relative errors (rel. err.) are given with respect to the stochastic simulation results. Population $N = 100$.

	Fluid/LNA cpl.		Fluid/LNA iter.		Moment cpl.		Moment iter.		SSA
	p bdy.	rel. err.	p bdy.	rel. err.	p bdy.	rel. err	p bdy.	rel. err.	p bdy.
Obj_1	0.670	1.5%	0.680	$< 0.1\%$	0.670	1.5%	0.680	$< 0.1\%$	0.680
Obj_2	0.713	5.7%	0.730	3.5%	0.717	5.2%	0.730	3.5%	0.756

6.4 Discussion

In this chapter we have presented a small example of policy synthesis which presents a framework which fits together the ideas presented in this thesis. In particular, we

Table 6.2: Calculated decision boundaries (p bdy.) for the logistic regression problem. Relative errors (rel. err.) are given with respect to the stochastic simulation results. Population $N = 200$.

	Fluid/LNA cpl.		Fluid/LNA iter.		Moment cpl.		Moment iter.		SSA
	p bdy.	rel. err.	p bdy.	rel. err.	p bdy.	rel. err	p bdy.	rel. err.	p bdy.
Obj_1	0.660	$< 0.1\%$	0.660	$< 0.1\%$	0.660	$< 0.1\%$	0.660	$< 0.1\%$	0.660
Obj_2	0.680	2.5%	0.690	1.1%	0.690	1.1%	0.690	1.1%	0.698

Table 6.3: Decision boundaries (p bdy.) for the two objectives under three different values of the congestion parameter a. Calculated values are based on the stochastic simulation of the population model.

pop. N	congestion param. a	p bdy. Obj_1	p bdy. Obj_2
100	0.5	0.661	0.731
100	0.7	0.680	0.756
100	0.9	0.701	0.793
200	0.5	0.648	0.684
200	0.7	0.660	0.698
200	0.9	0.679	0.722

described a model expressed in the CARMA-C language equipped with CTMDP semantics and set up a simple policy synthesis problem where a single parameter can be changed or controlled. The semantics of the language presented does not discriminate against more complex cases like time-dependent or probabilistic policies. With an appropriate choice of policy space we could, for example, consider scenarios where the movement rate of the robots further degrades with time. Similarly, it is possible to consider policies which put a distribution over the global store attributes mover and senser modelling parameters whose exact value is unknown and can fluctuate over time. The term imprecise has been previously used for such parameters [20] in the context of underspecified pCTMCs.

When considering more complex scenarios such as these it is interesting to consider the computational scalability of the presented analysis. For example, when considering time-dependent policies we are faced with a different highly non-trivial problem of conducting policy synthesis over a space of time-dependent functions. Constructing the fluid, linear noise and moment-based approximations for a given time-dependent function, however, presents few additional difficulties. Such approximation can then be used, for example, in statistical methods based policy synthesis in [11] for faster evaluations of individual policies.

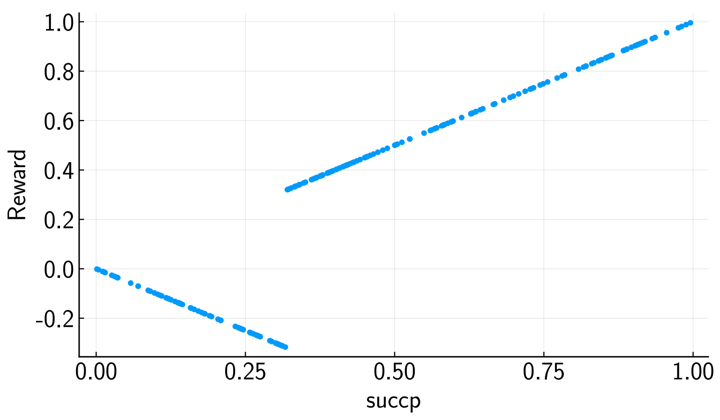


Figure 6.5: Example of a reward function for the model in this chapter sampled at 200 points.

The method of classifying these parameters into ones that satisfy the given objective and the ones that do not via logistic regression is only one of the possible ways we can consider such problems. Perhaps a more common approach from control theory would be to directly optimise with respect to a constructed a reward function. In the case of the above example we could quite easily construct a reward function on sample trajectories that returns 1 if the objective is satisfied and 0 otherwise, giving rise to a step function. As an additional refinement, a regularisation term can be added such that if the objective is satisfied we penalise higher values of `succp`. On the other hand if the objective is not satisfied we reward increasing `succp`. One such possible reward function for the example in this chapter sampled at 200 points is given in Figure 6.5. Note that even this simple reward function leads to an optimisation problem that is non-convex and discontinuous. Additionally note that many of the interesting cases would feature multiple objectives. Construction and optimisation of such multi-objective problems is a well-established and active research field.

Chapter 7

Conclusions

In this thesis we studied two different but intimately linked aspects of analysing collective dynamics in the context of formal modelling methods. In particular, we considered model specification and analysis motivated by policy synthesis problems for collective dynamics. Our starting point was that of formal quantitative modelling through the application of process algebra Carma.

As our first contribution we presented alternative semantics for the Carma process algebra that allow for convenient compositional specification of policy synthesis problems in the CTMDP framework. The existing CTMC-based semantics were shown to extend naturally to CTMDPs by allowing non-determinism in the definitions of the store attributes of the language.

The CTMDP models that arise from the defined high-level language Carma-C are not generally tractable by exact numerical methods. Thus as the second point of interest we considered the application of continuous state approximations to such models. The application domain of collective and, in particular, population dynamics provide many examples where fluid, linear noise and moment closure approximations can be used. However, there is a large class of population models arising from Carma-C where such approximations are not trivially applicable. In this thesis we studied situations where the agents are equipped with knowledge; moreover we assumed that the agents can learn about their environment through experience, and share that information within the population through broadcast communication. We considered a class of models where broadcast communication induces switches in the dynamics of the population. Each level of information available to the collective corresponds to a different mode in the underlying stochastic model. As a consequence we demonstrated an application of existing hybrid-continuous approximations to such mode-switching systems. Moreover we considered the computational difficulties arising when considering such models and proposed further approximations for computationally efficient solutions. By making a heuristic simplification to the filtering equations, which describe the probability of a given dynamic mode up to the history of the process, we examined two constructions: directly coupling the simplified filtering equations with the fluid, linear noise and moment-based approximation and an iterative construction. The results in Chapter 5 showed that in our cases the iterative construction offered significantly better approximation to the mean dynamics.

As a final step in this thesis we showed how the formal modelling language together with the considered approximations form a cohesive framework for policy synthesis in the case of collective dynamics. In particular, we constructed and analysed a simple policy synthesis problem for a model stated in the defined CARMA-C language.

There are a wealth of possible extensions that can be considered on the basis of the work presented in this thesis. Here we are going to propose a few. From the point of view of model specification and the constructed semantics for CARMA-C, the examples presented give only a small sample of the policy synthesis problems that we are able to consider. Firstly, the non-determinism induced via the definition of store attributes in CARMA-C models is amenable to various interpretations. In Chapter 6 we gave an example where the non-deterministic behaviour was considered as a controllable parameter of the robots. However, it is also possible to consider cases where the non-determinism is treated as uncertainty in parameters in the model identification context — a cost function is constructed as a comparison between the model output and measurement from a physical system. Similarly, we can combine interpretations of non-determinism. For example, consider policy synthesis for collective systems where we are uncertain about the precise effect of the environment on the dynamics. We can consider problems of optimising for the controllable parameters of the model up to uncertainty in the non-controllable ones. A deeper investigation of such problems in the context of the considered process algebraic framework is left as future work.

The approximation methods considered in this thesis are applicable to homogeneous population models. In many real-life collective systems the populations will be composed of heterogeneous components. While this limits the general applicability of the analysis considered in this thesis there are many interesting cases that can be explored further. For example, our models assumed "perfect" broadcast where component always receive the sent message. A more realistic view would result from the consideration of communication where components may fail to receive a message. Expressing such situations in CARMA-C is simple. The semantics of the language allow us to define a probability with which a given broadcast message is received by an eligible component. This probability is again dependent on the store attributes — for example, location of the receiver with respect to the sender. Such considerations would be a step towards being able to apply the methods considered in a more general setting. However, whether or not the mode-switching dynamics can then still be separated from the population dynamics in a way that leads to useful approximations remains to be seen. Similarly, much more can also be said about methods for solving the arising policy synthesis problems. For example, in this thesis we have only given a small example when the policy space consists of stationary and deterministic policies but more complex scenarios for the collective dynamics can be considered. It is moreover of interest how far do the model structure based approximations extend when considering real-world applications.

A further line of work we propose here deals with the iterative constructions for approximations of population dynamics in Chapter 5. We have seen in Chapter 5 that in most cases the iterative construction offers an improvement over directly coupling the filtering equations, detailing the mode-switching behaviour, with the continuous evolution of the population dynamics. For future work the general conditions and reasons under which the iterative construction can be expected to provide an improved approximation are of interest. The second aspect, related to the constructed approximations, to consider is efficient numerical solver methods for such approximations and consideration of larger number of modes. Note that even when the number of modes is potentially very large then in the context of transient evolution of the system we might be only interested in a small subset. For example, if the probability of reaching certain modes in a given finite time interval is low. This makes it possible to reduce the number of ODEs considered for a finite time interval. While in this thesis we concentrated on applying the discussed heuristic constructions in the context of models arising from broadcast communication, the iterative construction can also be considered as an initial coarse approximation in other more commonly considered settings: in models with a changing environment where the mode-switching behaviour of the system is not dependent on the population dynamics [127]; situations where mode-switching results from the behaviour of a small number of components in the system, leading to more commonly studied hybrid models [66, 39]; and stochastic hybrid systems more generally when the number of switches happening in a transient time period under study is small.

Bibliography

[1] CARMA Eclipse Plugin | QUANTICOL Toolset for the analysis of Collective Adaptive Systems. `http://quanticol.sourceforge.net/?page_id=27`.

[2] Github repository for the implementation of examples. `https://github.com/pihop/ThesisSupplement.jl`.

[3] A quantitative approach to management and design of collective and adaptive behaviours. `https://blog.inf.ed.ac.uk/quanticol/`.

[4] SymEngine. `https://github.com/symengine/symengine`.

[5] G. Agha and K. Palmskog. A survey of statistical model checking. *ACM Transactions on Modeling and Computer Simulation*, 28(1):6:1–6:39, 2018.

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

[7] D. F. Anderson. A modified next reaction method for simulating chemical systems with time dependent propensities and delays. *The Journal of Chemical Physics*, 127(21):214107, 2007.

[8] A. Aziz, K. Sanwal, V. Singhal, and R. Brayton. Model-checking continuous-time Markov chains. *ACM Transactions on Computational Logic*, 1(1):162–170, 2000.

[9] C. Baier, H. Hermanns, J. Katoen, and B. R. Haverkort. Efficient computation of time-bounded reachability probabilities in uniform continuous-time Markov decision processes. *Theoretical Computer Science*, 345(1):2–26, 2005.

[10] G. Balbo. Introduction to generalized stochastic petri nets. In *Proceedings of the 7th International Conference on Formal Methods for Performance Evaluation*, SFM 2007, page 83–131. Springer-Verlag, 2007.

[11] E. Bartocci, L. Bortolussi, T. Brázdil, D. Milios, and G. Sanguinetti. Policy learning in continuous-time Markov decision processes using Gaussian processes. *Performance Evaluation*, 116:84–100, 2017.

[12] J. Bect. A unifying formulation of the Fokker–Planck–Kolmogorov equation for general stochastic hybrid systems. *Nonlinear Analysis: Hybrid Systems*, 4(2):357 – 370, 2010.

[13] A. Bensoussan, J. Frehse, and P. Yam. *Mean Field Games and Mean Field Type Control Theory.* Springer New York, 2013.

[14] M. Bernardo and R. Gorrieri. Extended Markovian process algebra. In U. Montanari and V. Sassone, editors, *CONCUR '96: Concurrency Theory*, pages 315–330. Springer Berlin Heidelberg, 1996.

[15] Bortolussi and J. Hillston. Model checking single agent behaviours by fluid approximation. *Information and Computation*, 242:183–226, 2015.

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

[17] L. Bortolussi. Limit behavior of the hybrid approximation of stochastic process algebras. In *Analytical and Stochastic Modeling Techniques and Applications, 17th International Conference, ASMTA 2010, Cardiff, UK, June 14-16. Proceedings*, pages 367–381, 2010.

[18] L. Bortolussi. Hybrid behaviour of Markov population models. *Information and Computation*, 247:37–86, 2016.

[19] L. Bortolussi, L. Cardelli, M. Kwiatkowska, and L. Laurenti. Central limit model checking. *ACM Trans. Comput. Logic*, 20(4), 2019.

[20] L. Bortolussi and N. Gast. Mean field approximation of uncertain stochastic models. In *46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, DSN 2016, Toulouse, France, June 28 - July 1. Proceedings*, pages 287–298, 2016.

[21] L. Bortolussi and J. Hillston. Fluid model checking. In *Concurrency Theory - 23rd International Conference, CONCUR 2012, Newcastle upon Tyne, UK, September 4-7. Proceedings*, pages 333–347, 2012.

[22] L. Bortolussi and R. Lanciani. Model checking Markov population models by central limit approximation. In *Quantitative Evaluation of Systems. Proceedings*, pages 123–138, 2013.

[23] L. Bortolussi, D. Milios, and G. Sanguinetti. Efficient stochastic simulation of systems with multiple time scales via statistical abstraction. In *Computational Methods in Systems Biology - 13th International Conference, CMSB 2015, Nantes, France, September 16-18. Proceedings*, pages 40–51, 2015.

[24] L. Bortolussi and A. Policriti. Dynamical systems and stochastic programming: To ordinary differential equations and back. *Transactions on Computational Systems Biology*, 11:216–267, 2009.

[25] L. Bortolussi, A. Policriti, and S. Silvetti. Logic-based multi-objective design of chemical reaction networks. In *Hybrid Systems Biology - 5th International Workshop, HSB 2016, Grenoble, France, October 20-21. Proceedings*, pages 164–178, 2016.

[26] L. Bortolussi and S. Silvetti. Bayesian statistical parameter synthesis for linear temporal properties of stochastic models. In D. Beyer and M. Huisman, editors, *Tools and Algorithms for the Construction and Analysis of Systems*, pages 396–413. Springer International Publishing, 2018.

[27] P. Boutillier, M. Maasha, X. Li, H. F. Medina-Abarca, J. Krivine, J. Feret, I. Cristescu, A. G. Forbes, and W. Fontana. The Kappa platform for rule-based modeling. *Bioinformatics*, 34(13):i583–i592, 2018.

[28] L. Bronstein and H. Koeppl. Marginal process framework: A model reduction tool for Markov jump processes. *Physical Review E*, 97:062147, Jun 2018.

[29] J. Bruno, P. Downey, and G. N. Frederickson. Sequencing tasks with exponential service times to minimize the expected flow time or makespan. *Journal of the ACM*, 28(1):100–113, 1981.

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

[31] P. Buchholz, I. Dohndorf, and D. Scheftelowitsch. Optimal decisions for continuous time Markov decision processes over finite planning horizons. *Computers & Operations Research*, 77:267–278, 2017.

[32] C. J. Burke and M. Rosenblatt. A Markovian function of a Markov chain. *The Annals of Mathematical Statistics*, 29(4):1112–1122, 12 1958.

[33] Y. Butkova, H. Hatefi, H. Hermanns, and J. Krcál. Optimal continuous time Markov decisions. In *Automated Technology for Verification and Analysis, 2015. Proceedings*, pages 166–182, 2015.

[34] B. Caillaud, B. Delahaye, K. G. Larsen, A. Legay, M. L. Pedersen, and A. Wasowski. Constraint Markov chains. *Theoretical Computer Science*, 412(34):4373–4404, 2011.

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

[36] L. Cardelli, M. Kwiatkowska, and L. Laurenti. Stochastic analysis of chemical reaction networks using linear noise approximation. *Biosystems*, 149:26–33, 2016.

[37] L. Cardelli, M. Tribastone, M. Tschaikowski, and A. Vandin. Comparing chemical reaction networks: A categorical and algorithmic perspective. *Theoretical Computer Science*, 765:47 – 66, 2019.

[38] M. Ceska, F. Dannenberg, N. Paoletti, M. Kwiatkowska, and L. Brim. Precise parameter synthesis for stochastic biochemical systems. *Acta Informatica*, 54(6):589–623, 2017.

[39] A. Crudu, A. Debussche, A. Muller, and O. Radulescu. Convergence of stochastic gene networks to hybrid piecewise deterministic processes. *Annals of Applied Probability*, 22(5):1822–1859, 10 2012.

[40] P. Cuijpers and M. Reniers. Hybrid process algebra. *The Journal of Logic and Algebraic Programming*, 62(2):191 – 245, 2005.

[41] R. Darling and J. Norris. Differential equation approximations for Markov chains. *Probability Surveys*, 5:37–79, 2008.

[42] R. David and H. Alla. On hybrid petri nets. *Discrete Event Dynamic Systems*, 11(1-2):9–40, 2001.

[43] M. Davis. *Markov Models & Optimization.* Chapman & Hall/CRC Monographs on Statistics & Applied Probability. Taylor & Francis, 1993.

[44] R. De Nicola, D. Latella, M. Loreti, and M. Massink. A uniform definition of stochastic process calculi. *ACM Computing Surveys*, 46(1):5:1–5:35, 2013.

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

[46] A. de Palma and C. Lefevre. Individual decision-making in dynamic collective systems. *The Journal of Mathematical Sociology*, 9(2):103–124, 1983.

[47] C. Dehnert, S. Junges, J. Katoen, and M. Volk. A storm is coming: A modern probabilistic model checker. In *Computer Aided Verification - 29th International Conference, CAV 2017, Heidelberg, Germany. Proceedings, Part II*, pages 592–600, 2017.

[48] A. Duncan, R. Erban, and K. C. Zygalakis. Hybrid framework for the simulation of stochastic chemical kinetics. *Journal of Computational Physics*, 326:398–419, 2016.

[49] K. Elamvazhuthi and S. Berman. Mean-field models in swarm robotics: a survey. *Bioinspiration & Biomimetics*, 15(1):015001, 2019.

[50] S. N. Ethier and T. G. Kurtz. *Markov processes – characterization and convergence.* Wiley Series in Probability and Mathematical Statistics: Probability and Mathematical Statistics. John Wiley & Sons Inc., New York, 1986.

[51] E. A. Feinberg, M. Mandava, and A. N. Shiryaev. On solutions of Kolmogorov's equations for nonhomogeneous jump Markov processes. *Journal of Mathematical Analysis and Applications*, 411(1):261 – 270, 2014.

[52] C. Feng and J. Hillston. PALOMA: A process algebra for located Markovian agents. In *Quantitative Evaluation of Systems - 11th International Conference, QEST 2014, Florence, Italy. Proceedings*, pages 265–280, 2014.

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

[54] D. Gabelaia, R. Kontchakov, Á. Kurucz, F. Wolter, and M. Zakharyaschev. Combining spatial and temporal logics: Expressiveness vs. complexity. *Journal of Artificial Intelligence Research*, 23:167–243, 2005.

[55] V. Galpin, L. Bortolussi, and J. Hillston. HYPE: hybrid modelling by composition of flows. *Formal Aspects of Computing*, 25(4):503–541, 2013.

[56] V. Galpin, N. Zon, P. Wilsdorf, and S. Gilmore. Mesoscopic modelling of pedestrian movement using carma and its tools. *ACM Transactions on Modeling and Computer Simulation*, 28(2), 2018.

[57] N. Gast, B. Gaujal, and J. L. Boudec. Mean field for Markov decision processes: From discrete to continuous optimization. *IEEE Transactions on Automatic Control*, 57(9):2266–2280, 2012.

[58] A. Georgoulas, J. Hillston, D. Milios, and G. Sanguinetti. Probabilistic programming process algebra. In *Quantitative Evaluation of Systems - 11th International Conference, QEST 2014. Proceedings*, pages 249–264, 2014.

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

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

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

[62] M. Gribaudo and A. Remke. Hybrid petri nets with general one-shot transitions. *Performance Evaluation*, 105:22 – 50, 2016.

[63] M. C. Guenther, A. Stefanek, and J. T. Bradley. Moment closures for performance models with highly non-linear rates. In M. Tribastone and S. Gilmore, editors, *Computer Performance Engineering*, pages 32–47. Springer Berlin Heidelberg, 2013.

[64] H. Hansson and B. Jonsson. A logic for reasoning about time and reliability. *Formal Aspects of Computing*, 6(5):512–535, 1994.

[65] J. Hasenauer, V. Wolf, A. Kazeroonian, and F. J. Theis. Method of conditional moments (MCM) for the chemical master equation. *Journal of Mathematical Biology*, 69(3):687–735, 2014.

[66] A. Hellander and P. Lötstedt. Hybrid method for the chemical master equation. *Journal of Computational Physics*, 227(1):100–122, 2007.

[67] T. A. Henzinger. The theory of hybrid automata. In *Proceedings, 11th Annual IEEE Symposium on Logic in Computer Science, New Brunswick, New Jersey, USA, 1996*, pages 278–292. IEEE Computer Society, 1996.

[68] J. P. Hespanha. A model for stochastic hybrid systems with application to communication networks. *Nonlinear Analysis: Theory, Methods & Applications*, 62(8):1353 – 1383, 2005. Hybrid Systems and Applications.

[69] J. P. Hespanha. Modelling and analysis of stochastic hybrid systems. *IEE Proceedings - Control Theory and Applications*, 153(5):520–535, 2006.

[70] A. Hilfinger and J. Paulsson. Separating intrinsic from extrinsic fluctuations in dynamic biological systems. *Proceedings of the National Academy of Sciences of the United States of America*, 108 29:12167–72, 2011.

[71] J. Hillston. *A Compositional Approach to Performance Modelling.* Cambridge University Press, New York, NY, USA, 1996.

[72] J. Hillston. Fluid flow approximation of PEPA models. In *Second International Conference on the Quantitative Evaluaiton of Systems (QEST) 2005, Torino, Italy. Proceedings*, pages 33–43, 2005.

[73] A. C. Hindmarsh, P. N. Brown, K. E. Grant, S. L. Lee, R. Serban, D. E. Shumaker, and C. S. Woodward. SUNDIALS: suite of nonlinear and differential/algebraic equation solvers. *ACM Transactions on Mathematical Software*, 31(3):363–396, 2005.

[74] C. A. R. Hoare. *Communicating Sequential Processes*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1985.

[75] V. Holubec, P. Chvosta, M. Einax, and P. Maass. Attempt time Monte Carlo: An alternative for simulation of stochastic jump processes with time-dependent transition rates. *EPL (Europhysics Letters)*, 93(4):40003, 2011.

[76] F. Horn and R. Jackson. General mass action kinetics. *Archive for Rational Mechanics and Analysis*, 47(2):81–116, 1972.

[77] G. Horton, V. G. Kulkarni, D. M. Nicol, and K. S. Trivedi. Fluid stochastic petri nets: Theory, applications, and solution techniques. *European Journal of Operational Research*, 105(1):184 – 201, 1998.

[78] M. John, C. Lhoussaine, J. Niehren, and A. M. Uhrmacher. The attributed pi calculus. In *Computational Methods in Systems Biology, 6th International Conference, CMSB 2008, Rostock, Germany. Proceedings*, pages 83–102, 2008.

[79] B. Jonsson and K. G. Larsen. Specification and refinement of probabilistic processes. In *Proceedings of the Sixth Annual Symposium on Logic in Computer Science (LICS) 1991, Amsterdam, The Netherlands*, pages 266–277, 1991.

[80] A. Kazeroonian, F. Fröhlich, A. Raue, F. J. Theis, and J. Hasenauer. CERENA: ChEmical REaction Network Analyzer—A Toolbox for the Simulation and Analysis of Stochastic Chemical Kinetics. *PLOS ONE*, 11(1):1–15, 2016.

[81] A. Kazeroonian, F. J. Theis, and J. Hasenauer. Modeling of stochastic biological processes with non-polynomial propensities using non-central conditional moment equation. *IFAC Proceedings Volumes*, 47(3):1729 – 1735, 2014.

[82] J. G. Kemeny and J. L. Snell. *Finite Markov chains*. Undergraduate texts in mathematics. Springer, New York, 1976. Reprint of the 1960 ed. published by Van Nostrand, Princeton, N.J., in the University series in undergraduate mathematics.

[83] T. G. Kurtz. Solutions of ordinary differential equations as limits of pure jump Markov processes. *Journal of Applied Probability*, 7(1):49–58, 1970.

[84] M. Z. Kwiatkowska, G. Norman, and D. Parker. PRISM 4.0: Verification of probabilistic real-time systems. In *Computer Aided Verification - 23rd International Conference, CAV 2011, Snowbird, UT, USA. Proceedings*, pages 585–591, 2011.

[85] C. Lefèvre. Optimal control of a birth and death epidemic process. *Operations Research*, 29(5):971–982, 1981.

[86] A. Legay, B. Delahaye, and S. Bensalem. Statistical model checking: An overview. In *Runtime Verification - First International Conference, RV 2010, St. Julians, Malta. Proceedings*, pages 122–135, 2010.

[87] K. Lerman and A. Galstyan. Mathematical model of foraging in a group of robots: Effect of interference. *Autonomous Robots*, 13(2):127–141, 2002.

[88] M. Loreti and J. Hillston. Modelling and analysis of collective adaptive systems with CARMA and its tools. In *Formal Methods for the Quantitative Evaluation of Collective Adaptive Systems, Advanced Lectures*, pages 83–119, 2016.

[89] L. Luisa Vissat, J. Hillston, G. Marion, and M. J. Smith. MELA: modelling in ecology with location attributes. In *Proceedings 14th International Workshop Quantitative Aspects of Programming Languages and Systems, QAPL 2016, Eindhoven, The Netherlands*, pages 82–97, 2016.

[90] H. Lv, J. Hillston, P. Piho, and H. Wang. An attribute-based availability model for large scale IaaS clouds with CARMA. *IEEE Transactions on Parallel and Distributed Systems*, 31(3):733–748, 2020.

[91] L. S. Marcolino, Y. T. dos Passos, Á. A. F. de Souza, A. dos Santos Rodrigues, and L. Chaimowicz. Avoiding target congestion on the navigation of robotic swarms. *Autonomous Robots*, 41(6):1297–1320, 2017.

[92] C. Maus, S. Rybacki, and A. M. Uhrmacher. Rule-based multi-level modeling of cell biological systems. *BMC Systems Biology*, 5:166, 2011.

[93] S. Menz, J. C. Latorre, C. Schütte, and W. Huisinga. Hybrid stochastic-deterministic solution of the chemical master equation. *Multiscale Modeling & Simulation*, 10(4):1232–1262, 2012.

[94] R. C. Merton. Option pricing when underlying stock returns are discontinuous. *Journal of Financial Economics*, 3(1):125 – 144, 1976.

[95] M. Michaelides, J. Hillston, and G. Sanguinetti. Geometric fluid approximation for general continuous-time Markov chains. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 475(2229):20190100, 2019.

[96] R. Milner. *Communication and Concurrency.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1989.

[97] L. Nenzi, L. Bortolussi, V. Ciancia, M. Loreti, and M. Massink. Qualitative and quantitative monitoring of spatio-temporal properties with SSTL. *Logical Methods in Computer Science*, 14(4), 2018.

[98] J. Niño-Mora. Resource allocation and routing in parallel multi-server queues with abandonments for cloud profit maximization. *Computers & Operations Research*, 103:221–236, 2019.

[99] J. R. J. R. Norris. *Markov chains.* Cambridge series on statistical and probabilistic mathematics; 2. Cambridge University Press, Cambridge, 1998.

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

[101] B. Oksendal. *Stochastic Differential Equations (3rd Ed.): An Introduction with Applications.* Springer-Verlag, Berlin, Heidelberg, 1992.

[102] C. C. Pantelides. The consistent initialization of differential-algebraic systems. *SIAM Journal on Scientific and Statistical Computing*, 9(2):213–231, 1988.

[103] P. Piho and J. Hillston. Fluid approximation based analysis for mode switching population dynamics. (to be published).

[104] P. Piho and J. Hillston. Policy synthesis for collective dynamics. In *15th International Conference on Quantitative Evaluation of SysTems (QEST 2018)*. Springer, 2018.

[105] G. D. Plotkin. *A structural approach to operational semantics*. Report DAIMI-FN-19, Computer Science Dept, Aarhus University, Denmark, 1981.

[106] A. Pnueli. The temporal logic of programs. In *18th Annual Symposium on Foundations of Computer Science, Providence, Rhode Island, USA*, pages 46–57, 1977.

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

[108] M. L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, Inc., New York, NY, USA, 1st edition, 1994.

[109] C. Rackauckas and Q. Nie. DifferentialEquations.jl – a performant and feature-rich ecosystem for solving differential equations in julia. *Journal of Open Research Software*, 5, 05 2017.

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

[111] J. Ross. A stochastic metapopulation model accounting for habitat dynamics. *Journal of Mathematical Biology*, 52(6):788–806, Jun 2006.

[112] M. H. Rothkopf. Scheduling with random service times. *Management Science*, 12(9):707–713, 1966.

[113] C. Safta, K. Sargsyan, B. J. Debusschere, and H. N. Najm. Hybrid discrete/continuum algorithms for stochastic reaction networks. *Journal of Computational Physics*, 281:177–198, 2015.

[114] D. Schnoerr, G. Sanguinetti, and R. Grima. Approximation and inference methods for stochastic biochemical kinetics—a tutorial review. *Journal of Physics A: Mathematical and Theoretical*, 50(9):093001, 2017.

[115] A. Schroeder, B. Trease, and A. Arsie. Balancing robot swarm cost and interference effects by varying robot quantity and size. *Swarm Intelligence*, 13(1):1–19, 2019.

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

[117] W. Stewart. *Introduction to the Numerical Solution of Markov Chains*. Princeton University Press, 1994.

[118] D. W. Stroock. *An Introduction to Markov Processes*, volume 230 of *Graduate Texts in Mathematics*. Springer, Heidelberg, 2nd edition, 2014.

[119] P. Thomas, N. Popović, and R. Grima. Phenotypic switching in gene regulatory networks. *Proceedings of the National Academy of Sciences of the United States of America*, 111, 2014.

[120] N. M. van Dijk. Uniformization for nonhomogeneous Markov chains. *Operations Research Letters*, 12(5):283 – 291, 1992.

[121] N. M. van Dijk, S. P. J. van Brummelen, and R. J. Boucherie. Uniformization: Basics, extensions and applications. *Performance Evaluation*, 118:8–32, 2018.

[122] N. Van Kampen. *Stochastic Processes in Physics and Chemistry*. North-Holland Personal Library. Elsevier Science, 2011.

[123] M. Voliotis, P. Thomas, R. Grima, and C. G. Bowsher. Stochastic simulation of biomolecular networks in dynamic environments. *PLOS Computational Biology*, 12(6):1–18, 06 2016.

[124] A. Waserhole and V. Jost. Pricing in vehicle sharing systems: optimization in queuing networks with product forms. *EURO Journal on Transportation and Logistics*, 5(3):293–320, 2016.

[125] P. Whittle. On the use of the normal approximation in the treatment of stochastic processes. *Journal of the Royal Statistical Society. Series B (Methodological)*, 19(2):268–281, 1957.

[126] C. Zechner, S. Deb, and H. Koeppl. Marginal dynamics of stochastic biochemical networks in random environments. In *European Control Conference, ECC 2013, Zurich, Switzerland. Proceedings*, pages 4269–4274, 2013.

[127] C. Zechner and H. Koeppl. Uncoupled analysis of stochastic reaction networks in fluctuating environments. *PLOS Computational Biology*, 10(12):1–9, 12 2014.

[128] X. Zhang, G. Neglia, J. F. Kurose, and D. F. Towsley. Performance modeling of epidemic routing. *Computer Networks*, 51(10):2867–2891, 2007.

[129] N. Zon and S. Gilmore. Data-driven modelling and simulation of urban transportation systems using carma. In *Leveraging Applications of Formal Methods, Verification and Validation. Distributed Systems - 8th International Symposium, ISoLA 2018, Limassol, Cyprus. Proceedings, Part III*, pages 274–287, 2018.

Appendix A

Hybrid linear noise theorem

For convenience let us recall the hybrid linear noise theorem stated in Section 4.3.2.

Theorem 5 (Hybrid linear noise approximation). *If the initial conditions $\hat{\mathbf{X}}^N(0) \to \tilde{\mathbf{X}}(0)$ and $\hat{Z}^N(0) \to \tilde{Z}(0)$ almost surely then for all $t < T < \infty$ the sequence of normalised processes $(\hat{\mathbf{X}}^N, \hat{Z}^N)$ converges weakly to a process $(\tilde{\mathbf{X}}, \tilde{Z})$ given by*

$$\begin{pmatrix} \tilde{\mathbf{X}}(t) \\ \tilde{Z}(t) \end{pmatrix} = \begin{pmatrix} \tilde{\mathbf{X}}(0) + \int_0^t \mathbf{F}(\tilde{\mathbf{X}}(s))ds + N^{-\frac{1}{2}} \int_0^t \mathbf{G}(\tilde{\mathbf{X}}(s))d\mathbf{W}_s \\ \tilde{Z}(0) + \sum_{z,z'} (\mathbf{e}_{z'} - \mathbf{e}_z) N_{\mathbf{e}_{z'} - \mathbf{e}_z} \left(\int_0^t g_{zz'}(\tilde{\mathbf{X}}(s))ds \right) \end{pmatrix}$$

where $\mathbf{W}_s$ is the n-dimensional Wiener process and $\mathbf{G}(\hat{\mathbf{x}})$ is the diffusion matrix defined as follows.

$$\mathbf{G}(\hat{\mathbf{x}}) = \sum_{\tau \in \hat{\mathcal{T}}_z} \mathbf{v}_\tau \mathbf{v}_\tau^T f_\tau(\hat{\mathbf{x}}) \qquad \textit{if } \tilde{Z}(t) = z$$

As pointed out in Chapter 4 the proof of this fact is a straightforward application of the work presented in [18] and for that reason we are only going to sketch some of the main ideas for extending the existing result to our case. The proof relies on the following statement of the linear noise approximation theorem. As done in Section 2.4.2 let us define a stochastic process $\mathbf{V}^N(t) \overset{\text{def}}{=} N^{\frac{1}{2}} \left(\hat{\mathbf{X}}^N(t) - \hat{\mathbf{x}}(t) \right)$ capturing rescaled fluctuations of the CTMC $\hat{\mathbf{X}}^N$ around the fluid limit $\hat{\mathbf{x}}$. In addition, consider a Gaussian process $\{\mathbf{V}(t) \in \mathbb{R}^n \mid t \in \mathbb{R}\}$ with mean $\mathbb{E}(t)$ and covariance $\mathbb{C}(t)$ given as solutions to the following ODE systems

$$\begin{cases} \frac{d}{dt}\mathbb{E}(t) = \mathbf{J}_\mathbf{F}(\hat{\mathbf{x}}(t))\mathbb{E}(t) \\ \mathbb{E}(0) = 0 \end{cases}$$

$$\begin{cases} \frac{d}{dt}\mathbb{C}(t) = \mathbf{J}_\mathbf{F}(\hat{\mathbf{x}}(t))\mathbb{C}(t) + \mathbb{C}(t)\mathbf{J}_\mathbf{F}^T(\hat{\mathbf{x}}(t)) + \mathbf{G}(\hat{\mathbf{x}}(t)) \\ \mathbb{C}(0) = 0 \end{cases}$$

where $\mathbf{J}_\mathbf{F}(\hat{\mathbf{x}}(t))$ denotes the Jacobian of the limit drift $\mathbf{F}$ calculated along $\hat{\mathbf{x}}(t)$ and

$$\mathbf{G}(\hat{\mathbf{x}}) = \sum_{\tau \in \hat{\mathcal{T}}^N} \mathbf{v}_\tau \mathbf{v}_\tau^T f_\tau(\hat{\mathbf{x}})$$

Theorem 6 (Linear noise approximation theorem). *With the above notation suppose that*

$$\lim_{N \to \infty} \|\mathbf{V}^N(0) - \mathbf{V}(0)\| = 0 \qquad \textit{almost surely}$$

Then $\mathbf{V}^N(t)$ converges in distribution to $\mathbf{V}(t)$.

The above differs from the statement of Theorem 2 only in the condition set on the initial conditions. In particular, it is enough for the convergence of stochastic fluctuations at time $t = 0$ to be almost sure rather than sure. To see that let us observe that the proof of linear noise approximation given in [50] shows that the scaled fluctuations are follow the equation

$$\mathbf{V}^N(t) = \mathbb{E}(t)\mathbf{V}^N(0) + \mathbf{U}^N(t) + \varepsilon^N(t) + \int_0^t \mathbb{E}(s)\mathbf{J}_{\mathbf{F}}(\hat{\mathbf{x}}(s))(\mathbf{U}^N(s) + \varepsilon^N(s))ds$$

where $\varepsilon^N(t)$ are some defined error terms and $\mathbf{U}^N$ is a vector of Poisson processes centered around their expectation and scaled by $N^{-\frac{1}{2}}$. The weak convergence of $\mathbf{V}^N$ to $\mathbf{V}$ is then shown by proving that the convergence $\sup_{s\leq t}|\varepsilon^N(s)| \to 0$ almost surely as $N \to \infty$ holds for the error terms and by noting that $\mathbf{U}^N$ converges weakly to time-inhomogeneous Brownian motion. Continuous mapping theorem is then used to recover the weak convergence of $\mathbf{V}^N$ to $\mathbf{V}$. Note that the almost sure or convergence in probability of $\mathbf{V}^N(0)$ to $\mathbf{V}(0)$ is enough for the argument to work.

The next step of the proof is to show the following:

Lemma 2. *Let $\hat{\mathbf{X}}^N$ be a sequence of normalised pCTMCs and $\tilde{\mathbf{X}} = N^{-\frac{1}{2}}\mathbf{V} + \hat{\mathbf{x}}$ with $\mathbf{V}$ the limiting noise process defined according to Theorem 6 and $\hat{\mathbf{x}}$ the fluid limit of $\hat{\mathbf{X}}^N$. Moreover, let $\mathbf{V}^N = \sqrt{N}(\hat{\mathbf{X}}^N - \hat{\mathbf{x}})$. If $\hat{\mathbf{X}}^N(0)$ converges weakly to $\tilde{\mathbf{X}}(0)$ and $\mathbf{V}^N(0)$ converges weakly to $\mathbf{V}(0)$ as $N \to \infty$ then*

1. *$\hat{\mathbf{X}}^N$ converges weakly as $N \to \infty$ to $\tilde{\mathbf{X}}$.*
2. *supposing that T_0^N and T_0 are the first jump times of stochastic events with rates λ^N and λ such that $\lambda^N \to \lambda$ uniformly for each compact set of $\mathbb{R}^N$ then T_0^N converges weakly to T_0 as $N \to \infty$.*
3. *the state $\hat{\mathbf{X}}^N(T_0^N)$ converges weakly to $\tilde{\mathbf{X}}(T_0)$ as $N \to \infty$.*

Again, the proof of the above facts follows that of an almost identical result in [18]. The proof of Theorem 6 is then concluded via an inductive argument. The idea is to restrict the attention to the joint process $(\hat{\mathbf{X}}_m^N, \hat{Z}_m^N)$ and the hybrid noise process $(\tilde{\mathbf{X}}_m, \tilde{Z}_m)$ that perform at most m discrete jumps. Supposing that T_0^N and T_0 are the first jump times of the processes $\hat{Z}_m^N$ and $\tilde{Z}_m$ respectively note that the above lemma shows the convergence of the sequence of joint processes $(\hat{\mathbf{X}}_m^N, \hat{Z}_m^N)$ to $(\tilde{\mathbf{X}}_m, \tilde{Z}_m)$ up to the first discrete jump. The processes can then be restarted from the time T_0^N and T_0 repeating the arguments for the convergence up to the next jump and so on.

Appendix B

Conditional moments

B.1 Proof of Lemma 1

For convenience let us recall the statement of the lemma here.

Lemma 3. *For any sufficiently smooth test-function $h : \mathbb{R}^n \to \mathbb{R}^n$*

$$\frac{d}{dt}\mathbb{E}\left[h(\mathbf{X}(t)) \mid z;t\right]p(z;t) = \sum_{\mathbf{x}} h(\mathbf{X}(t)=\mathbf{x})\frac{d}{dt}p(\mathbf{x},z;t) + \sum_{\mathbf{x}} p(\mathbf{x},z;t)\frac{d}{dt}h(\mathbf{X}(t)=\mathbf{x}) \tag{B.1}$$

Proof. Consider a sufficiently smooth test function $h : \mathbb{R}^n \to \mathbb{R}^n$. First by the chain rule and the definition of conditional expectations we have the following.

$$\begin{aligned}
&\frac{d}{dt}(\mathbb{E}[h(\mathbf{X}(t)) \mid Z(t)=z])p(z;t)\\
&\qquad= \mathbb{E}\left[h(\mathbf{X}(t)) \mid Z(t)=z\right]\frac{d}{dt}p(z;t) + p(z;t)\frac{d}{dt}\mathbb{E}\left[h(\mathbf{X}(t)) \mid Z(t)=z\right]\\
&\qquad= \sum_{\mathbf{x}} h(\mathbf{X}(t)=\mathbf{x})p(\mathbf{x};t \mid z;t)\frac{d}{dt}p(z;t) + p(z;t)\frac{d}{dt}\sum_{\mathbf{x}} h(\mathbf{X}(t)=\mathbf{x})p(\mathbf{x};t \mid z;t)\\
&\qquad= \sum_{\mathbf{x}} h(\mathbf{X}(t)=\mathbf{x})p(\mathbf{x};t \mid z;t)\frac{d}{dt}p(z;t) + p(z;t)\sum_{\mathbf{x}} h(\mathbf{X}(t)=\mathbf{x})\frac{d}{dt}p(\mathbf{x},z;t)\\
&\qquad\qquad + p(\mathbf{x},z;t)\frac{d}{dt}h(\mathbf{X}(t)=\mathbf{x})
\end{aligned}$$

Second, again by the chain rule, note that

$$p(\mathbf{x};t \mid z;t)\frac{d}{dt}p(z;t) = \frac{d}{dt}p(\mathbf{x},z;t) - p(z;t)\frac{d}{dt}p(\mathbf{x};t \mid z;t)$$

Thus we get the following that concludes the proof.

$$\begin{aligned}
&\frac{d}{dt}(\mathbb{E}[h(\mathbf{X}(t)) \mid Z(t)=z])p(z;t)\\
&\qquad= \sum_{\mathbf{x}} h(\mathbf{X}(t)=\mathbf{x})\left[\frac{d}{dt}p(\mathbf{x},z;t) - p(z;t)\frac{d}{dt}p(\mathbf{x};t \mid z;t))\right]\\
&\qquad\quad + p(z;t)\sum_{\mathbf{x}} h(\mathbf{X}(t)=\mathbf{x})\frac{d}{dt}p(\mathbf{x},z;t) + p(\mathbf{x},z;t)\frac{d}{dt}h(\mathbf{X}(t)=\mathbf{x})\\
&\qquad= \sum_{\mathbf{x}} h(\mathbf{X}(t)=\mathbf{x})\frac{d}{dt}p(\mathbf{x},z;t) + \sum_{\mathbf{x}} p(\mathbf{x},z;t)\frac{d}{dt}h(\mathbf{X}(t)=\mathbf{x})
\end{aligned}$$

□

B.2 Conditional moments for the running example

Equations for probability density of the mode-switching process $Z(t)$.

$$\frac{d}{dt}p(z=0;t) = -p(z=0;t)\mu_4(0,t)r_s$$
$$\frac{d}{dt}p(z=1;t) = p(z=0;t)\mu_4(0,t)r_s$$

Equations for the first order conditonal moments (conditional means) of the population variable $\mathbf{X}(t)$.

$$p(z=0;t)\frac{d}{dt}\mu_1(0,t) = p(z=0;t)\Big[-r_s c_{14}(0,t) + \frac{1}{2}\mu_2(0,t)r_m - \mu_1(0,t)r_m\Big]$$
$$p(z=0;t)\frac{d}{dt}\mu_2(0,t) = p(z=0;t)\Big[-r_s c_{24}(0,t) - \mu_2(0,t)r_m + \mu_1(0,t)r_m + \frac{1}{2}\mu_3(0,t)r_m\Big]$$
$$p(z=0;t)\frac{d}{dt}\mu_3(0,t) = p(z=0;t)\Big[-r_s c_{34}(0,t) + \frac{1}{2}\mu_2(0,t)r_m + \mu_4(0,t)r_m - \mu_3(0,t)r_m\Big]$$
$$p(z=0;t)\frac{d}{dt}\mu_4(0,t) = p(z=0;t)\Big[-r_s c_{44}(0,t) + \frac{1}{2}\mu_3(0,t)r_m - \mu_4(0,t)r_m\Big]$$

$$p(z=1;t)\frac{d}{dt}\mu_1(1,t) = p(z=0;t)r_s\Big[c_{14}(0,t) - \mu_1(1,t)\mu_4(0,t) + \mu_1(0,t)\mu_4(0,t)r_s\Big] + p(z=1;t)\Big[-r_m\mu_1(1,t)\Big]$$
$$p(z=1;t)\frac{d}{dt}\mu_2(1,t) = p(z=0;t)r_s\Big[c_{24}(0,t) - \mu_2(1,t)\mu_4(0,t) + \mu_2(0,t)\mu_4(0,t)r_s\Big] + p(z=1;t)\Big[\mu_1(1,t)r_m - \mu_2(1,t)r_m\Big]$$
$$p(z=1;t)\frac{d}{dt}\mu_3(1,t) = p(z=0;t)r_s\Big[c_{34}(0,t) - \mu_3(1,t)\mu_4(0,t) + \mu_3(0,t)\mu_4(0,t)\Big] + p(z=1;t)\Big[\mu_2(1,t)r_m - \mu_3(1,t)r_m\Big]$$
$$p(z=1;t)\frac{d}{dt}\mu_4(1,t) = p(z=0;t)r_s\Big[c_{44}(0,t) - \mu_4(1,t)\mu_4(0,t) + \mu_4(0,t)^2\Big] + p(z=1;t)\Big[\mu_3(1,t)r_m\Big]$$

Appendix C

Filtering distribution

We are going to work with the hybrid fluid limit PDMP $(\hat{\mathbf{x}}, \hat{Z})$, introduced in Section 4.3.1, of a mode-switching population system and derive the evolution equation for the filtering distribution $\pi_t(z')$

$$\frac{d}{dt}\pi_t(z') = \frac{d}{dt}p(z';t \mid \mathbf{x}_{t^-}) \tag{C.1}$$

corresponding to the probability that the mode-switching process $\hat{Z}$ is in state z' given a sample trajectory $\mathbf{x}_{t^-}$ of the population process. As mentioned Section 5.3 of this thesis the derivation of the filtering distribution follows the same outline as given for continuous time Markov chains in [28] and is presented here mainly for completeness of presentation. The piece of background information needed in the following is how the transition densities $p(\hat{\mathbf{x}}, \hat{z}; t \mid \mathbf{x}, z; s)$ of the considered PDMPs depend on time t. This is characterised by the following forward or Fokker-Planck equation derived for example in [12].

$$\begin{aligned}\frac{\partial}{\partial t}p(\hat{\mathbf{x}}, \hat{z}; t \mid \mathbf{x}, z; s) &= \mathcal{L}^* p(\hat{\mathbf{x}}, \hat{z}; t \mid \mathbf{x}, z; s)\\ &= -\sum_i \partial_i \mathbf{F}_i^z(\mathbf{x}) p(\mathbf{x}, z'; t \mid \mathbf{x}_s, z; s)\\ &\qquad + \sum_{k, k \neq \hat{z}} \Big[p(\hat{\mathbf{x}}, k; t \mid \mathbf{x}, z; s) q_Z^{\mathbf{x}}(k, \hat{z}) - p(\hat{\mathbf{x}}, k; t \mid \mathbf{x}, z; s) q_Z^{\mathbf{x}}(\hat{z}, k)\Big]\end{aligned}$$

Evaluated at $t = s$ the above gives us

$$\mathcal{L}^* p(\hat{\mathbf{x}}, \hat{z}; s \mid \mathbf{x}, z; s) = \begin{cases} -\sum_i \partial_i \mathbf{F}_i^z(\mathbf{x}) + \sum_{k, k \neq z} \left[q_Z^{\mathbf{x}}(k, z) - q_Z^{\mathbf{x}}(z, k)\right] & \text{for } \hat{\mathbf{x}} = \mathbf{x}, \hat{z} = z \\ 0 & \text{otherwise} \end{cases}$$

The differences between the derivation in [28], that makes our case somewhat simpler, is that we are only going to consider continuous sample paths $\mathbf{x}_{t^-}$. The reason for that is given in Section 4.3.1 where we assumed that the transitions affecting the mode-switching process do no change the state of the population process directly. In order to derive the Equation C.1 we are going to consider the value of $\pi_{t+\Delta t}(z')$ and directly apply the definition of a derivative. First of all, by leveraging Bayes' rule, we get

$$\begin{aligned}\pi_{t+\Delta t}(z') &= p(z';t+\Delta t \mid \mathbf{x}_{(t+\Delta t)^-}) \\ &= \sum_z p((z';t+\Delta t)\cap(z;t) \mid \mathbf{x}_{t^-}\cap \mathbf{x}_{t+\Delta t}) \\ &= \sum_z \frac{p(z',\mathbf{x}_{t+\Delta t};t+\Delta t \mid z,\mathbf{x}_{t^-};t)\pi_t(z)}{p(\mathbf{x}_{t+\Delta t};t+\Delta t \mid \mathbf{x}_{t^-})} \\ &= \sum_z \frac{\pi_t(z)\left[p(z',\mathbf{x}_t;t \mid z,\mathbf{x}_t;t)+\Delta t\mathcal{L}^* p(z',\mathbf{x}_t;t \mid z,\mathbf{x}_t;t)+o(\Delta t)\right]}{1+\Delta t\mathbb{E}_{\hat{Z}|\mathbf{x}_{t^-}}\left[-\sum_i \partial_i \mathbf{F}_i^{\hat{Z}}(\mathbf{x})\right]+o(\Delta t)}\end{aligned} \tag{C.2}$$

The last line results from the Taylor expansion of $p(z',\mathbf{x}_{t+\Delta t};t+\Delta t \mid z,\mathbf{x}_{t^-};t)$ around t. Substituting Equation C.2 into the definition of a derivative we get

$$\begin{aligned}\frac{d}{dt}\pi_t(z') &= \lim_{\Delta t\to 0}\frac{\pi_{t+\Delta t}(z')-\pi_t(z')}{\Delta t} = \\ &= \lim_{\Delta t\to 0}\frac{1}{\Delta t}\left[\sum_z \frac{\pi_t(z)\left[p(z',\mathbf{x}_t;t \mid z,\mathbf{x}_{t^-};t)+\Delta t\mathcal{L}^* p(z',\mathbf{x}_t;t \mid z,\mathbf{x}_t;t)+o(\Delta t)\right]}{1+\Delta t\mathbb{E}_{\hat{Z}|\mathbf{x}_{t^-}}\left[-\sum_i \partial_i \mathbf{F}_i^{\hat{Z}}(\mathbf{x})\right]+o(\Delta t)}\right. \\ &\qquad \left. - \frac{\pi_t(z')\left[1+\Delta t\mathbb{E}_{\hat{Z}|\mathbf{x}_{t^-}}\left[-\sum_i \partial_i \mathbf{F}_i^{\hat{Z}}(\mathbf{x})\right]+o(\Delta t)\right]}{1+\Delta t\mathbb{E}_{\hat{Z}|\mathbf{x}_{t^-}}\left[-\sum_i \partial_i \mathbf{F}_i^{\hat{Z}}(\mathbf{x})\right]+o(\Delta t)}\right]\end{aligned} \tag{C.3}$$

Now let us concentrate on the numerator of the fraction insider the brackets of Equation C.3. First of all note that $p(z',\mathbf{x}_t;t \mid z,\mathbf{x}_{t^-};t)$ is 1 exactly when $z'=z$ and zero otherwise. In particular, we get

$$\begin{aligned}&\pi_t(z')+\sum_z \pi_t(z)\left[\Delta t\mathcal{L}^* p(z',\mathbf{x}_t;t \mid z,\mathbf{x}_t;t)+o(\Delta t)\right] \\ &\qquad\qquad -\pi_t(z')\left[1+\Delta t\mathbb{E}_{\hat{Z}|\mathbf{x}_{t^-}}\left[-\sum_i \partial_i \mathbf{F}_i^{\hat{Z}}(\mathbf{x})\right]+o(\Delta t)\right] \\ &= \sum_z \pi_t(z)\left[\Delta t\mathcal{L}^* p(z',\mathbf{x}_t;t \mid z,\mathbf{x}_t;t)+o(\Delta t)\right] \\ &\qquad\qquad -\pi_t(z')\left[\Delta t\mathbb{E}_{\hat{Z}|\mathbf{x}_{t^-}}\left[-\sum_i \partial_i \mathbf{F}_i^{\hat{Z}}(\mathbf{x})\right]+o(\Delta t)\right]\end{aligned}$$

The above gives the numerator in Equation C.3. Multiplying the numerator by $\frac{1}{\Delta t}$ and taking the limit $\Delta t\to 0$ in both the numerator and denominator of Equation C.3 then gives us

$$\begin{aligned}\frac{d}{dt}\pi_t(z') &= \sum_z \pi_t(z)\mathcal{L}^* p(z',\mathbf{x}_t;t \mid z,\mathbf{x}_t;t) - \pi_t(z')\mathbb{E}_{\hat{Z}|\mathbf{x}_{t^-}}\left[\sum_i \partial_i \mathbf{F}_i^{\hat{Z}}(\mathbf{x})\right] \\ &= \sum_z \pi_t(z)q(z,z') - \pi_t(z')\left[\sum_i \partial_i \mathbf{F}_i^{\hat{z}}(\mathbf{x}) - \mathbb{E}_{\hat{Z}|\mathbf{x}_{t^-}}\left[\sum_i \partial_i \mathbf{F}_i^{\hat{Z}}(\mathbf{x})\right]\right]\end{aligned}$$

as claimed in Section 5.3.

Appendix D

Carma-C model of policy synthesis example

The purpose of this section is to give an extended description of the example Carma-C model considered in Chapter 6. The only component in the model is one corresponding to the robots. We start off by defining the local stores γ_l of the robot components. The local stores hold the current location of the robot (loc), set of known target (target) and the success probability of navigation towards the target (succp).

$$\begin{aligned}\gamma_l = \{&\mathsf{loc} \mapsto \{\{(x,y)\}\},\\ &\mathsf{target} \mapsto \{\emptyset\}\\ &\mathsf{succp} \mapsto [0,1],\\ &\}\end{aligned}$$

Next, we set up the processes defining the behaviour of robots along with the relevant store updates.

$$\begin{aligned}\mathit{Explore} \overset{\mathsf{def}}{=} &[\pi_r]\mathit{random}^*[\circ]\langle\circ\rangle\{\mathsf{loc} \mapsto \{\{R(x,y)\}\}\\ &+ [\pi_d]\mathit{directed}^*[\circ]\langle\circ\rangle\{\mathsf{loc} \mapsto \{\{D(x,y)\}\}\\ &+ [\pi_r]\mathit{sense}^*[\circ]\langle M(x,y)\rangle\{\mathsf{target} \mapsto \{\emptyset \cup \{M(x,y)\}\}\\ \mathit{Listen} \overset{\mathsf{def}}{=} &[\pi_r]\mathit{sense}^*[\circ]((x,y))\{\mathsf{target} \mapsto \{\emptyset \cup \{(x,y)\}\}\\ \mathit{Robot} \overset{\mathsf{def}}{=} &\mathit{Explore} \parallel \mathit{Listen}\end{aligned}$$

The broadcast actions *random** and *directed** model a random walk and a directed walk on a graph structure respectively. This is achieved through the use of updates defined via functions R, D as described in Chapter 6. To recall, random function R maps the current location of the robot uniformly to any of the locations directly connected to it. The robot's location is updated to the singleton set containing the resulting value. The function D maps the current location to a connected location that is also closer to the target location $(1,1)$ with probability p corresponding to the attribute succp. With probability $1-p$ any of the other directly connected locations is used. Again the resulting singleton set is assigned as the value domain of location in the update. The broadcast output action *sense** models a robot gaining information about the target location and broadcasting to the collective. The helper function M maps the target location $(1,1)$ to $(1,1)$ and any other location to the empty set. Finally, the corresponding input action models the robot being able to receive the broadcast containing a known target locations.

The guards are evaluated up to a chosen control action f in the following way.

$$\pi_r = \begin{cases} \mathit{true} & \text{if } f(\gamma)(\mathsf{target}) = \emptyset \\ \mathit{false} & \text{otherwise} \end{cases}$$

$$\pi_d = \begin{cases} \mathit{true} & \text{if } f(\gamma)(\mathsf{target}) = \{(1,1)\} \\ \mathit{false} & \text{otherwise} \end{cases}$$

Thus, the guards make sure that when the target location is known to the robots they perform a directed walk towards the target while random walk is performed otherwise.

Components corresponding to the robots are defined as a pair composed of process description and a store.

$$\mathit{RobotComp} \overset{\mathsf{def}}{=} (\mathit{Robot}, \gamma_l)$$

Collective is defined as a composition of N robot components.

$$\mathit{Swarm} \overset{\mathsf{def}}{=} \mathit{RobotComp}[N]$$

Global store γ_g defines the attributes mover and senser in terms of their value domains.

$$\begin{aligned}\gamma_g = \{&\mathsf{mover} \mapsto [0,\infty), \\ &\mathsf{senser} \mapsto [0,\infty), \\ &\}\end{aligned}$$

Evolution rule is given in terms of resolved store values, process state of the collective and current time. Suppose f denotes the action chosen at time t and let γ_s and γ_r denote the sender and receiver store respectively. Firstly, we define

$$\mu_p(f(\gamma_s), f(\gamma_s), \mathit{sense}^*) = 1 \qquad \text{for all stores } \gamma_s \text{ and } \gamma_r$$

In particular, a broadcast message is received with probability 1 by all eligible receivers. There are no unicast actions in the model so the definition of μ_w is trivial. Supposing $f(\gamma)(\mathsf{mover}) = r_m$, $f(\gamma)(\mathsf{senser}) = r_s$ and a some fixed constant (chosen to be 0.7 in the example in Chapter 6 we say that the rates of the actions are given as follows.

$$\mu_r(f(\gamma), \mathit{move}^*) = r_m \times e^{-a\times\frac{x}{N}} \qquad \text{for all local stores } \gamma$$

$$\mu_r(f(\gamma), \mathit{sense}^*) = \begin{cases} r_s & \text{for all local stores } \gamma \text{ such that } f(\gamma)(\mathsf{loc}) = (1,1) \\ 0 & \text{otherwise} \end{cases}$$

The global store definitions do as well as the composition of the collective do not change so μ_u is again trivial completing the description of the Carma-C model.