Spatial and stochastic equivalence relations for PALOMA

Paul Piho

Master of Science by Research
CDT in Pervasive Parallelism
School of Informatics
University of Edinburgh
2016

Abstract

This dissertation studies behavioural equivalences of a recent stochastic process algebra PALOMA, which features primitives for both unicast and broadcast communication. PALOMA is motivated by modelling of spatially distributed systems and thus the components are parametrised by a location which affects their capability to communicate with each other.

In this work we give a new agent-based semantics for PALOMA in terms of the State to Function Labelled Transition System framework. This semantics complements the existing population-level semantics by allowing analysis of components at the level of individual agents.

We also propose several bisimulation-like equivalence relations with respect to isometric transformations of space that allow us to compare PALOMA models with respect to their relative rather than absolute locations. The properties of the equivalence relations are studied to see whether they have the desirable property of defining a congruence relation on PALOMA components. The defined bisimulations support the idea of being able to substitute one component, perhaps with a more efficient implementation, for another within a given system even though they may not exhibit exactly the same behaviour in arbitrary contexts.

Acknowledgements

I would like to thank my supervisor, Jane Hillston, for her encouragement, good advice and regular and frequent feedback.

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

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.

This dissertation includes content from the following published paper:

> Paul Piho and Jane Hillston. 'Stochastic and Spatial Equivalences for PALOMA'. in: *Proceedings of the Workshop on FORmal methods for the quantitative Evaluation of Collective Adaptive SysTems*. 2016

(*Paul Piho*)

Contents

List of Figures

Chapter 1

Introduction

Stochastic process algebras are an appealing modelling paradigm in performance and dependability analysis of concurrent systems due to the formally defined compositionality which allows for describing the full system as a collection of interacting components. When considering such formal models it is useful to know when two models can be considered equivalent or approximate each other with respect to some observable behaviour of the system. This allows for reasoning and model simplification up to a defined equivalence relation. This project studies process equivalence in the recent stochastic process algebra PALOMA (Process Algebra for Located Markovian Agents) [9, 10].

PALOMA captures the behaviour of agents that are distributed in space. The interactions of agents are affected by the relative positions of the agents in a system. This idea reflects many modern systems where, for example, the range of communication may be limited for devices using wireless communication technologies or some areas may be known “dead zones" from which no communication is possible. In this project we analyse what it means for PALOMA systems to be equivalent taking into account both their behaviour and location, and develop formal foundations so that such equivalences can be analysed.

The stochastic process algebras, including PALOMA, take the discrete state space view of the system and thus share the same problem as other discrete event modelling paradigms. Namely, that for realistic models of real-life systems the size of the state space can get too large to be computationally tractable. Thus, in the case of PALOMA the model simplification we are interested in relates to state space reduction of the generated mathematical model.

We will see that applying the usual notions of Markovian bisimulation for stochastic process algebras to equivalence checking in PALOMA is too strong, leaving little opportunity for a notion of equivalence that is not isomorphism. The approach taken here is to consider equivalence of components within the context of a given system. The aim is to devise an equivalence that supports being able to substitute one component, with a more efficient implementation, for another within a given system.

This dissertation makes the following contributions. Firstly, it equips the PALOMA language with new agent-based semantics in the FuTS framework for deriving the underlying mathematical model in the form of a continuous time Markov chain. This gives a rigorous foundation for defining the equivalence relations. Secondly, we define and analyse some bisimulation-like equivalences where we make use of the isometric transformations of metric spaces in order to consider the relative rather than absolute locations of PALOMA components.

The rest of the dissertation is structured as follows. In Chapter 2 we give a brief overview of the related background material and related work.

Chapter 3 is devoted to an in-depth introduction to PALOMA and laying down necessary formal ground-work to be used in later chapters. Firstly we will give an overview of the PALOMA syntax and an informal description of the language syntax features. In Section 3.2 we define conditional exit rates for components from their syntactic definitions. Section 3.3 reproduces the agent-based operational semantics for PALOMA in the FuTSs framework that were published in [23]. The rules give formal underpinnings for deriving the mathematical model for evolution of PALOMA components. The remainder of the chapter describes properties and an interpretation of the semantic rules via examples.

In Chapter 4 we define two sub-languages of PALOMA and equip them with bisimulation-based equivalence relations. We discuss some difficulties with extending those ideas to PALOMA.

In Chapter 5 we elaborate on our attempt to define bisimulation-like equivalence relations for PALOMA that we presented in [23]. We start with a rather naive way of defining a bisimulation and use examples and counter-examples to support strengthening the condition for our definitions. The aim of this chapter is to arrive at a bisimulation definition that gives rise to a weakened congruence relation on PALOMA component such that we are able to swap components in a system with bisimilar ones leaving the behaviour of the system as a whole unchanged up to the defined bisimilarity.

Finally, the findings and results are summarised in Chapter 6 with some possible directions for further work.

Chapter 2

Background

This chapter is dedicated to exposition of the background material and work related to equivalence theory of spatial stochastic process algebras.

2.1 Process algebras

Process algebras were proposed as a formal way to study concurrency in computer science in the 1980s and are well-established in modelling and reasoning about functional aspects, like reachability and deadlock behaviour, of systems. They allow relatively simple high level syntactic descriptions of systems and automation of the model construction using underlying formal semantic rules. The earlier and most well known examples of this are CCS (Calculus of Communicating Systems) by Milner [21], CSP (Communicating Sequential Processes) by Hoare [16] and ACP (Algebra of Communicating Processes) by Bergstra and Klop [2].

In general syntax of process algebras consists of a set of uninterpreted action names and basic operators (e.g. choice) to describe processes. In addition, they usually feature some communication combinators (e.g. parallel composition) to express concurrent behaviour of described processes.

The stochastic process algebras, like PEPA [15] and EMPA [3], extend these ideas by including a way to specify rates at which actions complete. Stochastic process algebras have been used to capture performance and quality of service properties of systems [11, 4, 26] as well as in modelling of biological system [19].

2.2 Operational semantics

The semantics to the process algebras are usually given via variants of structured operational semantics introduced by Plotkin in [24] which consist of rules in the form

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

These sets of rules give a rigorous meaning to the operators in the language and will allow for the construction of the mathematical model describing a system to be automated.

The operational semantic rules usually give rise to a labelled transition system (LTS) which is a set of triples $s_1 \xrightarrow{a} s_2$ denoting that there is a transition labelled a from the state s_1 of the system to the state s_2. The LTSs have a natural graphical representation as labelled directed graphs. An example of a simple transition system is given in Figure 2.1.

$$\begin{tikzcd}
idle \arrow[d, bend left, "accept\ job"] \\
busy \arrow[u, bend left, "complete\ job"]
\end{tikzcd}$$

Figure 2.1: Simple transition system.

In the case of stochastic process algebras the labels usually include the information about the rate parameters interpreted as the expected time needed to complete the action. Figure 2.2 gives a simple example of a transition system for a server that in state *idle* accepts jobs at rate λ and completes them at rate μ in state *busy*. In, PEPA for

$$\begin{tikzcd}
idle \arrow[d, bend left, "{accept\ job, \lambda}"] \\
busy \arrow[u, bend left, "{complete\ job, \mu}"]
\end{tikzcd}$$

Figure 2.2: Simple server with rates.

example, the rates λ and μ would be taken to be the parameters of exponential distribution meaning that the evolution of the described system in time can be modelled as a continuous-time Markov chain (CTMC).

2.2.1 FuTS framework

For PALOMA we are going to give the operational semantics using the FuTS (functional transition systems) framework presented in [7].

In general, the transition rules in FuTSs are given as triplets $s \xrightarrow{\lambda} f$ where s denotes a source state, λ the label of the transition and f the continuation function associating a value of suitable type to each state s'. The shorthand $[s_1 \mapsto v_1, \cdots, s_n \mapsto v_n]$ is used to denote a function f such that $f(s_i) = v_i$ for $i = 1, \cdots n$.

FuTS was chosen over Plotkin style structured operational semantics as the functional treatment of transition rules allowed us to give more concise definitions of semantic rules as many possible branches of model evolutions could be captured within a single rule.

2.3 Spatial representation

The ways of representing in model of real-life systems will fall into one of the following categories:

Discrete: The model will assume a number of distinct locations. The topology of the space is given through adjacency relations. That means we can think of space being represented by a graph (possibly directed and weighted).

Continuous: The locations are defined over a continuous space. A common example would be three-dimensional real space $\mathbb{R}^3$ which can be interpreted as an exact representation of real physical space.

From the domain of process algebras we have, for example, SpacePi [17] where locations are defined as continuous variables in real coordinate spaces. For wireless networks there is, for example, CWS [20] which makes no restrictions on the notion of location that can be used. A thorough overview of spatial representations in modelling from the point of view of collective adaptive systems can be found in [12, 13].

2.4 Bisimulation

Most process algebras allow for model comparison through some equivalence relations. These are usually based on some observational characteristic and thus the definitions one decides to give depend on how much information is assumed to be observable for each relation.

For classic process algebras, for example, there is the notion of bisimilarity. The observables, in this case, are taken to be actions the systems can perform. If two systems can always match each other's actions, then the systems are bisimilar.

For probabilistic process algebras, the situation is complicated by the addition of information for defining a probabilistic process. A common way of defining an equivalence relation on stochastic process algebras is akin to probabilistic bisimulation defined by Larsen and Skou in [18]. This idea relies, again, on considering actions as observables but also takes into account the probability or rate associated with each action. The described equivalence relations are not the only possible ones – different ways of defining observable behaviour will give rise to different equivalence relations.

The equivalence relations that are of particular interest are the ones that are also congruence relations – meaning that they are preserved by the operators of the language. An equivalence relation $\equiv$ is a congruence relation of a process algebra if for any $P \equiv Q$ and $K \equiv L$, one has $P \star K \equiv Q \star L$, where $\star$ is any binary relation of the algebra. This way one can consider observationally equivalent models by swapping in equivalent subsystems.

2.5 PALOMA

The process algebra under study in this project was PALOMA which belongs to the class of stochastic process algebras motivated by modelling of collective adaptive systems. In PALOMA the location is considered a primary feature of an agent which allows us to explicitly describe the spatial structure of a system. The model descriptions assume a discrete set of locations and each agent in the system is parametrised by a single location which can change in time. The spatial structure affects how components interact with each other.

Unlike CSP and PEPA, PALOMA does not feature direct communication or cooperation between agents. Instead, components affect each other's behaviour via message-passing through broadcast and unicast communication. Population-level operational semantics which give rise to a continuous-time Markov chain on the population counts of components at each defined location were presented in [10]. These semantics facilitate analysis through fluid approximations of the behaviour of systems.

2.6 Related work

There exist several process algebras targeting broadcasting communication systems, for example A Process Algebra for Wireless Mesh Networks introduced by Fehkner, et al. in [8] and Calculus of Broadcasting Systems (CBS) introduced by Prasad in [25].

CBS, for example, is a CCS-like process algebra where communication between agents happens through broadcasting – an agent can broadcast a message and every node instantaneously hears the message. Multicasting and point-to-point communication is achieved by adding a list of intended recipients to the broadcast message. Even though the agents in the system do not have explicit physical locations, the local communication is achieved through specifying which message can be heard by which agents. In [25] the author considered strong bisimulation and testing equivalence for CBS while [14] gives a thorough treatment of bisimulation equivalences of CBS. The broadcast-only sublanguage of PALOMA could be seen as a stochastic extension of CBS.

SpacePi [17] is an extension of π-calculus that takes its motivation from systems biology and deals with spatial dependecy of interactions. Similarly to the Real Space Process Algebra [1], in order to be able to describe continuous movement in space, SpacePi adds the explicit notion of time to the language. This means that the underlying mathematical model is very different to PALOMA where time is expressed implicity through exponentially distributed rates of actions.

Another example of a recent π-calculus based process algebra is 3π proposed in [5] where evolutions of geometric structures can be described. The dynamic geometric behaviour of processes in 3π is expressed through affine transformations. The equivalence theory for 3π, developed in [5], states that all processes in 3π remain equivalent under rigid body transformations. In the current dissertation a similar idea is studied from the perspective of the stochastic process algebra PALOMA.

There has not been much work in terms of equational theory in the framework of stochastic process algebras characterised by explicit spatial structure. Unicast and broadcast communication primitives provide a natural basis where dependencies of communication on spatial structure can be studied.

Chapter 3

Process algebra for located Markovian agents

In this chapter we give an introduction to PALOMA as described in [9, 10]. The new agent-based semantic rules will be presented in Section 3.3 for generating the continuous-time Markov chain model. Section 3.2 will be spent on defining conditional exit rates straight from the syntax to serve as the basis for the equivalence relations in Chapter 5.

The syntactic constructs in PALOMA are motivated by modelling, for example, scenarios of disease spread, performance of bike sharing systems, and wireless sensor networks. As already mentioned, PALOMA has primitives for both broadcast and unicast communication. The broadcast communication works as one would expect: the same message is sent to everyone within range.

The motivation for unicast is to allow modelling of contention for resources. For example, if there is a driver looking for a parking spot then, even though there may be several places available, he is going use just one of them. This can be modelled in PALOMA by taking the available parking spots to be unicast receivers and the driver, a unicast transmitter. Each parking spot would be given a weight representing whether it is preferred to others in the range and thus the competition between otherwise identical spots can be solved probabilistically. Successful unicast communication between a parking spot and a driver would represent the driver having parked the car.

3.1 Informal description

The spatial distribution of agents is a key feature of PALOMA models and we assume that there exists a finite set of locations, denoted *Loc*, and all agent expressions in PALOMA are parametrised by a location $\ell \in Loc$, indicating the current location of the agent.

The grammar of the language is as follows:

$$\pi ::= !!(\alpha,r)@\mathbf{IR}\{\vec{\ell}\} \;\mid\; ??(\alpha,p)@\mathbf{Wt}\{w\} \;\mid\; !(\alpha,r)@\mathbf{IR}\{\vec{\ell}\} \;\mid\; ?(\alpha,p)@\mathbf{Pr}\{q\} \;\mid\; (\alpha,r)$$
$$S(\ell) ::= \pi.S'(\ell') \;\mid\; S_1(\ell)+S_2(\ell) \;\mid\; C$$
$$P ::= S(\ell) \;\mid\; P \,\|\, P$$

The two-level grammar, defines individual agents $S(\ell)$, whose behaviours are specified by the actions they can undertake, with possible alternatives, and model components P, which are comprised of parallel compositions of agents. The behaviour of individual agents are given by actions of five distinct types:

Unicast output $!!(\alpha,r)@\mathbf{IR}\{\vec{\ell}\}$**:** Unicast is for point-to-point communication between a pair of agents and is included in the language to model contention for resources in systems. Each unicast output message has a label, α, and a rate r, that determines the rate at which the output is performed. The message is sent to locations specified by the set $\vec{\ell} \in 2^{Loc}$ interpreted as the *influence range*. Any agent located within that range, which enables the corresponding α-labelled unicast input action, is eligible to receive the action — that is, the label α is used to identify agents that can communicate with each other. Unicast actions are *blocking* meaning that the sending agent can only proceed when there is an eligible receiver.

Unicast input $??(\alpha,p)@\mathbf{Wt}\{w\}$**:** Each eligible receiver of a unicast message α must be located within the specified influence range, and each will have an associated *weight* w. The weights are used to define a probability distribution over the eligible receivers, i.e. if there are i potential receivers, each with weight w_i and $W = \sum_i w_i$ then the probability that the jth agent receives the message is w_j/W. Once the message is received the receiving agent may or may not act on the message (reflecting message failure, corruption etc.) with the specified probability p i.e. with probability $1-p$ the agent will not act on the message received. If this occurs then the message is lost — it is not the case that it is subsequently assigned to one of the other eligible receivers.

Broadcast output $!(\alpha,r)@\,\mathbf{IR}\{\overrightarrow{\ell}\}$**:** As its name suggests, a broadcast action allows its sender to influence multiple other agents. As with the unicast output action, a broadcast output message labelled α is sent with a specified influence range $\overrightarrow{\ell}$ and at a specified rate r. *All* agents with broadcast input prefix on label α located within that range may receive the message. Moreover, the output proceeds regardless of whether there are any eligible receivers so broadcast output is non-blocking for the sender.

Broadcast input $?(\alpha,p)@\,\mathbf{Pr}\{q\}$**:** Each eligible receiver of a broadcast message α must be located within the specified input range. Each such agent has a likelihood of receiving the message, recorded in the probability q. For example, agents closer to the sender may be more likely to receive the message. Each agent independently decides whether the broadcast is received or not (Bernoulli trials). As with unicast input, the receiving agent may or may not act on the message with the specified probability p i.e. with probability $1-p$ the agent will not act on the message received.

Spontaneous action (α,r)**:** These actions do not represent a communication but rather an individual action by the agent which may change the state of the agent, for example, its location. These can also be thought of as broadcast output actions whose influence range is the empty set.

The additional combinators given in the definition of the syntax are used to combine actions to define behaviours of processes. The following combinators are defined for PALOMA:

Prefix Provides a mechanism for describing ordered behaviour of a sequential component. The notation $(\alpha,r).S(\ell)$ says that after the component has performed the spontaneous action labelled α with expected rate r it continues behaving as component $S(\ell)$.

Choice The component $S_1(\ell)+S_2(\ell)$ describes a system which can behave as either $S_1(\ell)$ or $S_2(\ell)$. The choice is resolved by the race policy — the choice expression is going to behave as the first component to complete its actions while the other component in the expression is discarded.

Parallel The parallel composition $P \,||\, Q$ denotes that P and Q are acting concurrently. PALOMA does not feature co-operation between components — interaction between component happens through message passing via actions described previously.

Constant The expression $X \overset{\text{def}}{=} P$ gives the constant X the behaviour of process P.

All rates are assumed to be parameters of exponential distributions, meaning that the underlying stochastic model of a PALOMA model is a continuous time Markov chain (CTMC).

Example 1. Consider agents $Transmitter$ and $Receiver$ such that

$$\begin{aligned} Receiver(\ell) &:= ??(message, p) @ \mathbf{Wt}\{v\}.Receiver(\ell) \\ Transmitter(\ell) &:= !!(message, r) @ \mathbf{IR}\{\overrightarrow{\ell}\}.Transmitter(\ell) \end{aligned}$$

where ℓ denotes the current location of the agent and $\overrightarrow{\ell}$ denotes a set of locations in the range of the unicast message emitted by action *message*. In a system where no agent sends a *message* agent *Receiver* does not perform any action. On the other hand if there is a component, say $Tranmitter$, that outputs a *message* and the location of *Receiver* is in the influence range of the message then *Receiver* performs *message* with a rate dependent on the rate at which $Transmitter$ unicasts *message* and the probability that *Receiver* receives it. Similarly, if the component $Transmitter$ does not have a recipient for the *message*, it remains blocked and never performs an action.

Alternatively, we could consider broadcast communication instead. That is, define the components *Receiver* and $Transmitter$ in the following way:

$$\begin{aligned} Receiver(\ell) &:= ?(message, p) @ \mathbf{Pr}\{p\}.Receiver(\ell) \\ Transmitter(\ell) &:= !(message, r) @ \mathbf{IR}\{\overrightarrow{\ell}\}.Transmitter(\ell) \end{aligned}$$

The behaviour of $Transmitter$ in this case is always the same irrespective of the context it appears in. Here we are not going to check whether there are eligible receivers in the system. The action !*message* fires at a constant rate given by the rate parameter r. Hence, as mentioned before, the broadcast output is as treated a non-blocking action.

3.2 Conditional exit rates

Notions of equivalence in process algebras, such as bisimulation [21], are typically based on the idea of a pair of agents each being able to match the behaviour of the other. In the case of stochastic process algebras such as PEPA, not only the type of actions but also the rates at which they occur must in some sense be matched [15]. In order to make similar definitions for PALOMA we need to define some auxiliary functions which, given a syntactic expression, extract information about the rates and probabilities which may be exhibited by the term.

Denote the set of all sequential components of PALOMA parametrised by their location by $\mathscr{C}_S$ and the set of model components by $\mathscr{C}$. Let the set of action labels be defined as *Lab* and the set of action types as $\mathit{Type} = \{!!, ??, !, ?, \cdot\}$, where the interpretation of the symbols is clear, corresponding to the action types discussed above. Let *Act* denote the set of all actions. The actions in the set $\mathit{Act} = \mathit{Type} \times \mathit{Lab}$ are defined by their label and their type. Let $\mathscr{A}$ be the set of all syntactically defined actions. Define the function $\Pi_{\mathit{Act}} : \mathscr{A} \to \mathit{Act}$ as a projection returning the label of the action with its type, e.g. $\Pi_{\mathit{Act}}(??(\alpha, p)@\mathbf{Wt}\{v\}) = ??\alpha$. Similarly define the projection $\Pi_{\mathit{Lab}} : \mathit{Act} \to \mathit{Lab}$ returning just the label of the action and the function $\Pi_{\mathit{Type}} : \mathit{Act} \to \mathit{Type}$ returning the type of an action.

Denote by Π_{Loc} the function returning the set of locations spanned by a model component.

$$\Pi_{\mathit{Loc}}(S_1(\ell_1) \parallel \cdots \parallel S_n(\ell_n)) = \bigcup_{i=1}^{n} \{\ell_i\}$$

Note that in the case of sequential components Π_{Loc} will result in a singleton set — the location of the sequential component.

Suppose $\mathit{Sys} = S_1(\ell_1) \parallel \cdots \parallel S_n(\ell_n) \in \mathscr{C}$ for $n \in \mathbb{N}^+$. Let the function seq return the set of all sequential components of *Sys* in a set of locations *L*.

$$\mathrm{seq}(\mathit{Sys}, L) = \{S_i(\ell_i) \mid \Pi_{\mathit{Loc}}(S_i(\ell_i)) \in L\}$$

When we consider a PALOMA component in isolation we can use the syntax to find the potential rate, weight or probability associated with this component and a given action. Similar functions are defined for each form of prefix.

The simplest case to deal with is the spontaneous actions without message emission.

Definition 1. For all $\alpha \in \mathit{Lab}$, $a, b \in \mathscr{A}$ and $S(\ell) \in \mathscr{C}_S$ define the function s_α returning the rate of a spontaneous action **without** message emission labelled α as follows.

$$\begin{aligned}
s_\alpha((\beta,r).S(\ell)) &= \begin{cases} r & \text{for } \alpha=\beta \\ 0 & \text{otherwise} \end{cases} \\
s_\alpha(a.S(\ell)) &= 0 \quad \text{if } \Pi_{Type}(a) \neq \cdot \\
s_\alpha(S_1(\ell)+S_2(\ell)) &= s_\alpha(S_1(\ell))+s_\alpha(S_2(\ell))
\end{aligned}$$

3.2.1 Context unaware definitions: unicast

From the point of view of the originator of a unicast action, the important measure is the rate at which the action is preformed.

Definition 2. For all $\alpha \in Lab$, $a \in \mathscr{A}$, $\vec{\ell} \in 2^{Loc}$, and $S \in \mathscr{C}_S$ define the function $s^{!!}_\alpha$ returning the rate of a unicast output action labelled α as follows.

$$\begin{aligned}
s^{!!}_\alpha\left(!!(\beta,r)@\mathbf{IR}\{\vec{\ell}\}.S(\ell)\right) &= \begin{cases} r & \text{for } \alpha=\beta \\ 0 & \text{otherwise} \end{cases} \\
s^{!!}_\alpha(a.S(\ell)) &= 0 \quad \text{if } \Pi_{Type}(a) \neq !! \\
s^{!!}_\alpha(S_1(\ell)+S_2(\ell)) &= s^{!!}_\alpha(S_1(\ell))+s^{!!}_\alpha(S_2(\ell))
\end{aligned}$$

Example 2. Consider the following components

$$\begin{aligned}
Tester(\ell_0) &:= (message,r).Tester(\ell_0) \\
Transmitter(\ell_0) &:= !!(message,r).Transmitter(\ell_0) \\
Receiver(\ell_1) &:= ??(message,p)@\mathbf{Wt}\{v\}.Receiver(\ell_1)
\end{aligned}$$

Based on these definitions we can find:

$$\begin{aligned}
&s^{!!}_{message}(Tester(\ell_0)+Transmitter(\ell_0)) = 0+r=r \\
&s^{!!}_{message}(Receiver(\ell_1)) = 0
\end{aligned}$$

The rest of the context unaware definitions are given in a similar vein and just extract necessary syntactic information from the component definitions.

Definition 3. For all $\alpha \in Lab$, $a \in \mathscr{A}$, $S(\ell) \in \mathscr{C}_S$ define the function $p^{??}_\alpha$ returning the probability of reacting to a received unicast message as

$$\begin{aligned}
p^{??}_\alpha(??(\beta,p)@\mathbf{Wt}\{w\}.S(\ell)) &= \begin{cases} p & \text{for } \alpha=\beta \\ 0 & \text{otherwise} \end{cases} \\
p^{??}_\alpha(a.S(\ell)) &= 0 \quad \text{if } \Pi_{Type}(a) \neq ?? \\
p^{??}_\alpha(S_1(\ell)+S_2(\ell)) &= p^{??}_\alpha(S_1(\ell))+p^{??}_\alpha(S_2(\ell))
\end{aligned}$$

Definition 4. Define the function Π_{UniIR} returning the set of locations in the influence range of unicast action α. That is

$$\begin{array}{lcl}\Pi_{UniIR}(!!(\beta,r)@\,\mathbf{IR}\{\overrightarrow{\ell}\}.S(\ell),\alpha) = \begin{cases}\overrightarrow{\ell} & \text{if } \alpha=\beta\\ 0 & \text{otherwise}\end{cases} \\ \Pi_{UniIR}(a.S(\ell),\alpha) \quad = \quad \emptyset \quad \text{if } \Pi_{Type}(a)\neq\,!!\end{array}$$

That is, given that a sequential component has a unicast output prefix with label α, the function returns the influence range of unicast message α defined in the prefix. Otherwise, the function returns the empty set $\emptyset$.

Definition 5. For all $\alpha \in Lab$, $a \in \mathscr{A}$, $S(\ell) \in \mathscr{C}_S$ define the weight function in the following way.

$$\begin{array}{lcl}w_\alpha(??(\beta,r)@\,\mathbf{Wt}\{w\}.S(\ell)) = \begin{cases}w & \text{for } \alpha=\beta\\ 0 & \text{otherwise}\end{cases} \\ w_\alpha(a.S(\ell)) & = & 0 \quad \text{if } \Pi_{Type}(a)\neq\,??\\ w_\alpha(S_1(\ell)+S_2(\ell)) & = & w_\alpha(S_1(\ell))+w_\alpha(S_2(\ell))\end{array}$$

For all model components $P = S_1(\ell_1) \parallel \cdots \parallel S_n(\ell_n)$ with $S_i(\ell_i) \in \mathscr{C}_S$ for all $1 \leq i \leq n$ ($n \in \mathbb{N}$) define the weight function as

$$w_\alpha(P) = \sum_{i=1}^{n}(w_\alpha(S_i(\ell_i)))$$

Similarly suppose that C is a finite set of components and define

$$w_\alpha(C) = \sum_{S\in C}(w_\alpha(S)))$$

Example 3. Consider the following sequential components.

$$\begin{array}{lcl}Transmitter(\ell_0) & := & !!(message,r)@\,\mathbf{IR}\{\overrightarrow{\ell}\}.Transmitter(\ell_0)\\ Receiver1(\ell_1) & := & ??(message,p)@\,\mathbf{Wt}\{w_{r1}\}.Receiver1(\ell_1)\\ Receiver2(\ell_2) & := & ??(message,q)@\,\mathbf{Wt}\{w_{r2}\}.Receiver2(\ell_2)\end{array}$$

For the system given by $Sys = Transmitter(\ell_0) \parallel Receiver1(\ell_1) \parallel Receiver2(\ell_2)$ the weight for receiving a unicast message *message* is calculated as

$$w_{message}(Sys) = w_{message}(Transmitter(\ell_0) \parallel Receiver1(\ell_1) \parallel Receiver2(\ell_2)) = w_{r1}+w_{r2}$$

3.2.2 Context unaware definitions: broadcast

Definition 6. For all $\alpha \in Lab$, $a \in \mathscr{A}$, $\vec{\ell} \in 2^{Loc}$, and $S(\ell) \in \mathscr{C}_S$ define the function returning the rate of broadcast of a sequential component in the following way

$$
\begin{aligned}
s^{!}_{\alpha}(!(\beta,r)@\,\mathbf{IR}\{\vec{\ell}\}.S(\ell)) &= \begin{cases} r & \text{for } \alpha=\beta \\ 0 & \text{otherwise} \end{cases} \\
s^{!}_{\alpha}(a.S(\ell)) &= 0 \quad \text{if } \Pi_{Type}(a) \neq\, ! \\
s^{!}_{\alpha}(S_1(\ell)+S_2(\ell)) &= s^{!}_{\alpha}(S_1(\ell)) + s^{!}_{\alpha}(S_2(\ell))
\end{aligned}
$$

Definition 7. For all $\alpha \in Lab$, $a \in \mathscr{A}$, $\vec{\ell} \in 2^{Loc}$, and $S(\ell) \in \mathscr{C}_S$ define the function returning the probability of firing an action after receiving a broadcast message labelled α as follows.

$$
\begin{aligned}
p^{?}_{\alpha}(?(\beta,p)@\,\mathbf{Pr}\{q\}.S(\ell)) &= \begin{cases} pq & \text{for } \alpha=\beta \\ 0 & \text{otherwise} \end{cases} \\
p^{?}_{\alpha}(a.S(\ell)) &= 0 \quad \text{if } \Pi_{Type}(a) \neq\, ? \\
p^{?}_{\alpha}(S_1(\ell)+S_2(\ell)) &= p^{?}_{\alpha}(S_1(\ell)) + p^{?}_{\alpha}(S_2(\ell))
\end{aligned}
$$

The assumption is made that receiving a broadcast message and having it induce an action are independent – we denote the probability of both of the events happening by $p^{?}_{\alpha}$.

Definition 8. Define the function Π_{BrIR} returning the set of locations in the influence range of broadcast action α.

$$
\begin{aligned}
\Pi_{BrIR}\left(!(\beta,r)@\,\mathbf{IR}\{\vec{\ell}\}.S(\ell),\alpha\right) &= \begin{cases} \vec{\ell} & \text{if } \alpha=\beta \\ 0 & \text{otherwise} \end{cases} \\
\Pi_{BrIR}(a.S(\ell),\alpha) &= \emptyset \quad \text{if } \Pi_{Type}(a) \neq\, !
\end{aligned}
$$

3.2.3 Context-aware conditional exit rates

Unfortunately the syntactic information alone is not sufficient to determine the rate at which an action will be witnessed in a PALOMA system. The spatial aspect, as captured by the influence range, plays an important role in determining both which actions are possible and potentially their rates and probabilities. Thus we also define some context-dependent functions.

Definition 9. Let α be an action label in *Lab*. Let $S(\ell) \in \mathscr{C}_S$ be a sequential component. Define the rate at which a component is capable of unicasting a message labelled α to a location ℓ' as follows:

$$\begin{aligned} u_\alpha(\ell', !!(\beta, r)@\mathbf{IR}\{\vec{\ell}\}.S(\ell)) &= \begin{cases} s^{!!}_{\alpha}(!!(\beta, r)@\mathbf{IR}\{\vec{\ell}\}.S(\ell)) & \text{if } \ell' \in \vec{\ell} \text{ and } \alpha = \beta \\ 0 & \text{otherwise} \end{cases} \\ u_\alpha(\ell', S_1(\ell) + S_2(\ell)) &= u_\alpha(\ell', S_1(\ell)) + u_\alpha(\ell', S_2(\ell)) \end{aligned}$$

Definition 10. Suppose $P = S_1(\ell_1) \,\|\, \cdots \,\|\, S_n(\ell_n) \in \mathscr{C}$ for $n \in \mathbb{N}^+$ is a model component with $S_i(\ell_i) \in \mathscr{C}_S$ for all $1 \leq i \leq n$. Let *Sys* be any other system serving as context. Let $u_\alpha(\ell, Sys, P)$ be the rate at which a model component P unicasts a message labelled α to location ℓ in the context of *Sys*, defined as

$$u_\alpha(\ell, Sys, P) = \sum_{S \in \text{seq}(P)} u_\alpha(\ell, S) \times \mathbb{1}_{>0}\{w_\alpha(\text{seq}(Sys \,\|\, P, \Pi_{UniIR}(S, \alpha)))\}$$

$$\text{where } \mathbb{1}_{>0}(x) = \begin{cases} 1 & x > 0 \\ 0 & \text{otherwise} \end{cases}$$

For each sequential component S of P we calculate the total weight over the components in the influence range of S. The indicator function $\mathbb{1}_{>0}$ is set to 1 if this weight is greater than 0 — meaning there are eligible receivers in the influence range. The rate at which P unicasts a message α to location ℓ is then defined as the sum of rates at which each sequential component S of P is *capable* of said unicast multiplied by the indicator function ensuring that the blocking nature of unicast is taken into account.

The next definition deals with determining the probability of a sequential component receiving the unicast message.

Definition 11. Let $S_1(\ell)$ and $S_2(\ell')$ be sequential components and $Sys \in \mathscr{C}$ any model component. Suppose $!!(\alpha, r)@\mathbf{IR}\{\vec{\ell}\}.S_2'(\ell'')$ is a prefix guarded term in the expression of $S_2(\ell')$. Then we define the probability of $S_1(\ell)$ receiving a unicast message with label α from $S_2(\ell')$, when composed in parallel with *Sys* and $S_2(\ell')$, to be:

$$p_\alpha(S_1(\ell), Sys, S_2(\ell')) = \begin{cases} \frac{w_\alpha(S_1(\ell))}{w_\alpha(\text{seq}(Sys \| S_1(\ell), \vec{\ell}))} & \text{if } \ell \in \vec{\ell} \\ 0 & \text{otherwise} \end{cases}$$

Similar definitions are given for broadcast and spontaneous actions.

Definition 12. Let $a \in \mathscr{A}$. Let $S(\ell) \in \mathscr{C}_S$ be a sequential component. Define the rate at which a component is capable of broadcasting a message labelled α to a location ℓ' as follows:

$$\begin{aligned} b_\alpha(\ell', !(\beta,r)@\mathbf{IR}\{\vec{\ell}\}.S(\ell)) &= \begin{cases} s^!_\alpha(!(\beta,r)@\mathbf{IR}\{\vec{\ell}\}.S(\ell)) & \text{if } \ell' \in \vec{\ell} \text{ and } \alpha = \beta \\ 0 & \text{otherwise} \end{cases} \\ b_\alpha(\ell', S_1(\ell) + S_2(\ell)) &= b_\alpha(\ell', S_1(\ell)) + b_\alpha(\ell', S_2(\ell)) \end{aligned}$$

Now let $P = S_1(\ell_1) \parallel \cdots \parallel S_n(\ell_n)$ to be a model component with $S_i(\ell_i) \in \mathscr{C}_S$ for all $1 \leq i \leq n$ $(n \in \mathbb{N})$. Define

$$b_\alpha(\ell, P) = \sum_{i=1}^{n} b_\alpha(\ell, S_i(\ell_i))$$

Finally we have set up all the necessary definitions in order to be able to define the conditional exits rates of components.

Definition 13. Let $Sys \in \mathscr{C}$ be a system in which the component $S \in \mathscr{C}_S$ appears. Let $a \in \mathscr{A}$ be any action with label α. Define the *context-aware exit rate R* for agents by the following:

$$R_a(Sys, S(\ell_0)) = \begin{cases} s_\alpha(S(\ell_0)) & \text{if } \Pi_{Type}(a) = \cdot \\ b_\alpha(S(\ell_0)) & \text{if } \Pi_{Type}(a) = ! \\ b_\alpha(\ell_0, Sys) p^?_\alpha(S(\ell_0)) & \text{if } \Pi_{Type}(a) = ? \\ \max\limits_{\ell \in \Pi_{Loc}(Sys)} \{u_\alpha(\ell, Sys, S(\ell_0))\} & \text{if } \Pi_{Type}(a) = !! \\ \sum\limits_{T \in Seq(Sys)} u_\alpha(\ell_0, Sys, T) p_\alpha(S(\ell_0), Sys, T) p^{??}_\alpha(S(\ell_0)) & \text{if } \Pi_{Type}(a) = ?? \end{cases}$$

Now consider a model component $P = S_1(\ell_1) \parallel \cdots \parallel S_n(\ell_n)$ with $S_i(\ell_i) \in \mathscr{C}_S$ for all $1 \leq i \leq n$ $(n \in \mathbb{N})$ and suppose it is a part of the system Sys. Then define

$$R_a(Sys, P) = \sum_{i=1}^{n} R_a(Sys \parallel (P \setminus S_i(\ell_i)), S_i(\ell_i))$$

where $P \setminus S_i(\ell_i)$ denotes the model component P with $S_i(\ell_i)$ removed.

The first three rates given for spontaneous actions and broadcast communication in Definition 13 are clear. For the unicast ouput we have to take into account that it should be interpreted as a blocking action. Note that the rate at which the message is broadcast does not depend on how many listeners there are in the system. This means that the rate which a component sends out a unicast message is either equal to the syntactically defined rate if there are eligible listeners in the influence range of the message or zero if there are no eligible listeners in the influence range. The definition of the observed rate for unicast output takes the maximum of the rate at which the component unicasts a message to a location in the system. Note that if there are no eligible listeners of a unicast message in the system then by Definition 10 all the calculated u_α are 0 and so is the observed rate. This results in the blocking nature of unicast output being treated correctly.

The corresponding unicast input action definition takes into account the output actions from all sources for which the component is in the influence range.

Finally we define the rate at which action $a \in Act$ is performed over a set of locations.

Definition 14. Consider a model component $P = S_1(\ell_1) \parallel \cdots \parallel S_n(\ell_n)$ with $S_i(\ell_i) \in \mathscr{C}_S$ for all $1 \leq i \leq n$ ($n \in \mathbb{N}$) and suppose it is a part of the system *Sys*. Let L be a set of locations of interest. We define $R_a(L, Sys, P)$, the rate at which action a is performed by P in locations L, within the context of system *Sys* to be:

$$R_a(L, Sys, P) = \sum_{S \in \mathrm{seq}(P,L)} R_a(Sys \parallel (P \setminus S), S)$$

3.3 Agent-based operational semantics

This section is devoted to presenting the new agent-based semantic rules for the PALOMA language that will serve as a basis for analysis of the possible equivalence relations in the later chapters. The definition of the semantics of PALOMA will proceed in the FuTS (State to Function Labelled Transition Systems) framework as already introduced in Chapter 2.

In the case of PALOMA semantics we are going to define the set of states as the set of all model components $\mathscr{C}$. For convenience, the treatment of semantic rules is split into two steps where the following types of transition relations are considered separately:

Capability relation Denoted by $s \overset{\lambda}{\rightarrowtail}_c f$ where $f : \mathscr{C} \rightarrow [0, 1]$. The aim is to describe actions that a defined model component is capable of and introduce probabilities for all possible states resulting from the said action firing. For example, a component including a prefix for unicast input will be capable of the unicast input action firing with some probability dependent on the context. The function f will assign a probability for possible continuation states.

Stochastic relation Denoted by $s \xrightarrow{\lambda}_s f$ where $f : \mathscr{C} \to \mathbb{R}^{\geq 0}$. These rules are used to generate the CTMC and thus need to assign rates to each available transition.

As we have already seen, the calculation of rates of actions for each component depend the system they appear in (a PALOMA model component) and thus in this section we use *Sys* as a place-holder for any such PALOMA model component that contains the component under consideration. That is the rules are given for PALOMA components assuming they are part of *Sys*. In the following, we use $P_1 \equiv P_2$ to denote that model components P_1 and P_2 are syntactically equivalent.

3.3.1 Capability relations

The only capability relations of interest here are ones for broadcast and unicast input actions as these are the only ones that can either succeed or fail depending on the rest of the system *Sys*.

The labels λ_c, of the FuTSs rules are given by the following grammar where $\alpha \in$ Lab denotes the action labels:

$$\begin{aligned} \lambda_c ::= \;& (?\alpha, \vec{\ell}, Sys) && \text{Broadcast input} \\ \mid\; & (??\alpha, \vec{\ell}, Sys) && \text{Unicast input} \end{aligned}$$

The semantic rules given in Figure 3.1 use the definitions from Section 3.2 to extract necessary information from the syntactic definitions of components.

The rules BrIn and UniIn are the primitive rules describing the capability of sequential components to perform a broadcast or unicast input action, respectively, given the set of locations $\vec{\ell}$ denoting the influence range of the message and the complete system *Sys*. In both cases the function f, which is defined over all states, gives the probability of a transition to each state given the action has fired. For BrIn the calculation only depends on the parameters p and q given explicitly in the syntactic definition of the component. For UniIn the likelihood of the component receiving the message, $\frac{w}{w_\alpha(Sys)}$, is calculated on the basis that there may be many eligible receivers of the given message in *Sys*.

The rule BrSystem is used to deal with parallel compositions of model components that can act as broadcast message receivers. Note that the outcomes of all the broadcast input actions in a system are independent of each other. Thus the probability of $P_1 \parallel P_2$ transitioning to $P_1' \parallel P_2'$ due to a broadcast input action is the product of the probabilities of P_1 and P_2 respectively making the corresponding transitions.

BrIn $$?(\alpha,p)@\mathbf{Pr}\{q\}.S \overset{(?\alpha,\vec{\ell},Sys)}{\rightarrowtail}_c f \qquad \text{if } \Pi_{Loc}\,(?(\alpha,p)@\mathbf{Pr}\{q\}.S) \in \vec{\ell}$$

$$\text{where } f(s) = \begin{cases} pq & \text{if } s \equiv S \\ 1-pq & \text{if } s \equiv ?(\alpha,p)@\mathbf{Pr}\{q\}.S \\ 0 & \text{otherwise} \end{cases}$$

UniIn $$??(\alpha,p)@\mathbf{Wt}\{w\}.S \overset{(??\alpha,\vec{\ell},Sys)}{\rightarrowtail}_c f \qquad \text{if } \Pi_{Loc}\,(??(\alpha,p)@\mathbf{Wt}\{w\}.S) \in \vec{\ell}$$

$$\text{where } f(s) = \begin{cases} \dfrac{wp}{w_\alpha(Seq)} & \text{if } s \equiv S \\ \dfrac{w(1-p)}{w_\alpha(Seq)} & \text{if } s \equiv ??(\alpha,p)@\mathbf{Wt}\{w\}.S \\ 0 & \text{otherwise} \end{cases}$$

$$Seq = \text{seq}(Sys,\, \vec{\ell}\,)$$

BrSystem $$\frac{P_1 \overset{(?\alpha,\vec{\ell},Sys)}{\rightarrowtail}_c f_1 \quad P_2 \overset{(?\alpha,\vec{\ell},Sys)}{\rightarrowtail}_c f_2}{P_1 \parallel P_2 \overset{(?\alpha,\vec{\ell},Sys)}{\rightarrowtail}_c g}$$

$$\text{where } g(s) = \begin{cases} f_1(P_1')f_2(P_2') & \text{if } s \equiv P_1' \parallel P_2' \\ 0 & \text{otherwise} \end{cases}$$

ParllelUniIn $$\frac{S_1 \overset{(??\alpha,\vec{\ell},Sys)}{\rightarrowtail}_c f_1 \quad S_2 \overset{(??\alpha,\vec{\ell},Sys)}{\rightarrowtail}_c f_2}{S_1 \parallel S_2 \overset{(??\alpha,\vec{\ell},Sys)}{\rightarrowtail}_c g}$$

$$\text{where } g(s) = \begin{cases} f_1(S_1') & \text{if } s \equiv S_1' \parallel S_2 \\ f_2(S_2') & \text{if } s \equiv S_1 \parallel S_2' \\ 0 & \text{otherwise} \end{cases}$$

Choice $$\frac{P_1 \overset{\lambda_c}{\rightarrowtail}_c f}{P_1 + P_2 \overset{\lambda_c}{\rightarrowtail}_c f} \qquad\qquad \frac{P_2 \overset{\lambda_c}{\rightarrowtail}_c f}{P_1 + P_2 \overset{\lambda_c}{\rightarrowtail}_c f}$$

Constant $$\frac{P \overset{\lambda_c}{\rightarrowtail}_c f \qquad X \overset{\text{def}}{=} P}{X \overset{\lambda_c}{\rightarrowtail}_c f}$$

Figure 3.1: Capability rules for communication

For unicast input actions the rule ParllelUniIn is just saying that no two components can perform the unicast input on the same label simultaneously.

3.3.2 Stochastic relations

Firstly we need to define a set of labels for stochastic relations. It will be necessary to carry around the set of locations $\vec{\ell}$ in the labels to distinguish between actions having the same label and type but affecting a different set of components due to their influence range. In addition, including the system *Sys* in the labels ensures that the communication rules are only applied to components in the same system. The set of labels for stochastic relations is thus defined as follows:

$$\begin{array}{rll}\lambda_s ::= & (\alpha,\ \emptyset,\ Sys) & \text{Spontaneous action}\\ & \mid (!\alpha,\ \vec{\ell}, Sys) & \text{Broadcast communication}\\ & \mid (!!\alpha, \vec{\ell}, Sys) & \text{Unicast communication}\end{array}$$

The stochastic rules are summarised in Figure 3.2. Firstly we have rules Br, Uni and SpAct that just define the primitive rules for all spontaneous actions and give the rates at which the defined transitions can happen.

For the rule Uni the side-condition is needed to ensure that there are eligible receivers available in the system.

The rules BrCombo and UniPair are to combine the capability rules with stochastic rules to give rates of system state transitions that are induced by broadcast or unicast message passing. BrCombo takes as premise the existence of components S and P such that S can perform the broadcast communication action defined by stochastic relations and P is capable of broadcast input. The rate at which the parallel composition $S \parallel P$ reaches the next state $S' \parallel P'$ is given by the function $f \otimes g$ which is defined as the product of f applied to the S' and g applied to P. The unicast case is treated similarly.

Note. Suppose we have a component defined as follows:

$$S \stackrel{\text{def}}{=} ??(\alpha, p)@\,\mathbf{Wt}\{w\}.S$$

For such a recursively defined component communication success and failure result in syntactically equivalent components. Strictly speaking this would make the defined continuation functions one-to-many but from the semantics point of view we are still going to treat $??(\alpha, p)@Wt\{w\}.S$ and S as distinct terms.

Br $\quad !(\alpha,r)@\mathbf{IR}\{\vec{\ell}\}.S \xrightarrow{(!\alpha,\vec{\ell},Sys)}_{s} [S \mapsto r]$

Uni $\quad !!(\alpha,r)@\mathbf{IR}\{\vec{\ell}\}.S \xrightarrow{(!!\alpha,\vec{\ell},Sys)}_{s} [S \mapsto r]$

if there exists S such that $S \xrightarrow{(??\alpha,\vec{\ell},Sys)}_{c} f$

SpAct $\quad (\alpha,r).S \xrightarrow{(\alpha,\emptyset,Sys)}_{s} [S \mapsto r]$

(a) Primitive rules

BrCombo
$$\frac{S \xrightarrow{(!\alpha,\vec{\ell},Sys)}_{s} f \quad P \xrightarrow{(?\alpha,\vec{\ell},Sys)}_{c} g}{S \parallel P \xrightarrow{(!\alpha,\vec{\ell},Sys)}_{s} f \otimes g}$$

$$(f \otimes g)(s) = \begin{cases} f(S')g(P') & \text{if } s \equiv S' \parallel P' \\ 0 & \text{otherwise} \end{cases}$$

UniPair
$$\frac{S \xrightarrow{(!!\alpha,\vec{\ell},Sys)}_{s} f \quad P \xrightarrow{(??\alpha,\vec{\ell},Sys)}_{c} g}{S \parallel P \xrightarrow{(!!\alpha,\vec{\ell},Sys)}_{s} f \otimes g}$$

$$(f \otimes g)(s) = \begin{cases} f(S')g(P') & \text{if } s \equiv S' \parallel P' \\ 0 & \text{otherwise} \end{cases}$$

(b) Combining with capabilities

Parallel
$$\frac{P_1 \xrightarrow{\lambda_s}_{s} f}{P_1 \parallel P_2 \xrightarrow{\lambda_s}_{s} f \otimes Id} \qquad (f \otimes Id)(s) = \begin{cases} f(P_1') & \text{if } s \equiv P_1' \parallel P_2 \\ 0 & \text{otherwise} \end{cases}$$

$$\frac{P_2 \xrightarrow{\lambda_s}_{s} f}{P_1 \parallel P_2 \xrightarrow{\lambda_s}_{s} Id \otimes f} \qquad (Id \otimes f)(s) = \begin{cases} f(P_2') & \text{if } s \equiv P_1 \parallel P_2' \\ 0 & \text{otherwise} \end{cases}$$

Choice
$$\frac{P_1 \xrightarrow{\lambda_s}_{s} f \qquad P_2 \xrightarrow{\lambda_s}_{s} f}{P_1 + P_2 \xrightarrow{\lambda_s}_{s} P_1 + P_2 \xrightarrow{\lambda_s}_{s} f}$$

Constant
$$\frac{P \xrightarrow{\lambda_s}_{s} f \qquad X := P}{X \xrightarrow{\lambda_s}_{s} f}$$

(c) Rules for composition

Figure 3.2: Stochastic rules for rates

3.4 Properties of semantics

The aim of the described semantics is to capture the behaviour of a system. We start by giving the definition for a derivative of a component.

Definition 15. If $P \overset{\lambda}{\rightarrowtail}_s f$ and $P' \in \mathscr{C}$ such that $f(P') = r \neq 0$ then P' is a one-step derivative of P via λ, denoted $P \xrightarrow{\lambda,r} P'$.

This allows us to analyse the reachable states of any given system by defining the derivative set of the component. We denote by $\Lambda(Sys)$ the set of all stochastic transition labels λ that correspond to the system Sys.

Definition 16. Define the derivative set of a PALOMA system component Sys $ds(Sys)$, as the smallest set of components such that

1. if $Sys_0 \stackrel{\text{def}}{=} Sys$ then $Sys_0 \in ds(Sys)$.
2. if $Sys_i \in ds(Sys)$ and there exists $\lambda \in \Lambda(Sys_i)$ and $r > 0$ such that $Sys_i \xrightarrow{\lambda,r} Sys_j$ then $Sys_j \in ds(Sys)$.

From that we are going to define the derivation graphs as was done for PEPA in [15].

Definition 17. Given a PALOMA system Sys and its derivative set $ds(Sys)$ we define the derivation graph $\mathscr{D}(Sys)$ as the labelled multigraph with the set of nodes given by $ds(Sys)$ and the set of arcs A defined by

- $A \subset ds(Sys) \times ds(Sys) \times \Lambda \times \mathbb{R}^{>0}$
- $(s, s', \lambda, r) \in A$ if there exists $s \xrightarrow{\lambda,r} s'$ such that $\lambda \in \Lambda(s)$.

Example 4. In this example we will go through constructing the derivation graph for a simple system. Consider the following system:

$$\begin{aligned}
Receiver(\ell) &\stackrel{\text{def}}{=} ??(message, p)@\,\mathbf{Wt}\{w\}.Receiver(\ell)\\
Transmitter(\ell) &\stackrel{\text{def}}{=} !!(message, r)@\,\mathbf{IR}\{\ell\}.(process, q)(\ell).Transmitter(\ell)\\
Sys &\stackrel{\text{def}}{=} Receiver(\ell) \parallel Transmitter(\ell)
\end{aligned}$$

Firstly, we apply the capability relation rules exhaustively to Sys. In particular, applying the rule UniIn to $Receiver(\ell)$ we get the following:

$$??(message, p)@\,\mathbf{Wt}\{w\}.Receiver(\ell) \overset{(??message,\, \overrightarrow{\ell},\, Sys)}{\rightarrowtail}_c$$
$$[Receiver(\ell) \mapsto p,\ ??(message, p)@\,\mathbf{Wt}\{w\}.Receiver(\ell) \mapsto 1 - p]$$

As there are no other agents with input prefixes then we have already exhaustively applied the capability relation rules and we can move to the stochastic relations. Applying the stochastic rule Uni to $Transmitter(\ell)$ gives

$$!!(message,r)@\mathbf{IR}\{\ell\}.(process,q)(\ell).Transmitter(\ell) \xrightarrow{(!!message,\{\ell\},Sys)}_s$$
$$[(process,q).Transmitter(\ell) \mapsto r]$$

Finally, we pair up the unicast sender and the unicast receiver using the rules UniPair. This results in the following transition:

$$Transmitter(\ell) \,||\, Receiver(\ell) \xrightarrow{(!!message,\{\ell\},Sys)}_s f$$

where, according to the defined semantic rules, we have

$$f(s) = \begin{cases} rp & \text{if } s \equiv (process,q)(\ell).Transmitter(\ell) \,||\, Receiver(\ell) \\ r(1-p) & \text{if } s \equiv Transmitter(\ell) \,||\, Receiver(\ell) \\ 0 & \text{otherwise} \end{cases}$$

This gives us that only two states are reachable from $Sys \equiv Transmitter(\ell) \,||\, Receiver(\ell)$, namely

$$Transmitter(\ell) \,||\, Receiver(\ell) \equiv Sys$$
$$(process,q)(\ell).Transmitter(\ell) \,||\, Receiver(\ell) \equiv Sys'$$

From Sys' we can only get back to Sys by noting that the only applicable stochastic rule is SpAct giving

$$(process,q)(\ell).Transmitter(\ell) \,||\, Receiver(\ell) \xrightarrow{(process,\emptyset,Sys')}_s f$$

where

$$f(s) = \begin{cases} q & \text{if } s \equiv Sys \\ 0 & \text{otherwise} \end{cases}$$

Thus by the above reasoning we get the derivative set $ds(Sys) = \{Sys, Sys'\}$. The associated derivation graph $\mathscr{D}(Sys)$ will then have the following arcs.

$$\begin{aligned} A = \{&(Sys, Sys', (!!message, \{\ell\}, Sys), rp) \\ &(Sys', Sys, (process, \emptyset, Sys'), q) \\ &(Sys, Sys, (!!message, \{\ell\}, Sys), r(p-1))\} \end{aligned}$$

The graphical representation of the derivation graph $\mathscr{D}(Sys)$ is given below as Figure 3.3.

$$\begin{tikzcd}
Transmitter(\ell) \parallel Receiver(\ell) \arrow[loop left, "(!!message,\{\ell\},Sys),r(1-p)"] \arrow[r, bend left, "(!!message,\{\ell\},Sys),rp"] & (process,q)(\ell).Transmitter(\ell) \parallel Receiver(\ell) \arrow[l, bend left, "(process,\emptyset,Sys'),q"]
\end{tikzcd}$$

Figure 3.3: The graphical representation of the derivation graph.

3.5 Markov model

As mentioned the purpose of the given semantic rules is to provide a framework from which the mathematical model in the form of a CTMC can be rigorously derived for any PALOMA model. We already have the required foundations, namely the definition of derivation graph from the FuTS transition system, in place and thus the only thing left to do is to relate a given derivation graph to a CTMC.

Informally, the procedure used to derive a CTMC from the derivation graph is obvious with rates of different transitions from the same origin state to the same end state added together and the transitions inducing self-jumps of components in the derivative set forgotten.

Example (Example 4 revisited)**.** The derivation graph gives rise to a two state CTMC with the following generator matrix Q

$$Q = \begin{pmatrix} -rp & rp \\ q & -q \end{pmatrix}$$

Conventional methods of analysing CTMCs can be used to then extract performance characteristics of the described system. For example, one can find the steady-state distribution π of the system described by Q as a solution to

$$\pi Q = 0$$

In such a small example the steady-state distirbution is easy to find analytically and turns out to be

$$\pi = \left[\frac{q}{rp+q}, \frac{rp}{rp+q}\right]$$

From this we can derive some perfomance measures, for example the utilisation of the receiver in the system. We can say that the receiving component is utilised when it is processing a message — that is the utilisation of $Receiver(\ell)$ is simply the steady-state probability of the system being in state $(process,q).Transmitter(\ell) \parallel Receiver(\ell)$. In our case this is $\frac{rp}{rp+q}$.

3.6 Context

We have noted that the behaviour of a model component P depends on the context it appears in. In the following we are going to define what is meant by "P in the context of Sys". In our case P appearing in the context of Sys will simply mean that the behaviour of P is characterised by the behaviour of the parallel composition $Sys \,||\, P$ or any reordering of the terms as reordering leaves the behaviour of $Sys \,||\, P$ invariant.

We are going to denote the transitions of a component P within the context of a system Sys by $P \xrightarrow{a,Sys} P'$ where $a \in Act$. In terms of semantic rules introduced in Section 3.3 we are going to say that $P \xrightarrow{a,Sys} P'$ holds if there is a stochastic transition $P \rightarrowtail^{\lambda_s}_{s} f$ and a component P' such that $f(P') \neq 0$. In addition the label λ_s is required to be such that

$$\lambda_s = \begin{cases} (\alpha, \emptyset, Sys \,||\, P) & \text{if } a = \alpha \\ (!\alpha, \vec{\ell}, Sys \,||\, P) & \text{if } a = ?\alpha \vee !\alpha \\ (!!\alpha, \vec{\ell}, Sys \,||\, P) & \text{if } a = ??\alpha \vee !!\alpha \end{cases}$$

For a an input action, we also require the existence of a corresponding capability relation with label

$$\lambda_c = \begin{cases} (?\alpha, \vec{\ell}, Sys \,||\, P) & \text{if } a = ?\alpha \\ (??\alpha, \vec{\ell}, Sys \,||\, P) & \text{if } a = ??\alpha \end{cases}$$

That is, the label needs to indicate that P is taken to be in the context of system Sys.

If we are interested in the behaviour of model component Sys on its own we consider it in the context of an empty system denoted $\emptyset$. For completeness we are going to define the syntactic equivalence $Sys \,||\, \emptyset \equiv Sys$.

Chapter 4

Equivalences of PALOMA sub-languages

This chapter consists of some preliminary work on the analysis of the equivalence theory of PALOMA from the perspective of two unicast-based sub-languages. In Section 4.1 we define a sub-language of PALOMA where we only consider unicast communication with no location attributes. We give a definition of a bisimulation-like relation for the said sub-language and give a working example of how it can be used for model reduction.

Section 4.2 expands the sub-language to include the location parameter. We will see that the treatment of bisimulation equivalences in Section 4.2 is too discriminating when the locations of components are taken into account — namely it turns out that finding bisimilar components becomes very difficult if two components are considered equivalent only if they exhibit equivalent behaviour in all possible contexts.

4.1 Unicast-only sub-language *without* locations

In the following we consider the sub-language where only unicast communication is enabled. In addition we are going to assume that all components are within the influence range of each other — we can think of all communication happening with influence range defined as $\mathbf{IR}\{all\}$ or alternatively that all components are at the same location with communication happening with influence range $\mathbf{IR}\{local\}$.

The sub-language under consideration is defined by the following grammar.

$$\pi ::= !!(\alpha, r) \ \mid \ ??(\alpha, r)@\mathbf{Wt}\{v\} \ \mid \ (\alpha, r)$$
$$S ::= \pi.S' \ \mid \ S_1 + S_2 \ \mid \ C$$
$$P ::= S \ \mid \ P \,\|\, P$$

Figure 4.1: Grammar for unicast-only sub-language of PALOMA, PALOMA-uo

Let us call the sub-language given in Figure 4.1 PALOMA-uo (for unicast only). The most obvious way to define a bisimulation on PALOMA-uo components would be the following: two components are equivalent if for any considered context system they perform the same set of actions with equal rates with the same holding for the derivatives of components. This is formally given in the definition below.

Definition 18. A binary relation $\mathscr{R}^{[1]}$ is a bisimulation over model components if, and only if, $(P,Q) \in \mathscr{R}^{[1]}$ implies for all $a \in Act$ and all system components Sys.

1. $R_a(Sys, P) = R_a(Sys, Q)$.
2. $P \xrightarrow{a,Sys} P'$ implies there exists Q', $Q \xrightarrow{a,Sys} Q'$ and $(P', Q') \in \mathscr{R}^{[1]}$.
3. $Q \xrightarrow{a,Sys} Q'$ implies there exists P', $P \xrightarrow{a,Sys} P'$ and $(P', Q') \in \mathscr{R}^{[1]}$.

Example 5. As a simple example consider the following components

$$\begin{aligned} Transmitter &\stackrel{\text{def}}{=} !!(message, r).Transmitter \\ Receiver_0 &\stackrel{\text{def}}{=} ??(message, p)@\mathbf{Wt}\{1\}.Receiver_0 \\ Receiver_1 &\stackrel{\text{def}}{=} ??(message, p)@\mathbf{Wt}\{2\}.Receiver_1 \end{aligned}$$

As we can see the difference between $Receiver_0$ and $Receiver_1$ is only in the weight parameter. If we consider the systems $Transmitter \,\|\, Receiver_0$ and $Transmitter \,\|\, Receiver_1$ then both systems and in particular $Receiver_0$ and $Receiver_1$ exhibit the same behaviour. On the other hand we can consider the following scenarios:

$$\begin{aligned} Scenario_0 &\stackrel{\text{def}}{=} Transmitter \,\|\, Receiver_0 \,\|\, Receiver_1 \\ Scenario_1 &\stackrel{\text{def}}{=} Transmitter \,\|\, Receiver_0 \,\|\, Receiver_0 \end{aligned}$$

That is we are considering the components $Receiver_0$ and $Receiver_1$ in the context of the system $Sys \stackrel{\text{def}}{=} Transmitter \,\|\, Receiver_0$. The observed rates of $Receiver_0$ and $Reciever_1$ performing the unicast input action on label $message$ will differ as $Receiver_1$

has the priority in $Scenario_0$ due to greater weight parameter. In $Scenario_1$ on the other hand the two identical receivers $Receiver_0$ are equally likely to receive the unicast message. In particular the probability of $Receiver_1$ getting the preference as the receiver of the unicast message labelled *message* in $Scenario_0$ is $\frac{2}{3}$ (weight of $Receiver_1$ over the total weight of the receivers in the system). Similarly the probability of $Receiver_0$ begin preferred in $Scenario_1$ is $\frac{1}{2}$. Thus the bisimulation 18 differentiates between $Reciever_0$ and $Receiver_1$.

The example above shows that although the the components $Receiver_0$ and $Receiver_1$ have equivalent behaviour thoughout the evolutions of the systems $Transmitter \,||\, Receiver_0$ and $Transmitter \,||\, Receiver_1$ they are not considered bisimilar. This is unnecessarily strong as when modelling a system we are not going to be interested in the behaviour of a component in all possible contexts but rather in the context of the system we are modelling. This is one of the ideas we are going to be dealing with in Chapter 5.

Let us assume we can define the bisimilarity in the standard way as the largest bisimulation as given by Definition 18. The following example is going to demonstrate that although intuitively we might consider the above bisimulation for PALOMA-uo too strong it can still be used to reduce models up to a bisimilarity.

Example 6. We consider the following system:

$$\begin{aligned}
Transmitter &\overset{\text{def}}{=} !!(message_1, r).Transmitter' + !!(message_2, r).Transmitter' \\
Transmitter' &\overset{\text{def}}{=} !!(message\_nonsense, r).Transmitter' \\
Receiver_1 &\overset{\text{def}}{=} ??(message_1, 1) @ \mathbf{Wt}\{1\}.(process, q).Receiver_1 \\
Receiver_2 &\overset{\text{def}}{=} ??(message_2, 1) @ \mathbf{Wt}\{1\}.(process, q).Receiver_2 \\
Sys &\overset{\text{def}}{=} Transmitter \,||\, Receiver_1 \,||\, Receiver_2
\end{aligned}$$

In the system *Sys* given above we have a component *Transmitter*, transmitting messages $message_1$ and $message_2$, composed in parallel with $Receiver_1$ expecting message $message_1$ and $Receiver_2$ expecting message $message_2$. All communication in this case happens with no failure. Once $Receiver_1$ or $Receiver_2$ get their respective messages they process the message, represented by the spontaneous action *process*, and return to the state where they are ready to accept another message. After *Transmitter* has sent out a message it arrives at a state *Transmitter'* where it starts sending out a *message_empty* which none of the components in the system is expecting. This means that the system *Sys* will reach an absorbing state where no transitions are possible.

$$\begin{tikzcd}
 & Transmitter' \,\|\, (process, q).Receiver_1 \,\|\, Receiver_2 \arrow[dr, "process{,}\, q"] & \\
Transmitter \,\|\, Receiver_1 \,\|\, Receiver_2 \arrow[ur, "!!message_1{,}\, r"] \arrow[dr, "!!message_2{,}\, r"'] & & Transmitter' \,\|\, Receiver_1 \,\|\, Receiver_2 \\
 & Transmitter' \,\|\, Receiver_1 \,\|\, (process, q).Receiver_2 \arrow[ur, "process{,}\, q"'] &
\end{tikzcd}$$

Figure 4.2: Derivation graph of *Sys* in Example 6

We can give an alternative implementation of the two receivers in the system in the following way:

$$\begin{aligned}AltReceiver \overset{\text{def}}{=} & ??(message_1, 1) @ \mathbf{Wt}\{1\}.(process, q).AltReceiver \\ & + ??(message_2, 1) @ \mathbf{Wt}\{1\}.(process, q).AltReceiver\end{aligned}$$

AltReceiver and $Receiver_1 \,\|\, Receiver_2$ being bisimilar according to Definition 18 is immediate. Namely, adding components transmitting or receiving messages $message_1$ or $message_2$ is going to scale the rates at which *AltReceiver* and $Receiver_1 \,\|\, Receiver_2$ perform the actions in the same way. The derivatives are also always going to be bisimilar as the behaviours of

$$(process, q).Receiver_1,\ (process, q).Receiver_2,\ \text{and}\ (process, q).AltReceiver$$

are not going to depend on the rest of the system.

Now consider the following system where the parallel composition of $Receiver_1$ and $Receiver_2$ is substituted with the new implementation *AltReceiver*.

$$AltSys \overset{\text{def}}{=} Transmitter \,\|\, AltReceiver$$

The aim is to show that *AltSys* is bisimilar to *Sys*. First note that the set of actions available from both *Sys* and *AltSys* are $\{!!message_1, !!message_2, ??message_1, ??message_2\}$. All of the observed rates are equal to r due to the chosen parameters.

The behaviours of *Sys* and *AltSys* are represented in Figures 4.2 and 4.3 in terms of the derivation graphs. This shows that the action *process* firing at rate q is available from one-step derivatives of *Sys* and *AltSys* with no other actions happening. Thus we can say that *Sys* and *AltSys* are bisimilar with respect to Definition 18. The benefit of using *AltReceiver* instead of *Receiver* is that the state space generated collapses to three states from four.

$$\begin{tikzcd}
Transmitter \,\|\, AltReceiver \arrow[r, bend left, "!!message_1{,}r"] \arrow[r, bend right, "!!message_2{,}r"'] & Transmitter' \,\|\, (process,q).AltReceiver \arrow[r, bend left, "process{,}q"] & Transmitter' \,\|\, AltReceiver
\end{tikzcd}$$

Figure 4.3: Derivation graph of *AltSys* in Example 6

Note that in general verifying the bisimulation of Definition 18 is not a trivial task as it needs reasoning about an infinite set of possible context systems. In addition we are not generally interested in whether the components behave equivalently in all the possible contexts. Instead the interest lies in whether in the given system we can replace certain components with ones that have more efficient representation in terms of the size of the generated derivation graph. This becomes more apparent in the next section when components with locations are considered and the analogously defined bisimulation will be too discriminating.

4.2 Unicast-only sub-language *with* locations

In this section we are going to put back the location parameter in the component definitions. The sub-language, called PALOMA-uowl, we are going to work with is defined by the following grammar.

$$\begin{aligned}
\pi &::= !!(\alpha,r)@\mathbf{IR}\{\vec{\ell}\} \;\mid\; ??(\alpha,r)@\mathbf{Wt}\{v\} \;\mid\; (\alpha,r)\\
S(\ell) &::= \pi.S'(\ell') \;\mid\; S_1(\ell)+S_2(\ell) \;\mid\; C\\
P &::= S(\ell) \;\mid\; P \,\|\, P
\end{aligned}$$

Figure 4.4: Grammar for unicast-only sub-language of PALOMA with locations, PALOMA-uowl

We can define an equivalence relation on PALOMA-uowl in exactly the same way as in Section 4.1 where model components are defined by the grammar in Figure 4.4.

Definition 19. A binary relation $\mathscr{R}^{[1*]}$ is a bisimulation over model components if, and only if, $(P,Q) \in \mathscr{R}^{[1*]}$ implies for all $a \in Act$ and all system components Sys.

1. $R_a(Sys,P) = R_a(Sys,Q)$.
2. $P \xrightarrow{a,\,Sys} P'$ implies there exists Q', $Q \xrightarrow{a,\,Sys} Q'$ and $(P',Q') \in \mathscr{R}^{[1*]}$.

3. $Q \xrightarrow{a, Sys} Q'$ implies there exists P', $P \xrightarrow{a, Sys} P'$ and $(P', Q') \in \mathscr{R}^{[1*]}$.

Definition 19 checks whether two components have equivalent behaviour with respect to all context systems. This gives rise to a very strict equivalence on PALOMA components as can be seen from the following example.

Example 7. Consider the component $Receiver_1(\ell_1) \parallel Receiver_2(\ell_1)$ from Example 6 but this time parametrised by location ℓ_1. The transmitting component now needs an influence range $\overrightarrow{\ell}$ in its definition

$$\begin{aligned} Transmitter(\ell_2) \stackrel{\text{def}}{=} & !!(message_1, r)@\mathbf{IR}\{\overrightarrow{\ell}\}.Transmitter'(\ell_2) \\ & + !!(message_2, r)@\mathbf{IR}\{\overrightarrow{\ell}\}.Transmitter'(\ell_2) \end{aligned}$$

Assuming $\ell_1 \in \overrightarrow{\ell}$ then *Sys* behaves in the same way as described in the previous section. Suppose we want to implement a bisimilar component in the same way as the one given in Example 6.

$$AltReceiver(\ell_3) \stackrel{\text{def}}{=} ??(message_1, 1)@\mathbf{Wt}\{1\}.(process, q).AltReceiver(\ell_3)$$

Now if $\ell_3 \in \overrightarrow{\ell}$ then the system $Transmitter(\ell_2) \parallel Receiver_1(\ell_1) \parallel Receiver_2(\ell_1)$ and $Transmitter(\ell_2) \parallel AltReceiver(\ell_3)$ can match each other's behaviour when no additional context is considered. However the components $Receiver_1(\ell_1) \parallel Receiver_2(\ell_2)$ and $AltReceiver(\ell_3)$ are no longer considered bisimilar as their behaviour is not equivalent in all the possible contexts. For example in the case where $\ell_1 \in \overrightarrow{\ell}$ but $\ell_3 \notin \overrightarrow{\ell}$.

The example above demonstrates that a bisimulation relation that requires the components to behave equivalently in all possible contexts is unlikely to give rise to a useful relation for model reduction. In particular, when we add the location back to the language it becomes rather easy to find counter-examples that show components to behave non-bisimilarly. For example, even two components that can perform identical unicast output and actions but are parametrised by different locations will be considered non-bisimilar just because we can always consider a context system which can only interact with one of the components considered.

A way to relax the definitions is treated in the following chapter where we move towards defining a bisimilarity for a given context, and also consider a less strict form of location matching. The broadcast communication sub-languages were not explored in this chapter due to two reasons. Firstly, the location aspect of the sub-language is the same as for unicast-only sub-languages. Secondly, the broadcast communication on its own could be seen as a subset of CBS for which the bisimulation theory is well-developed and could be extended to our case.

Chapter 5

Observational equivalences for PALOMA

In this chapter we will define several bisimulation-like equivalence relations for PALOMA. The aim is to arrive at a bisimulation relation which satifies the following weaker congruence property:

> If the behaviours of two components P and Q are equivalent with respect to the system *Sys* then the behaviour of $P \parallel Sys$ is equivalent to $Q \parallel Sys$ with respect to the empty context.

5.1 Contextual bisimulation

This chapter will concentrate on exploring bisimulation-like relations for the PALOMA language where we make use of the observed exit rates defined in Section 3.2. Firstly we will briefly cover a naive attempt to define a bisimulation on sequential components of PALOMA to demonstrate why it is not entirely trivial to deal with spatial properties of PALOMA models. The aim for the rest of the chapter is to refine the bisimulation relation to allow for substituting components in context systems. That is if two components are bisimilar with respect to some context then in this specific context we would be allowed to substitute one for the other.

5.1.1 Location as part of state

As the behaviour of the PALOMA sequential component is parametrised by its location the natural interpretation would be to consider locations as an inherent part of a component's state. This would lead to the following definition, making use of the syntax-derived rate function defined in Section 3.2.

Definition 20. Let $Sys \in \mathscr{C}$ be any model component serving as a context. A binary relation $\mathscr{R}^{[0]}_{Sys}$ is a bisimulation over sequential components if, and only if, $(S(\ell_1), T(\ell_2)) \in \mathscr{R}^{[0]}_{Sys}$ implies, for all $a \in Act$

1. $R_a(Sys, S(\ell_1)) = R_a(Sys, T(\ell_2))$.
2. $\ell_1 = \ell_2$.
3. $S(\ell_1) \xrightarrow{a, Sys} S'(\ell_1')$ implies $\exists\, T'(\ell_2'),\ T(\ell_2) \xrightarrow{a, Sys} T'(\ell_2'))$ and $(S'(\ell_1'), T'(\ell_2')) \in \mathscr{R}^{[0]}_{Sys}$.
4. $T(\ell_2) \xrightarrow{a, Sys} T'(\ell_2')$ implies $\exists\, S'(\ell_1'),\ S(\ell_1) \xrightarrow{a, Sys} S'(\ell_1'))$ and $(S'(\ell_1'), T'(\ell_2')) \in \mathscr{R}^{[0]}_{Sys}$.

Definition 20 is rather restrictive due to the way in which location is treated. Specifically, two sequential components which have identical behaviour in different locations will be considered non-equivalent in this setting even if they do not communicate with *Sys*. This would lead to a very strict equivalence being defined on the model components of PALOMA. A more interesting idea would be to shift to considering relative locations between the sequential components. This will be explored in the following section.

5.1.2 Relative locations

In order to consider relative locations between sequential components we need a notion of distance between the components. Thus we consider the case where *Loc* denotes a metric space. Specifically we will consider the Euclidean plane $\mathbb{R}^2$ (extensions to different metric spaces are immediate).

The notion we make use of in the following discussion is that of isometries – that is, maps between metric spaces that preserve the distances between points. In particular we are interested in the set of Euclidean plane isometries of which we have four types: translations, rotations, reflections and glide reflections. Denote the set of Euclidean plane isometries by $E(2)$.

The first definition we are going to give mimics Definition 20 but allows the locations of the sequential components under consideration to differ by an element in $E(2)$.

Definition 21. Let $Sys \in \mathscr{C}$ be a system component serving as context. A binary relation $\mathscr{R}^{[1]}_{Sys}$ is a spatial bisimulation over components if, and only if, $(S(\ell_1), T(\ell_2)) \in \mathscr{R}^{[1]}_{Sys}$ implies there exists $\phi \in E(2)$ such that for all $a \in Act$

1. $R_a(Sys, S(\ell_1)) = R_a(Sys, T(\ell_2))$.
2. $\phi(\ell_1) = \ell_2$.
3. $S(\ell_1) \xrightarrow{a,\,Sys} S'(\ell_1')$ implies $\exists\, T'(\ell_2')$, $T(\ell_2) \xrightarrow{a,\,Sys} T'(\ell_2')$ and $(S'(\ell_1'), T'(\ell_2')) \in \mathscr{R}^{[1]}_{Sys}$.
4. $T(\ell_2) \xrightarrow{a,\,Sys} T'(\ell_2')$ implies $\exists\, S'(\ell_1')$, $S(\ell_1) \xrightarrow{a,\,Sys} S'(\ell_1')$ and $(S'(\ell_1'), T'(\ell_2')) \in \mathscr{R}^{[1]}_{Sys}$.

In this definition for sequential components the location plays little role. The situation becomes more interesting when we to extend the definition to model components $\mathscr{C}$ of PALOMA.

Definition 22. Let $Sys \in \mathscr{C}$ a model component serving as context. A binary relation $\mathscr{R}^{[1*]}_{Sys}$ is a spatial bisimulation over model components if, and only if, $(P, Q) \in \mathscr{R}^{[1*]}_{Sys}$ implies there exists $\phi \in E(2)$ such that for all $a \in Act$ and all sets of locations L

1. $R_a(L, Sys, P) = R_a(\phi(L), Sys, Q)$.
2. $P \xrightarrow{a,\,Sys} P'$ implies for some Q', $Q \xrightarrow{a,\,Sys} Q'$ and $(P', Q') \in \mathscr{R}^{[1*]}_{Sys}$.
3. $Q \xrightarrow{a,\,Sys} Q'$ implies for some P', $P \xrightarrow{a,\,Sys} P'$ and $(P', Q') \in \mathscr{R}^{[1*]}_{Sys}$.

In Definition 22 we allow the locations of the actions performed by components P and Q to differ up to an isometry.

Next we are going to consider an example where, for the time being, omitting the proofs, we have supposed that one is going to be able to define a bisimilarity relation $\sim^{[1]}_{Sys}$ as the largest bisimulation as given by Definition 22. The simplest case we can consider is bisimilarity with respect to empty context system Sys denoted by $\emptyset$. We illustrate this in the following example.

Example 8.

$$\begin{aligned}
Transmitter(\ell_0) &:= !!(message\_move, r)@\mathbf{IR}\{all\}.Transmitter(\ell_1)\\
Transmitter(\ell_1) &:= !!(message\_move, r)@\mathbf{IR}\{all\}.Transmitter(\ell_0)\\
Receiver(\ell_1) &:= ??(message\_move, p)@\mathbf{Wt}\{v\}.Receiver(\ell_0)\\
Receiver(\ell_0) &:= ??(message\_move, q)@\mathbf{Wt}\{v\}.Receiver(\ell_1)
\end{aligned}$$

For this example take $\ell_0 = (-1,0)$ and $\ell_1 = (1,0)$. The two systems we are going to analyse are

$$Scenario_1 := Transmitter(\ell_0) \,||\, Receiver(\ell_1)$$
$$Scenario_2 := Transmitter(\ell_1) \,||\, Receiver(\ell_0)$$

It is clear that the systems are symmetric in the sense that if the locations in $Scenario_1$ are reflected along the y-axis we get $Scenario_2$. Denote the reflection along the y-axis as ϕ. This give $\phi(\ell_0) = \ell_1$ and $\phi(\ell_1) = \ell_0$.

It it intuitively clear that the two systems behave in the same way up to the starting location of the $Transmitter$ and $Receiver$ in both systems. Thus it makes sense to abstract away the absolute locations and consider the given systems observationally equivalent up to spatial transformation ϕ. In the following we verify that applying Definition 22 to these examples indeed agrees with the intuition. The two systems are considered on their own with no additional context – that is the Sys in Definiton 22 becomes $\emptyset$.

$$R_{!!message\_move}(\ell_0, \emptyset, Scenario_1) = r$$
$$R_{??message\_move}(\ell_1, \emptyset, Scenario_2) = rp$$

and

$$R_{!!message\_move}(\phi(\ell_0), \emptyset, Scenario_2) = R_{!!message\_move}(\ell_1, \emptyset, Scenario_1) = r$$
$$R_{??message\_move}(\phi(\ell_1), \emptyset, Scenario_2) = R_{??message\_move}(\ell_0, \emptyset, Scenario_1) = rp$$

As the rest of the rates are 0 then the first condition in Definition 22 holds. To get the second and third conditions requires verifying that the rates also match for derivatives of the systems $Scenario_1$ and $Scenario_2$. This is not going to be done here but one can easily see that the same symmetries are going to hold throughout the evolution of the systems and thus

$$Scenario_1 \sim_{\emptyset}^{[1]} Scenario_2$$

In the example we gave no additional context to the systems under study but the Definition 22 allows for reasoning about the equivalence of the two components in the context of any given system. The following example demonstrates that components being equivalent with respect to one context system does not imply equivalence with respect to other contexts.

Example 9.

$$\begin{aligned}
Transmitter(\ell_0) &:= !!(message, r)@\,\mathbf{IR}\{all\}.Transmitter(\ell_0)\\
Receiver(\ell_0) &:= ??(message, r)@\,\mathbf{Wt}\{w\}.Receiver(\ell_0)\\
Sys &:= Transmitter(\ell_0)
\end{aligned}$$

It can be verified that according to Definition 22 we have

$$Transmitter \sim^{[1]}_{\emptyset} Receiver$$

as neither component can perform an action due to the blocking nature of unicast communication. On the other hand we have

$$Transmitter \not\sim^{[1]}_{Sys} Receiver$$

as the system $Transmitter(\ell_0) \,||\, Transmitter(\ell_0)$ would not perform an action while in $Transmitter(\ell_0) \,||\, Receiver(\ell_0)$ we have unicast communication happening.

The Examples 8 and 9 give some confidence in the definitions. However there turns out to be a flaw in the way context is treated making $\sim^{[1]}_{Sys}$ fail to be a congruence.

Example 10. The congruence property we are looking for would be that if $P_1 \sim^{[1]}_{Sys} P_2$ then $P_1 \,||\, Sys \sim^{[1]}_{\emptyset} P_2 \,||\, Sys$. That is, if two components are bisimilar with respect to some context system then one can substitute the bisimilar components in the context and the behaviour of the system as a whole will remain the same with respect to the empty context. The following system will provide a counter-example to this.

$$\begin{aligned}
Transmitter(\ell_0) &\stackrel{\text{def}}{=} !!(message, r)@\,\mathbf{IR}\{all\}.Transmitter(\ell_0)\\
Receiver(\ell_0) &\stackrel{\text{def}}{=} ??(message, r)@\,\mathbf{IR}\{all\}.Receiver(\ell_0)\\
DelayedRec(\ell_0) &\stackrel{\text{def}}{=} (wait, q).??(message, r)@\,\mathbf{Pr}\{p\}.Receiver(\ell_0)\\
Sys &\stackrel{\text{def}}{=} DelayedRec(\ell_0)
\end{aligned}$$

According to the given bisimulation definition we have $Transmitter(\ell_0) \sim^{[1]}_{Sys} Receiver(\ell_0)$ as neither will perform an action in parallel with $DelayedRec(\ell_0)$. However, the following systems

$$\begin{aligned}
Scenario_1 &:= Transmitter(\ell_0) \,||\, DelayedRec(\ell_0)\\
Scenario_2 &:= Receiver(\ell_0) \,||\, DelayedRec(\ell_0)
\end{aligned}$$

will clearly have non-bisimilar behaviour after $DelayedRec(\ell_0)$ has completed the spontaneous action *wait*.

In conclusion there is a flaw in how context is dealt with in the given definitions. Example 8 provides us with a special case where the preservation under substitution trivially holds. The aim of the following section is to see whether context can be dealt with in more general cases in such a way that does not lead us to considering an isomorphism between components.

5.1.3 Dealing with changing context

Example 10 will provide us with a starting point for the discussion in this section. As before we are looking for conditions where the following implication holds:

$$P \sim_{Sys} Q \implies P \parallel Sys \sim_{\emptyset} Q \parallel Sys$$

We already noted, based on Example 10, that for the bisimilarity $\sim_{Sys}^{[1]}$ the implication above does not hold. The question now is what extra conditions have to be enforced. The new definition would have to distinguish between $Transmitter(\ell_0)$ and $Receiver(\ell_0)$ in the context of system $DelayedRec(\ell_0)$. For this we would have to look ahead at the possible resulting states of Sys. We are going to propose the following bisimulation definition.

Definition 23. Let $Sys \in \mathscr{C}$ a model component serving as context. A binary relation $\mathscr{R}_{Sys}^{[2]}$ is a spatial bisimulation over model components if, and only if, $(P,Q) \in \mathscr{R}_{Sys}^{[2]}$ implies there exists $\phi \in E(2)$ such that for all $a \in Act$ and all sets of locations L

1. $R_a(L, Sys, P) = R_a(\phi(L), Sys, Q)$.
2. $P \xrightarrow{a, Sys} P'$ implies for some Q', $Q \xrightarrow{a, Sys} Q'$ and $(P', Q') \in \mathscr{R}_{Sys'}^{[2]}$ for all context systems Sys' such that $P \parallel Sys \xrightarrow{a, \emptyset} P' \parallel Sys'$.
3. $Q \xrightarrow{a, Sys} Q'$ implies for some P', $P \xrightarrow{a, Sys} P'$ and $(P', Q') \in \mathscr{R}_{Sys'}^{[2]}$ for all context systems Sys' such that $Q \parallel Sys \xrightarrow{a, \emptyset} Q' \parallel Sys'$.

The conditions on derivatives of P and Q will ensure that if an action performed by either of those components causes the context system Sys to change to Sys' then the resulting continuations P' of P, and Q' of Q are bisimilar with respect to the context system Sys'. Again we define a bisimilarity $\sim_{Sys}^{[2]}$ as the largest bisimulation satisfying Definition 23 omitting the proofs.

Note that the implication

$$P_1 \sim_{Sys}^{[2]} P_2 \implies P_1 \parallel Sys \sim_{\emptyset}^{[2]} P_2 \parallel Sys$$

would also have to imply for some $\phi \in E(2)$ that $R_a(L, \emptyset, Sys \,||\, P) = R_a(\phi(L), \emptyset, Sys \,||\, Q)$ for all actions a and sets of location L. In the following we are going to provide a counter-example to this demonstrating the asymmetry between how components are affected by the context and how they affect the context. We are going to see that one of the reasons why the situation is asymmetric is due to the differing influence ranges of bisimilar components according to Definition 23.

Example 11. Consider the following components capable of broadcast communication.

$$
\begin{aligned}
&Trasmitter_a(1,0) \overset{\text{def}}{=} !(message, r) @ \mathbf{IR}\{(0,0),(0,1)\}.Transmitter(1,0)\\
&Trasmitter_b(1,0) \overset{\text{def}}{=} !(message, r) @ \mathbf{IR}\{(0,1)\}.Transmitter(1,0)\\
&Receiver(0,0) \quad\; \overset{\text{def}}{=} ?(message, r) @ \mathbf{Pr}\{p\}.Receiver(0,0)\\
&Receiver(0,1) \quad\; \overset{\text{def}}{=} ?(message, r) @ \mathbf{Pr}\{p\}.Receiver(1,0)\\
&Sys \qquad\qquad\quad\; \overset{\text{def}}{=} Receiver(0,0) \,||\, Receiver(1,0)
\end{aligned}
$$

We can see that $Transmitter_a(1,0) \sim^{[2]}_{Sys} Transmitter_b(1,0)$. Consider

$$
\begin{aligned}
Scenario_1 &\overset{\text{def}}{=} Transmitter_a(1,0) \,||\, Sys\\
Scenario_2 &\overset{\text{def}}{=} Transmitter_b(1,0) \,||\, Sys
\end{aligned}
$$

Now note that, due to $Transmitter_a(1,0)$ having an extra location in the influence range, the $Scenario_1$ will have two locations with action named $?\alpha$ firing at non-zero rate while as $Scenario_2$ has just one. This is supported by derivation graphs for the two scenarios given in Figures 5.1 and 5.2 where the $Scenari_1$ can be seen to have an two extra of broadcast communication transitions. Thus there will not exist an isometry such that $Scenario_1 \sim^{[2]}_{\emptyset} Scenario_2$.

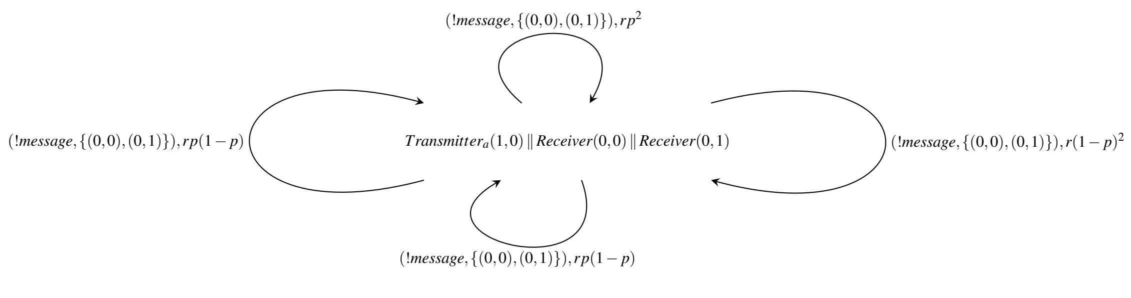
$(!message, \{(0,0),(0,1)\}), rp^2$
$(!message, \{(0,0),(0,1)\}), rp(1-p)$
$Transmitter_a(1,0) \,||\, Receiver(0,0) \,||\, Receiver(0,1)$
$(!message, \{(0,0),(0,1)\}), r(1-p)^2$
$(!message, \{(0,0),(0,1)\}), rp(1-p)$

Figure 5.1: Derivation graph of $Scenario_1$.

The example above shows that the bisimulation relation definition would have to be even stronger in order to be a congruence.

$$\begin{tikzcd}
Transmitter_b(1,0) \,\|\, Receiver(0,0) \,\|\, Receiver(0,1) \arrow[loop left, "{(!message,\{(0,0),(0,1)\}),r(1-p)}"] \arrow[loop right, "{(!message,\{(0,0),(0,1)\}),rp}"]
\end{tikzcd}$$

Figure 5.2: Derivation graph of $Scenario_2$.

5.1.4 Symmetry between components and context

In this section we continue refining the bisimulation definition given in Definition 23 to give rise to a bisimilarity $\sim_{Sys}$ such that the following implication holds

$$P_1 \sim_{Sys} P_2 \implies P_1 \,\|\, Sys \sim_{\emptyset} P_2 \,\|\, Sys$$

First note that we would like that, for some ϕ, the following holds for all $a \in Act$ and $L \in Loc$:

$$R_a(L,\emptyset,P_1 \,\|\, Sys) = R_a(\phi(L),\emptyset,P_2 \,\|\, Sys)$$

Writing these out based on Definition 14 gives

$$\begin{aligned}
R_a(L,\emptyset,P_1 \,\|\, Sys) \quad &= R_a(L,P_1,Sys) + R_a(L,Sys,P_1) \\
R_a(\phi(L),\emptyset,P_2 \,\|\, Sys) &= R_a(\phi(L),P_2,Sys) + R_a(\phi(L),Sys,P_2)
\end{aligned}$$

Note that $R_a(L,Sys,P_1) = R_a(\phi(L),Sys,P_2)$ holds whenever $P_1 \sim^{[2]}_{Sys} P_2$. If the equality was to hold $R_a(L,P_1,Sys) = R_a(\phi(L),P_2,Sys)$ for all $a \in Act$ and $L \in Loc$ then

$$R_a(L,\emptyset,P_1 \,\|\, Sys) = R_a(\phi(L),\emptyset,P_2 \,\|\, Sys)$$

would also hold. However we have already seen in Example 11 that this cannot be implied from $P_1 \sim^{[2]}_{Sys} P_2$ and thus we arrive at the following refined definition of a bisimulation relation.

Definition 24. Let $Sys \in \mathscr{C}$ be a model component serving as context. A binary relation $\mathscr{R}^{[3]}_{Sys}$ is a spatial bisimulation over model components if, and only if, $(P,Q) \in \mathscr{R}^{[3]}_{Sys}$ implies there exists $\phi \in E(2)$ such that for all $a \in Act$ and all locations $L \in Loc$

1. $R_a(L,\emptyset,P \,\|\, Sys) = R_a(\phi(L),\emptyset,Q \,\|\, Sys)$.
2. $P \xrightarrow{a,\,Sys} P'$ implies there exists Q', $Q \xrightarrow{a,\,Sys} Q'$ and $(P',Q') \in \mathscr{R}^{[3]}_{Sys'}$ for all context systems Sys' such that $P \,\|\, Sys \xrightarrow{a,\emptyset} P' \,\|\, Sys'$.

3. $Q \xrightarrow{a,Sys} Q'$ implies there exists P', $P \xrightarrow{a,Sys} P'$ and $(P',Q') \in \mathscr{R}^{[3]}_{Sys'}$ for all context systems Sys' such that $P \,||\, Sys \xrightarrow{a,\emptyset} P' \,||\, Sys'$.

Note that the definition above is much stronger than Definition 23 given in the previous section as it now checks the rates of the entire system rather than just the rates of actions involving components P and Q. We can also note that the two transmitters in Example 11 are no longer considered bisimilar according to Definition 24.

In order to completely avoid the case where $P \,||\, Sys$ can perform an action that $Q \,||\, Sys$ can not match, for example through differing behaviours of Sys due to contexts P and Q, we are going to make sure all the possible transitions from these systems are considered. That is, we are going to give the following definition.

Definition 25. Let $Sys \in \mathscr{C}$ a model component serving as context. A binary relation $\mathscr{R}_{Sys}$ is a spatial bisimulation over model components if, and only if, $(P,Q) \in \mathscr{R}_{Sys}$ implies there exists $\phi \in E(2)$ such that for all $a \in Act$ and all locations $L \in Loc$

1. $R_a(L,\emptyset,P \,||\, Sys) = R_a(\phi(L),\emptyset,Q \,||\, Sys)$.
2. $P \,||\, Sys \xrightarrow{a,\emptyset} P' \,||\, Sys'$ implies there exists Q', $Q \,||\, Sys \xrightarrow{a,\emptyset} Q' \,||\, Sys'$ and $(P',Q') \in \mathscr{R}_{Sys'}$.
3. $Q \,||\, Sys \xrightarrow{a,\emptyset} Q' \,||\, Sys'$ implies there exists P', $P \,||\, Sys \xrightarrow{a,\emptyset} P' \,||\, Sys'$ and $(P',Q') \in \mathscr{R}_{Sys'}$.

Informally, the above definition states that two components P and Q are bisimilar with respect to a context system Sys if for any action the rate at which $P \,||\, Sys$ performs it is equal to the rate at which $Q \,||\, Sys$ performs the same action. The bisimilarity conditions on the derivatives of P and Q are also stated in terms of the whole systems $P \,||\, Sys$ and $Q \,||\, Sys$. That is, in this chapter we made a step-by-step move towards considering equivalence in completely closed systems.

Definition 25 is very strict but allows us to show that the defined relation $\mathscr{R}_{Sys}$ is an equivalence relation of PALOMA components.

Lemma 1. *Spatial bisimulation $\mathscr{R}_{Sys}$, as given in Definition 25, is an equivalence relation on PALOMA components $\mathscr{C}$.*

Proof. The reflexivity and symmetry are clear by just noting that the $E(2)$ contains an identity map and each $\phi \in E(2)$ has an inverse in $E(2)$. For the transitivity suppose

$P,Q,R \in \mathscr{C}$ such that $(P,Q) \in \mathscr{R}_{Sys}$ and $(Q,R) \in \mathscr{R}_{Sys}$. Thus there exist $\phi, \psi \in E(2)$ such that for any L and a we have

$$R_a(L,\emptyset,P \parallel Sys) = R_a(\phi(L),\emptyset,Q \parallel Sys)$$

And likewise

$$R_a(L,\emptyset,Q \parallel Sys) = R_a(\psi(L),\emptyset,R \parallel Sys)$$

By the properties of isometries we know that $\psi \circ \phi \in E(2)$ and thus for any L and a we have

$$R_a(L,\emptyset,P \parallel Sys) = R_a(\psi(\phi(L)),\emptyset,R \parallel Sys)$$

All systems $P \parallel Sys$, $Q \parallel Sys$, and $R \parallel Sys$ are closed and from the conditions on derivatives we get that $Q \parallel Sys$ can match every action of $P \parallel Sys$, and $R \parallel Sys$ can match every action of $Q \parallel Sys$ and thus $R \parallel Sys$ can match all actions of $P \parallel Sys$. This clearly holds also in the other direction and thus we get $(P,R) \in \mathscr{R}_{Sys}$. □

Lemma 1 shows that the defined spatial bisimulation indeed satisfies the conditions of being an equivalence relation on PALOMA components. The following lemma is going to set us up for defining the spatial bisimilarity as the largest bisimulation relation. Namely, we are going to show that the union of bisimulations is a bisimulation.

Lemma 2. *Suppose that $\mathscr{R}_{Sys,i}$ (for $i = 1,2,\cdots$) are bisimulations. The union of bisimulation $\mathscr{R}_{Sys} = \cup_i \mathscr{R}_{Sys,i}$ is a bisimulation.*

Proof. Note that $\mathscr{R}_{Sys}$ is just a set of pairs of bisimilar components and thus $(P,Q) \in \mathscr{R}_{Sys}$ if, and only if, for some i we have $(P,Q) \in \mathscr{R}_{Sys,i}$. This means $(P,Q) \in \mathscr{R}_{Sys}$ implies that there exists $\phi \in E(2)$ such that for all $a \in Act$ and all sets of locations L we have

$$R_a(L,\emptyset,P \parallel Sys) = R_a(\phi(L),\emptyset,Q \parallel Sys)$$

In addition, we have that if

$$P \parallel Sys \xrightarrow{a,\emptyset} P' \parallel Sys'$$

then there exists Q' such that

$$Q \parallel Sys \xrightarrow{a,\emptyset} Q' \parallel Sys'$$

and $(P',Q') \in \mathscr{R}_{Sys'}$.

Similarly for $Q \parallel Sys \xrightarrow{a,\emptyset} Q' \parallel Sys'$ and hence $(P,Q) \in \mathscr{R}_{Sys}$ if, and only if, the bisimulation conditions hold. □

The Lemma 2 above lets us define a new relation of PALOMA components by taking the union of all bisimulations $\mathscr{R}_{Sys}$ in the following way.

Definition 26. Define the bisimilarity with respect to context system Sys, denoted $\sim_{Sys}$, as the largest spatial bisimulation relation as defined by Definition 25. We say two model components P_1, P_2, parametrised by locations in $\mathbb{R}^2$ are considered spatial bisimilar with respect to system Sys, denoted $P_1 \sim_{Sys} P_2$ if there exists spatial bisimulation $\mathscr{R}_{Sys}$ such that $(P_1, P_2) \in \mathscr{R}_{Sys}$.

The final parts of this section are devoted to analysing the properties of the defined bisimulation $\sim_{Sys}$. Namely we are interested in whether $\sim_{Sys}$ is preserved by the combinators of PALOMA.

Proposition 1. *Let $P_1, P_2 \in \mathscr{C}$ be components such that $P_1 \sim_{Sys} P_2$ where Sys is some context system then*

$$P_1 \,||\, Sys \sim_{\emptyset} P_2 \,||\, Sys$$

Proof. The rate condition

$$R_a(L, \emptyset, P_1 \,||\, Sys \,||\, \emptyset) = R_a(\phi(L), \emptyset, P_2 \,||\, Sys \,||\, \emptyset)$$

is clear. The conditions on derivatives of $P_1 \,||\, Sys$ and $P_2 \,||\, Sys$ are also clear from the definition of spatial bisimulation as for both $P_1 \sim_{Sys} P_2$ and $P_1 \,||\, Sys \sim_{\emptyset} P_2 \,||\, Sys$ we are considering the actions of the same systems, namely $P_1 \,||\, Sys$ and $P_2 \,||\, Sys$. □

Finally we have the question of whether $\sim_{Sys}$ is preserved by adding a prefix to the existing component. That is, suppose $P_1 \sim_{Sys} P_2$ and consider

$$Q_1(\ell) \stackrel{\text{def}}{=} a.P_1$$
$$Q_2(\ell) \stackrel{\text{def}}{=} a.P_2$$

We have that, in general,

$$Q_1(\ell) \not\sim_{Sys} Q_2(\ell)$$

The action a can be involved in changing the context system Sys to Sys' such that $P_1 \sim_{Sys'} P_2$ does not hold. An example of this is given below.

Example 12.

$$\begin{aligned}
Transmitter(\ell) &\stackrel{\text{def}}{=} !!(message_a, r) @ \mathbf{IR}\{all\}.Transmitter(\ell)\\
Receiver(\ell) &\stackrel{\text{def}}{=} ??(message_a, r) @ \mathbf{Pr}\{p\}.Receiver(\ell)\\
Sys &\stackrel{\text{def}}{=} ??(message_b, p) @ \mathbf{Pr}\{p\}.Receiver(\ell)
\end{aligned}$$

We have that $Transmitter(\ell) \,||\, Sys$ and $Receiver(\ell) \,||\, Sys$ never perform an action and thus $Transmitter(\ell) \sim_{Sys} Receiver(\ell)$. If we now consider

$$!!(message_b, r) @ \mathbf{IR}\{all\}.Transmitter(\ell)$$
$$!!(message_b, r) @ \mathbf{IR}\{all\}.Receiver(\ell)$$

and the parallel compositions

$$Sys || !!(message_b, r) @ \mathbf{IR}\{all\}.Transmitter(\ell)$$
$$Sys || !!(message_b, r) @ \mathbf{IR}\{all\}.Receiver(\ell)$$

then after the action $!!message_b$ fires Sys ends up in the state $Sys' \equiv Receiver(\ell)$ for which the components $Transmitter(\ell)$ and $Receiver(\ell)$ exhibit different behaviour.

The final case, namely $\sim_{Sys}$ being preserved by the choice operator is discussed separately in the next section.

5.1.5 Preservation under choice

In order to be able to prove congruence with respect to the choice operator we need to introduce back the rate condition from Definition 23.

Definition 27. Let $Sys \in \mathscr{C}$ be a model component serving as context. A binary relation $\mathscr{R}^{[4]}_{Sys}$ is a spatial bisimulation over model components if, and only if, $(P,Q) \in \mathscr{R}^{[4]}_{Sys}$ implies there exists $\phi \in E(2)$ such that for all $a \in Act$ and all locations $L \in Loc$

1. $R_a(L, Sys, P) = R_a(\phi(L), Sys, Q)$.

2. $R_a(L, \emptyset, P \,||\, Sys) = R_a(\phi(L), \emptyset, Q \,||\, Sys)$.

3. $P \,||\, Sys \xrightarrow{a,\emptyset} P' \,||\, Sys'$ implies there exists Q', $Q \,||\, Sys \xrightarrow{a,\emptyset} Q' \,||\, Sys'$ and $(P',Q') \in \mathscr{R}^{[4]}_{Sys'}$.

4. $Q \,||\, Sys \xrightarrow{a,\emptyset} Q' \,||\, Sys'$ implies there exists P', $P \,||\, Sys \xrightarrow{a,\emptyset} P' \,||\, Sys'$ and $(P',Q') \in \mathscr{R}^{[4]}_{Sys'}$.

The proofs for $\mathscr{R}^{[4]}_{Sys}$ defining an equivalence and being closed under taking unions are almost identical to the ones given in the previous section and will not be recreated here. We are going to define a spatial bisimilarity, denoted $\sim^{*}_{Sys}$ as the largest bisimulation relation $\mathscr{R}^{[4]}_{Sys}$.

Additionally we are going to need a well-formedness condition on PALOMA components.

Definition 28. A sequential component P is called well-formed if

1. for any two syntactically defined actions a and b of P we have $\Pi_{Act}(a) \neq \Pi_{Act}(b)$.

2. any continuation component of P is well-formed.

This means that well-formed components will not have choice between the same action type with the same label.

Example 13.

- $P(\ell) \overset{\text{def}}{=} (\alpha, r).P(\ell) + (\alpha, q).P(\ell)$ *is not* well-formed as there is choice between spontaneous actions with identical labels.

- $P(\ell) \overset{\text{def}}{=} (\alpha, r).P(\ell) + (\beta, q).P(\ell)$ for $\alpha \neq \beta$ *is* well-formed.

In the following we are going to restrict ourselves to considering model components composed of well-formed sequential components. This will allow us to reason about the observed exit rates in the following way.

Lemma 3. *For well-formed sequential components P, Q, $P+Q$, and model component Sys we have*

$$R_a(L, P+Q, Sys) = \begin{cases} \max\{R_a(L,P,Sys), R_a(L,Q,Sys)\} & \text{if } \Pi_{Type}(a) \in \{!!, !, ?, \cdot\} \\ \min\{R_a(L,P,Sys), R_a(L,Q,Sys)\} & \text{if } \Pi_{Type}(a) = ?? \end{cases}$$

Proof. If a is a spontaneous action ($\Pi_{Type}(a) = \cdot$) or a broadcast output action ($\Pi_{Type}(a) =$!) then it is clear that

$$R_a(L, P+Q, Sys) = R_a(L, P, Sys) = R_a(L, Q, Sys) = R_a(L, \emptyset, Sys) \tag{5.1}$$

as P, Q and $P+Q$ taken as contexts for Sys will have no effect on how fast we observe Sys performing action a.

Now consider the case where a is a broadcast input action ($\Pi_{Type}(a) =$?). If neither P nor Q have the relevant broadcast output action enabled then clearly Equation 5.1 holds again.

On the other hand if P has the corresponding output action enabled then we observe a firing at a faster rate. This is due to the fact that in addition to broadcasting sequential components in Sys we also have broadcast from component P. As $P+Q$ is well-formed then Q is not broadcasting. The same argument holds if we consider Q as the broadcasting component. Thus we can use function max to deal with the possible cases.

The observed rate of a if $\Pi_{Type}(a) = !!$ is only affected by whether there are listeners in the system. Thus the contribution from $P+Q$ can again be represented by the max function as only one of P and Q may have the relevant input action enabled.

In the case of a being a unicast input action we have the opposite situation as the context $P+Q$ can potentially add another listener competing for the relevant unicast message. In particular, if P can also perform a then the observed rate $R_a(L,P+Q,Sys)$, where $P+Q$ is a well-formed component, is $R_a(L,P,Sys)$. This however is smaller than the rate $R_a(L,Q,Sys)$ as P is now also competing for the unicast messages that are sent out by sequential components in Sys. Similarly for Q having a enabled. $\square$

Definition 3 allows us to prove the following lemma about the choice operator for well-formed components.

Lemma 4. *Let $S_1(\ell_1), S_2(\ell_2) \in \mathscr{C}_S$ be sequential components such that $S_1(\ell_1) \sim^*_{Sys} S_2(\ell_2)$, then for well-formed components $S_1(\ell_1)+T(\ell)$ and $S_2(\ell_2)+T(\ell)$ we have*

$$S_1(\ell_1)+T(\ell) \sim^*_{Sys} S_2(\ell_2)+T(\ell)$$

Proof. Sequential components in PALOMA are defined over a single location and thus $S_1(\ell_1)$ and $S_2(\ell_2)$ are restricted to having the same location by $T(\ell)$. This means that $S_1(\ell)$ and $S_2(\ell)$ are bisimilar via the identity isometry. For an action a such that $\Pi_{Type}(a) = \{!!, !, ?, \cdot\}$ we have the following

$$\begin{aligned}
&R_a(L,\emptyset,(S_1(\ell)+T(\ell)) \parallel Sys) \\
&\quad= \quad R_a(L,(S_1(\ell)+T(\ell)),Sys)+R_a(L,Sys,(S_1(\ell)+T(\ell))) \\
&\quad= \quad R_a(L,(S_1(\ell)+T(\ell)),Sys)+R_a(L,Sys,S_1(\ell))+R_a(L,Sys,T(\ell)) \\
&\quad= \quad \max\{R_a(L,S_1(\ell),Sys)+R_a(L,Sys,S_1(\ell)), \\
&\qquad\qquad R_a(L,T(\ell),Sys)+R_a(L,Sys,S_1(\ell))\}+R_a(L,Sys,T(\ell)) \\
&\quad\overset{S_1(\ell_1)\sim^*_{Sys}S_2(\ell_2)}{=} \quad \max\{R_a(L,\emptyset,S_2(\ell) \parallel Sys),R_a(L,T(\ell),Sys)+R_a(L,Sys,S_2(\ell))\} \\
&\qquad\qquad +R_a(L,Sys,T(\ell)) \\
&\quad= \quad \max\{R_a(L,S_2(\ell),Sys)+R_a(L,Sys,S_2(\ell)), \\
&\qquad\qquad R_a(L,T(\ell),Sys)+R_a(L,Sys,S_2(\ell))\}+R_a(L,Sys,T(\ell)) \\
&\quad= \quad \max\{R_a(L,S_2(\ell),Sys),R_a(L,T(\ell),Sys)\} \\
&\qquad\qquad +R_a(L,Sys,S_2(\ell))+R_a(L,Sys,T(\ell)) \\
&\quad= \quad R_a(L,\emptyset,(S_2(\ell)+T(\ell)) \parallel Sys))
\end{aligned}$$

We have used the property

$$\max(a,b)+c = \max(a+b,b+c)$$

Note that the same property holds for the min function and thus the above works for $\Pi_{Type}(a) =$?? with function max replaced with min. For the derivatives we note that the set of actions $(S_1(\ell)+T(\ell)) \parallel Sys$ can perform is exactly the union of actions $S_1(\ell) \parallel Sys$ and $T(\ell) \parallel Sys$ can perform. Thus if there is a transition $(S_1(\ell)+T(\ell)) \parallel Sys \xrightarrow{a,\emptyset} (S_1(\ell)+T(\ell))' \parallel Sys'$ then, as we are dealing with well-formed components, there are two cases

$$(S_1(\ell)+T(\ell))' = \begin{cases} S_1'(\ell') \\ T'(\ell') \end{cases}$$

In both cases there exists a matching transition of $(S_2(\ell)+T(\ell)) \parallel Sys$ due to $S_1(\ell) \sim_{Sys} S_2(\ell)$ by the assumption and $T(\ell) \sim_{Sys} T(\ell)$ by reflexivity. $\square$

In conclusion, we have found that when considering closed systems we can construct a bisimulation-like equivalence that behaves well with respect parallel composition. For the prefix combinator we could quite easily find a counter-example showing that currently defined equivalences will not be preserved by the combinator. We could solve this by strengthening the bisimulation relations even further by requiring the components to be bisimilar with respect to all possible context systems in PALOMA. As discussed in Chapter 4 this would lead to a much stricter equivalence than the ones discussed in this chapter. We also found that the defined bisimilarity $\sim^*_{Sys}$ can be shown to be preserved by the choice operator if the extra well-formedness conditions are imposed on PALOMA components. Namely, we prohibit choice between actions with the same name — that is type as well as label.

Chapter 6

Conclusions

In this chapter we are going to summarise the main results of this project. We are also going to give a brief evaluation of the defined bisimulation-like equivalence relations from the point of view of model comparison and reduction, as well as discuss some possible further research directions.

6.1 Summary

PALOMA language was introduced in its current form in [10] where the emphasis was on the fluid analysis of CTMCs. The paper gave population-level semantic rules for generating a model in the Multi-message Multi-class Markovian Agents Model framework [6] which were used to generate CTMCs on population counts of agents at each location. This approach lent itself well to fluid-approximations of the behaviour of systems.

In order to have a rigorous foundation for bisimulation definitions we have introduced the new agent level semantics in the FuTSs framework [7]. We have given an example of applying the defined semantics to construction of a derivation graphs and a CTMC, and the subsequent calculation of performance measures.

In Chapter 5 we presented several bisimulation relations of PALOMA models which allow us to abstract away explicitly defined locations and consider relative locations of components instead. The idea was to work over the Euclidean plane and use the properties of isometries of the Eucliden plane to argue about the relative structure of the system. To deal with context dependence we have defined a bisimilarity $\sim_{Sys}$ with respect to a given context system *Sys*. The relation was consistent with the idea of being able to swap out components for bisimilar ones in the systems that we are interested in. The refinement of $\sim_{Sys}$ denoted $\sim^{*}_{Sys}$ was also shown to be preserved by the choice operator if the set of sequential component under study was restricted to ones with not unicast communication capabilities. The definitions and analysis in Chapters 4 and 5 were supported by examples and counter-examples.

6.2 Evaluation and further work

The semantics presented in Chapter 3 seem to be suitable for generation of numerical models in terms of CMTCs of systems described in PALOMA. The set of labels chosen, however, loses some information about the actions being performed. Namely, the labels only reflect whether there was a communication failure through the action rates. In the case of broadcast communication that does not require there to be any available listeners in the system the labels do not give any information about which agents were part of the communication. This becomes an issue when we want to define equivalence relations directly in terms of the transition system as the connection between syntactic definitions of components and their transition relations becomes less clear.

The bisimilarities $\sim_{Sys}$ and $\sim^{*}_{Sys}$ we arrived at in Chapter 5 were shown to have the basic algebraic properties to make them theoretically meaningful in the context of model comparison and component substitution. The construction of those relations were supported by non-trivial examples. However, in order to be able to show substitution properties of the defined relations the bisimulation definitions had to be strengthened to consider the entire behaviour of closed systems rather than just the behaviour of individual components under study. That is, in order to say that two components are equivalent in some context system the scenarios where the components are composed in parallel with the context have to behave equivalently. This is a very strong requirement as the full derivation graphs of both scenarios would have to be constructed in full detail.

The bisimulation $\sim^{*}_{Sys}$ was found to be preserved by the choice operator in restricted cases. In particular, we prohibit choice between actions with same type and label. Note that from a modelling perspective this is a reasonable assumption to make as we are getting rid of redundancy when describing the actions available to a component.

Whether or not the given relations give rise to significant model reductions remains a topic for further work. However due to the complexity of PALOMA language it will be difficult to find non-trivial alternative implementations of components. The difficulty stems from the large number of different parameters that need to be tracked as well as the way agents communicate in PALOMA.

Finally, we would like to note that only equivalence stemming from bisimulation as described in [21] were considered. Alternative notions of equivalence like trace equivalence from automata theory as well as strong equivalence as defined for PEPA [15] could also be considered in the future.

Bibliography

[1] Jos C. M. Baeten and Jan A. Bergstra. ‘Real space process algebra’. In: *Formal Asp. Comput.* 5.6 (1993), pp. 481–529. DOI: 10.1007/BF01211247.

[2] Jan A. Bergstra and Jan. W. Klop. ‘Algebra of communicating processes with abstraction’. In: *Theoretical Computer Science* 37 (1985), pp. 77–121. ISSN: 0304-3975. DOI: 10.1016/0304-3975(85)90088-X.

[3] Marco Bernardo and Roberto Gorrieri. ‘Extended Markovian Process Algebra’. In: *CONCUR ’96, Concurrency Theory, 7th International Conference, Pisa, Italy, August 26-29, 1996, Proceedings*. 1996, pp. 315–330. DOI: 10.1007/3-540-61604-7_63.

[4] Howard Bowman, Jeremy Bryans and John Derrick. ‘Analysis of a Multimedia Stream using Stochastic Process Algebra’. In: *Comput. J.* 44.4 (2001), pp. 230–245. DOI: 10.1093/comjnl/44.4.230.

[5] Luca Cardelli and Philippa Gardner. ‘Processes in space’. In: *Theor. Comput. Sci.* 431 (2012), pp. 40–55. DOI: 10.1016/j.tcs.2011.12.051.

[6] Davide Cerotti et al. ‘A Markovian Agent Model for Fire Propagation in Outdoor Environments’. In: *Computer Performance Engineering - 7th European Performance Engineering Workshop, EPEW 2010, Bertinoro, Italy, September 23-24, 2010. Proceedings*. 2010, pp. 131–146. DOI: 10.1007/978-3-642-15784-4_9.

[7] Rocco De Nicola et al. ‘A uniform definition of stochastic process calculi’. In: *ACM Comput. Surv.* 46.1 (2013), p. 5. DOI: 10.1145/2522968.2522973.

[8] Ansgar Fehnker et al. ‘A Process Algebra for Wireless Mesh Networks’. In: *Programming Languages and Systems - 21st European Symposium on Programming, ESOP 2012, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2012, Tallinn, Estonia, March 24 - April 1, 2012. Proceedings*. 2012, pp. 295–315. DOI: 10.1007/978-3-642-28869-2_15.

[9] Cheng Feng and Jane Hillston. ‘PALOMA: A Process Algebra for Located Markovian Agents’. In: *Quantitative Evaluation of Systems - 11th International Conference, QEST 2014, Florence, Italy, September 8-10, 2014. Proceedings*. 2014, pp. 265–280. DOI: 10.1007/978-3-319-10696-0_22.

[10] Cheng Feng, Jane Hillston and Vashti Galpin. ‘Automatic Moment-Closure Approximation of Spatially Distributed Collective Adaptive Systems’. In: *ACM Trans. Model. Comput. Simul.* 26.4 (2016), p. 26. DOI: 10.1145/2883608.

[11] Jean-Michel Fourneau, Leïla Kloul and Fabrice Valois. ‘Performance modelling of hierarchical cellular networks using PEPA’. In: *Perform. Eval.* 50.2/3 (2002), pp. 83–99. DOI: 10.1016/S0166-5316(02)00101-3.

[12] Vashti Galpin. ‘Spatial Representations and Analysis Techniques’. In: *Formal Methods for the Quantitative Evaluation of Collective Adaptive Systems - 16th International School on Formal Methods for the Design of Computer, Communication, and Software Systems, SFM 2016, Bertinoro, Italy, June 20-24, 2016, Advanced Lectures*. 2016, pp. 120–155. DOI: 10.1007/978-3-319-34096-8_5.

[13] Vashti Galpin, Cheng Feng and Jane Hillston. *Language constructs for spatial representation*. Internal QUANTICOL report. University of Edinburgh, 2015.

[14] Matthew Hennessy and Julian Rathke. ‘Bisimulations for a Calculus of Broadcasting Systems’. In: *Theor. Comput. Sci.* 200.1-2 (1998), pp. 225–260. DOI: 10.1016/S0304-3975(97)00261-2.

[15] Jane Hillston. *A Compositional Approach to Performance Modelling*. New York, NY, USA: Cambridge University Press, 1996. ISBN: 0521571898.

[16] C. A. R. Hoare. *Communicating Sequential Processes*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1985. ISBN: 0131532715.

[17] Mathias John, Roland Ewald and Adelinde M. Uhrmacher. ‘A Spatial Extension to the pi Calculus’. In: *Electr. Notes Theor. Comput. Sci.* 194.3 (2008), pp. 133–148. DOI: 10.1016/j.entcs.2007.12.010.

[18] Kim Guldstrand Larsen and Arne Skou. ‘Bisimulation through Probabilistic Testing’. In: *Inf. Comput.* 94.1 (1991), pp. 1–28. DOI: 10.1016/0890-5401(91)90030-6.

[19] Mylène Maurin, Morgan Magnin and Olivier H. Roux. ‘Modeling of Genetic Regulatory Network in Stochastic pi-Calculus’. In: *Bioinformatics and Computational Biology, First International Conference, BICoB 2009, New Orleans, LA, USA, April 8-10, 2009. Proceedings*. 2009, pp. 282–294. DOI: 10.1007/978-3-642-00727-9_27.

[20] Nicola Mezzetti and Davide Sangiorgi. ‘Towards a Calculus For Wireless Systems’. In: *Electr. Notes Theor. Comput. Sci.* 158 (2006), pp. 331–353. DOI: 10.1016/j.entcs.2006.04.017.

[21] Robin Milner. *Communication and Concurrency*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1989. ISBN: 0131150073.

[22] Paul Piho and Jane Hillston. ‘Stochastic and Spatial Equivalences for PALOMA’. In: *Proceedings of the Workshop on FORmal methods for the quantitative Evaluation of Collective Adaptive SysTems*. 2016.

[23] Paul Piho and Jane Hillston. ‘Stochastic and Spatial Equivalences for PALOMA’. In: *Proceedings of the Workshop on FORmal methods for the quantitative Evaluation of Collective Adaptive SysTems, FORECAST@STAF 2016, Vienna, Austria, 8 July 2016*. 2016, pp. 69–80. DOI: 10.4204/EPTCS.217.9.

[24] Gordon D Plotkin. *A structural approach to operational semantics*. Report DAIMI-FN-19, Computer Science Dept, Aarhus University, Denmark, 1981.

[25] K. V. S. Prasad. 'A Calculus of Broadcasting Systems'. In: *TAPSOFT'91: Proceedings of the International Joint Conference on Theory and Practice of Software Development, Brighton, UK, April 8-12, 1991, Volume 1: Colloquium on Trees in Algebra and Programming (CAAP'91)*. 1991, pp. 338–358. DOI: 10.1007/3-540-53982-4_19.

[26] Xian Yang et al. 'Modelling and performance analysis of clinical pathways using the stochastic process algebra PEPA'. In: *BMC Bioinformatics* 13.S-14 (2012), S4. DOI: 10.1186/1471-2105-13-S14-S4.