# Probabilistic Supervisory Control of Probabilistic Discrete Event Systems

By

Vera Pantelic, B.Eng., M.A.Sc.

A Thesis
Submitted to the School of Graduate Studies
in Partial Fulfilment of the Requirements
for the Degree of
Doctor of Philosophy

McMaster University

DOCTOR OF PHILOSOPHY (2011) MCMASTER UNIVERSITY
(Software Engineering) Hamilton, Ontario

TITLE: Probabilistic Supervisory Control of Probabilistic Discrete Event Systems

AUTHOR: Vera Pantelic
B.Eng., University of Belgrade, Serbia
M.A.Sc., McMaster University, Canada

SUPERVISOR: Dr. Mark Lawford

NUMBER OF PAGES: ix, 109

*To Ivan and Stefan*

# Abstract

This thesis considers probabilistic supervisory control of probabilistic discrete event systems (PDES). PDES are modeled as generators of probabilistic languages. The probabilistic supervisors employed are a generalization of the deterministic ones previously employed in the literature. At any state, the supervisor enables/disables events with certain probabilities. The probabilistic supervisory control problem (PSCP) that has previously been considered in the literature is revisited: find, if possible, a supervisor under whose control the behavior of a plant is identical to a given probabilistic specification. The existing results are unified, complemented with a solution of a special case and the computational analysis of synthesis problem and the solution.

The central place in the thesis is given to the solution of the optimal probabilistic supervisory control problem (OPSCP) in the framework: if the conditions for the existence of probabilistic supervisor for PSCP problem are not satisfied, find a probabilistic supervisor such that the achievable behaviour is as close as possible to the desired behaviour. The proximity is measured using the concept of pseudometric on states of generators. The distance between two systems is defined as the distance in the pseudometric between the initial states of the corresponding generators.

The pseudometric is adopted from the research in formal methods community and is defined as the greatest fixed point of a monotone function. Starting from this definition, we suggest two algorithms for finding the distances in the pseudometric. Further, we give a logical characterization of the same pseudometric such that the distance between two systems is measured by a formula that distinguishes between the systems the most. A trace characterization of the pseudometric is then derived from the logical characterization by which the pseudometric measures the difference of (appropriately discounted) probabilities of traces and sets of traces generated by systems, as well as some more complicated properties of traces. Then, the solution to the optimal probabilistic supervisory control problem is presented.

Further, the solution of the problem of approximation of a given probabilistic generator with another generator of a prespecified structure is suggested such that the new model is as close as possible to the original one in the pseudometric (probabilistic model fitting). The significance of the approximation is then discussed. While other applications are briefly discussed, a special attention is given to the use of ideas of probabilistic model fitting in the solution of a modified optimal probabilistic supervisory control problem.

# Acknowledgments

I would like to express my gratitude and appreciation to Dr. Mark Lawford, my advisor, for his guidance, and support during my Ph.D. journey.

Also, I would like to thank Dr. Ryszard Janicki for his constant encouragement since my first days at McMaster University.

I offer my sincere gratitude to the other members of my Ph.D. committee, Dr. Tom Maibaum, and Dr. Doug Down, for their valuable suggestions and advices.

Above all, I thank my family and friends, especially my beloved parents Divna and Milos, for being by my side all these years.

# Contents

# List of Figures

# Chapter 1

# Introduction

## 1.1 Motivation

The supervisory control theory of *discrete event systems* (DES) was developed in the seminal work of Ramadge and Wonham (Ramadge and Wonham, 1987). A supervisor (controller) controls a plant by enabling/disabling controllable events based on the observation of the previous behaviour of the plant. DES are most often modeled by *generators*, finite automata whose transitions are labeled with events: therefore, the behaviour of a DES can be represented as a regular language. The supervisory control problem typically considered is to supervise the plant so it generates a given specification language.

On the other hand, probabilistic models have attracted considerable attention in modeling systems with uncertainty. They are of interest in many application areas, e.g., communication protocols, distributed computing, performance analysis, and fault tolerance. Many probabilistic logics are used in the specification and verification of probabilistic systems (excellent overviews can be found in (Rutten et al., 2004; Kwiatkowska et al., 2007)). Probabilistic model checking provides for limited guarantees when conventional model checking is not possible (e.g., the state space is too large) (Huth and Kwiatkowska, 1998). Many of probabilistic models have been widely researched and applied. While reactive models have been predominantly used in probabilistic model checking tools as well as in the control of probabilistic systems, generative models have also found their applications, especially in the control and modeling of robot systems (Li et al., 1998; Mallapragada et al., 2009; Chattopadhyay et al., 2009), human sequence prediction (Feldman and Hanna, 1966), etc. The difference between reactive and generative systems is in the treatment of events (Schröder and Mateus, 2002; Glabbeek et al., 1995). In a reactive system, an event is seen as an input from environment: the system reacts to it by choosing a next state according to a probability distribution on the states of the system. On the other hand, in a generative system, an event is seen as an output: the system chooses a transition according to a probability distribution, and generates as an output the event the chosen transition is labeled with. In terms of expressiveness, in general, a generative model is more expressive than a reactive model.

We seek for a comprehensive theoretical framework for probabilistic supervisory control theory of *probabilistic discrete event systems* (PDES). We build upon the framework introduced in (Lawford and Wonham, 1993; Postma and Lawford, 2004). Finite state machines with transitions labeled with events, *generators*, commonly used in classical supervisory theory to represent discrete event systems, are generalized to a generative probabilistic model called *probabilistic generators*, which are generative models. Roughly speaking, probabilistic generators are generators extended with probabilities attached to each transition. A probability attached to an event that can occur from a state represents the probability of occurrence of that event from the state. A probabilistic generator generates a *probabilistic language*: each string generated by the underlying (nonprobabilistic) generator has a probability attached to it that represents the probability of occurrence of that particular string in the language. The standard supervisory control problem is accordingly generalized to the *probabilistic supervisory control problem (PSCP)*: find, if possible, a supervisor for a plant so that the plant under control generates a given probabilistic language. The problem has been solved in (Lawford and Wonham, 1993; Postma and Lawford, 2004).

The most widely used framework for control of PDES is that of Markov decision processes (MDPs, also known as Markov controlled processes). An MDP is a reactive model. Most state-of-the-art probabilistic model checkers support this model directly. The generative model that will be used in this thesis can be straightforwardly transformed into an MDP (with the introduction of a special event) so that the probabilistic model checking tools can be used for the analysis of these systems. The details of the transformation will be discussed later in this chapter. Also, as will be shown, probabilistic supervisor in our framework can be represented as an MDP, and probabilistic policy as defined in the MDP framework can be represented using an augmented probabilistic generator. This convertibility of the components of one framework to the components of the other has a potential in the reuse of the existing knowledge base of one framework in the other one, in terms of both control theory and verification. However, simply transforming our model to an MDP and applying the control as defined in the MDP framework on the resulting MDP is pointless: the control applied to the transformed model in the MDP framework cannot change the dynamics of the system.

A distinguishing feature of our framework is *probabilistic control* (as opposed to deterministic control). Probabilistic control means the employment of the control method of *random disablement*: after observing a string $s$, the probabilistic supervisor enables an event $\sigma$ with a certain probability. Although deterministic control of PDES is easier to deal with than probabilistic control (both from the viewpoint of analysis, and practice), probabilistic control of PDES is much more powerful. It has been shown in (Lawford and Wonham, 1993) that probabilistic supervisory control can generate a much larger class of probabilistic languages than deterministic control. In the sense of the probabilistic supervisory control problem mentioned, the use of deterministic control might be too restrictive for a designer. Hence, probabilistic control is employed in this framework.

The main goals of the thesis are, firstly, to merge the existing results of (Lawford and Wonham, 1993) and (Postma and Lawford, 2004) while completing them with the solution of a special case and complexity analysis, and, secondly, and more importantly, to solve the *optimal probabilistic supervisory control problem* (OPSCP, also known as the *closest approximation problem*) inside the framework. Analogous to a problem in classical supervisory control theory, it can happen that, given a plant to be controlled and a probabilistic specification language, no supervisor exists such that the plant under probabilistic control generates the prespecified (probabilistic) language. In this case, when the exact solution is not achievable, a designer tries to find a supervisor such that the plant under control generates a *closest approximation* of the desired behaviour. The nonprobabilistic behaviour of the requirements specification is considered to be a safety constraint in the standard supervisory control sense similar to (Kumar and Garg, 2001; Kumar and Garg, 1998a; Kumar and Garg, 1998b). Therefore, the supremal controllable sublanguage of the specification with respect to the plant is generated as the maximal achievable legal nonprobabilistic behaviour of the plant under control. Then, the closest approximation is calculated by minimizing the distance between the achievable probabilistic behaviour of the plant under control and the probabilistic behaviour of the specification whose nonprobabilistic behaviour is reduced to the mentioned supremal controllable sublanguage. The distance between (generators representing) PDES is measured using a *pseudometric* on states of probabilistic generators.

While our initial focus was on its use in supervisory control, the concept of pseudometric is obviously useful outside of supervisory control theory as a tool to measure the behavioural similarity of systems represented as probabilistic generators. As (Giacalone et al., 1990; Desharnais et al., 1999; Desharnais et al., 2002) (to name a few) pointed out, probabilistic bisimulation is not robust as it requires the exact matching of the values of probabilities of corresponding transitions. It is too sensitive to small changes in probabilities: a slight change of probabilities makes bisimilar systems nonbisimilar. Similarly, two systems with only slightly different probabilities of corresponding transitions would be as different as two systems with disjoint event sets (Deng et al., 2006). Further, as the values of probabilities are often only approximations, using either probabilistic bisimulation or reasoning in a boolean-valued logic is not sensible (van Breugel and Worrell, 2005). The notion of a pseudometric is hence used to approximate the notion of equivalence. The pseudometric we use is based on the pseudometric introduced in (Deng et al., 2006). It measures behavioural similarity between two states: the smaller the distance, the greater similarity between the states. The pseudometric subsumes probabilistic bisimulation: two states are at distance 0 in the pseudometric if and only if they are probabilistic bisimilar. No algorithms have been previously suggested for the calculation of distances in the pseudometric for probabilistic generators, and previously only a fixed point characterization has been offered. We suggest two algorithms for the calculation of the distance between two generators in the pseudometric, and then present different characterizations of the pseudometric that offer more information about its nature. Further, we explore the approximation of a PDES with a probabilistic generator with prespecified structure such that the original probabilistic behaviour of the PDES is preserved as much as possible (the approximation will be referred to as *probabilistic model fitting*). One of the possible applications of this approximation is the state reduction of PDES. However, as it turns out, the approximation has far more reaching applications with regards to the optimal probabilistic supervisory control problem. More precisely, some of the ideas of probabilistic model fitting are used to modify the OPSCP algorithm so that the distance between the achievable probabilistic behaviour of the plant under control and the original (probabilistic) requirements specification (without constraining the specification to its supremal controllable sublanguage) is minimized.

## 1.2 Related Work

Many models of stochastic behaviour of discrete event systems have been proposed. Markov chains (Cassandras, 1993), Markov decision processes (also referred to as controlled Markov chains) (Bellman, 1957; Howard, 1960), Rabin's probabilistic automata (Rabin, 1963), and stochastic Petri nets (Molloy, 1982) are some of the most referenced and widely applied. Markov chains extend automata with probability distributions induced by states: each transition has a certain probability of occurrence. Markov decision processes (MDPs) are Markov chains with actions. For each state, and an action, a probability distribution is induced: the sum of transition probabilities of one event at a state is 1. MDPs are a classical reactive model: for each state, probabilities are distributed over the outgoing transitions labeled with the same event. In a generative system, on the other hand, for each state, probabilities are distributed over all outgoing transitions. Another popular reactive model is that of Rabin's probabilistic automata (Rabin, 1963). The model is similar to that of MDPs, with the addition of accepting states and no cost/reward typically associated with transitions in MDPs. The motivation of Rabin's automata was to model *cut-languages*: the sets of strings whose probability of occurrence is greater than a certain value, $\lambda$.

The work of (Garg et al., 1999; Garg, 1992a; Garg, 1992b) models probabilistic systems using probabilistic languages: a map assigns each trace in a system a value that represents its probability of occurrence. The following two constraints are imposed on the probabilities: (i) the probability of the trace of zero length is 1, and (ii) the probability of any trace is greater than or equal to the cumulative probability of all of its extensions. The second constraint allows for modeling of termination. As (Garg et al., 1999) states, probabilistic languages can also be viewed as formal power series (Salomaa, 1990), or fuzzy sets (Lee and Zadeh, 1969). A probabilistic language is a formal power series with the two constraints stated above. The difference of the approach of (Garg et al., 1999; Garg, 1992a; Garg, 1992b) and formal power series is in the operator definitions (for details, see (Garg et al., 1999)). On the other hand, fuzzy languages are sets of event traces with a membership grade in the interval $[0, 1]$. Compared to fuzzy languages, probabilistic languages allow only the membership grades that satisfy certain constraints, such that membership grades can be viewed as the probability of occurrences of traces. Further, in (Garg et al., 1999; Garg, 1992a; Garg, 1992b), probabilistic languages are represented using probabilistic automata.

We use the approach of (Garg et al., 1999; Garg, 1992a; Garg, 1992b) and model PDES as probabilistic languages. Further, a probabilistic language can be generated by a finite state automaton with transitions labeled with events and probabilities. The model is generative: the probabilities of all the events in a certain state add up to at most one. When the probabilities add up to less than one, the remaining probability is the probability of termination. Terminating automata can be transformed into nonterminating with introduction of a special, terminating event that would lead to a dump state. Generative models are more general than reactive models: for every state, a generative model contains not only the information about relative probabilities of transitions on the same event, but also information on relative probabilities of transitions on different events (Schröder and Mateus, 2002; Glabbeek et al., 1995). However, as opposed to the automata of (Garg et al., 1999; Garg, 1992a; Garg, 1992b), our probabilistic automaton (probabilistic generator) is deterministic in the sense that, for each state of the automaton, there is at most one next state to which the automaton can move to on a given event. Also, unlike Markov chains, MDPs or stochastic Petri nets, the emphasis of the approach of (Garg et al., 1999; Garg, 1992a; Garg, 1992b) is on event traces rather than state traces. As previously mentioned, the model has been used in control and modeling of robot systems (Li et al., 1998; Mallapragada et al., 2009; Chattopadhyay et al., 2009), human sequence prediction (Feldman and Hanna, 1966), etc.

The most widely used framework for the study of control of probabilistic systems is that of MDPs or controlled Markov chains (for details on the control of MDPs, and its comparison with our chosen setting, see Section 1.3). The control typically studied in this framework is optimal in the sense that a certain performance criterion is optimized (Gihman and Skorohod, 1979; Bertsekas, 1987; Hernández-Lerma and Lasserre, 1996). Probabilistic control (usually referred to as randomized control in the literature) has been extensively studied in this framework. The famous result (Blackwell, 1962) states that for every MDP with a finite state space and set of actions, there is a deterministic and stationary policy that is optimal. In the case of partial observability things become more complicated (Arapostathis et al., 1993). It has been shown in (Rosenberg et al., 2000) that an optimal policy for MDP with partial observation is, in the general case, neither deterministic nor stationary. However, if the model is deterministic, then the optimal policy can be chosen to be deterministic and stationary too. But, if we add the constraint for a supervisor to have bounded memory, a deterministic policy is outperformed by a randomized policy (Kalai and Solan, 2003) (without taking the cost of randomization into account). In (Beauquier et al., 1995), partially observed MDPs are considered. The goal is to find optimal control where the criterion of optimality is described in terms of a regular language that represents the admissible behavior of the system. The paper shows that, when it comes to finite memory policies, randomized policies can do better than deterministic ones. Optimality of the trunk reservation problem (the problem of serving customers using finite-capacity queues, where the number of customer types is fixed) with a different type of constraints is considered in (Feinberg and Reiman, 1994; Fan-Orzechowski and Feinberg, 2007). E.g., in (Fan-Orzechowski and Feinberg, 2007), the goal is to maximize the average rewards per unit time subject to multiple constraints on the average costs per unit time. It is shown that an optimal policy is randomized.

In (Arapostathis et al., 2003), state feedback control of controlled Markov chains is studied, where the requirements specification is given as a unit-interval vector that represents an upper bound on the state probability vector. First, the set of state feedback controllers that satisfy the requirement for any given safe initial state probability distribution is identified, and, then, the set of all safe initial state probability distributions for a given state feedback controller is found. The work of (Arapostathis et al., 2005) further extends the results of (Arapostathis et al., 2003) to a more general class of Markov chains. Also, a safety requirement is given by two vectors representing lower and upper bounds on the state probability vector. Also, (Arapostathis et al., 2005) presents a more general iterative algorithm to find safe initial distributions, and provides the number of iterations needed for the result to be reached.

Controller synthesis for probabilistic systems has also attracted attention in the formal methods community. E.g., (Baier et al., 2004; Kučera and Stražovský, 2008) consider different control policies: deterministic or randomized (probabilistic) on one hand; memoryless (Markovian) or history-independent on the other. The systems considered are finite Markov decision processes with the state space divided into two disjoint sets: controllable states and uncontrollable states. In (Baier et al., 2004), the controller synthesis problem for a requirements specification given as a probabilistic computation tree logic (PCTL) formula is shown to be NP-hard, and a synthesis algorithm for automata specifications is presented. Controller synthesis was considered in (Kučera and Stražovský, 2008) for a requirements specification given as a formula of PCTL extended with long-run average propositions. It is shown that the existence of such a controller is decidable, and an algorithm for the synthesis of a controller, when it exists, is presented. Further, controller robustness with respect to slight changes in the probabilities of the plant is discussed. The paper shows that the existence of robust controllers is decidable and the controller, if it exists, is effectively computable.

Rabin's probabilistic automata are used in (Mortazavian, 1993) as the underlying model. A requirements specification is given as a cut-language, and the classical supervisory control definitions of controllability and observability are modified accordingly. E.g., a language is $\lambda$-controllable if the probability that uncontrollable events leave the set of legal strings legal is greater than $\lambda$. Necessary and sufficient conditions for the existence of a supervisor are given.

A deterministic supervisory control framework for stochastic discrete event systems was developed in (Kumar and Garg, 2001; Kumar and Garg, 1998a; Kumar and Garg, 1998b) using the model of (Garg et al., 1999; Garg, 1992a; Garg, 1992b). Controllable events are disabled dynamically as first suggested in (Lawford and Wonham, 1993), so that the probabilities of their execution become zero, and the probabilities of the occurrence of other events proportionally increase. The control objective considered is to construct a supervisor such that the controlled plant does not execute specified illegal traces, and occurrences of the legal traces in the system are greater than or equal to specified values. While (Kumar and Garg, 2001; Kumar and Garg, 1998a) give necessary and sufficient conditions for the existence of a supervisor, (Kumar and Garg, 2001; Kumar and Garg, 1998b) offer an algorithm to compute a maximally permissive supervisor on-line. An optimal supervisory theory of probabilistic systems was considered in (Li et al., 1998), where the system is allowed to violate the specification, but with a probability lower than a prespecified value. PDES are modeled as the deterministic version of probabilistic automata used in (Garg et al., 1999; Garg, 1992a; Garg, 1992b). In (Chattopadhyay and Ray, 2007a; Chattopadhyay and Ray, 2007b), the same model is used. The requirements specification is given by weights assigned to states of a plant and the control goal is, roughly speaking, to reach the states with more weight (more desirable states) more often. A deterministic control is synthesized for a given requirements specification so that a measure based on the specification and probabilities of the plant is optimized. In (Chattopadhyay et al., 2009), an algorithm for robot path planning is suggested based on this measure. Further, this measure-theoretic approach is generalized to partially observed probabilistic transition systems modeled as probabilistic generators in (Chattopadhyay and Ray, 2010).

The framework we adopt was first suggested in (Lawford and Wonham, 1993). PDES are modeled as the deterministic version of probabilistic automata from (Garg et al., 1999; Garg, 1992a; Garg, 1992b). As mentioned before, in the sense of probabilistic languages generated, it has been shown in (Lawford and Wonham, 1993) that probabilistic supervisory control is much more powerful: it can generate a much larger class of probabilistic languages than deterministic control. Hence, (Lawford and Wonham, 1993), (Postma and Lawford, 2004) investigate probabilistic supervisory control: conditions under which a probabilistic control can generate a prespecified probabilistic language, and, if the supervisor exists, the algorithm for its synthesis. Further, (Lawford and Wonham, 1993) gives the necessary and sufficient conditions for the existence of a supervisor for a class of PDES. The conditions reduce to checking whether certain linear equalities and inequalities hold. A formal proof of the necessity and sufficiency of the conditions and an algorithm for the calculation of the supervisor, if it exists, are presented in (Postma and Lawford, 2004).

## 1.3 Comparison with MDPs

MDPs represent a widely used framework for study of control of Markov chains. They have been applied in many fields (engineering, economics, statistics, control of epidemics). An MDP is a tuple $(S, A, \{A(s)|s \in S\}, P, c)$, where $S$ is a set of states, $A$ is a set of actions, $\{A(s)|s \in S\}$ is the family of nonempty subsets of $A$, where $A(s)$ is the set of admissible actions at a state $s$. Let $K$ be a set of admissible state-action pairs, $K = \{(s,a)|s \in S, a \in A(s)\}$. Then, $P$ is a transition function, $P : K \times S \to [0,1]$, such that for every $(s,a) \in K$, it holds that $\sum_{s' \in S} P((s,a),s') = 1.$ $P((s,a),s')$ is interpreted as the probability of the action $a$ taking the system from the state $s$ to state $s'$. The function $c$ is a one-stage cost function, $c : K \rightarrow \mathbb{R}$. Alternatively, a reward function can be defined.

Let $s_t$ be the state at time $t \in \mathbb{N}$. If the action $a$ is chosen ($(s_t, a)$ is an admissible pair), then the system moves to the state $s_{t+1}$ with the probability $P((s_t, a), s_{t+1})$, and with the cost $c(s_t, a)$. A *control policy* $\pi$ is a sequence $\{\pi_t\}$, where $\pi_t$ is the mapping $\pi_t : K \rightarrow [0, 1]$, such that $\pi_t(A(s_t)|h_t) = 1$, where $h_t$ is the *admissible history* of the form $h_t = (s_0, a_0, \ldots, s_{t-1}, a_{t-1}, s_t)$ with $(s_i, a_i) \in K$ for $i = 0, \ldots, t-1$ and $s_t \in S$. Note that the defined policy is probabilistic: the deterministic one is just its special case (the choice of action to be taken is deterministic). The period of time over which the system is observed is called the *planning (or decision making or control) horizon* $T$. It can be finite or infinite.

The optimal control problem is to find the admissible control policy such that a certain criterion is optimized (possibly with some additional constraints, as already mentioned). Different cost evaluation criteria have been used. Some of the most commonly used are total cost, discounted cost, average cost, and sample path average cost. For more details, the interested reader is referred to (Borkar, 1991; Arapostathis et al., 1993).

As already stated, compared to classical reactive models, a generative model is the more general one: for every state, it contains not only the information about relative probabilities of transitions on a particular event, but also information on the relative probabilities of transitions on different events (Schröder and Mateus, 2002; Glabbeek et al., 1995). Therefore, as a generative model contains more information than a reactive one, a direct transformation of a generative model to a reactive model would abstract from this information. However, a probabilistic generator can be transformed into a reactive model by the introduction of a special event $\tau$ (see Figure 3.3), with a state expansion factor of $O(|\Sigma|)$. The new model is an MDP, with a special event $\tau$.

Let $G$ be a probabilistic generator representing a plant, and $G_{MDP}$ its equivalent MDP. For every state $s \in S$ of $G_{MDP}$, the set $A(s)$ is a singleton as only one event is admissible from a state. That means that a controller in the

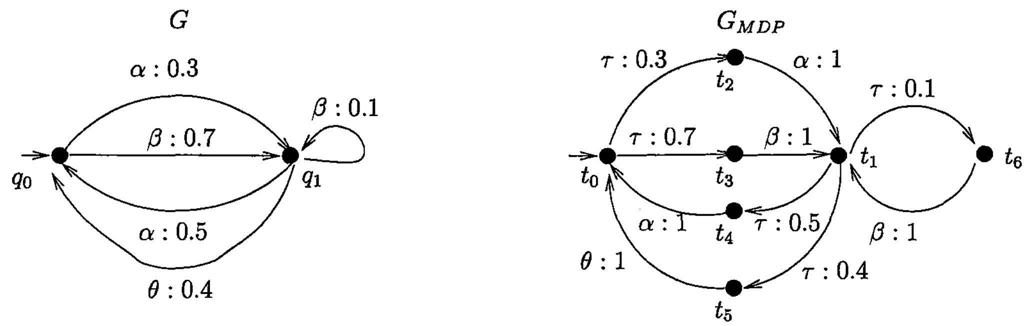

Figure 1.1: Transformation from a probabilistic generator to an MDP

MDP framework would always pick that one event available, and, consequently, would not be able to affect the probabilistic behaviour of the plant. Effectively, the control in MDP framework would not make sense for an MDP equivalent to a probabilistic generator.

## 1.4 Contributions of the Thesis

The contributions of this thesis are the following:

- The framework for probabilistic supervisory control of probabilistic discrete event systems, initiated in the work of (Lawford and Wonham, 1993; Postma and Lawford, 2004), is further refined by solving a special case previously unconsidered. The main results of (Lawford and Wonham, 1993; Postma and Lawford, 2004) are updated to reflect the special case. The formal proofs are reworked accordingly. Then, time complexity analyses of both the controller synthesis problem and the proposed synthesis algorithm are discussed.

- An optimal supervisory control problem inside the probabilistic framework is posed and solved. Again, as in classical supervisory theory, if there does not exist a (probabilistic) supervisor such that the controlled plant's behaviour can exactly match a prespecified probabilistic behaviour, a supervisor is synthesized such that the controlled plant's behaviour is "as close as possible" to the desired behaviour. The measure of proximity is a pseudometric on states of probabilistic systems. The distance between two generators in the pseudometric is the distance between the initial states of the generators.

- Significant results regarding the chosen pseudometric are presented. The pseudometric that is used to measure the proximity of two probabilistic systems (represented as two generators) is defined based on the pseudometric of (Deng et al., 2006) (more precisely, our pseudometric is equal to the pseudometric of (Deng et al., 2006) up to a constant). First, we simplify the fixed point characterization of the pseudometric for probabilistic generators. Then, we suggest two efficient algorithms for the calculation of distances in the pseudometric. Further, both logical and trace characterizations of the pseudometric are given. The logical characterization measures the distance between two systems by a real-valued formula that distinguishes between the systems the most. The trace characterization offers an insight into the similarity of (appropriately discounted) probabilistic traces of the systems (whose distance is measured by the pseudometric), certain sets of the traces and certain properties of the traces.

- Probabilistic model fitting is presented: a probabilistic generator can be approximated (under certain conditions) with another one with a prespecified structure such that the new representation is as close as possible to the original one in the pseudometric. Applications of ideas of model fitting are numerous, starting with PDES state space reduction. Then, the solution to model fitting is used to justify a criterion for the optimal probabilistic supervisory control problem of PDES. The most significant use of an idea of model fitting is in the solution of a modified optimal probabilistic supervisory control problem for PDES.

## 1.5 Thesis Outline

Now, we present the organization of the thesis.

Chapter 2 introduces PDES modeled as generators of probabilistic languages. Then, probabilistic control and the probabilistic supervisory control problem (PSCP, or the probabilistic model matching problem) are presented. Also, the intuition behind the necessary and sufficient conditions for the existence of a supervisor for solving the PSCP is given. Then, the concept of a pseudometric is explained, and a detailed literature review of pseudometrics on probabilistic transition systems is offered.

Chapter 3 presents the PSCP and its solution in detail. Also, the time complexity analyses of both the synthesis problem and algorithm are presented. Probabilistic supervisor is then represented using an MDP. The chapter further presents the formulation of the optimal probabilistic supervisory control problem (the closest approximation problem): if a supervisor for the PSCP does not exist, an optimal control should be synthesized such that the resulting controlled plant is as close as possible to the (appropriately modified) requirements specification. Also, an application of the research is depicted.

Chapter 4 first presents the pseudometric to be used in the solution of the OPSCP. Then, it derives and proves the correctness of two algorithms for the calculation of the distances between the states of a PDES in this pseudometric. Then, the pseudometric is also given a logical characterization and a trace characterization: these characterizations help understand the pseudometric itself.

Chapter 5 solves the optimal probabilistic supervisory control problem: the algorithm for finding the closest approximation within a prespecified accuracy is presented.

Chapter 6 introduces probabilistic model fitting, and some of its main applications. The main application of model fitting presented is that a slightly different optimal probabilistic supervisory control problem is solved by a straightforward modification of the OPSCP algorithm derived in Chapter 5.

Chapter 7 concludes the thesis and offers avenues for future work.

*Note:* Most of this thesis has already been published. The material of Chapter 3 (except for Section 3.2 and Section 3.4) was introduced in (Pantelic et al., 2009; Pantelic et al., 2008). Chapters 4 and 5 present the work that was published in (Pantelic and Lawford, 2010b; Pantelic and Lawford, 2009; Pantelic and Lawford, 2010a; Pantelic and Lawford, 2011). Finally, the contents of the Chapter 6 were presented in (Pantelic and Lawford, 2010b), and (Pantelic and Lawford, 2011).

# Chapter 2

# Preliminaries

In this chapter, we first set the notation to be used throughout the thesis (Section 2.1). Next, PDES modeled as generators of probabilistic languages are presented in Section 2.2. The method of probabilistic control (random disablement) is introduced in Section 2.3. Then, the probabilistic supervisory control problem is presented, and its solution is informally sketched in Section 2.4. In Section 2.5, transformation of terminating to nonterminating generators is given. The notion of a pseudometric as a dominant concept in the solution of the optimal probabilistic supervisory control problem is introduced in Section 2.6, and the related literature is reviewed.

## 2.1 Notation

Small Greek letters will be used to denote events and capital Greek letters will denote sets of events. To denote sets whose elements are not necessarily events, capital Roman letters will be used, and small Roman letters will denote functions. Given sets $A$, and $B$, the power set of $A$ will be denoted by $\mathcal{P}(A)$, and the set difference of $A$ and $B$ by $A \backslash B$. Also, we assume the set difference operation to be left-associative. Further, the set of functions from $A$ to $B$ will be denoted as $B^A$.

## 2.2 Modeling PDES

Following (Lawford and Wonham, 1993), a probabilistic DES can be modeled as a probabilistic generator $G = (Q, \Sigma, \delta, q_0, p)$, where $Q$ is the nonempty finite set of states, $\Sigma$ is a finite alphabet whose elements we will refer to as event labels, $\delta : Q \times \Sigma \rightarrow Q$ is the (partial) transition function, $q_0 \in Q$ is the initial state, and $p : Q \times \Sigma \rightarrow [0,1]$ is the statewise event probability distribution, i.e. for any $q \in Q$, $\sum_{\sigma\in\Sigma} p(q,\sigma) \leq 1$. The probability that the event $\sigma \in \Sigma$ is going to occur at the state $q \in Q$ is $p(q,\sigma)$. For the generator $G$ to be well-defined, $(i)$ $p(q,\sigma) = 0$ should hold if and only if $\delta(q,\sigma)$ is undefined and $(ii)$ $\forall q \sum_{\sigma\in\Sigma} p(q,\sigma) \leq 1$. The probabilistic generator $G$ is nonterminating if, for every reachable state $q \in Q$, $\sum_{\sigma\in\Sigma} p(q,\sigma) = 1$. Conversely, $G$ is terminating if there is at least one reachable state $q \in Q$ such that $\sum_{\sigma\in\Sigma} p(q,\sigma) < 1$. The probability that the system terminates at state $q$ is $1 - \sum_{\sigma\in\Sigma} p(q,\sigma)$. Throughout the sequel, unless stated otherwise, we assume nonterminating generators. If a PDES is terminating, it can easily be transformed into a nonterminating one using the technique described in Section 2.5.

The state transition function is traditionally extended by induction on the length of strings to $\delta : Q \times \Sigma^* \rightarrow Q$ in a natural way. For a state $q$, and a string $s$, the expression $\delta(q,s)!$ will denote that $\delta$ is defined for the string $s$ in the state $q$. Note that the definition of probabilistic generators does not contain marking states since the probabilistic specification languages considered in this thesis are prefix closed languages.

The language $L(G)$ generated by a probabilistic DES generator $G = (Q, \Sigma, \delta, q_0, p)$ is $L(G) = \{s \in \Sigma^* \mid \delta(q_0, s)!\}$. The probabilistic language generated by $G$ is defined as:

$$L_p(G)(\epsilon) = 1,$$

$$L_p(G)(s\sigma) = \begin{cases} L_p(G)(s) \cdot p(\delta(q_0,s),\sigma), & \text{if } \delta(q_0,s)! \\ 0, & \text{otherwise.} \end{cases}$$

Informally, $L_p(G)(s)$ is the probability that the string $s$ is executed in $G$. Also, $L_p(G)(s) > 0$ iff $s \in L(G)$.

For each state $q \in Q$, we define the function $\rho_q : \Sigma \times Q \to [0,1]$ such that for any $q' \in Q$, $\sigma \in \Sigma$, we have $\rho_q(\sigma, q') = p(q, \sigma)$ if $q' = \delta(q, \sigma)$, and 0 otherwise. The function $\rho_q$ is a probability distribution on the set $\Sigma \times Q$ induced by $q$. Also, for a state $q$, we define *the set of possible events* to be $Pos(q) := \{\sigma \in \Sigma | p(q, \sigma) > 0\}$.

Next, the synchronous product of (nonprobabilistic) discrete event systems (DES) that underlie PDES is defined in a standard manner. For a probabilistic generator $G = (Q, \Sigma, \delta, q_0, p)$, the (nonprobabilistic) discrete event system (DES) that underlies $G$ will be denoted $G^{np}$, i.e., $G^{np} = (Q, \Sigma, \delta, q_0)$ throughout this thesis. Let $G_1^{np}$ and $G_2^{np}$ be the nonprobabilistic generators (DES) underlying $G_1 = (Q_1, \Sigma, \delta_1, q_{0_1}, p_1)$ and $G_2 = (Q_2, \Sigma, \delta_2, q_{0_2}, p_2)$, respectively, i.e., $G_1^{np} = (Q_1, \Sigma_1, \delta_1, q_{0_1})$ and $G_2^{np} = (Q_2, \Sigma, \delta_2, q_{0_2})$.

**Definition 2.1.** *The synchronous product of $G_1^{np} = (Q_1, \Sigma, \delta_1, q_{0_1})$ and $G_2^{np} = (Q_2, \Sigma, \delta_2, q_{0_2})$, denoted $G_1^{np} \parallel G_2^{np}$, is the reachable sub-DES of DES $G_a = (Q_a, \Sigma, \delta, q_0)$, where $Q_a = Q_1 \times Q_2$, $q_0 = (q_{0_1}, q_{0_2})$, and, for any $\sigma \in \Sigma$, $q_i \in Q_i$, $i = 1, 2$, it holds that $\delta((q_1, q_2), \sigma) = (\delta_1(q_1, \sigma), \delta_2(q_2, \sigma))$ whenever $\delta_1(q_1, \sigma)!$ and $\delta_2(q_2, \sigma)!$.*

## 2.3 Probabilistic Supervisory Control of PDES

As in classical supervisory control theory, the set $\Sigma$ is partitioned into $\Sigma_c$ and $\Sigma_u$, the sets of controllable and uncontrollable events, respectively. Also, let plant $G$ under the supervision of supervisor $V$ be denoted as $V/G$. Given probabilistic generator $G_2$ of a probabilistic specification language $E$ (i.e. $L_p(G_2) = E$) and probabilistic generator $G$ of probabilistic language $L_p(G)$ representing a plant, the goal is to find a supervisor $V$ such that the language generated by the plant under supervision, $L_p(V/G)$, is equal to $E$. After observing a string $s$, a classical, deterministic supervisor must consistently either enable or disable each controllable event $\sigma \in \Sigma_c$. Let $G = (Q, \Sigma, \delta, q_0, p)$. Then, a deterministic supervisor can be defined using a function $V : L(G) \to \{0, 1\}^{\Sigma}$, where:

$$(\forall s \in L(G))(\forall \sigma \in \Sigma)V(s)(\sigma) = \begin{cases} 1, & \text{if } \sigma \in \Sigma_u \text{ or } s\sigma \in E \\ 0, & \text{otherwise.} \end{cases}$$

We now explore the limited effect a classical supervisor can have on a PDES. Figure 2.1 shows two PDES: the first one, $G$, represents a plant, and the second one, $G_2$, is a requirements specification. Controllable events are marked with a bar on their edges. A number next to an event represents the probability distribution of that event. $G$ has alphabet $\Sigma = \{\alpha, \beta, \gamma\}$ and is nonterminating. The event $\gamma$ is uncontrollable, and, therefore, always enabled. An important assumption about the behaviour of a supervisor is made: *After an event is disabled, the probabilities of the remaining enabled events proportionally increase.* The question to be answered is: Does there exist a deterministic supervisor $V$ such that $L_p(V/G) = L_p(G_2)$?

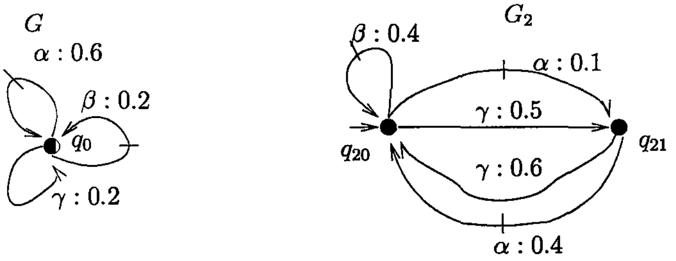

Figure 2.1: Plant $G$ and requirements specification $G_2$

We first consider the case when the PDES $G$ is in state $q_0$ and the PDES $G_2$ is in state $q_{20}$. The required probabilities of all events in state $q_{20}$

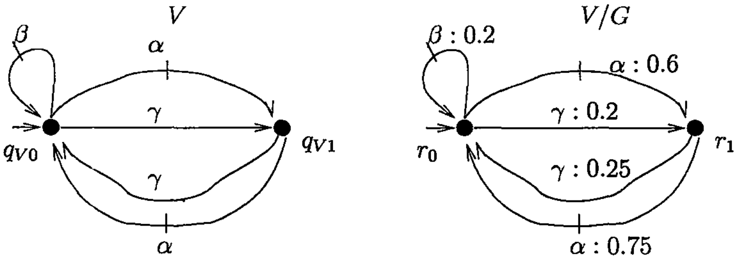

Figure 2.2: Deterministic supervisor $V$ and controlled plant $V/G$

are nonzero. Therefore, the deterministic supervisor $V$ should enable all (controllable) events (state $q_{V_0}$ of DES $V$ in Figure 2.2). Hence, the probabilities of events in the controlled plant remain unchanged (see state $r_0$ of the controlled plant $V/G$ in Figure 2.2).

Next, after an odd number of $\alpha$ or $\gamma$ events (PDES $G$ is in state $q_0$ and PDES $G_2$ is in state $q_{21}$), the supervisor should disable $\beta$. When $V$ disables only $\beta$, the plant can choose between $\alpha$ and $\gamma$. The probabilities of these events occurring in the resulting system are increased proportional to their original probabilities. Therefore, the probability of $\alpha$ occurring in state $r_1$ of the controlled plant is equal to:

$$P(\alpha|\sigma \in \{\gamma, \alpha\}) = \frac{p(q_0, \alpha)}{p(q_0, \alpha) + p(q_0, \gamma)} = \frac{0.6}{0.6 + 0.2} = 0.75$$

Similarly, the probability of $\gamma$ occurring is 0.25.

Therefore, although the requirement was met nonprobabilistically (meaning $L(V/G) = L(G_2)$), it is obvious that there is no deterministic control such that $L_p(V/G) = L_p(G_2)$. This example illustrates that application of deterministic supervisors to PDES results in a rather limited class of probabilistic languages. Hence, applying a deterministic supervisor to a PDES might be unacceptable for a designer.

Now, deterministic supervisors for DES are generalized to *probabilistic supervisors*. The control technique used is called *random disablement*. Instead of deterministically enabling or disabling controllable events, probabilistic supervisors enable events with certain probabilities. This means that, upon reaching a certain state $q$, the control pattern (the set of events to be enabled) is chosen according to the supervisor's probability distributions of controllable events. Consequently, the controller does not always enable the same events when in the state $q$.

For a PDES $G = (Q, \Sigma, \delta, q_0, p)$, a *probabilistic supervisor* is a function $V_p : L(G) \rightarrow [0, 1]^{\Sigma}$ such that for $s \in L(G), \sigma \in \Sigma$:

$$V_p(s)(\sigma) = \begin{cases} 1, & \text{if } \sigma \in \Sigma_u \\ x(s)(\sigma), & \text{otherwise, where } x(s)(\sigma) \in [0, 1]. \end{cases}$$

Therefore, after observing a string $s$, the supervisor enables the event $\sigma$ with probability $V_p(s)(\sigma)$. After a set of controllable events to be enabled, $\Theta$, has been decided upon (uncontrollable events are always enabled), the system acts as if supervised by a deterministic supervisor. An example of a probabilistic supervisor is given in Figure 2.3. Note that the probabilities of all the events that can execute in a state of this generator do not, in general, add up to 1. This is because those are not the probabilities of events occurring, but rather being enabled.

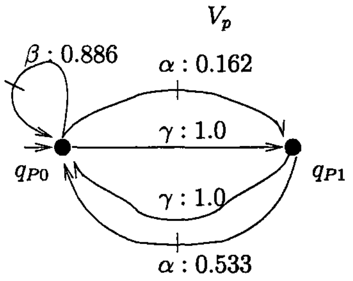

Figure 2.3: Probabilistic supervisor $V_p$

What is the probability that an event $\alpha$ will occur in a plant $G$ under the control of probabilistic supervisor $V_p$ when the string $s \in L(G)$ has been observed? First, the control pattern is chosen according to the controllable event probabilities of the supervisor, and then, under that pattern, the plant makes a choice according to its events probabilities. Let $q \in Q$ be the state of the plant after $s$.

The probability that the event $\alpha \in \Sigma$ will occur after string $s$ has been observed is equal to:

$$\begin{aligned} & P(\alpha \text{ in } V_p/G|s) \\ & = \sum_{\Theta\in\mathcal{P}(Pos(q)\cap\Sigma_c)} P(\alpha|V_p \text{ enables } \Theta \text{ after } s)\cdot P(V_p \text{ enables } \Theta|s) \end{aligned} \tag{2.1}$$

where

$$P(\alpha|V_p \text{ enables } \Theta \text{ after } s) = \begin{cases} \dfrac{p(q,\alpha)}{\sum\limits_{\sigma\in\Theta\cup\Sigma_u} p(q,\sigma)}, & \text{if } \alpha\in\Theta\cup\Sigma_u \\ 0, & \text{otherwise} \end{cases}$$

$$P(V_p \text{ enables } \Theta|s) = \prod_{\sigma\in\Theta} V_p(s)(\sigma) \quad \cdot \prod_{\sigma\in(Pos(q)\cap\Sigma_c)\setminus\Theta} (1 - V_p(s)(\sigma)).$$

For the probabilistic controller from Figure 2.3, the possible control patterns in state $q_{P0}$ are $\Theta = \emptyset$, $\Theta = \{\alpha\}$, $\Theta = \{\beta\}$, and $\Theta = \{\alpha, \beta\}$. Let $P_\sigma(s)$ be the probability of $\sigma$ occurring in the plant under control after the string $s$ is observed, $P_\sigma(s) = P(\sigma \in V_p/G|s)$. If we apply (2.1) to the controller from Figure 2.3 for a string $s$ such that the controller is at state $q_{P0}$, then :

$$\begin{aligned}
P_\alpha(s) &= p(q_0,\alpha)V_p(s)(\alpha)V_p(s)(\beta) + P(\alpha|\sigma \in \{\gamma,\alpha\})V_p(s)(\alpha)(1 - V_p(s)(\beta)) \\
&= 0.6(0.162)(0.886) + 0.75(0.162)(1 - 0.886) \qquad (2.2) \\
&= 0.1
\end{aligned}$$

$$\begin{aligned}
P_\beta(s) &= p(q_0,\beta)V_p(s)(\alpha)V_p(s)(\beta) + P(\beta|\sigma \in \{\gamma,\beta\})(1 - V_p(s)(\alpha))V_p(s)(\beta) \\
&= 0.2(0.162)(0.886) + 0.5(1 - 0.162)(0.886) \qquad (2.3) \\
&= 0.4 \\
P_\gamma(s) &= p(q_0,\gamma)V_p(s)(\alpha)V_p(s)(\beta) + P(\gamma|\sigma \in \{\gamma,\alpha\})V_p(s)(\alpha)(1 - V_p(s)(\beta)) \\
&\quad + P(\gamma|\sigma \in \{\gamma,\beta\})(1 - V_p(s)(\alpha))V_p(s)(\beta) \\
&\quad + P(\gamma|\sigma \in \{\gamma\})(1 - V_p(s)(\alpha))(1 - V_p(s)(\beta)) \qquad (2.4) \\
&= 0.2(0.162)(0.886) + 0.25(0.162)(1 - 0.886) \\
&\quad + 0.5(1 - 0.162)(0.886) + 1.0(1 - 0.162)(1 - 0.886) \\
&= 0.5
\end{aligned}$$

Similarly, for a string $t$ that corresponds to the supervisor in Figure 2.3 being at state $q_{P1}$, we have $P_\alpha(t) = 0.4$, $P_\beta(t) = 0$ and $P_\gamma(t) = 0.6$. Therefore, the plant under probabilistic control succeeds in generating the probabilistic language $L_p(G_2)$ from Figure 2.1 whereas a deterministic controller failed. However, in the general case, for a given plant, there might not exist a probabilistic supervisor for a given probabilistic specification language. In the next section, we will explore the conditions under which a probabilistic supervisor exists.

## 2.4 Probability Matching

The goal is to match the behaviour of the controlled plant with a given probabilistic specification language. We call this problem the *Probabilistic Super-*

*visory Control Problem (PSCP).* More formally:
*Given a plant PDES $G_p$ and a requirements specification PDES $G_r$, find, if possible, a probabilistic supervisor $V_p$ such that $L_p(V_p/G_p) = L_p(G_r)$.*

Let $G = (Q, \Sigma, \delta, q_0, p)$ be the plant as seen in Figure 2.1. Let $x(s) \in [0,1]^{Pos(q)\cap\Sigma_c}$ be the control input after string $s$ has been observed, and the plant is in state $q \in Q$.

First, we note that the ratio between the probabilities of two uncontrollable events in an uncontrolled plant should remain the same in the plant under control. Here is the informal reasoning for this claim. Let $\alpha, \beta \in \Sigma_u$. We compare the values of $P(\alpha \in V_p/G|s)$ and $P(\beta \in V_p/G|s)$ when calculated using (2.1). For a control policy $\Theta$, the value of $P(V_p \text{ enables } \Theta|s)$ is the same for both $\alpha$ and $\beta$. Also, the factor $1/\sum_{\sigma\in\Theta\cup\Sigma_u} p(q,\sigma)$ is the same for both events. Therefore, the only distinguishing factor is $p(q,\sigma)$. The formal proof of this claim will be presented in Section 3.1.2.

Further, we consider controllable events. We slightly abuse the notation for $P_\sigma(s)$ in order to explicitly relate $P_\alpha(s)$ and $P_\beta(s)$ of (2.2) and (2.3) to supervisor probabilities, $x(s)(\alpha)$ and $x(s)(\beta)$. If we apply (2.1) to the plant $G$ from Figure 2.1, then:

$$\begin{aligned}
P_\alpha(x(s)(\alpha), x(s)(\beta)) &= 0.6x(s)(\alpha)x(s)(\beta) + 0.75x(s)(\alpha)(1 - x(s)(\beta)) \\
&= 0.75x(s)(\alpha) - 0.15x(s)(\alpha)x(s)(\beta) \qquad (2.5) \\
P_\beta(x(s)(\alpha), x(s)(\beta)) &= 0.2x(s)(\alpha)x(s)(\beta) + 0.5(1 - x(s)(\alpha))x(s)(\beta) \\
&= 0.5x(s)(\beta) - 0.3x(s)(\alpha)x(s)(\beta) \qquad (2.6)
\end{aligned}$$

For notational convenience, in the sequel of this section, let us denote $P_\sigma(x(s)(\alpha), x(s)(\beta))$ with $P_\sigma$, where $\sigma \in \{\alpha, \beta, \gamma\}$. Our goal is to find constraints on $P_\alpha$ and $P_\beta$ such that (2.5) and (2.6) are satisfied, and $(x(s)(\alpha), x(s)(\beta)) \in [0,1] \times [0,1]$. To map this region from the $(x(s)(\alpha), x(s)(\beta))$ plane to the $(P_\alpha, P_\beta)$ plane (as shown at Figure 2.4), we use the following logic. If $x(s)(\alpha)$ is equal to 0, then, according to (2.5) and (2.6), $P_\alpha = 0$ ($\alpha$ is disabled), and $P_\beta = 0.5x(s)(\beta)$. So, in this case, $P_\alpha = 0$ and $P_\beta \in [0, 0.5]$. Similarly, when $x(s)(\beta) = 0$, then $P_\beta = 0$ and $P_\alpha \in [0, 0.75]$. For $x(s)(\alpha) = 1$, we solve for $x(s)(\beta)$ in one of (2.5) or (2.6) and substitute it into other equation to get

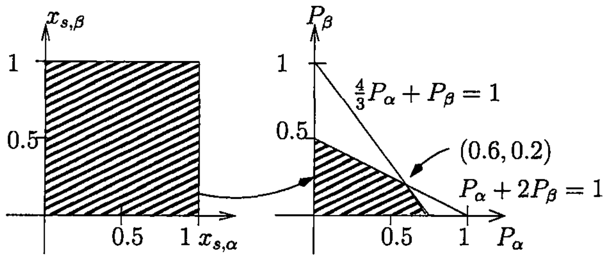

Figure 2.4: Mapping from $(x(s)(\alpha), x(s)(\beta))$ to $(P_\alpha, P_\beta)$ plane

$\frac{4}{3}P_\alpha + P_\beta = 1$. Similarly, for $x(s)(\beta) = 1$, we get $P_\alpha + 2P_\beta = 1$. Those two lines intersect at the point $(0.6, 0.2)$, that corresponds to the probabilities of $\alpha$ and $\beta$ in the original, uncontrolled system.

There exists another way to derive those bounds. Since $G$ is nonterminating and the controller never disables all the events, then

$$P_\alpha + P_\beta + P_\gamma = 1. \tag{2.7}$$

Since $\gamma$ is uncontrollable, then $V_p(s)(\gamma) = 1$. Let us consider a case when $x(s)(\alpha) = 1$. This means that $\alpha$ is effectively uncontrollable, so:

$$\frac{P_\gamma}{P_\alpha} = \frac{p(q,\gamma)}{p(q,\alpha)} = \frac{1}{3}$$

Also, as $x(s)(\alpha)$ decreases, $P(\alpha)$ decreases too. Therefore:

$$P_\gamma \geq \frac{1}{3}P_\alpha$$

with equality holding when $x(s)(\alpha) = 1$. We plug this back into (2.7) to get $\frac{4}{3}P_\alpha + P_\beta \leq 1$. Similarly, if we assume that $x(s)(\beta) = 1$, we get $P_\alpha + 2P_\beta \leq 1$.

## 2.5 Handling terminating PDES

The results and proofs presented in this thesis apply only to nonterminating systems. We now present their extension to terminating systems as introduced in (Lawford and Wonham, 1993). The terminating PDES $G = (Q, \Sigma, \delta, q_0, p)$ can be extended to nonterminating $G' = (Q \cup \{q_\perp\}, \Sigma \cup \{\sigma_\perp\}, \delta', q_0, p')$, where

$$\begin{array}{rlcrl} p'(q,\sigma) &= p(q,\sigma), \sigma \in \Sigma & & \delta'(q,\sigma) &= \delta(q,\sigma), \sigma \in \Sigma \\ p'(q,\sigma_\perp) &= 1 - \sum_{\sigma\in\Sigma} p(q,\sigma) & \text{and} & \delta'(q,\sigma_\perp) &= q_\perp \text{ if } p'(q,\sigma_\perp) > 0 \\ p'(q_\perp,\sigma_\perp) &= 1 & & \delta'(q_\perp,\sigma_\perp) &= q_\perp. \end{array}$$

An example of a terminating generator and its extension to a nonterminating one is shown in Figure 2.5.

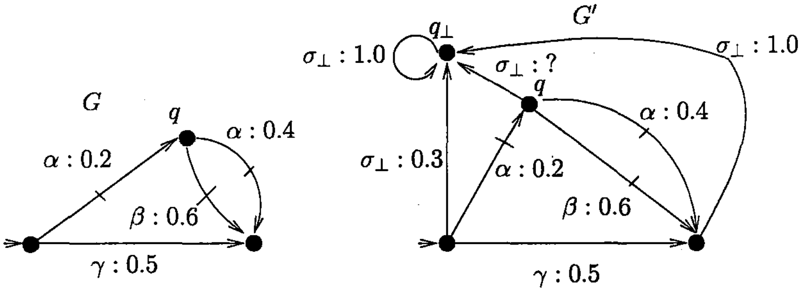

Figure 2.5: Terminating PDES $G$ and resulting nonterminating PDES $G'$

There is one obstacle to a straightforward extension of presented results to terminating generators. At the state $q$, as depicted in Figure 2.5, all the events that can occur are controllable. Then, the termination in the controlled plant can come about not only as a consequence of a termination in the original plant, but also as a consequence of the controller disabling all the (controllable) events. This problem is addressed in Section 3.1.2.

## 2.6 Probabilistic Pseudometrics

Central to the solution of the optimal probabilistic supervisory control problem is the concept of a pseudometric. The concept is defined next, and the relevant literature is reviewed.

*Probabilistic bisimulation*, introduced in (Larsen and Skou, 1991), is commonly used to define an equivalence relation between probabilistic systems. However, probabilistic bisimulation is not a robust relation: roughly speaking, two states of probabilistic systems are bisimilar if and only if they have the same transitions with exactly the same probabilities to states in the same equivalence classes. The formal definition follows and represents a modified version of the definition of probabilistic bisimulation given in (Barrett and Lafortune, 1997).

**Definition 2.2.** *Let $G = (Q, \Sigma, \delta, q_0, p)$ be a PDES. A probabilistic bisimulation on $Q$ is the binary relation $\equiv$ such that for any $\sigma \in \Sigma$, and any $q_1, q_2 \in Q$ such that $q_1 \equiv q_2$, the following holds:*

1. *For every $q_1' \in Q$ such that $\delta(q_1, \sigma) = q_1'$, there is $q_2' \in Q$ such that $\delta(q_2, \sigma) = q_2'$, $p(q_1, \sigma) = p(q_2, \sigma)$, and $q_1' \equiv q_2'$.*

2. *For every $q_2' \in Q$ such that $\delta(q_2, \sigma) = q_2'$, there is $q_1' \in Q$ such that $\delta(q_1, \sigma) = q_1'$, $p(q_1, \sigma) = p(q_2, \sigma)$, and $q_1' \equiv q_2'$.*

*States $q_1$ and $q_2$ are probabilistic bisimilar if there exists a probabilistic bisimulation $\equiv$ such that $q_1 \equiv q_2$.*

As a more robust way to compare probabilistic systems, a notion of pseudometric is introduced. A pseudometric on a set of states $Q$ is a function $d : Q \times Q \to \mathbb{R}$ that defines a distance between two elements of $Q$, and satisfies the following conditions: $d(x, y) \geq 0$, $d(x, x) = 0$, $d(x, y) = d(y, x)$, and $d(x, z) \leq d(x, y) + d(y, z)$, for any $x, y, z \in Q$. A pseudometric generalizes a metric in that two distinct points are allowed to be at the distance 0. If all distances are in $[0, 1]$, the pseudometric is *1-bounded.* In the sequel, we will use the terms metric and pseudometric interchangeably.

Little work on metrics has focused on generative models. The first paper that discussed the use of a metric as a way to measure the distance between two probabilistic processes is (Giacalone et al., 1990). This early work considers deterministic generative probabilistic systems. The distance between processes is a number between 0 and 1, and represents a measure of a behavioural proximity between the processes: the smaller the number, the smaller the distance. The work of (Garg et al., 1999) suggests a metric based on probabilities of occurrence of strings in languages generated by two automata. More precisely, the distance between two automata in the metric is defined as a maximal difference in occurrence probabilities of strings in the corresponding languages. Probabilistic generators are used to model probabilistic systems in (Chattopadhyay and Ray, 2008). In a symbolic pattern recognition application, a metric is introduced to measure the distance between the original model and the transformed one, where the transformed model has the same long term distribution over the states as the original one.

The work of (Deng et al., 2006) introduces a pseudometric on states for a large class of probabilistic automata, including reactive and generative probabilistic automata. Further, this metric is inspired by the Kantorovich metric (Kantorovich, 1942) which is used in transport problems, and more recently has been used by Hutchinson in his theory of fractals (Hutchinson, 1981). The metric is also known as Wasserstein metric (Wasserstein, 1969). The metric is characterized as the greatest fixed point of a function. Two states are at distance 0 in this metric if and only if they are probabilistic bisimilar. They also introduce two process calculi, and show that process combinators are non-expansive: they do not increase distance. This is the metric we will use in the solution of our problem. For reactive systems, the work of (Deng et al., 2006) is closely related to (Desharnais et al., 1999; Desharnais et al., 2002; Desharnais et al., 2004; van Breugel and Worrell, 2001b; van Breugel and Worrell, 2001a; van Breugel and Worrell, 2005; van Breugel et al., 2005; van Breugel and Worrell, 2006; Ferns et al., 2004; Ferns et al., 2005; Ferns et al., 2006).

The work of (Desharnais et al., 1999) considers partial labeled Markov chains (Markov decision processes of Section 1.3, possibly with termination, and no cost function defined). The motivation of the paper is to explore the possibility of substituting one process with another that is sufficiently close in a metric space. Two states are at distance 0 in this metric if and only if they are probabilistic bisimilar. The pseudometric is given via a real-valued logic that is motivated by the well-known result that the Hennessy-Milner logic is complete for bisimulation (Arnold, 1994). More concretely, the ideas of (Kozen, 1985) are used to generalize a logic so that reasoning about probabilistic systems is supported. Let $\mathcal{F}$ be a set of functions such that a function $f \in \mathcal{F}$ evaluated at a state takes truth values in the interval $[0, 1]$, instead of $\{0, 1\}$. Then, the distance between two states is defined as a pseudometric:

$$d(q_q, q_r) = sup\{|f(q_q) - f(q_r)| f \in \mathcal{F}\}.$$

This paper also offers an algorithm to calculate a distance between two systems in the introduced metric with a prespecified accuracy. The algorithm runs in exponential time. Further, asymptotic metrics have been considered as a side topic. Also, compositional reasoning is shown to be possible by showing the non-expansiveness of process combinators. Non-expansiveness means that the distance between processes does not increase when they are put in the same context.

An extension of (Desharnais et al., 1999) is given in (Desharnais et al., 2004). It considers not only discrete probabilistic systems (partial labeled Markov chains of (Desharnais et al., 1999)), but also continuous systems (labeled Markov processes). A pseudometric analogue to weak bisimulation is presented in (Desharnais et al., 2002) for labeled concurrent Markov chains: the set of states is divided into two disjoint sets of probabilistic and nondeterministic states. The transitions from probabilistic states are not labeled, while the transitions from nondeterministic states are. Two systems can be differently designed to satisfy the same requirement: they would use different actions to reach the same goal. Then, it would not make sense to compare them logically, as they would be completely different, and use of the metric analogue of strong bisimulation would not be appropriate. However, if internal actions are abstracted away using an appropriate metric analogue of weak bisimulation, the systems would be similar in the logical, metric, and quantitative senses. The metric is given a fixed-point characterization that allows for a coinductive proof.

A pseudometric as a measure of behavioral similarity between the systems is suggested in (van Breugel and Worrell, 2001a; van Breugel and Worrell, 2006) for reactive nonlabeled probabilistic systems, but the results can be easily generalized to labeled reactive systems (partial labeled Markov chains) (van Breugel and Worrell, 2005). The pseudometric is coalgebraic. An algorithm to approximate (with a prespecified accuracy) the distances in the presented pseudometric between states of a system is presented. The algorithm is polynomial and uses linear programming techniques. Comparison of this metric with a number of metrics (e.g., (de Vink and Rutten, 1999; Norman, 1997; Baier and Kwiatkowska, 2000; den Hartog, 1998)) is given in (van Breugel and Worrell, 2001b). As opposed to the metric (de Vink and Rutten, 1999) (and also those of (Baier and Kwiatkowska, 2000) and (den Hartog, 1998)), the metric of (van Breugel and Worrell, 2001a) reflects the fact that two systems with probabilities of corresponding transitions only slightly different, are metrically very close. The distance between systems in the metric of (Norman, 1997) can be zero, even though the systems are not probabilistic bisimilar. Also, the metric of (van Breugel and Worrell, 2001a) can be recovered from the metric of (Desharnais et al., 1999) by adding negation to the logic of (Desharnais et al., 1999) (see (van Breugel and Worrell, 2001b)). The work of (Ferns et al., 2004; Ferns et al., 2005; Ferns et al., 2006) builds on the aforementioned bulk of research closely related to the pseudometric of (Deng et al., 2006) by considering pseudometrics on states of MDPs, in optimization context.

In the probabilistic model checking literature, the model of (van Breugel and Worrell, 2001a) is extended with a state labeling function that assigns to each state a set of atomic propositions valid in that state, e.g., *probabilistic transition systems* of (van Breugel and Worrell, 2001a) are extended to *(labeled) discrete time Markov chains* of (Rutten et al., 2004; Kwiatkowska et al., 2007) (although, the latter does not allow for termination, while the former does). Our generative models can be transformed to (labeled) discrete time Markov chains of (Rutten et al., 2004; Kwiatkowska et al., 2007), but with a state space expansion by a factor of $O(|\Sigma|)$. Also, with the same state expansion factor, our generators can be converted to the *labeled concurrent Markov chains* of (Desharnais et al., 2002) and *partial labeled Markov chains* of (Desharnais et al., 1999; Desharnais et al., 2004). However, as the current mathematical apparatus allows for direct reasoning about distance between our generators, no benefits in regards to the optimal supervisory control of PDES would have been gained by a transformation to one of the aforementioned models. Furthermore, keeping generators as the primary model allows for a smooth integration of use of classical supervisory control theory and probabilistic methods.

# Chapter 3

# The Framework

In (Lawford and Wonham, 1993), the probabilistic supervisory control problem (PSCP, or the probability matching problem) is formulated, and necessary and sufficient conditions for the existence of a probabilistic supervisor such that a given specification is satisfied are given. The results of that work were presented in Chapter 2. In this chapter, we present the material first introduced in (Postma and Lawford, 2004; Pantelic et al., 2009; Pantelic et al., 2008). The core of the material was first presented in (Postma and Lawford, 2004), where the formal proof of necessity and sufficiency of the conditions for the existence of a probabilistic supervisor for the PSCP is given, and an algorithm for synthesis of a supervisor is presented. Then, in (Pantelic et al., 2009), the main results of (Lawford and Wonham, 1993) and (Postma and Lawford, 2004) are modified to include a special case when only controllable events can occur from a state in a plant. This case has not been explicitly handled in any of the previous work. Further, the time complexity analyses of both the controller synthesis problem and the synthesis algorithm are given. The work of (Pantelic et al., 2008) represents a long version of (Pantelic et al., 2009), with a detailed formal proof being a reworked version of proofs from (Postma and Lawford, 2004). This is the proof that will be presented in this chapter. Then, in Section 3.2, previously unpublished material on a probabilistic supervisor being modeled by a reactive model is introduced. In Section 3.3, the OPSCP problem is formulated. At last, an application is presented in Section 3.4.

## 3.1 Solution of the PSCP

In Chapter 2, the intuition behind the conditions for the existence of a probabilistic supervisor (that were first presented in (Lawford and Wonham, 1993)) was given. Now, in Section 3.1.1, the conditions are presented formally together with an algorithm for the computation of the supervisor which is the main result of (Postma and Lawford, 2004). A detailed formal proof of the results of Section 3.1.1 is presented in Section 3.1.2. The modification of the results of (Lawford and Wonham, 1993) and (Postma and Lawford, 2004) that accounts for the special case when all events possible from a state are controllable is discussed in detail in this section. An example is presented in Section 3.1.3. The complexity analysis of both the synthesis problem and its solution is given in Section 3.1.4.

### 3.1.1 Supervisor for PSCP: Existence and Synthesis

First, we state necessary and sufficient conditions for the existence of a solution to the PSCP problem for nonterminating PDES.

**Theorem 3.1.** *Let $G_p = (Q_p, \Sigma, \delta_p, q_{p0}, p_p)$ and $G_r = (Q_r, \Sigma, \delta_r, q_{r0}, p_r)$ be two nonterminating PDES with disjoint state sets $Q_p$ and $Q_r$. There exists a probabilistic supervisor $V_p$ such that $L_p(V_p/G_p) = L_p(G_r)$ iff for all $s \in L(G_r)$ there exists $q \in Q_p$ such that $\delta_p(q_{p0}, s) = q$ and, letting $r = \delta_r(q_{r0}, s)$, the following two conditions hold:*

**(i)** *$Pos(q) \cap \Sigma_u = Pos(r) \cap \Sigma_u$, and for all $\sigma \in Pos(q) \cap \Sigma_u$,*

$$\frac{p_p(q,\sigma)}{\sum\limits_{\alpha\in\Sigma_u} p_p(q,\alpha)} = \frac{p_r(r,\sigma)}{\sum\limits_{\alpha\in\Sigma_u} p_r(r,\alpha)}$$

**(ii)** *$Pos(r) \cap \Sigma_c \subseteq Pos(q) \cap \Sigma_c$, and, if $Pos(q) \cap \Sigma_u \neq \emptyset$, then for all $\sigma \in Pos(q) \cap \Sigma_c$,*

$$\frac{p_r(r,\sigma)}{p_p(q,\sigma)} \sum_{\alpha\in\Sigma_u} p_p(q,\alpha) + \sum_{\alpha\in Pos(q)\cap\Sigma_c} p_r(r,\alpha) \leq 1.$$

The first part of both conditions of Theorem 3.1 corresponds to controllability from classical supervisory control theory (namely, the condition $Pos(q) \cap \Sigma_u = Pos(r) \cap \Sigma_u$ of (i), and $Pos(r) \cap \Sigma_c \subseteq Pos(q) \cap \Sigma_c$ of (ii)). The remaining equations and inequalities correspond to the conditions for probability matching.

For each uncontrollable event possible from a state in a plant, the equation to be checked reflects the fact that the ratio of probabilities of uncontrollable events remains the same under supervision. As elaborated in Chapter 2, this comes from the fact that after a control pattern has been chosen, the probabilities of disabled events in the plant are redistributed over enabled events in proportion to their probabilities. All possible uncontrollable events are always enabled, hence the ratios of their probabilities remain unchanged. Also, as shown in Chapter 2, an inequality for each possible controllable event $\sigma$ is derived from the upper bound on the probability of the occurrence of $\sigma$ in the supervised plant, that is reached when the controllable event is always enabled.

When the conditions are satisfied, a solution to the PSCP exists. After a string has been observed, the control input is given as a solution to the system of nonlinear equations given by (2.1). This solution can be approximated by the fixpoint iteration algorithm as presented in the following theorem.

**Theorem 3.2.** *Assume that conditions (i) and (ii) of Theorem 3.1 are satisfied. Let $\Gamma(s) = Pos(q) \cap \Sigma_c$ if $Pos(q) \cap \Sigma_u \neq \emptyset$, and $\Gamma(s) = (Pos(q) \cap \Sigma_c) \setminus \{\gamma\}$ otherwise, where $\gamma \in Pos(q)$ is chosen such that for every $\sigma \in Pos(q)$, $\frac{p_r(r,\gamma)}{p_p(q,\gamma)} \geq \frac{p_r(r,\sigma)}{p_p(q,\sigma)}$ is satisfied. Let $x^0(s) \in [0,1]^{\Gamma(s)}$ and $f(s) : \mathbb{R}^{\Gamma(s)} \to \mathbb{R}^{\Gamma(s)}$. For $x^0(s) = 0$, the sequence*

$$x^{k+1}(s) = f(s)(x^k(s)), \qquad k = 0, 1, \ldots, \text{ where} \tag{3.1}$$

$$f(s)(x)(\sigma) = \frac{p_r(r,\sigma)}{p_p(q,\sigma)h(s)(x)(\sigma)}, \sigma \in \Gamma(s), x \in \mathbb{R}^{\Gamma(s)} \text{ and}$$

$$\begin{aligned} &h(s)(x)(\sigma) \\ &= \sum_{\Theta \in \mathcal{P}(\Gamma(s) \setminus \{\sigma\})} \frac{1}{1 - \sum\limits_{\alpha \in \Theta} p_p(q,\alpha)} \prod_{\alpha \in \Theta} (1 - x(s)(\alpha)) \prod_{\alpha \in \Gamma(s) \setminus \{\sigma\} \setminus \Theta} x(s)(\alpha) \end{aligned}$$

*converges to the control input $x^*(s)$ (i.e., $V_p(s) = x^*(s)$).*

In the theorem, in the case when all possible events from a state are controllable, event $\gamma$ is chosen to be the one with the greatest ratio of required probability to uncontrolled probability - in some sense, the controllable event whose probability is "farthest" from its desired value.

### 3.1.2 Formal Proof

If there exists a probabilistic supervisor $V_p$ such that $L_p(V_p/G_p) = L_p(G_r)$, then $L(G_r) \subseteq L(G_p)$. Therefore, let $s \in L(G_r)$ and assume there exists $q \in Q_p$ such that $q = \delta_p(q_{p_0}, s)$, and $r = \delta_r(q_{r_0}, s)$. For notational convenience, whenever obvious from the context, we will omit the symbols for strings and states so that, e.g., instead of $p_p(q, \sigma)$, we shall write $p_{p,\sigma}$, and instead of $p_r(r, \sigma)$, we shall write $p_{r,\sigma}$.

Further, without loss of generality, we assume that in state $q$, not all the possible events are controllable, that is $\sum_{\sigma \in \Sigma_c} p_{p,\sigma} < 1$ (equivalently, $Pos(q) \cap \Sigma_c \neq \emptyset$). This assumption is safe since if $p_{p,\sigma} = 0$ for all $\sigma \in \Sigma_u$, then the PSCP reduces to the PSCP with only controllable events which can be transformed into a problem with exactly one uncontrollable event (we will discuss this further later in this section). Note that in the case of at least one possible uncontrollable event, we have $\Gamma(s) = Pos(q) \cap \Sigma_c$, where $\Gamma(s)$ is defined in Theorem 3.2. We will write $\Gamma$ instead of $\Gamma(s)$.

After a string $s \in L(G_r)$ has been observed, the supervisory problem is effectively the problem of finding the control input vector $x(s) \in [0,1]^\Gamma$ such that $P(\sigma \text{ in } V_p/G_p|s) = p_{r,\sigma}$, where $P(\sigma \text{ in } V_p/G_p|s)$ is given by (2.1), for all $\sigma \in \Sigma$.

#### Proof

For the purposes of the proof, we will write $x$ instead of $x(s)$, $x_\sigma$ instead of $x(s)(\sigma)$, $f_\sigma(x)$ instead of $f(s)(x)(\sigma)$, $h_\sigma(x)$ instead of $h(s)(x)(\sigma)$, for $\sigma \in \Sigma$. Also, we will denote $P(\sigma \text{ in } V_p/G|s)$ by $P_\sigma(x)$.

**Lemma 3.1.** *Let $\Psi \subseteq \Gamma$ and $x \in \mathbb{R}^\Psi$. Then,* $\sum_{\Phi \subseteq \Psi} \prod_{\sigma \in \Phi} x_\sigma \prod_{\sigma \in \Psi \setminus \Phi} (1 - x_\sigma) = 1.$

*Proof.* We prove the lemma by the induction on the size of $\Psi$. Let $k = |\Psi|$. For $k = 0$, the left hand side of the identity reduces to an empty product; therefore, the identity is satisfied. Assume that it holds for $k = n - 1$. We now prove that it holds for $k = n$. Let $\sigma \in \Gamma$, but $\sigma \not\subseteq \Psi$. Then:

$$\begin{aligned}
& \sum_{\Phi \subseteq \Psi \cup \{\sigma\}} \prod_{\alpha \in \Phi} x_\alpha \prod_{\alpha \in (\Psi \cup \{\sigma\}) \setminus \Phi} (1 - x_\alpha) \\
= & \sum_{\substack{\Phi \subseteq \Psi \cup \{\sigma\} \\ \sigma \in \Phi}} \prod_{\alpha \in \Phi} x_\alpha \prod_{\alpha \in (\Psi \cup \{\sigma\}) \setminus \Phi} (1 - x_\alpha) + \sum_{\substack{\Phi \subseteq \Psi \cup \{\sigma\} \\ \sigma \notin \Phi}} \prod_{\alpha \in \Phi} x_\alpha \prod_{\alpha \in (\Psi \cup \{\sigma\}) \setminus \Phi} (1 - x_\alpha) \\
= & \; x_\sigma \sum_{\Phi \subseteq \Psi} \prod_{\alpha \in \Phi} x_\alpha \prod_{\alpha \in \Psi \setminus \Phi} (1 - x_\alpha) + (1 - x_\sigma) \sum_{\Phi \subseteq \Psi} \prod_{\alpha \in \Phi} x_\alpha \prod_{\alpha \in \Psi \setminus \Phi} (1 - x_\alpha) \\
= & \; 1
\end{aligned}$$

□

**Lemma 3.2.** *Let $x \in [0,1]^\Gamma$. Then, $P_\sigma(x) = p_{p,\sigma} x_\sigma h_\sigma(x)$ for every $\sigma \in \Sigma$, where $h_\sigma : \mathbb{R}^\Gamma \to \mathbb{R}$ is given by*

$$h_\sigma(x) = \sum_{\Theta \in \mathcal{P}(\Gamma \setminus \{\sigma\})} \frac{1}{1 - \sum_{\alpha \in \Theta} p_{p,\alpha}} \prod_{\alpha \in \Theta} (1 - x_\alpha) \prod_{\alpha \in \Gamma \setminus \{\sigma\} \setminus \Theta} x_\alpha \tag{3.2}$$

*Proof.* Let $x \in [0,1]^\Gamma$. For $\sigma \in \Sigma$, Equation 2.1 can be equivalently expressed as:

$$P_\sigma(x) = \sum_{\Theta \in \mathcal{P}(\Gamma \setminus \{\sigma\})} \frac{p_{p,\sigma}}{\sum_{\alpha \in \Theta \cup \{\sigma\} \cup \Sigma_u} p_{p,\alpha}} \prod_{\alpha \in \Theta \cup \{\sigma\}} x_\alpha \prod_{\alpha \in \Gamma \setminus (\Theta \cup \{\sigma\})} (1 - x_\alpha)$$

If we apply the substitution $\Omega = \Gamma \setminus (\Theta \cup \{\sigma\})$ to the previous equation, it becomes

$$P_\sigma(x) \;=\; \sum_{\Omega \in \mathcal{P}(\Gamma \setminus \{\sigma\})} \frac{p_{p,\sigma}}{1 - \sum_{\alpha \in \Omega} p_{p,\alpha}} \prod_{\alpha \in \Omega} (1 - x_\alpha) \prod_{\alpha \in \Gamma \setminus \Omega} x_\alpha$$

The previous equation is well-defined as, for any $\Omega \in \mathcal{P}(\Gamma \setminus \{\sigma\})$, we have $1 - \sum_{\alpha \in \Omega} p_{p,\alpha} > 0$ since we assumed that there is at least one event $\alpha \in \Sigma_u$ such that $p_{p,\alpha} > 0$ . Therefore, for $\sigma \in \Sigma_c$, we have:

$$P_\sigma(x) = \sum_{\Omega \in \mathcal{P}(\Gamma \setminus \{\sigma\})} \frac{p_{p,\sigma}}{1 - \sum_{\alpha \in \Omega} p_{p,\alpha}} \prod_{\alpha \in \Omega} (1 - x_\alpha) \cdot \left( x_\sigma \prod_{\alpha \in \Gamma \setminus \{\sigma\} \setminus \Omega} x_\alpha \right) = p_{p,\sigma} x_\sigma h_\sigma(x)$$

For $\sigma \in \Sigma_u$, since $\sigma \notin \Gamma$ and $x_\sigma = 1$, then:

$$P_\sigma(x) = \sum_{\Omega \in \mathcal{P}(\Gamma \backslash \{\sigma\})} \frac{p_{p,\sigma}}{1 - \sum\limits_{\alpha \in \Omega} p_{p,\alpha}} \prod_{\alpha \in \Omega} (1 - x_\alpha) \prod_{\alpha \in \Gamma \backslash \{\sigma\} \backslash \Omega} x_\alpha = p_{p,\sigma} x_\sigma h_\sigma(x)$$

. □

Next, we introduce a partial order on $\mathbb{R}^\Gamma$. For $x, y \in \mathbb{R}^\Gamma$, $x \leq y$ iff $\forall \sigma \in \Gamma \; x_\sigma \leq y_\sigma$. Let $f : (X, \leq) \to (Y, \leq)$ be a mapping between posets. This mapping is monotone if whenever $x \leq x'$, then $f(x) \leq f(x')$; it is antitone if whenever $x \leq x'$, then $f(x) \geq f(x')$. Also, for $a \in \mathbb{R}$, let $\bar{a}$ denote $x \in \mathbb{R}^\Gamma$ such that $x_\sigma = a$ for all $\sigma \in \Gamma$.

**Lemma 3.3.** *Let $\Delta \subseteq \Gamma$ and $l : \mathcal{P}(\Gamma) \to \mathbb{R}$ be positive and monotone. The function $f_\Delta : \mathbb{R}^\Gamma \to \mathbb{R}$ given by $f_\Delta(x) = \sum\limits_{\Phi \in \mathcal{P}(\Delta)} l(\Phi) \prod\limits_{\sigma \in \Phi} (1 - x_\sigma) \prod\limits_{\sigma \in \Delta \backslash \Phi} x_\sigma$ is positive and antitone on $[0,1]^\Gamma$.*

*Proof.* First, we find the derivative of the function $f_\Delta(x)$ with respect to $x_\alpha$, for $\alpha \in \Sigma$. When $\alpha \notin \Delta$, $f_\Delta$ does not depend on $x_\alpha$ and $\frac{\partial f_R}{\partial x_\alpha}(x) = 0$. For the case when $\sigma \in \Delta$,

$$\begin{aligned} f_\Delta(x) &= \sum_{\substack{\Phi \in \mathcal{P}(\Delta) \\ \alpha \in \Phi}} l(\Phi) \prod_{\sigma \in \Phi} (1 - x_\sigma) \prod_{\sigma \in \Delta \backslash \Phi} x_\sigma + \sum_{\substack{\Phi \in \mathcal{P}(\Delta) \\ \alpha \notin \Phi}} l(\Phi) \prod_{\sigma \in \Phi} (1 - x_\sigma) \prod_{\sigma \in \Delta \backslash \Phi} x_\sigma \\ &= \sum_{\Phi \in \mathcal{P}(\Delta \backslash \{\alpha\})} (l(\Phi \cup \{\alpha\})(1 - x_\alpha) + l(\Phi) x_\alpha) \prod_{\sigma \in \Phi} (1 - x_\sigma) \prod_{\sigma \in \Delta \backslash \{\alpha\} \backslash \Phi} x_\sigma \end{aligned}$$

and so

$$\frac{\partial f_\Delta}{\partial x_\alpha}(x) = (-1) \cdot \sum_{\Phi \in \mathcal{P}(\Delta \backslash \{\alpha\})} (l(\Phi \cup \{\alpha\}) - l(\Phi)) \prod_{\sigma \in \Phi} (1 - x_\sigma) \prod_{\sigma \in \Delta \backslash \{\alpha\} \backslash \Phi} x_\sigma$$

which is always non-positive on $x \in [0,1]^\Gamma$ since $l$ is monotone. Therefore, $f_\Delta$ is antitone on $[0,1]^\Gamma$ and, consequently, $f_\Delta(x) \geq f_\Delta(\bar{1}) = l(\emptyset) > 0$. Hence, $f_\Delta$ is positive on $[0,1]^\Gamma$. □

**Lemma 3.4.** *The functions $h_\sigma(x)$, $\sigma \in \Sigma$, as defined in Lemma 3.2 are positive antitone, and such that $x_\sigma h_\sigma(x) \leq h_\gamma(x)$ on $[0,1]^\Gamma$ for all $\sigma \in \Sigma_c, \gamma \in \Sigma_u$.*

*Proof.* Let $\Delta \subseteq \Gamma$ be a nonempty set. Let $l : \mathcal{P}(\Delta) \to \mathbb{R}$ be given by $l(\Delta) = \frac{1}{1-\sum\limits_{\sigma\in\Delta} p_{p,\sigma}}$. Let $\Phi, \Lambda \in \mathcal{P}(\Delta)$ be such that $\Phi \subseteq \Lambda$. Since $0 \le \sum_{\sigma\in\Phi} p_{p,\sigma} \le \sum_{\sigma\in\Lambda} p_{p,\sigma} < 1$, $l$ is monotone. Therefore, according to Lemma 3.3, $h_\sigma$ are positive antitone. We now prove that for all $\sigma \in \Sigma_c, \gamma \in \Sigma_u$ we have $x_\sigma h_\sigma(x) \le h_\gamma(x)$:

$$\begin{aligned}
x_\sigma h_\sigma(x) &= x_\sigma \sum_{\Theta\in\mathcal{P}(\Gamma\backslash\{\sigma\})} l(\Theta) \prod_{\alpha\in\Theta}(1-x_\alpha) \prod_{\alpha\in\Gamma\backslash\{\sigma\}\backslash\Theta} x_\alpha \\
&= \sum_{\Theta\in\mathcal{P}(\Gamma\backslash\{\sigma\})} l(\Theta) \prod_{\alpha\in\Theta}(1-x_\alpha) \prod_{\alpha\in\Gamma\backslash\Theta} x_\alpha \\
&\le \sum_{\Theta\in\mathcal{P}(\Gamma)} l(\Theta) \prod_{\alpha\in\Theta}(1-x_\alpha) \prod_{\alpha\in\Gamma\backslash\Theta} x_\alpha \\
&= \sum_{\Theta\in\mathcal{P}(\Gamma\backslash\{\gamma\})} l(\Theta) \prod_{\alpha\in\Theta}(1-x_\alpha) \prod_{\alpha\in\Gamma\backslash\Theta\backslash\{\gamma\}} x_\alpha, \text{ for any } \gamma \in \Sigma_u, \text{ since } \gamma \notin \Gamma \\
&= h_\gamma(x), \quad \text{for any } \gamma \in \Sigma_u.
\end{aligned}$$

□

Let $\{x^k\}$ be a sequence of real numbers, and $x \in \mathbb{R}$. We will write $x^k \uparrow x$ iff $x^k \le x^{k+1}$ for all $k \in \mathbb{N}$, and $x^k \to x$ as $k \to \infty$. The following lemma gives sufficient conditions for the existence of a fixpoint of the function $f$.

**Lemma 3.5.** *Let $f : D \to \mathbb{R}^\Gamma$ be a monotone function on $D \subseteq \mathbb{R}^\Gamma$, $a_0, b_0 \in \mathbb{R}$, such that $a_0 \le b_0$, $[a_0, b_0]^\Gamma \subseteq D$ and $\bar{a}_0 \le f(\bar{a}_0)$. Assume that for every $x \in [a_0, b_0]^\Gamma$ such that $x \le f(x)$, we have $f(x) \le \bar{b}_0$. Then, the sequence $\{x^k\}$ given by*

$$x^0 = \bar{a}_0, \quad x^{k+1} = f(x^k), \quad k = 0, 1, \ldots$$

*exists and is such that $x^k \uparrow x^*$ for some $x^* \in [a_0, b_0]^\Gamma$. If, furthermore, $f$ is lower continuous $(\lim\limits_{x^k\uparrow x} f(x^k) = f(x))$, then $x^* = f(x^*)$.*

*Proof.* We first use induction to show that the sequence $\{x^k\}$ is a monotone chain contained in $[a_0, b_0]^\Gamma$. Because of the assumptions, $\bar{a}_0 \le f(\bar{a}_0) \le \bar{b}_0$, so that the basis step is true. Assume that $\{x^i\}$ for $i \le k$ forms a monotone chain in $[a_0, b_0]^\Gamma$. Since $x^k$ is in $[a_0, b_0]^\Gamma$, $x^{k+1} = f(x^k)$ is defined. By the induction hypothesis $x^{k-1} \le x^k$ holds. Hence, since $f$ is monotone, $f(x^{k-1}) \le f(x^k)$ also holds; consequently, $x^k \le x^{k+1}$. Since $\bar{a}_0 \le x^k$, then $\bar{a}_0 \le x^{k+1}$. Also, since $f(x^k) \le \bar{b}_0$, then $x^{k+1} \le \bar{b}_0$. Therefore, for $i \le k+1$, $\{x^i\}$ is a monotone chain contained in $[a_0, b_0]^\Gamma$.

Since $\{x^k\}$ is monotone and has a finite upper bound $\bar{b}_0$, it converges to a point $x^* \le \bar{b}_0$. Since $f$ is lower continuous, from $x^k \uparrow x^*$ follows $x^{k+1} \to f(x^*)$; therefore $x^* = f(x^*)$. □

The following theorem presents necessary and sufficient conditions for controllable events' probabilities to be assignable to given probabilities. If the conditions hold, the fixpoint algorithm to calculate the control input is given.

**Theorem 3.3.** *Assume that $Pos(r) \cap \Sigma_c \subseteq \Gamma$, and, for every $\sigma \in \Gamma$:*

$$\frac{p_{r,\sigma}}{p_{p,\sigma}} \sum_{\alpha \in \Sigma_u} p_{p,\alpha} + \sum_{\alpha \in \Gamma} p_{r,\alpha} \le 1 \tag{3.3}$$

*Then, the sequence $\{x^k\}$ given by*

$$x^0 = 0, \quad x^{k+1} = f(x^k) \quad k = 0, 1, \ldots \textit{ where } \; f_\sigma(x) = \frac{p_{r,\sigma}}{p_{p,\sigma} h_\sigma(x)}, \sigma \in \Gamma \tag{3.4}$$

*exists and is such that $x^k \uparrow x^*$ for some $x^* \in [0,1]^\Gamma$. Furthermore, $P_\sigma(x^*) = p_{r,\sigma}$ for all $\sigma \in \Sigma_c$. Conversely, for any $x \in [0,1]^\Gamma$, if $p_{r,\sigma} \triangleq P_\sigma(x)$ for all $\sigma \in \Sigma_c$, then $Pos(r) \cap \Sigma_c \subseteq \Gamma$ and (3.3) holds.*

*Proof.* First, we show that $f$ is defined on $[0,1]^\Gamma$ and monotone. By Lemma 3.3, $h_\sigma$ is positive and antitone on $[0,1]^\Gamma$. Therefore, $f_\sigma$ is positive and monotone on $[0,1]^\Gamma$, and $f(\bar{0}) \ge \bar{0}$. We show that whenever $x \le f(x)$, then $f(x) \le \bar{1}$. For $x \in [0,1]^\Gamma$ assume that $x \le f(x)$. Then $P_\sigma(x) = p_{p,\sigma} x_\sigma h_\sigma(x) \le p_{r,\sigma}$ for $\sigma \in \Gamma$. For $\sigma \in \Gamma$ let

$$l_\sigma = \frac{1 - \sum\limits_{\alpha \in \Gamma \setminus \{\sigma\}} p_{p,\alpha}}{p_{p,\sigma}}.$$

Note that $l_\sigma$ is well-defined since $p_{p,\sigma} > 0$.

$$l_\sigma p_{p,\sigma} h_\sigma(x) = \sum_{\Omega\in\mathcal{P}(\Gamma\backslash\{\sigma\})} \left(1 - \frac{\sum\limits_{\alpha\in\Gamma\backslash\{\sigma\}\backslash\Omega} p_{p,\alpha}}{1-\sum\limits_{\alpha\in\Omega} p_{p,\alpha}}\right) \prod_{\alpha\in\Omega}(1-x_\alpha) \prod_{\alpha\in\Gamma\backslash\Omega} x_\alpha$$

$$= 1 - \sum_{\Omega\in\mathcal{P}(\Gamma\backslash\{\sigma\})} \sum_{\alpha\in\Gamma\backslash\{\sigma\}\backslash\Omega} \frac{p_{p,\alpha}}{1-\sum\limits_{\alpha\in\Omega} p_{p,\alpha}} \prod_{\alpha\in\Omega}(1-x_\alpha) \prod_{\alpha\in\Gamma\backslash\{\sigma\}\backslash\Omega} x_\alpha \quad \text{(from Lemma 3.1)}$$

$$\geq 1-$$

$$\sum_{\Omega\in\mathcal{P}(\Gamma\backslash\{\sigma\})} \sum_{\alpha\in\Gamma\backslash\{\sigma\}\backslash\Omega} \left(\frac{p_{p,\alpha}}{1-\sum\limits_{\alpha\in\Omega} p_{p,\alpha}} x_\sigma + \frac{p_{p,\alpha}}{1-p_{p,\sigma}-\sum\limits_{\alpha\in\Omega} p_{p,\alpha}}(1-x_\sigma)\right) \prod_{\alpha\in\Omega}(1-x_\alpha) \prod_{\alpha\in\Gamma\backslash\{\sigma\}\backslash\Omega} x_\alpha$$

$$\left(\text{since } \frac{p_\alpha}{1-\sum\limits_{\alpha\in\Omega} p_\alpha} x_\sigma + \frac{p_\alpha}{1-p_\alpha-\sum\limits_{\alpha\in\Omega} p_\alpha}(1-x_\sigma) \geq \frac{p_\alpha}{1-\sum\limits_{\alpha\in\Omega} p_\alpha}\right)$$

$$= 1 - \sum_{\Omega\in\mathcal{P}(\Gamma\backslash\{\sigma\})} \sum_{\alpha\in\Gamma\backslash\{\sigma\}\backslash\Omega} \frac{p_{p,\alpha}}{1-\sum\limits_{\alpha\in\Omega} p_{p,\alpha}} \prod_{\alpha\in\Omega}(1-x_\alpha) \prod_{\alpha\in\Gamma\backslash\Omega} x_\alpha -$$

$$- \sum_{\substack{\Omega\in\mathcal{P}(\Gamma)\\ \sigma\in\Omega}} \sum_{\alpha\in\Gamma\backslash\{\sigma\}\backslash\Omega} \frac{p_{p,\alpha}}{1-\sum\limits_{\alpha\in\Omega} p_{p,\alpha}} \prod_{\alpha\in\Omega}(1-x_\alpha) \prod_{\alpha\in\Gamma\backslash\Omega} x_\alpha$$

$$= 1 - \sum_{\Omega\in\mathcal{P}(\Gamma)} \sum_{\alpha\in\Gamma\backslash\{\sigma\}\backslash\Omega} \frac{p_{p,\alpha}}{1-\sum\limits_{\alpha\in\Omega} p_{p,\alpha}} \prod_{\alpha\in\Omega}(1-x_\alpha) \prod_{\alpha\in\Gamma\backslash\Omega} x_\alpha$$

$$= 1 - \sum_{\alpha\in\Gamma\backslash\{\sigma\}} \sum_{\Omega\in\mathcal{P}(\Gamma\backslash\{\alpha\})} \frac{p_{p,\alpha}}{1-\sum\limits_{\alpha\in\Omega} p_{p,\alpha}} \prod_{\alpha\in\Omega}(1-x_\alpha) \prod_{\alpha\in\Gamma\backslash\Omega} x_\alpha$$

(Suppose $\Omega \in \mathcal{P}(\Gamma)$ and $\alpha \in \Gamma\backslash\{\sigma\}\backslash\Omega$. Then $\alpha \in \Gamma\backslash\{\sigma\}$. Since $\alpha \notin \Omega, \Omega\backslash\{\alpha\} = \Omega$, so $\Omega \in \mathcal{P}(\Gamma\backslash\{\alpha\})$)

$$= 1 - \sum_{\alpha\in\Gamma\backslash\{\sigma\}} P_\alpha(x) \qquad \text{(from Lemma 3.2)}$$

$$\geq 1 - \sum_{\alpha\in\Gamma\backslash\{\sigma\}} p_{r,\alpha} \qquad (\text{since } P_\sigma \leq p_{r,\sigma} \text{ for } \alpha \in \Gamma)$$

Therefore,

$$f_\sigma(x) = \frac{p_{r,\sigma}}{p_{p,\sigma} h_\sigma(x)} = \frac{l_\sigma p_{r,\sigma}}{l_\sigma p_{p,\sigma} h_\sigma(x)} \leq \frac{l_\sigma p_{r,\sigma}}{1 - \sum\limits_{\alpha\in\Gamma\backslash\{\sigma\}} p_{r,\alpha}} \leq 1$$

Using Lemma 3.5, the sequence $x^0 = \bar{0}$ and $x^{k+1} = f(x^k)$ for $k = 0, 1, \ldots$, exists and is such that $x^k \uparrow x^*$ for some $x^* \in [0,1]^\Gamma$. Since $f$ is continuous on $[0,1]^\Gamma$, it follows that $f(x^*) = x^*$, or equivalently that $P_\sigma(x^*) = p_{r,\sigma}$ for $\sigma \in \Gamma$. Also, since $Pos(r) \cap \Sigma_c \subseteq \Gamma$, then for $\sigma \in \Sigma_c$, but $\sigma \notin \Gamma$, we have $P_\sigma(x^*) = p_{p,\sigma} x^*_\sigma h_\sigma = 0 = p_{r,\sigma}$. Therefore, $P_\sigma(x^*) = p_{r,\sigma}$ for $\sigma \in \Sigma_c$.

We now prove the necessity of the condition. Suppose that there is $x \in [0,1]^\Gamma$ such that $P_\sigma(x) = p_{r,\sigma}$ for any $\sigma \in \Sigma_c$. Then, for $\sigma \notin \Gamma$ we have $P_\sigma(x) = 0 = p_{r,\sigma}$. Therefore, $Pos(r) \cap \Sigma_c \subseteq \Gamma$. Next, we note that for any $\sigma, \alpha \in \Sigma_u$ we have that $h_\sigma = h_\alpha$. We therefore introduce the function $g : \mathbb{R}^\Gamma \to \mathbb{R}$ such that $g(x) = h_\sigma(x)$ for any $\sigma \in \Sigma_u$. Then, since our generators are nonterminating,

$$\begin{aligned} 1 &= \sum_{\alpha \in \Sigma_u} P_\alpha(x) + \sum_{\alpha \in \Sigma_c} P_\alpha(x) = \sum_{\alpha \in \Sigma_u} p_{p,\alpha} x_\alpha h_\alpha(x) + \sum_{\alpha \in \Sigma_c} p_{r,\alpha} \\ &= g(x) \sum_{\alpha \in \Sigma_u} p_{p,\alpha} + \sum_{\alpha \in \Gamma} p_{r,\alpha}. \end{aligned} \tag{3.5}$$

Since $p_{r,\sigma} = p_{p,\sigma} x_\sigma h_\sigma(x)$ for $\sigma \in \Sigma$, and, according to Lemma 3.4, $x_\sigma h_\sigma(x) \leq g(x)$, it follows that $g(x) \geq \frac{p_{r,\sigma}}{p_{p,\sigma}}$. By plugging this inequality into Equation 3.5, we get the condition (3.3). □

So far we have considered the conditions under which the specified probabilities can be assigned to probabilities of controllable events. Next, we consider uncontrollable events as well.

**Lemma 3.6.** *There exists $x \in [0,1]^\Gamma$ such that $P_\sigma(x) = p_{r,\sigma}$ for every $\sigma \in \Sigma$ iff $Pos(r) \cap \Sigma_c \subseteq \Gamma$, $Pos(q) \cap \Sigma_u = Pos(r) \cap \Sigma_u$, (3.3) holds for every $\sigma \in \Gamma$, and for every $\sigma \in Pos(q) \cap \Sigma_u$*

$$\frac{p_{p,\sigma}}{\sum_{\alpha \in \Sigma_u} p_{p,\alpha}} = \frac{p_{r,\sigma}}{\sum_{\alpha \in \Sigma_u} p_{r,\alpha}}. \tag{3.6}$$

*Proof.* Assume that $Pos(r) \cap \Sigma_c \subseteq \Gamma$, $Pos(q) \cap \Sigma_u = Pos(r) \cap \Sigma_u$, (3.3) holds for $\sigma \in \Gamma$, and (3.6) holds for $\sigma \in Pos(q) \cap \Sigma_u$. Then, according to Theorem 3.3, for all $\sigma \in \Sigma_c$, we have $P_\sigma(x) = p_{r,\sigma}$, and, according to Lemma 3.2, for all $\sigma \in \Sigma_u$, $P_\sigma(x) = p_{p,\sigma} x_\sigma h_\sigma(x)$, that is $P_\sigma(x) = p_{p,\sigma} g(x)$. Therefore, for all

$\sigma \in \Sigma_u$:

$$P_\sigma(x) = p_{p,\sigma} g(x) = p_{p,\sigma} \cdot \frac{\sum\limits_{\alpha\in\Sigma_u} p_{p,\alpha} g(x)}{\sum\limits_{\alpha\in\Sigma_u} p_{p,\alpha}} = p_{p,\sigma} \cdot \frac{\sum\limits_{\alpha\in\Sigma_u} P_\alpha(x)}{\sum\limits_{\alpha\in\Sigma_u} p_{p,\alpha}}$$

$$= p_{p,\sigma} \cdot \frac{1 - \sum\limits_{\alpha\in\Sigma_c} P_\alpha(x)}{\sum\limits_{\alpha\in\Sigma_u} p_{p,\alpha}} = p_{p,\sigma} \cdot \frac{\sum\limits_{\alpha\in\Sigma_u} p_{r,\alpha}}{\sum\limits_{\alpha\in\Sigma_u} p_{p,\alpha}} = p_{r,\sigma}$$

For the converse, assume that there is a supervisory controller $x$ such that $P_\sigma(x) = p_{r,\sigma}$ for every $\sigma \in \Sigma$. Then, according to Theorem 3.3, $Pos(r) \cap \Sigma_c \subseteq \Gamma$ and (3.3) holds. For $\sigma \in Pos(q) \cap \Sigma_u$, if $p_{p,\sigma} = 0$, then $P_\sigma(x) = p_{p,\sigma} g(x) = 0 = p_{r,\sigma}$. Also, for $\sigma \in Pos(r) \cap \Sigma_u$, if $p_{r,\sigma} = 0 = P_\sigma(x) = p_{p,\sigma} g(x)$, then $p_{p,\sigma} = 0$ (since $g(x) \neq 0$). Therefore, $Pos(q) \cap \Sigma_u = Pos(r) \cap \Sigma_u$. Then, for $\sigma \in Pos(q) \cap \Sigma_u$:

$$\frac{p_{r,\sigma}}{\sum\limits_{\alpha\in\Sigma_u} p_{r,\alpha}} = \frac{P_\sigma(x)}{\sum\limits_{\alpha\in\Sigma_u} p_{r,\alpha}} = \frac{p_{p,\sigma}}{\sum\limits_{\alpha\in\Sigma_u} p_{r,\alpha}} g(x) = \frac{p_{p,\sigma}}{\sum\limits_{\alpha\in\Sigma_u} p_{r,\alpha}} \cdot \frac{\sum\limits_{\alpha\in\Sigma_u} p_{p,\alpha} g(x)}{\sum\limits_{\alpha\in\Sigma_u} p_{p,\alpha}}$$

$$= \frac{p_{p,\sigma}}{\sum\limits_{\alpha\in\Sigma_u} p_{r,\alpha}} \cdot \frac{\sum\limits_{\alpha\in\Sigma_u} P_\alpha(x)}{\sum\limits_{\alpha\in\Sigma_u} p_{p,\alpha}} = \frac{p_{p,\sigma}}{\sum\limits_{\alpha\in\Sigma_u} p_{r,\alpha}} \cdot \frac{\sum\limits_{\alpha\in\Sigma_u} p_{r,\alpha}}{\sum\limits_{\alpha\in\Sigma_u} p_{p,\alpha}} = \frac{p_{p,\alpha}}{\sum\limits_{\alpha\in\Sigma_u} p_{p,\alpha}}$$

□

**Special Case:** $Pos(q) \cap \Sigma_u = \emptyset$

We now address the issue previously mentioned: in a certain state, only controllable events can happen in the plant. Then, a probabilistic supervisor can disable them all which would cause termination. However, as we consider nonterminating generators, this is not allowed. An elegant solution is to always enable one event: this event effectively becomes uncontrollable and the problem reduces to the one already proved. We now show that, if an event $\gamma$ with the maximal ratio $p_{r,\gamma}/p_{p,\gamma}$ is chosen, then condition (3.3) is satisfied.

Formally, let $Pos(q) \cap \Sigma_u = \emptyset$. Then, only for this local problem, we declare event $\gamma \in Pos(q) \cap \Sigma_c$ to be uncontrollable. Then, $\Gamma(s) = (Pos(q) \cap \Sigma_c) \backslash \{\gamma\}$, denoted $\Gamma$ for simplicity. The left hand side of the condition in (3.3), for $\sigma \in \Gamma$, becomes:

$$\begin{aligned}\frac{p_{r,\sigma}}{p_{p,\sigma}}p_{p,\gamma} + \sum_{\alpha\in\Gamma} p_{r,\alpha} &= \frac{p_{r,\sigma}}{p_{p,\sigma}}p_{p,\gamma} - p_{r,\gamma} + \sum_{\alpha\in\Gamma} p_{r,\alpha} + p_{r,\gamma} \\ &= \frac{p_{r,\sigma}}{p_{p,\sigma}}p_{p,\gamma} - p_{r,\gamma} + \sum_{\alpha\in Pos(q)\cap\Sigma_c} p_{r,\alpha} = \frac{p_{r,\sigma}}{p_{p,\sigma}}p_{p,\gamma} - p_{r,\gamma} + 1\end{aligned}$$

Then, the condition (3.3) becomes

$$\frac{p_{r,\sigma}}{p_{p,\sigma}}p_{p,\gamma} - p_{r,\gamma} \leq 0$$

which, since $p_{p,\gamma} > 0$, is equivalent to

$$\frac{p_{r,\sigma}}{p_{p,\sigma}} - \frac{p_{r,\gamma}}{p_{p,\gamma}} \leq 0.$$

If the event $\gamma$ is one with the maximal ratio $p_{r,\gamma}/p_{p,\gamma}$ (meaning, for every $\sigma \in Pos(q)$, $\frac{p_r(r,\gamma)}{p_p(q,\gamma)} \geq \frac{p_r(r,\sigma)}{p_p(q,\sigma)}$), it is obvious that the condition is satisfied for any $\sigma \in \Gamma$. Now, it is easy to show that this case and Lemma 3.6 result in Theorem 3.1. Further, if the conditions of Theorem 3.1 are satisfied, the algorithm for computation of control input from Theorem 3.3 can be applied to this special case, with $\gamma$ considered an uncontrollable event.

### 3.1.3 Example

Let $G = (Q_p, \Sigma, \delta_p, q_0, p_p)$ and $G_2 = (Q_r, \Sigma, \delta_r, q_{20}, p_r)$ for the example from Figure 3.1 (the same as Figure 2.1), where $\Sigma_c = \{\alpha, \beta\}$, and $\Sigma_u = \{\gamma\}$.

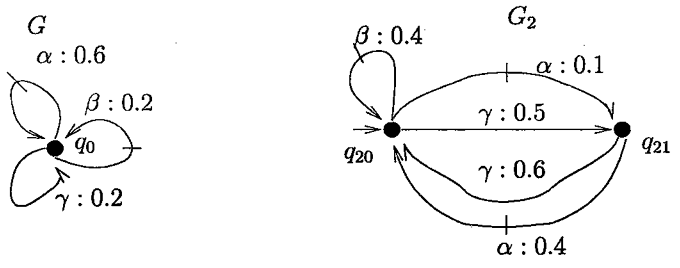

Figure 3.1: Plant $G$ and requirements specification $G_2$

We now present the calculation of a probabilistic supervisor for the example. The case when the string $s \in L(G_2)$ has been observed such that

$G$ is at the state $q_0$, and $G_2$ is at $q_{20}$ will be presented in detail. Again, for notational convenience, we shall write $p_{p,\sigma}$ instead of $p_p(q_0, \sigma)$, and $p_{r,\sigma}$ instead of $p_r(q_{20}, \sigma)$ where $\sigma \in \Sigma$. With a slight abuse of notation, let $p_p = (p_{p,\alpha}, p_{p,\beta}, p_{p,\gamma})$, $p_r = (p_{r,\alpha}, p_{r,\beta}, p_{r,\gamma})$, and $x(s) = (x(s)(\alpha), x(s)(\beta))$. From Figure 3.1, it follows $p_p = (0.6, 0.2, 0.2), p_r = (0.1, 0.4, 0.5)$. First, we check the conditions of Theorem 3.1. The equality of Theorem 3.1 is trivially satisfied. We then check if the inequalities of Theorem 3.1 are satisfied:

$$\frac{1}{6}p_{p,\gamma} + p_{r,\alpha} + p_{r,\beta} = 0.53 \leq 1,$$
$$2p_{p,\gamma} + p_{r,\alpha} + p_{r,\beta} = 0.9 \leq 1.$$

Then, the control input $x^*(s)$ can be calculated by the fixpoint iteration where $x^0(s) = (0, 0)$, $x^k(s) = f(s)(x^{k-1}(s))$, and, for $x \in \mathbb{R}^{\{\alpha,\beta\}}$:

$$f(s)(x)(\alpha) = \frac{p_{r,\alpha}}{p_{p,\alpha}\left(x(\beta) + \frac{1}{1-p_{p,\beta}}(1 - x(\beta))\right)},$$
$$f(s)(x)(\beta) = \frac{p_{r,\beta}}{p_{p,2}\left(x(\alpha) + \frac{1}{1-p_{p,\alpha}}(1 - x(\alpha))\right)}.$$

After just a few iterations, the sequence $\{x^k(s)\}$ converges to $x^*(s) = (0.162, 0.886)$ (see Figure 3.2).

The calculation of the supervisor after the string $t \in L(G_2)$ has been observed such that $G_2$ is in the state $q_{21}$ gives the result $x^*(t) = (0.533, 0)$. The supervisor is shown in Figure 2.3.

### 3.1.4 Complexity Analysis of Synthesis Problem and Algorithm

Let $G_p^{np}$ and $G_r^{np}$ be the nonprobabilistic automata underlying generators $G_p$ and $G_r$, i.e. $G_p^{np} = (Q_p, \Sigma, \delta_p, q_{p0})$ and $G_r^{np} = (Q_r, \Sigma, \delta_r, q_{r0})$. Also, let the nonprobabilistic automaton $G_s = (Q_s, \Sigma, \delta_s, q_{s0})$ be the synchronous product of $G_p^{np}$ and $G_r^{np}$. The check of the conditions of Theorem 3.1 can be implemented by checking conditions (i) and (ii) of Theorem 3.1 for each $(q, r) \in Q_s$. The construction of $G_s$ and the check of the conditions of Theorem 3.1 can be performed in time $O(|Q_p| \cdot |Q_r| \cdot |\Sigma|)$. Assume that the conditions of Theorem 3.1

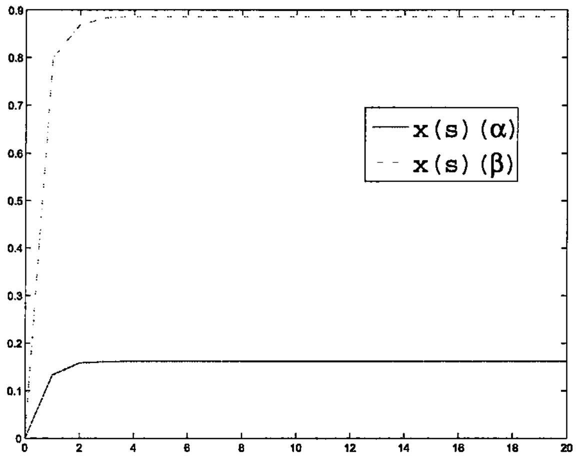

Figure 3.2: Fixpoint iteration

are satisfied. Then, for each state $(q,r) \in Q_s$ of the automaton $G_s$ (there are at most $|Q_p| \cdot |Q_r|$ states), control input $x(s) \in [0,1]^{\Gamma(s)}$, where $s \in L(G_s)$ such that $q = \delta_p(q_{p0}, s)$, and $r = \delta_r(q_{r0}, s)$, is the solution of the system of nonlinear polynomial equations

$$x = f(s)(x), \tag{3.7}$$

on the interval $[0,1]^{\Gamma(s)}$, where $f(s)$ is defined as in (3.1). This control input can be calculated using the algorithm from Theorem 3.2.

We assume that the probabilities of both the plant and specification are rational. Even in this case, control inputs are, in general, irrational. E.g., let us consider the plant and specification in Figure 2.1, after string $s$ has been observed such that PDES $G$ is in state $q_0$ and the PDES $G_2$ is in state $q_{20}$. Then, solving for $x = x(s)(\alpha)$, the system of equations (3.7) reduces to the equation $45x^2 - 69x + 10 = 0$, whose roots are irrational numbers, and, therefore, cannot be computed exactly. Hence, the best we could do is approximate the supervisor's probabilities to a certain accuracy. The theoretical complexity of this problem is equal to the theoretical complexity of approximating the solution of the system of nonlinear polynomial equations (3.7). It is known that even for systems of quadratic equations, the problem is at least exponentially hard (Kreinovich et al., 1998).

For deriving the upper bounds on complexity of the problem, we use reasoning similar to that presented in (Etessami and Yannakakis, 2009). We resort to results on complexity of decision procedures on the Existential Theory of Reals, $ExTh(\mathbb{R})$. Results of (Canny, 1988; Renegar, 1992) give the upper time complexity bounds for deciding sentences in $ExTh(\mathbb{R})$. A sentence in $ExTh(\mathbb{R})$ is of the form: $\Phi \equiv \exists x_1, \ldots, x_n P(x_1, \ldots, x_n)$, where $P$ is a quantifier free boolean formula with "atomic predicates" of the form $g_i(x_1, \ldots, x_n) \triangle 0$, where $g_i$ is a (multivariate) polynomial with rational coefficients, and $\triangle \in \{>, \geq, =, \neq, \leq, <\}$. Let $m$ be the number of atomic predicates $g_i$, and $d$ the maximal degree of polynomials $g_i$. Then, there is an algorithm that decides if the sentence $\Phi$ is true over real numbers, that runs in PSPACE, and in time $O((md)^{O(n)})$. This complexity result contains an implicit assumption that the validity of $P$ can be decided in constant time (given the truth values of its atomic predicates); this assumption serves to simplify the result and does not have a significant impact on the following complexity results.

It is easy to construct a sentence in $ExTh(\mathbb{R})$ that compares $x(s)(\alpha)$ ($\alpha \in \Gamma(s)$) to a rational number. The sentence

$$\Phi(s)(\alpha) \equiv \exists x(s) \left( x(s) = f(s)(x(s)) \wedge \bigwedge_{\sigma \in \Gamma(s)} 0 \leq x(s)(\sigma) \leq 1 \wedge x(s)(\alpha) < u \right)$$

checks if there exists control input $x(s)$ that is a solution of (3.7) such that $x(s)(\alpha)$ is less than a rational number $u$. Since each $x(s)(\sigma)$ ($\sigma \in \Gamma(s)$) is in the interval $[0, 1]$, we can use binary search and queries similar to $\Phi(s)(\alpha)$ to close in on the value of a control input up to an accuracy $10^{-i}$, $i \in \mathbb{N}$. In order to reach this accuracy, we need to use $O(i \cdot |\Gamma(s)|)$ queries. Therefore, there is an algorithm that approximates the solution of (3.7) up to prespecified accuracy $10^{-i}$, and it runs in time $O(i \cdot |\Gamma(s)|^{O(|\Gamma(s)|)})$.

On the other hand, a straightforward analysis of (3.7) suggests that the worst-case running time of one iteration of the fixpoint algorithm of Theorem 3.2 that approximates the solution of (3.7) is $O(|\Gamma(s)|^2 \cdot 2^{|\Gamma(s)|})$. Since the number $|\Gamma(s)|$ is typically small in practical applications, this complexity does not represent a practical limitation of the algorithm.

We next discuss the rate of convergence of the algorithm. Let $\Lambda(s) =$

$Pos(r) \cap \Sigma_c$ if $Pos(q) \cap \Sigma_u \neq \emptyset$, and $\Lambda(s) = (Pos(r) \cap \Sigma_c) \backslash \{\gamma\}$ otherwise, where $\gamma \in Pos(q)$ is chosen, as before, such that for every $\sigma \in Pos(q)$, $p_r(r,\gamma)/p_p(q,\gamma) \geq p_r(r,\sigma)/p_p(q,\sigma)$ is satisfied. Also, let $x_\sigma(s)$ denote $x(s)(\sigma)$ for any $\sigma \in \Gamma(s)$. The exact rate of convergence is not yet known. However, experimental results indicate that the speed of convergence can be given as:

$$\underset{\sigma \in \Gamma(s)}{max}\{|x_\sigma(s) - x_\sigma^k(s)|\} \leq K(s) \cdot \underset{\sigma \in \Gamma(s)}{max}\{|x_\sigma(s) - x_\sigma^{k-1}(s)|\}, \quad k = 1, 2, \ldots, \quad (3.8)$$

where $K(s)$ is at most $\sum_{\sigma \in \Lambda(s)} p_r(r,\sigma) < 1$. The estimate was obtained in the following manner. For a number of controllable events $N$, we randomly generated controllable events' probabilities of the plant $p_p \in [0,1]^N$, such that the elements of $p_p$ sum to less than 1. Also, supervisor probabilities $x \in [0,1]^N$ were randomly generated. Then, the resulting controllable events' probabilities $p_r$ of the plant under the control were calculated, and then, in turn, used to calculate $x^k$ using the algorithm of Theorem 3.2. In this manner, the inequality (3.8) was checked for a thousand problems for each number of controllable events up to 10. Practically, the convergence of (3.8) means that, when the value $K(s) \leq 0.6$, the algorithm converges fast (gaining one decimal of precision in at most five iterations). When $0.6 < K(s) < 0.9$, it takes not more than 22 iterations per decimal of precision. For $K(s)$ very close to 1, the number of iterations per decimal of precision (at most $1/\log(1/K(s))$) can become quite large.

## 3.2 Reactive Model of Probabilistic Supervisor

In general, a probabilistic supervisor is not a probabilistic generator (an example is the probabilistic supervisor in Figure 2.3). This comes from the fact that, with a probabilistic supervisor, the probabilities attached to events are the probabilities of the events being enabled, not occurring (see Section 2.3). Therefore, the probabilities of occurrences of events from a state of the supervisor in Figure 2.3 are not distributed according to a probability distribution.

A probabilistic supervisor, however, can be represented using a reactive model. In Figure 3.3, probabilistic supervisor $V_p$ from Figure 2.3 is represented using an MDP. The transformation illustrated in Figure 3.3 is next explained briefly.

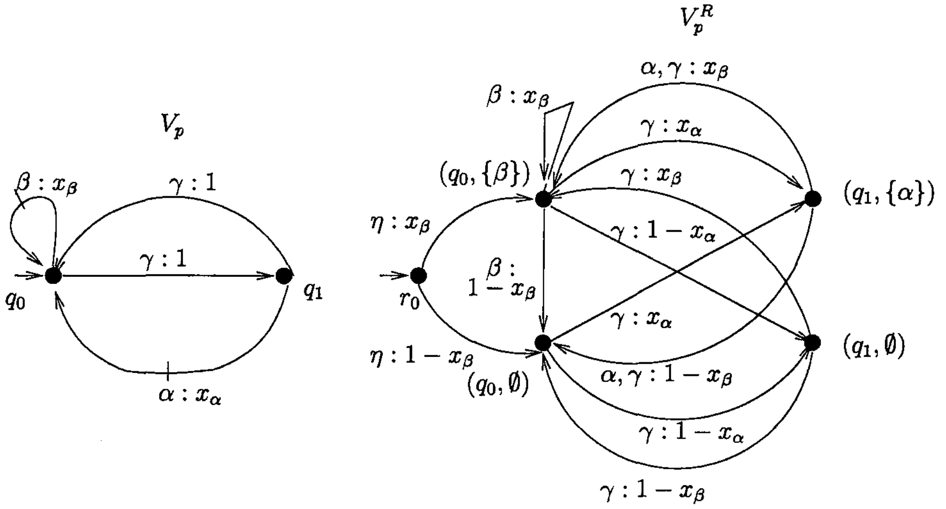

Figure 3.3: Probabilistic supervisor $V_p$ as MDP $V_p^R$

Each state $q \in Q$ of probabilistic supervisor $V_p$ maps to a set of states $\{(q, \Phi_1) | \Phi_1 \subseteq Pos(q) \backslash \Sigma_u\}$. Events admissible from state $(q, \Phi_1)$ are exactly those from $\Phi_1$ together with uncontrollable events possible from state $q$. The probability attached to event $\sigma$ going from state $(q, \Phi_1)$ to $(r, \Phi_2)$ (where $r \in Q$, $\Phi_2 \subseteq Pos(r) \backslash \Sigma_u$) is the probability of $\Phi_2$ being enabled at state $r$ (not the probability of $\Phi_1$ being enabled at state $q$). Event $\eta \notin \Sigma$ and state $r_0 \notin Q$ are used to initialize the supervisor properly.

On the other hand, a randomized control policy in the MDP framework (see Section 1.3) cannot be represented with a probabilistic generator in a straightforward manner. This comes from the fact that, even when a Markov policy is being implemented (decision at a state of a system depends only on that state), there might be two different paths corresponding to the same sequence of events leading to two different states of the system. In this case, we would not be able to encode two different decisions at these two states using our probabilistic generator without event augmentation. I.e., it would be possible to encode information about the different states reached by same strings through event labeling: transitions would be labeled not only with events, but also the states of the system that these events lead to.

Therefore, probabilistic supervisor in our framework can be represented as an MDP. Similarly, probabilistic policy as defined in the MDP framework can be represented using an augmented probabilistic generator. This fact can turn out to be extremely useful, as it would potentially allow for the exchange of some results between the models and the frameworks.

## 3.3 Optimal Probabilistic Supervisory Control Problem: Formulation

In the case when the conditions for the existence of a solution of the probabilistic supervisory control problem are not satisfied, we search for a suitable approximation. We define the problem as follows.

*The Optimal Probabilistic Supervisory Control Problem (OPSCP)*: Let $G_p = (Q_p, \Sigma, \delta_p, q_{p0}, p_p)$ be a plant PDES, and let $G_r = (Q_r, \Sigma, \delta_r, q_{r0}, p_r)$ be a requirements specification represented as a PDES. If there is no probabilistic supervisor $V_p$ such that $L_p(V_p/G_p) = L_p(G_r)$ (i.e., the conditions of Theorem 3.1 fail), find, if it exists, $V_p$ such that

1. $L(V_p/G_p) \subseteq L(G_r)$ and supervisor $V_p$ is maximally permissive and deadlock-free in the nonprobabilistic sense (i.e., $L(V_p/G_p)$ is the supremal deadlock-free and controllable sublanguage of $L(G_r)$ with respect to $G_p$).

2. The probabilistic behaviour of the controlled plant is "as close as possible" to the probabilistic behaviour of the requirements specification restricted to the supremal deadlock-free and controllable sublanguage of $L(G_r)$ with the respect to $G_p$.

Let $G = V_p/G_p = (S, \Sigma, \delta, s_0, p)$ be the closest approximation.

The first criterion is straightforward. The requirement $G_r$ represents a safety constraint: the controlled plant is not allowed to generate strings not in $L(G_r)$ even with the smallest of probabilities. Further, the criterion of maximal permissiveness is a standard one for optimality of supervisory control. The controlled plant should be deadlock-free as nonterminating generators are considered. The second criterion, on the other hand, is probabilistic: the distance in a pseudometric between the initial states of the probabilistic generators $G$ and an appropriately modified $G_r$ is chosen as a measure of probabilistic similarity. The requirements specification $G_r$ is modified such that its nonprobabilistic behaviour is reduced to the maximal permissible deadlock-free legal nonprobabilistic behaviour of the plant under control. In other words, the (nonprobabilistic) language of the modified specification is the supremal deadlock-free and controllable sublanguage of $L(G_r)$ with respect to $G_p$. Consequently, the probabilities of the specification are revised so that the probabilities of the events inadmissible for not satisfying the first criterion are redistributed over the admissible ones. The rationale behind the modified specification is as follows:

- It makes sense for a designer to modify the (probabilistic) requirements specification as she cannot do better in a nonprobabilistic sense. So, after realizing that only a subset of the desired nonprobabilistic behaviour is achievable, the designer sees no reason in insisting on probabilities suggested for the behaviour that cannot be achieved. We assume that the designer wants to, for each state, distribute the probabilities of the events not possible anymore over the remaining events so that the new probabilities are proportional to the old ones. However, the designer might want to rebalance the probabilities any way it suites her.

- Obviously, it might be the case that the designer prefers to leave the specification intact. Then, the problem to solve becomes the OPSCP with criterion (2) modified so that the difference between the controlled plant and the original specification is minimized. As it turns out, the solution of the original OPSCP is an important step en route to the solution of this modified OPSCP. The solution of the modified OPSCP will be presented in this thesis, too.

## 3.4 Applications: An Example

We now present a real-world application of the research presented in this thesis. We adapt the simplified model of a distributed robot control system presented in (Li et al., 1998) to our setting. The example is simple, but illustrative of one class of application. A processor processes the readings from two sensors. The sensors models are shown at the top of Figure 3.4. For $i = 1, 2$, event $\alpha_i$ stands for "sensor $i$ requests the processor", event $\beta_i$ for "sensor $i$ uses the processor", and $\gamma_i$ is "sensor $i$ releases the processor". The resulting plant is given in the same figure. Its nonprobabilistic behaviour is given as the shuffle of two DES that represent the sensors. Sensor 1 uses the processor more frequently. E.g., sensor 1 reads the speed of the robot at a fixed rate, while sensor 2 sends warning signals when the robot approaches an obstacle. Probabilities attached to transitions are such that the probabilities of $\alpha_1, \beta_1, \gamma_1$ are 0.95, while the probabilities of $\alpha_2, \beta_2, \gamma_2$ are 0.05.

The requirements specification $G_{r_1}$ is given in Figure 3.5. The nonprobabilistic part of the specification expresses the mutual exclusion requirement: two sensors cannot use the processor at the same time. Therefore, state $(2, 2')$ is the forbidden state of the plant. The probabilistic part of the requirements specification reflects a need for a prioritization of sensor 2 when both sensors have requested the processor. More precisely, at state $r_4$, after both sensors have requested the processor, sensor 2 should be four times more likely to use the processor than sensor 1.

In order to solve the PSCP for the given plant and requirements specification, the results of Section 3.1.1 are used. The probabilistic supervisor for the PSCP exists, and it disables event $\beta_1$ at state $(1, 2')$ and event $\beta_2$ at state $(2, 1')$, while event $\beta_1$ is enabled with probability 0.2105 at state $(1, 1')$.

As already mentioned, in general, the solution of the PSCP might not exist. The requirements specification from Figure 3.6 is different from $G_{r_1}$ in that at state $r_2$, it is more likely that sensor 2 will be allowed to apply for the processor. For the plant from Figure 3.4 and the requirements specification from Figure 3.6, there is no probabilistic supervisor such that the probabilistic language of the plant under control is the same as the probabilistic language

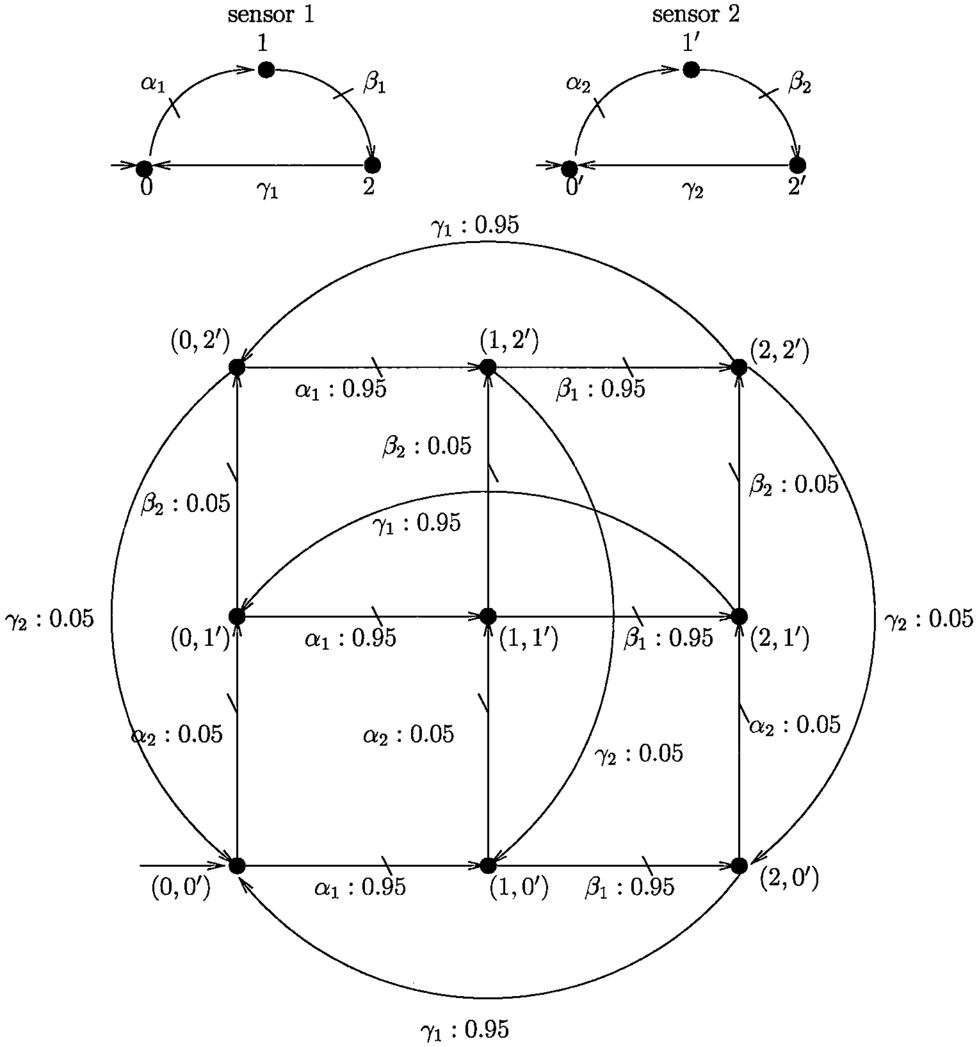

Figure 3.4: Sensors and resulting plant

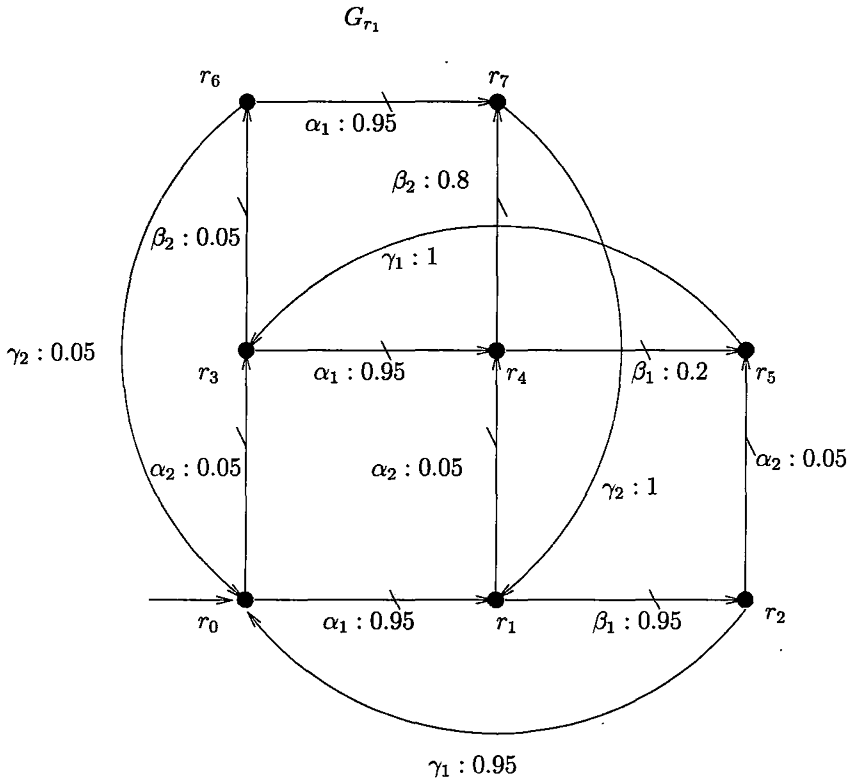

Figure 3.5: Requirements specification $G_{r_1}$

generated by the requirements specification (for state $(2, 0')$ of the plant and $r_2$ of the specification $G_{r_2}$, the conditions of Theorem 3.1 do not hold). In this case, the closest approximation is sought after (see Section 3.3 for the problem formulation).

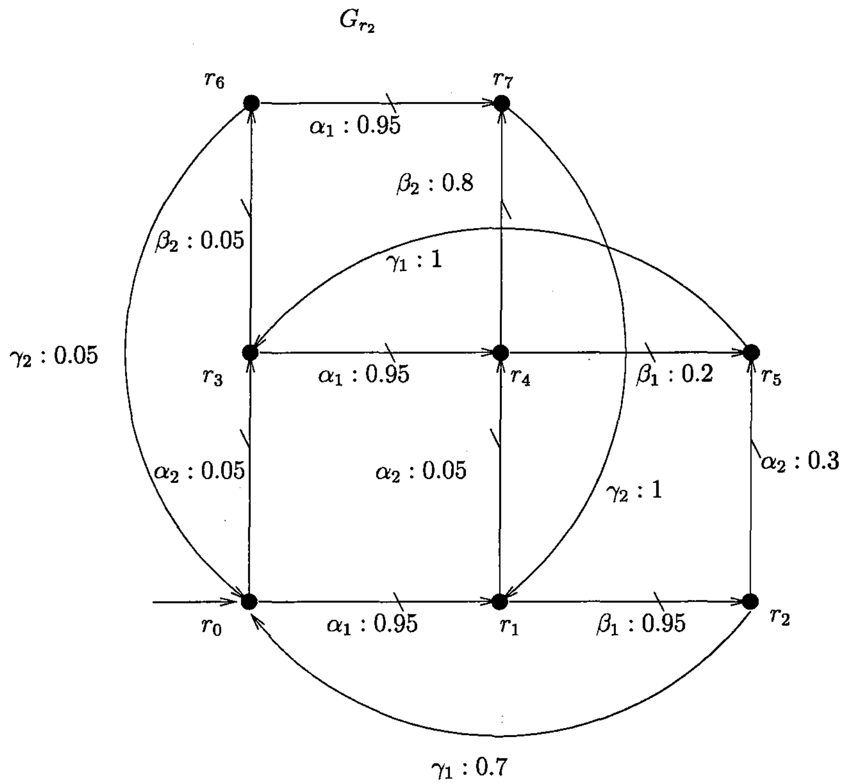

Figure 3.6: Requirements specification $G_{r_2}$

## 3.5 Summary

The focus of this chapter was on the solution of the probabilistic supervisory control problem. First, necessary and sufficient conditions for the existence of a probabilistic supervisor for the problem were presented. The conditions consist of two different types of conditions: the conditions for classical (non-probabilistic) controllability in classical supervisory control theory, and the conditions for probabilistic matching. The conditions for probabilistic matching are represented by a set of equalities and inequalities.

Next, an algorithm for the calculation of a supervisor, if one exists, and its correctness proof were introduced. The algorithm is iterative, and linear in the number of states of both the plant and the requirements specification. Also, although each iteration runs in time exponential in the maximal number of events possible from any state, this is not a limitation of the algorithm as this number is typically small.

Next, probabilistic supervisors were modeled using a reactive model, more concretely, using MDPs. Then, the chapter introduced a problem central to the thesis: the optimal probabilistic supervisory control problem. Finally, a simplified robot control system was used to illustrate an application of the research presented in this thesis.

# Chapter 4

# The Metric: Definition, Algorithms and Alternative Characterizations

This chapter introduces a metric as a means of quantifying the behavioural difference between two systems in the solution of the OPSCP as formulated in Section 3.3. The metric, based on the metric suggested in (Deng et al., 2006), is defined in Section 4.1 using a fixed point characterization. Then, two algorithms for the calculation of distances in the metric are given in Section 4.2. Further, we develop two alternative characterizations of the metric. First, in Section 4.3, the logical characterization measures how similar the systems are in terms of how closely they satisfy real-valued formulae of a logic. Section 4.4 offers a trace characterization by which the metric measures the difference of (appropriately discounted) probabilities of traces and sets of traces generated by systems, as well as some more complicated properties of traces. Section 4.5 justifies the use of the metric in the solution of the OPSCP.

## 4.1 The Metric: Definition

Let $G = (Q, \Sigma, \delta, q_0, p)$ be a PDES, where $Q = \{q_0, q_1, \dots q_{N-1}\}$. This is the system that will be used throughout the sequel. Our pseudometric is based on the metric suggested in (Deng et al., 2006) for a large class of automata which includes our generator: the two metrics are the same up to a constant.

First, (Deng et al., 2006) introduces the class $\mathcal{M}$ of 1-bounded pseudometrics on states with the (partial) ordering

$$d_1 \preceq d_2 \text{ if } \forall q_q, q_r \in Q \quad d_1(q_q, q_r) \geq d_2(q_q, q_r) \tag{4.1}$$

as was initially suggested in (Desharnais et al., 2002). Note that the ordering in (4.1) is reverse for the purpose of characterizing bisimilarity as the greatest fixed point of a function. It is proved that $(\mathcal{M}, \preceq)$ is a complete lattice.

Then, let $d \in \mathcal{M}$, and let the constant $e \in (0, 1]$ be a *discount factor* that determines the degree to which the difference in the probabilities of transitions further in the future is discounted: the smaller the value of $e$, the greater the discount on future transitions. Next, we introduce some useful notation. Let $q_q, q_r \in Q$ and let $\rho_{q_q}$ and $\rho_{q_r}$ be the distributions on $\Sigma \times Q$ induced by the states $q_q$ and $q_r$, respectively. Assume $0 \leq i, j \leq N - 1$. For notational convenience, we will write $\rho_{\sigma,i}$ instead of $\rho_{q_q}(\sigma, q_i)$, and, similarly, $\rho'_{\sigma,j}$ instead of $\rho_{q_r}(\sigma, q_j)$. Without loss of generality, we assume that the total mass of $\rho_{q_q}$ is greater than or equal to the total mass of $\rho_{q_r}$:

$$\sum_{\substack{\sigma \in \Sigma \\ 0 \leq i \leq N-1}} \rho_{\sigma,i} \geq \sum_{\substack{\sigma \in \Sigma \\ 0 \leq i \leq N-1}} \rho'_{\sigma,i}.$$

This assumption is not needed for nonterminating automata. Then, the distance between the distributions $\rho_{q_q}$ and $\rho_{q_r}$, $d(\rho_{q_q}, \rho_{q_r})$ (note the slight abuse of notation) for our generators is given as:

$$\begin{aligned} &\text{Maximize} \left( \sum_{\substack{\sigma \in \Sigma \\ 0 \leq i \leq N-1}} a_{\sigma,i} \rho_{\sigma,i} \right) - \left( \sum_{\substack{\sigma \in \Sigma \\ 0 \leq i \leq N-1}} a_{\sigma,i} \rho'_{\sigma,i} \right) \\ &\text{subject to} \quad 0 \leq a_{\sigma,i} \leq 1, \qquad \sigma \in \Sigma,\ 0 \leq i \leq N-1 \\ &\qquad\qquad a_{\sigma,i} - a_{\alpha,j} \leq c_{ij}^{\sigma\alpha}, \quad \sigma, \alpha \in \Sigma,\ 0 \leq i, j \leq N-1 \\ &\qquad\qquad \text{where} \\ &\qquad\qquad c_{ij}^{\sigma\alpha} = \begin{cases} e \cdot d(q_i, q_j) & \text{if } \sigma = \alpha \\ 1 & \text{otherwise} \end{cases} \end{aligned} \tag{4.2}$$

If the total mass of $\rho_{q_q}$ is strictly less than the total mass of $\rho_{q_r}$, $d(\rho_{q_q}, \rho_{q_r})$ is defined to be $d(\rho_{q_r}, \rho_{q_q})$.

The pseudometric on states, $d_{fp}$, is then given as the greatest fixed-point of the function $\mathcal{D}$ on $\mathcal{M}$ (here, for probabilistic generators, we give a simplified version of that in (Deng et al., 2006)):

$$\mathcal{D}(d)(q_q, q_r) = d(\rho_{q_q}, \rho_{q_r}), \quad d \in \mathcal{M}, q_q, q_r \in Q \tag{4.3}$$

The definition of the metric on distributions is a modified version of that of (Deng et al., 2006): the metric is changed such that the distances between the states in $d_{fp}$ are larger by a factor of $1/e$ than the distances in the metric defined in (Deng et al., 2006). This is done so that the distances in our metric are in the $[0, 1]$ interval instead of $[0, e]$. A number of existing results can be reused in reasoning about our metric. The distance between distributions (4.2) is a 1-bounded pseudometric, and is consistent with the ordering (4.1) (see (Desharnais et al., 2002), (van Breugel and Worrell, 2001a)). The proofs that the function defined by (4.3) is monotone on $\mathcal{M}$, and that it does have a greatest fixed point originate from (Desharnais et al., 2002). Also, according to Tarski's fixed point theorem, the greatest fixed point of function $\mathcal{D}$ can be reached through an iterative process that starts from the greatest element. As the number of transitions from a state of a probabilistic generator is finite, the greatest fixed point of the function $\mathcal{D}$ is reached after at most $\omega$ iterations (Deng et al., 2006; Desharnais et al., 2002) (equivalently, the closure ordinal of $\mathcal{D}$ is $\omega$). Therefore, the metric $d_{fp}$ can be reached through the following iterative process.

**Definition 4.1.** *The distance function $d^0_{fp} : Q \times Q \to [0, \infty)$ is defined as:*

$$d^0_{fp} = 0,$$

*and the distance function $d^{n+1}_{fp} : Q \times Q \to [0, \infty), n \in \mathbb{N}$, is given as:*

$$d^{n+1}_{fp} = \mathcal{D}(d^n_{fp}),$$

*where $\mathcal{D}$ is defined as in (4.3).*

**Remark 4.1.** *An important feature of $d_{fp}$ is to be noted: metric $d_{fp}$ is defined on any two states of a single PDES, not on two states that belong to different PDES. In order to define the distance between two PDES (with disjoint sets of states) as the distance between their initial states, a new PDES is created that represents the union of the two PDES (the union is defined in a natural way as will be presented formally in Section 5.2). However, in the sequel of the thesis, most often, the union will not be formalized as it does not change the distance between the states.*

The work of (Deng et al., 2006) does not offer any algorithms for the calculation of distances in their metric. In the following section, two algorithms for calculating distances in our metric are proposed.

## 4.2 Calculating the Metric

In this section, function $\mathcal{D}$ as defined by (4.2) and (4.3) is simplified for our probabilistic generators. Then, two algorithms for the calculation of the metric are suggested. The first algorithm calculates exact distances in the metric. The second algorithm approximates the distances with a prespecified accuracy. It is iterative and better suited for large systems.

### 4.2.1 Simplifying Function $\mathcal{D}$ for Deterministic Generators

The function that represents the pseudometric on distributions is defined as the linear programming problem (4.2). We now show that, for deterministic generators, this function, and consequently, function $\mathcal{D}$ as defined by (4.3), can be simplified by explicitly solving the linear programming problem (4.2).

First, recall that our generators are deterministic: for an event $\sigma$ and a state $q$, there is at most one state $q'$ such that $q' = \delta(q, \sigma)$. For the purposes of the following analysis of our deterministic generators, we rewrite the objective function of the optimization problem of (4.2) as:

$$\sum_{\sigma \in \Sigma} (a_{\sigma,i(q_q,\sigma)} \rho_{\sigma,i(q_q,\sigma)} - a_{\sigma,j(q_r,\sigma)} \rho'_{\sigma,j(q_r,\sigma)}) \tag{4.4}$$

where $i(q_q, \sigma) = i$ such that $q_i = \delta(q_q, \sigma)$ if $\delta(q_q, \sigma)!$, and $i(q_q, \sigma) = 0$, otherwise. We arbitrarily choose $i(q_q, \sigma)$ to be 0 when $\delta(q_q, \sigma)$ is not defined although we could have chosen any other $i \in \{1, \ldots, N-1\}$. This is because when $\delta(q_q, \sigma)!$ does not hold, then $\rho_{\sigma, i(q_q, \sigma)} = 0$ for any $i(q_q, \sigma) \in \{1, \ldots, N-1\}$. Similarly, $j(q_r, \sigma) = j$ such that $q_j = \delta(q_r, \sigma)$ if $\delta(q_r, \sigma)!$, and $j(q_r, \sigma) = 0$, otherwise. For readability purposes, we will write $i$ instead of $i(q_q, \sigma)$, and $j$ instead of $j(q_r, \sigma)$.

We are now ready to state our first result.

**Lemma 4.1.** *Let $G = (Q, \Sigma, \delta, q_0, p)$ be a PDES. Then, the function $\mathcal{D}$ simplifies to:*

$$\mathcal{D}(d)(q_q, q_r) = \sum_{\sigma \in \Sigma} max(\rho_{\sigma,i} - \rho'_{\sigma,j} + c_{ij}\rho'_{\sigma,j}, c_{ij}\rho_{\sigma,i}) \tag{4.5}$$

*where $c_{ij} = e \cdot d(q_i, q_j)$, and $i$ and $j$ denote $i(q_q, \sigma)$ and $j(q_r, \sigma)$, respectively, as defined in (4.4).*

*Proof.* The objective function (4.4) can be maximized by maximizing each of its summands separately. In order to explain this observation, we consider a summand $a_{\sigma,i}\rho_{\sigma,i} - a_{\sigma,j}\rho'_{\sigma,j}$. Due to the generator's determinism, there is no other nonzero summand containing $a_{\sigma,k}$, $0 \leq k \leq N-1$, $k \neq i$, $k \neq j$. Therefore, the last constraint of (4.2) for any two coefficients $a_{\sigma,i}$ and $a_{\alpha,j}$ $(0 \leq i, j \leq N-1)$ from different summands becomes $a_{\sigma,i} - a_{\alpha,j} \leq 1$. This constraint is already implied by the first constraint, so we can independently pick the coefficients $a$ in different summands, and, consequently, independently maximize the summands in order to maximize the sum.

In order to maximize a summand of the objective function (4.4), we solve the following linear programming problem for $\sigma \in \Sigma$:

$$\begin{aligned} &\text{Maximize } (a_{\sigma,i}\rho_{\sigma,i} - a_{\sigma,j}\rho'_{\sigma,j}) \\ &\text{subject to } 0 \leq a_{\sigma,i}, a_{\sigma,j} \leq 1, \\ &\qquad\qquad a_{\sigma,i} - a_{\sigma,j} \leq c_{ij} \end{aligned}$$

where $i$ and $j$ are defined as in (4.4), and $c_{ij} = ed(q_i, q_j)$ as before. Also, note that the set of constraints does not contain the inequality $a_{\sigma,j} - a_{\sigma,i} \leq c_{ji}$. In order to maximize the given function, the coefficient $a_{\sigma,i}$ is to be chosen to be greater than $a_{\sigma,j}$ since the given constraints allow it. In that case, since $c_{ij} = c_{ji}$, if $a_{\sigma,i} - a_{\sigma,j} \leq c_{ij}$, then $a_{\sigma,j} - a_{\sigma,i} \leq c_{ji}$ follows, so the latter constraint is redundant. Further, it is not hard to see that the solution of the given linear programming problem for $\rho_{\sigma,i} \geq \rho'_{\sigma,j}$ is equal to $\rho_{\sigma,i} - \rho'_{\sigma,j} + c_{ij}\rho'_{\sigma,j}$. We can solve this problem using graphical method, simplex method or using the following line of reasoning. In order to maximize the given function, we can either choose $a_{\sigma,i}$ to be 1 and then pick $a_{\sigma,j}$ so that it has the minimal value for the given constraints, or we choose $a_{\sigma,j}$ to be 0, and then pick $a_{\sigma,i}$ so that it has the maximal value under the given constraints. In the first case, we pick $a_{\sigma,i}$ to be 1, $a_{\sigma,j}$ to be $1 - c_{ij}$, and value of the objective function is $\rho_{\sigma,i} - \rho'_{\sigma,j} + c_{ij}\rho'_{\sigma,j}$. In the second case, since $a_{\sigma,j}$ is 0, then $a_{\sigma,i}$ is equal to $c_{ij}$, and the objective function becomes $c_{ij}\rho_{\sigma,i}$. The latter is our solution, since $c_{ij}\rho_{\sigma,i} = c_{ij}\left(\rho_{\sigma,i} - \rho'_{\sigma,j} + \rho'_{\sigma,j}\right) \leq \rho_{\sigma,i} - \rho'_{\sigma,j} + c_{ij}\rho'_{\sigma,j}$ (for $\rho_{\sigma,i} \geq \rho'_{\sigma,j}$ and $c_{ij} \in [0,1]$). Using the same reasoning, for $\rho_{\sigma,i} < \rho'_{\sigma,j}$, the maximum is reached at $(a_i, a_j) = (c_{ij}, 0)$ and its value is $c_{ij}\rho_{\sigma,i}$.

Now, we put together the presented solution of the linear programming problem (4.2). The distance between the distributions $\rho_{q_q}$ and $\rho_{q_r}$ is then:

$$d(\rho_{q_q}, \rho_{q_r}) = \sum_{\sigma\in\Sigma} f(d, q_q, q_r, \sigma), \text{ where} \tag{4.6}$$

$$f(d, q_q, q_r, \sigma) = \begin{cases} \rho_{\sigma,i} - \rho'_{\sigma,j} + c_{ij}\rho'_{\sigma,j}, & \text{if } \rho_{\sigma,i} \geq \rho'_{\sigma,j} \\ c_{ij}\rho_{\sigma,i}, & \text{otherwise} \end{cases}$$

*or, equivalently,*

$$f(d, q_q, q_r, \sigma) = max(\rho_{\sigma,i} - \rho'_{\sigma,j} + c_{ij}\rho'_{\sigma,j}, c_{ij}\rho_{\sigma,i}), \tag{4.7}$$

where $c_{ij} = e \cdot d(q_i, q_j)$ as before, and $i$ and $j$ denote $i(q_q, \sigma)$ and $j(q_r, \sigma)$, respectively, as defined as in (4.4). Equation (4.7) comes from the fact that, for any $d \in \mathcal{M}$, it holds that:

$$max(\rho_{\sigma,i} - \rho'_{\sigma,j} + c_{ij}\rho'_{\sigma,j}, c_{ij}\rho_{\sigma,i}) = \begin{cases} \rho_{\sigma,i} - \rho'_{\sigma,j} + c_{ij}\rho'_{\sigma,j}, & \text{if } \rho_{\sigma,i} \geq \rho'_{\sigma,j} \\ c_{ij}\rho_{\sigma,i}, & \text{otherwise.} \end{cases} \tag{4.8}$$

To summarize, the function $\mathcal{D}(d)$ for our model is given as:

$$\mathcal{D}(d)(q_q, q_r) = \sum_{\sigma\in\Sigma} max(\rho_{\sigma,i} - \rho'_{\sigma,j} + c_{ij}\rho'_{\sigma,j}, c_{ij}\rho_{\sigma,i})$$

where, again, $c_{ij} = e \cdot d(q_i, q_j)$ as before, and $i$ and $j$ denote $i(q_q, \sigma)$ and $j(q_r, \sigma)$, respectively, as defined in (4.4). □

**Example**

For the systems from the figure:

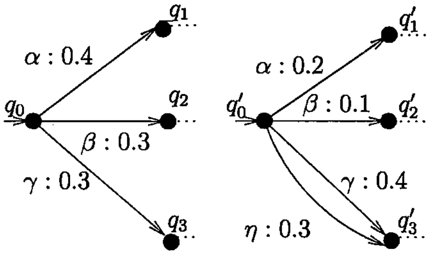

Figure 4.1: Function $\mathcal{D}$: Example

we have:

$$\begin{aligned}\mathcal{D}(d)(q_0, q'_0) &= (0.2 + 0.2ed(q_1, q'_1)) + (0.2 + 0.1ed(q_2, q'_2)) + 0.3ed(q_3, q'_3) \\ &= 0.4 + 0.2ed(q_1, q'_1) + 0.1ed(q_2, q'_2) + 0.3ed(q_3, q'_3) \\ &= \mathcal{D}(d)(q'_0, q_0)\end{aligned}$$

As stated in Section 4.1, the pseudometric $d_{fp}$ is now characterized as the greatest fixed point of function $\mathcal{D}$.

### 4.2.2 Calculating the Metric: Algorithms

For $e \in (0, 1)$, we will prove that the function $\mathcal{D}$ has only one fixed point, $d^*$, and, consequently, $d_{fp} = d^*$. Then, two algorithms for calculating the distances in metric $d_{fp}$ are suggested.

First, some useful definitions and results from linear algebra are introduced.

A real $n \times n$ matrix $A = (a_{ij})$ defines a linear mapping from $\mathbb{R}^n$ to $\mathbb{R}^n$, and we will write $A \in L(\mathbb{R}^n)$ to denote either the matrix or the linear function, as no distinction between the two will be made. Also, the absolute value of column vector $x = (x_1, \dots, x_n)^T \in \mathbb{R}^n$ will be denoted by $|x|$, and defined as $|x| = (|x_1|, \dots, |x_n|)^T$. A partial ordering on $\mathbb{R}^n$ is defined in a natural way:

$$\forall x, y \in \mathbb{R}^n \;\; x \leq y \Leftrightarrow (\forall i = 1, \dots, n \;\; x_i \leq y_i).$$

**Definition 4.2.** *For any complex $n \times n$ matrix $A$, the spectral radius of $A$ is defined as the maximum of $|\lambda_1|, \cdots, |\lambda_n|$, where $\lambda_1, \cdots, \lambda_n$ are the eigenvalues of $A$.*

The spectral radius of $A$, denoted $\varphi(A)$, satisfies $\varphi(A) \leq ||A||$, where $||A||$ is an arbitrary norm on $\mathbb{R}^n$. During the course of the following proof, we will make use of infinity norm $||A||_\infty = \underset{1 \leq i \leq n}{max} \sum_{j=1}^{n} |a_{ij}|$. Also, the proof will use functions $div : N \times N \rightarrow N$ and $mod : N \times N \rightarrow N$ defined in the standard manner to be the quotient and remainder, respectively, of the division of the first argument with the second.

**Definition 4.3.** *(Ortega and Rheinboldt, 1970) An operator $G : D \subseteq \mathbb{R}^n \rightarrow \mathbb{R}^n$ is called a $P$-contraction on a set $D_0 \subseteq D$ if there exists a linear operator $P \in L(\mathbb{R}^n)$ such that $P \geq 0$, $\varphi(P) < 1$ and*

$$|G(x) - G(y)| \leq P\,|x - y| \;\; \textit{for all } x, y \in D_0. \tag{4.9}$$

Now, let $d \in \mathcal{M}$. Next, we define the function $\mathcal{V} : \mathcal{M} \rightarrow [0, 1]^{N^2}$:

$$\mathcal{V}(d) = (d(q_0, q_0), d(q_0, q_1), \dots, d(q_{N-1}, q_{N-2}), d(q_{N-1}, q_{N-1}))^T.$$

Note that the vector $\mathcal{V}(d)$ could be further cut down, as $d(s, s) = 0$ and $d(s, t) = d(t, s)$ for any $s, t \in Q$. However, for ease of presentation, we will not decrease the size of the vector. Therefore, $\mathcal{V}(d) = (\mathcal{V}_1(d), \mathcal{V}_2(d), \cdots, \mathcal{V}_{N^2}(d))^T$, where $\mathcal{V}_k(d)$ for $k \in \{1, \dots, N^2\}$ is given as:

$$\mathcal{V}_k(d) = d(q_i, q_j), \;\; i = k \; div \; (N + 1), \;\; j = (k - 1) \; mod \; N.$$

Now, the function $\mathcal{D}$ is redefined in a natural way as $\mathcal{D}(\mathcal{V}(d)) = (\mathcal{D}_1(\mathcal{V}(d)), \dots, \mathcal{D}_{N^2}(\mathcal{V}(d)))^T$, where for any $k \in \{1, \dots, N^2\}$:

$$\mathcal{D}_k(\mathcal{V}(d)) = d(\rho_{q_i}, \rho_{q_j}), \;\; i = k \; div \; (N + 1), j = (k - 1) \; mod \; N. \tag{4.10}$$

Further, let $D_0 = \{\mathcal{V}(d) | d \in \mathcal{M}\}$.

**Lemma 4.2.** *The function $\mathcal{D}$ is a $P$-contraction on $D_0$.*

*Proof.* Let $d', d'' \in \mathcal{M}$, and $\mathfrak{d}' = \mathcal{V}(d')$, and $\mathfrak{d}'' = \mathcal{V}(d'')$. Let $k \in \{1, \dots, N^2\}$, and let $i$ and $j$ $(0 \leq i, j \leq N-1)$ be given as in (4.10). Also, let $t(i,\sigma) = t$ such that $\delta(q_i, \sigma) = q_t$ if $\delta(q_i,\sigma)!$, and $t(i,\sigma) = 0$, otherwise. Similarly, $l(j,\sigma) = l$ such that $\delta(q_j,\sigma) = q_l$ if $\delta(q_j,\sigma)!$, and $l(j,\sigma) = 0$, otherwise. Again, for notational convenience, we will write $t$ instead of $t(i,\sigma)$, and $l$ instead of $l(j,\sigma)$. Also, we will write $\rho_{\sigma,t}$ instead of $\rho_{q_i}(\sigma, q_t)$, and, similarly, $\rho'_{\sigma,l}$ instead of $\rho_{q_j}(\sigma, q_l)$ for $q_t, q_l \in Q$. Then:

$$
\begin{aligned}
&|\mathcal{D}_k(\mathfrak{d}') - \mathcal{D}_k(\mathfrak{d}'')| = \left|d'(\rho_{q_i}, \rho_{q_j}) - d''(\rho_{q_i}, \rho_{q_j})\right| \\
&= |\sum_{\sigma\in\Sigma} max(\rho_{\sigma,t} - \rho'_{\sigma,l} + ed'(q_t,q_l)\rho'_{\sigma,l}, ed'(q_t,q_l)\rho_{\sigma,t}) \\
&\qquad - \sum_{\sigma\in\Sigma} max(\rho_{\sigma,t} - \rho'_{\sigma,l} + ed''(q_t,q_l)\rho'_{\sigma,l}, ed''(q_t,q_l)\rho_{\sigma,t})| \\
&= |\sum_{\sigma\in\Sigma}(max(\rho_{\sigma,t} - \rho'_{\sigma,l} + ed'(q_t,q_l)\rho'_{\sigma,l}, ed'(q_t,q_l)\rho_{\sigma,t}) \\
&\qquad - max(\rho_{\sigma,t} - \rho'_{\sigma,l} + ed''(q_t,q_l)\rho'_{\sigma,l}, ed''(q_t,q_l)\rho_{\sigma,t}))| \\
&= \left|\sum_{\sigma\in\Sigma} e(d'(q_t,q_l) - d''(q_t,q_l)) min(\rho_{\sigma,t}, \rho'_{\sigma,l})\right| \\
&\left(\begin{array}{l}
\text{since, for any } d \in \mathcal{M}, \text{according to (4.8), it holds that:} \\
max(\rho_{\sigma,t} - \rho'_{\sigma,l} + ed(q_t,q_l)\rho'_{\sigma,l}, ed(q_t,q_l)\rho_{\sigma,t}) = \\
\qquad\qquad \left\{\begin{array}{ll} \rho_{\sigma,t} - \rho'_{\sigma,l} + ed(q_t,q_l)\rho'_{\sigma,l}, & \text{if } \rho_{\sigma,t} \geq \rho'_{\sigma,l} \\ \quad ed(q_t,q_l)\rho_{\sigma,t}, & \text{otherwise} \end{array}\right.
\end{array}\right) \\
&\leq e\sum_{\sigma\in\Sigma} min(\rho_{\sigma,t}, \rho'_{\sigma,l}) \, |d'(q_t,q_l) - d''(q_t,q_l)| \qquad (4.11) \\
&\leq e \sum_{\substack{\sigma\in\Sigma \\ m=tN+l+1}} min(\rho_{\sigma,t}, \rho'_{\sigma,l}) \, |\mathfrak{d}'_m - \mathfrak{d}''_m| \, . \qquad (4.12)
\end{aligned}
$$

Note that $t = t(i,\sigma)$ and $l = l(j,\sigma)$ are also functions of $k$ (since $i$ and $j$ are functions of $k$). Now, without the explicit construction of matrix $P$, we can see from (4.12) that there exists $P$ such that $|\mathcal{D}(\mathfrak{d}') - \mathcal{D}(\mathfrak{d}'')| \leq P\,|\mathfrak{d}' - \mathfrak{d}''|$, where

$$\begin{aligned}||P||_\infty &= \max_k\{e\sum_{\sigma\in\Sigma} min(\rho_{\sigma,t},\rho'_{\sigma,l})\}\\ &\le e\\ &\text{(since } \sum_{\substack{\sigma\in\Sigma\\ t\in\{0,\ldots,N-1\}}} \rho_{\sigma,t}=1,\quad \sum_{\substack{\sigma\in\Sigma\\ l\in\{0,\ldots,N-1\}}} \rho'_{\sigma,l}=1)\\ &< 1 \text{ (since } e\in(0,1))\end{aligned}$$

Therefore, $\varphi(P) < 1$ and, since, obviously, $P \geq 0$, then $\mathcal{D}$ is $P$-contraction. □

**Lemma 4.3.** *Let $d', d'' \in \mathcal{M}$, and $\mathfrak{d}' = \mathcal{V}(d')$, and $\mathfrak{d}'' = \mathcal{V}(d'')$. For any $k \in \{1,\ldots,N^2\}$, there exists $m \in \{1,\ldots,N^2\}$ such that:*

$$|\mathcal{D}_k(\mathfrak{d}') - \mathcal{D}_k(\mathfrak{d}'')| \leq e\,|\mathfrak{d}'_m - \mathfrak{d}''_m|$$

*Proof.*

$$\begin{aligned}|\mathcal{D}_k(\mathfrak{d}') - \mathcal{D}_k(\mathfrak{d}'')| &\leq e\sum_{\sigma\in\Sigma} min(\rho_{\sigma,t},\rho'_{\sigma,l})\,|d'(q_t,q_l)-d''(q_t,q_l)| \text{ (from (4.11))}\\ &\leq e\sum_{\sigma\in\Sigma} min(\rho_{\sigma,t},\rho'_{\sigma,l}) \max_{\substack{(t,l)\in\\ \{(t(\sigma,i),l(\sigma,j))\mid\sigma\in\Sigma\}}}\{|d'(q_t,q_l)-d''(q_t,q_l)|\}\\ &\leq e\sum_{\sigma\in\Sigma} min(\rho_{\sigma,t},\rho'_{\sigma,l})\,|d'(q_r,q_s)-d''(q_r,q_s)|\\ &\qquad\qquad \text{for some } r,s\in\{0,\ldots,N-1\}\\ &\leq e|d'(q_r,q_s)-d''(q_r,q_s)|\\ &\qquad \text{(since } \sum_{\substack{\sigma\in\Sigma\\ t\in\{0,\ldots,N-1\}}} \rho_{\sigma,t} = \sum_{\substack{\sigma\in\Sigma\\ l\in\{0,\ldots,N-1\}}} \rho'_{\sigma,l}=1)\\ &\leq e\,|\mathfrak{d}'_m - \mathfrak{d}''_m| \text{ for some } m\in\{1,\ldots,N^2\}\end{aligned}$$

□

**Theorem 4.1.** *For any $\mathfrak{d}^0 \in D_0$, the sequence*

$$\mathfrak{d}^{n+1} = \mathcal{D}(\mathfrak{d}^n),\ \ n = 0,1,\ldots$$

*converges to the unique fixed point of $\mathcal{D}$ in $D_0$, $\mathfrak{d}^*$, and the error estimate is given componentwise ($k \in \{1, \ldots, N^2\}$) as:*

$$|\mathfrak{d}_k^n - \mathfrak{d}_k^*| \leq e^n, \;\; n = 0, 1 \ldots \tag{4.13}$$

*Proof.* Note that this is a variant of the contraction-mapping theorem extended to $P$-contractions (see (Ortega and Rheinboldt, 1970), Theorem 13.1.2.). A similar proof technique is employed.
Let $n, m \geq 1$. Then:

$$\begin{aligned}
&|\mathfrak{d}_k^{n+m} - \mathfrak{d}_k^n| && (4.14)\\
&\leq \sum_{t=1}^{m} |\mathfrak{d}_k^{n+t} - \mathfrak{d}_k^{n+t-1}| \\
&\leq \sum_{t=1}^{m} e^t |\mathfrak{d}_{i(t)}^n - \mathfrak{d}_{i(t)}^{n-1}| \\
&\text{(applying Lemma 4.3 } t \text{ times, where } i(t) \in \{1, \ldots, N^2\}) \\
&\leq \sum_{t=1}^{m} e^t \underset{i(t)}{max} \{|\mathfrak{d}_{i(t)}^n - \mathfrak{d}_{i(t)}^{n-1}|\} \\
&\leq \left(\sum_{t=1}^{m} e^t\right) |\mathfrak{d}_j^n - \mathfrak{d}_j^{n-1}| \text{ (for some } j \in \{1, \ldots, N^2\}) \\
&\leq (1-e)^{-1}\, e |\mathfrak{d}_j^n - \mathfrak{d}_j^{n-1}| \text{ (since } \sum_{t=0}^{m} e^t \leq (1-e)^{-1} \text{ for } m \geq 0) && (4.15)\\
&\leq (1-e)^{-1} e^n |\mathfrak{d}_l^1 - \mathfrak{d}_l^0| && (4.16)\\
&\text{(for some } l \in \{1, \ldots, N^2\}\text{, using Lemma 4.3 } (n-1) \text{ times)}
\end{aligned}$$

Therefore, the sequence $\{\mathfrak{d}_k^n\}_{n \geq 0}$ is a Cauchy sequence and hence converges to some $\mathfrak{d}_k^*$, and, consequently, the sequence $\{\mathfrak{d}^n\}_{n \geq 0}$ converges to some $\mathfrak{d}^* \in D_0$. Also, we have:

$$|\mathfrak{d}^* - \mathcal{D}(\mathfrak{d}^*)| \leq |\mathfrak{d}^* - \mathfrak{d}^{n+1}| + |\mathcal{D}(\mathfrak{d}^n) - \mathcal{D}(\mathfrak{d}^*)| \leq |\mathfrak{d}^* - \mathfrak{d}^{n+1}| + P|\mathfrak{d}^n - \mathfrak{d}^*|$$

When we let $n \longrightarrow \infty$, we see that $\mathfrak{d}^* = \mathcal{D}(\mathfrak{d}^*)$.

Then, it should be proven that $\eth^*$ is the only fixed point in $D_0$. Assume that there is another fixed point of $\mathcal{D}$ in the same set $D_0$, $\eth^+$. Then,

$$|\eth^* - \eth^+| = |\mathcal{D}(\eth^*) - \mathcal{D}(\eth^+)| \leq P|\eth^* - \eth^+|$$

Hence, $(I-P)|\eth^*-\eth^+| \leq 0$. However, since $\varphi(P) < 1$, $(I-P)^{-1} = \sum_{i=0}^{\infty} P^i \geq 0$ (see (Ortega and Rheinboldt, 1970), 2.4.5.), then $|\eth^* - \eth^+| \leq 0$. Therefore, $\eth^* = \eth^+$.

The error estimate (4.13) can be easily proven by induction and using Lemma 4.3. The base case (for $n = 0$) is trivially satisfied. Assume that $|\eth_k^n - \eth_k^*| \leq e^n$ for any $n \in \mathbb{N}$, and $k = 1, \ldots, N^2$. Then:

$$\begin{aligned}|\eth_k^{n+1} - \eth_k^*| &= |\mathcal{D}_k(\eth^n) - \mathcal{D}_k(\eth^*)| \\ &\leq e|\eth_m^n - \eth_m^*|, \text{ for some } m = 1, \ldots, N^2 \quad \text{(according to Lemma 4.3)} \\ &\leq e^{n+1} \text{ (from induction hypothesis)}\end{aligned}$$

□

**Remark 4.2.** *It is important to note that the error in the n-th iteration ($n = 1, 2 \ldots$) can also be estimated as ($k = 1, \ldots, N^2$)*

$$|\eth_k^n - \eth_k^*| \leq (1-e)^{-1} e^n \underset{i}{max}\{|\eth_i^1 - \eth_i^0|\},$$

*or, as:*

$$|\eth_k^n - \eth_k^*| \leq (1-e)^{-1} e \ \underset{i}{max}\{|\eth_i^n - \eth_i^{n-1}|\},$$

*that follow from (4.16), and (4.15), respectively, when $m \to \infty$ in (4.14).*

Now, using the presented analysis, the following two algorithms for the calculation of the distances between the states of PDES in the chosen pseudometric are suggested.

### Algorithm 1

Theorem 4.1 proves that the system of equations

$$\eth = \mathcal{D}(\eth) \tag{4.17}$$

has a unique solution. The system (4.17) is a system of linear equations. Therefore, the system (4.17) can be rewritten into the standard form $A\mathfrak{d} = b$, where $A$ is a $N^2 \times N^2$ matrix and $b$ is a column vector of dimension $N^2$. Therefore, the distances in the metric $d_{fp}$ can be calculated by solving this system of linear equations.

### Algorithm 2

Theorem 4.1 suggests an iterative algorithm to approximate distances in the metric $d_{fp}$ between the states of a probabilistic generator. Let $d^0(q_q, q_r) = 0$ for any two states $q_q, q_r \in Q$. As before, let $\rho_{q_q}$ and $\rho_{q_r}$ be the distributions induced by the states $q_q$ and $q_r$, respectively. The $n$-th iteration of the algorithm calculates the distance $d^n$ between each two states $q_q, q_r \in Q$:

$$d^n(q_q, q_r) = \sum_{\sigma \in \Sigma} max(\rho_{\sigma,i} - \rho'_{\sigma,j} + c_{ij}\rho'_{\sigma,j}, c_{ij}\rho_{\sigma,i})$$

where $c_{ij} = e \cdot d^{n-1}(q_i, q_j)$, and $i = i(q_q, \sigma)$ and $j = j(q_r, \sigma)$ are defined as in (4.4). The accuracy of the solution found at the $n$-th iteration is $e^n$.

The iterative method can be useful for systems with large $N^2$, where the direct method can be rather expensive. Furthermore, the mathematical apparatus used to reach the iterative method will be reused in the solution of the OPSCP in Chapter 5. The number of iterations sufficient to reach the accuracy of $\epsilon$ is $\lceil \log_e \epsilon \rceil$. This term is obtained from the fact that the number of iterations $n$ for which an accuracy $\epsilon$ is achieved should be the smallest natural number for which $\epsilon \geq e^n$ is satisfied.

The pseudometric of (Deng et al., 2006) is closely related to the ones suggested in (van Breugel and Worrell, 2001a; van Breugel and Worrell, 2006; Ferns et al., 2004; Ferns et al., 2005). It is not a surprise then that our iterative algorithm turns out to be similar to those of (van Breugel and Worrell, 2001a; van Breugel and Worrell, 2006; Ferns et al., 2004; Ferns et al., 2005) that calculate distances in similar pseudometrics suggested for different kinds of probabilistic system. Those algorithms are similar to ours in that they all approximate the distances by fixed point iterations. In each iteration, however, the algorithms of (van Breugel and Worrell, 2001a; van Breugel and Worrell, 2006; Ferns et al., 2004; Ferns et al., 2005) solve the special case of the linear programming problem - the transshipment problem - for each pair of states. The transshipment problem can be solved in time polynomial in the number of states. On the other hand, in each iteration, for each pair of states, our algorithm simply evaluates an expression. The evaluation is done in time linear in $|\Sigma|$. The simplification is possible due to the nature of our generators. Also, while our algorithm is derived from the characterization of the pseudometric as the greatest fixed point of a monotone function, the pseudometric of (van Breugel and Worrell, 2001a; van Breugel and Worrell, 2006) is defined as the pseudometric kernel induced by the unique map from the probabilistic transition system, viewed as a coalgebra, to the terminal coalgebra. In that regard, the derivation of (Ferns et al., 2005) is more similar to ours since it starts from the fixed point characterization, and then uses the Banach fixed point theorem, whereas we use its generalization to $P$-contractions.

Also, it should be stressed that the presented algorithms work for $e \in (0,1)$. However, (van Breugel et al., 2008) presented an algorithm for calculating distances in the pseudometric of (Desharnais et al., 2004) for a variant of Markov chains for the case when $e = 1$. The key element in the algorithm is Tarski's decision procedure for the first order theory of real closed fields that solves the satisfiability problem for the existential fragment of the first order of the real closed fields. We believe that this algorithm can be modified to calculate $d_{fp}$ between the states of probabilistic generators. However, as the satisfiability problem for the existential fragment of the first order of the real closed fields is PSPACE, an efficient calculation of distances in the metric for $e = 1$ is still an open problem.

## 4.3 Logical Characterization

The metric with the fixed point characterization as presented in Section 4.1 is now given a logical characterization, along the lines of (Desharnais et al., 2002). The logic used is real-valued so that it can handle probabilities. However, the logic itself is different than that of (Desharnais et al., 2002) as our models are generative. Also, the main part of the characterization proof is, to the best of our knowledge, novel. The idea behind the logical characterization is that the distance between two systems is measured by a logical formula that distinguishes between the systems the most. If the systems are probabilistic bisimilar, there should not be a formula that distinguishes between the systems.

As before, let $G = (Q, \Sigma, \delta, q_0, p)$ be a nonterminating generator, where $Q = \{q_0, q_1, \ldots q_{N-1}\}$.

**Definition 4.4.** *The logic $\mathcal{L}$ is defined as follows:*

$$\phi ::= \mathbf{1} \mid \langle\sigma\rangle\phi \mid \bigvee_{\sigma\in\Theta} \langle\sigma\rangle\phi \mid 1-\phi \mid \phi\ominus p,$$

*where $p$ is a rational number in $[0,1]$, $\sigma \in \Sigma$, and $\Theta \subseteq \Sigma$.*

The formula $\phi$ evaluated at a state $q \in Q$, $\phi(q)$, is a measure of how much $\phi$ is satisfied at the state. The semantics of the logic $\mathcal{L}$ is given next.

**Definition 4.5.** *Let $q \in Q$, and $\rho_q$ be the probability distribution on $\Sigma \times Q$ induced by state $q$. Let $\phi \in \mathcal{L}$, and $\psi : \Sigma \to \mathcal{L}$. The notation $\psi_\sigma$ will be used for $\psi(\sigma)$, $\sigma \in \Sigma$. Then:*

$$\begin{aligned}
\mathbf{1}(q) &= 1\\
\langle\sigma\rangle\phi(q) &= e\rho_q(\sigma, q_{i(q,\sigma)})\phi(q_{i(q,\sigma)})\\
\bigvee_{\sigma\in\Theta}\langle\sigma\rangle\psi_\sigma(q) &= \sum_{\sigma\in\Theta} e\rho_q(\sigma, q_{i(q,\sigma)})\psi_\sigma(q_{i(q,\sigma)})\\
(1-\phi)(q) &= 1-\phi(q)\\
(\phi\ominus p)(q) &= max(\phi(q)-p, 0)
\end{aligned}$$

*where $\sigma \in \Sigma$, and, as before, $i(q,\sigma) = i$ such that $q_i = \delta(q,\sigma)$ if $\delta(q,\sigma)!$, and $i(q,\sigma) = 0$, otherwise.*

The presented logic represents a probabilistic modification of Hennessy-Milner logic (Hennessy and Milner, 1985). The formula $\mathbf{1}$ corresponds to the constant *true*, $\langle\sigma\rangle\phi$ is the next operator, $1-\phi$ corresponds to negation, and $\phi\ominus p$ provides for the testing of the value of $\phi$ (Desharnais et al., 2002). The logic supports only a specific disjunction of form $\bigvee\langle\sigma\rangle\phi$; extending it to $\bigvee\phi$

would require a more complicated formalization not necessary for the main result to be presented.

The metric $d_L$ is defined next. The distance between two states is measured by a formula that differentiates them the most.

**Definition 4.6.** *For every $q_q, q_r \in Q$, the metric $d_L$ is defined as:*

$$d_L(q_q, q_r) = \sup_{\phi \in \mathcal{L}} \{ |\phi(q_q) - \phi(q_r)| \}.$$

In this logical setting, the smaller the factor $e$ is, the more discounted the difference is for complex formulae.

An example is given in Figure 4.2. The states $q_0$ and $q_0'$ are at the distance $0.25e + 0.75e^2$ in the metric $d_L$, witnessed by the formula $\phi = \bigvee_{\sigma \in \{\alpha,\beta\}} \langle\sigma\rangle\phi_\sigma$, where $\phi_\alpha = 1 - \langle\gamma\rangle\mathbf{1}$, and $\phi_\beta = \langle\tau\rangle\mathbf{1}$. Further, states $q_1$ and $q_1'$ (also, $q_1$ and $q_2'$) are at the distance $e$ as witnessed by the formula $\phi = \langle\tau\rangle\mathbf{1}$.

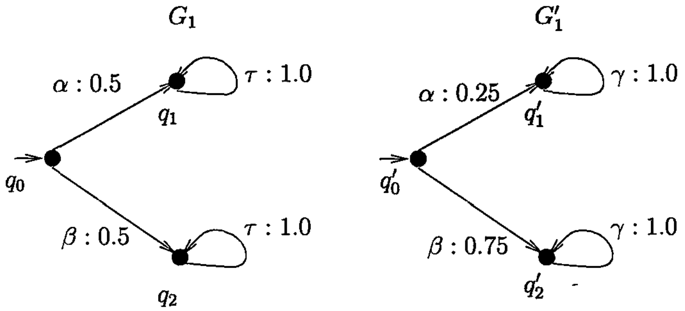

Figure 4.2: Example

The goal is to show that the metric $d_{fp}$ is equal to the metric $d_L$ up to constant $e$.

**Lemma 4.4.** *Let $q_q, q_r \in Q$. For a function $\psi : \Sigma \longrightarrow \mathcal{L}$, the shorthand notation $\psi_\sigma$ will be used for $\psi(\sigma)$. Then:*

$$d_L(q_q, q_r) = \sup_{\psi_\sigma \in \mathcal{L}} \left\{ \left| \bigvee_{\sigma \in \Sigma} \langle\sigma\rangle\psi_\sigma(q_q) - \bigvee_{\sigma \in \Sigma} \langle\sigma\rangle\psi_\sigma(q_r) \right| \right\}.$$

*Proof.* The idea of the proof is similar to that of (Desharnais et al., 2002), Lemma 4.4. As before, for a function $\varphi : \Sigma \longrightarrow \mathcal{L}$, the shorthand notation $\varphi_\sigma$

will be used for $\varphi(\sigma)$. It should be proven that there exist $\varphi_\sigma \in \mathcal{L}$, $\sigma \in \Sigma$, such that

$$\left|\bigvee_{\sigma\in\Sigma}\langle\sigma\rangle\varphi_\sigma(q_q) - \bigvee_{\sigma\in\Sigma}\langle\sigma\rangle\varphi_\sigma(q_r)\right| \geq |\phi(q_q) - \phi(q_r)|,$$

for any $\phi \in \mathcal{L}$. Induction on the structure of $\phi$ is used. The base case ($\phi = \mathbf{1}$) is satisfied. Next, the case when $\phi = \langle\alpha\rangle\phi'$, $\phi' \in \mathcal{L}$, is investigated. It should be shown that

$$\left|\bigvee_{\sigma\in\Sigma}\langle\sigma\rangle\varphi_\sigma(q_q) - \bigvee_{\sigma\in\Sigma}\langle\sigma\rangle\varphi_\sigma(q_r)\right| \geq |\langle\alpha\rangle\phi'(q_q) - \langle\alpha\rangle\phi'(q_r)|.$$

If, for $\sigma \neq \alpha$, $\varphi_\sigma = 1 - \mathbf{1} = 0$, and $\varphi_\sigma = \phi'$ for $\sigma = \alpha$, the inequality is obviously satisfied. The case for $\phi = \bigvee_{\sigma\in\Theta}\langle\sigma\rangle\varphi_\sigma$, for $\Theta \subseteq \Sigma$, is proven in the same manner.

The functions $\phi = 1 - \phi'$ and $\phi = \phi' \ominus p$ are non-expansive (easily shown), so

$$\begin{aligned}|\phi(q_q) - \phi(q_r)| &\leq |\phi'(q_q) - \phi'(q_r)| \\ &\leq \left|\bigvee_{\sigma\in\Sigma}\langle\sigma\rangle\varphi_\sigma(q_q) - \bigvee_{\sigma\in\Sigma}\langle\sigma\rangle\varphi_\sigma(q_r)\right|\end{aligned}$$

by the induction hypothesis on $\phi'$. □

The following two definitions will be used for the proof of the main result. First, the depth of a formula $\phi \in \mathcal{L}$ is defined (in a manner similar to that of (Desharnais et al., 2002)). Then, the formula $\phi^n_{q_q,q_r}$ is introduced.

**Definition 4.7.** *The depth of a formula of logic $\mathcal{L}$ is defined as:*

$$\begin{aligned} depth(\mathbf{1}) &= 0, \\ depth(\langle\sigma\rangle\phi) &= depth(\phi) + 1, \\ depth(\bigvee_{\sigma\in\Theta}\langle\sigma\rangle\psi_\sigma(q)) &= max\{depth(\psi_\sigma)|\sigma \in \Theta\} + 1, \\ depth(1 - \phi) &= depth(\phi), \\ depth(\phi \ominus p) &= depth(\phi). \end{aligned}$$

**Definition 4.8.** *Let $q_q, q_r \in Q$. The notation adopted for (4.3) is used here. Then, formula $\phi^0_{q_q,q_r}$ is defined as*

$$\phi^0_{q_q,q_r} = \mathbf{1},$$

*and, for $n \in \mathbb{N}$, formula $\phi^{n+1}_{q_q,q_r}$ is defined as*

$$\phi^{n+1}_{q_q,q_r} = \bigvee_{\sigma \in \Sigma} \langle \sigma \rangle \psi^n_{\sigma,q_q,q_r}, \ \textit{where}$$

$$\psi^n_{\sigma,q_q,q_r} = \begin{cases} 1 - ((1 - \phi^n_{q_i,q_j}) \ominus (1 - \phi^n_{q_i,q_j}(q_i))) & \textit{if } \rho_{\sigma,i} \geq \rho'_{\sigma,j} \\ \phi^n_{q_i,q_j} \ominus \phi^n_{q_i,q_j}(q_j) & \textit{otherwise.} \end{cases}$$

The main result relating the two metrics is presented next. It states that $d_L$ and $d_{fp}$ are equal up to constant $e$.

**Theorem 4.2.** $d_L = ed_{fp}$

*Proof.* The proof consists of two parts. In the first part, it is proven that, for every $q_q, q_r$, there exists $\phi \in \mathcal{L}$ such that $\phi(q_q) - \phi(q_r) = ed_{fp}(q_q, q_r)$. Consequently, $d_L(q_q, q_r) \geq ed_{fp}(q_q, q_r)$. In the second part, inequality $d_L(q_q, q_r) \leq ed_{fp}(q_q, q_r)$ is proven.

First, let us prove that for every $q_q, q_r$, there exists $\phi \in \mathcal{L}$ such that $\phi(q_q) - \phi(q_r) = ed_{fp}(q_q, q_r)$. Given Definition 4.1, it is sufficient to prove that $\phi^n_{q_q,q_r}(q_q) - \phi^n_{q_q,q_r}(q_r) = ed^n_{fp}(q_q, q_r)$, for every $n \in \mathbb{N}$, where $\phi^n_{q_q,q_r}$ is given as in Definition 4.8. The proof is by induction. The base case is satisfied, since $\phi^0_{q_q,q_r}(q_q) = \phi^0_{q_q,q_r}(q_r) = 1$, and $d^0_{fp}(q_q, q_r) = 0$ according to Definition 4.1. Let assume that, for every $q_q, q_r \in Q$, $n \in \mathbb{N}$:

$$\phi^n_{q_q,q_r}(q_q) - \phi^n_{q_q,q_r}(q_r) = ed^n_{fp}(q_q, q_r).$$

Also, let $\rho_{q_q}$ and $\rho_{q_r}$ be the distributions on $\Sigma \times Q$ induced by the states $q_q$ and $q_r$, respectively. Also, for notational convenience, we will write $\rho_{\sigma,i}$ instead of $\rho_{q_q}(\sigma, q_i)$, and, similarly, $\rho'_{\sigma,j}$ instead of $\rho_{q_r}(\sigma, q_j)$ for any $i, j$ such that $0 \leq i, j \leq N - 1$. Then, for $\sigma \in \Sigma$, let $i(q_q, \sigma) = i$ such that $q_i = \delta(q_q, \sigma)$

if $\delta(q_q,\sigma)!$, and $i(q_q,\sigma)=0$, otherwise. Similarly, let $j(q_r,\sigma)=j$ such that $q_j=\delta(q_r,\sigma)$ if $\delta(q_r,\sigma)!$, and $j(q_r,\sigma)=0$, otherwise. For readability purposes, we will write $i$ instead of $i(q_q,\sigma)$, and $j$ instead of $j(q_r,\sigma)$. Then:

$$\begin{aligned}
&\phi^{n+1}_{q_q,q_r}(q_q)-\phi^{n+1}_{q_q,q_r}(q_r)\\
&=\left(\sum_{\sigma\in\{\sigma\in\Sigma|\rho_{\sigma,i}\geq\rho'_{\sigma,j}\}} e\rho_{\sigma,i}+\sum_{\sigma\in\{\sigma\in\Sigma|\rho_{\sigma,i}<\rho'_{\sigma,j}\}} e\rho_{\sigma,i}ed^n_{fp}(q_i,q_j)\right)\\
&-\sum_{\sigma\in\{\sigma\in\Sigma|\rho_{\sigma,i}\geq\rho'_{\sigma,j}\}} e\rho'_{\sigma,j}(1-ed^n_{fp}(q_i,q_j))\\
&\text{(by the definition of } \phi^{n+1}_{q_q,q_r} \text{ and the induction hypothesis)}\\
&=\sum_{\sigma\in\{\sigma\in\Sigma|\rho_{\sigma,i}\geq\rho'_{\sigma,j}\}}\left(e(\rho_{\sigma,i}-\rho'_{\sigma,j})+e^2\rho'_{\sigma,j}d^n_{fp}(q_i,q_j)\right)\\
&+\sum_{\sigma\in\{\sigma\in\Sigma|\rho_{\sigma,i}<\rho'_{\sigma,j}\}} e^2\rho_{\sigma,i}d^n_{fp}(q_i,q_j)\\
&=e\sum_{\sigma\in\{\sigma\in\Sigma|\rho_{\sigma,i}\geq\rho'_{\sigma,j}\}}\left(\rho_{\sigma,i}-\rho_{\sigma,j}+e\rho'_{\sigma,j}d^n_{fp}(q_i,q_j)\right)\\
&+e\sum_{\sigma\in\{\sigma\in\Sigma|\rho_{\sigma,i}<\rho'_{\sigma,j}\}} e\rho_{\sigma,i}d^n_{fp}(q_i,q_j)\\
&=ed^{n+1}_{fp}(q_q,q_r)\ \text{(follows from (4.1))}
\end{aligned}$$

Next, the induction on the depth of formula is used to prove that $d_L(q_q,q_r)\leq ed_{fp}(q_q,q_r)$ by proving that $d^n_L(q_q,q_r)\leq ed^n_{fp}(q_q,q_r)$ for any $n\in\mathbb{N}$, where

$$d^n_L(q_q,q_r)=\sup_{\phi\in\mathcal{L}}\{|\phi(q_q)-\phi(q_r)||depth(\phi)\leq n\}.$$

The base case is satisfied as $d^0_L(q_q,q_r)=d^0_{fp}(q_q,q_r)=0$. For $n\in\mathbb{N}$, assume:

$$d^n_L(q_q,q_r)\leq ed^n_{fp}(q_q,q_r).$$

Then, according to Lemma 4.4, and the definition of function *depth*:

$$\begin{aligned}
&d_L^{n+1}(q_q, q_r)\\
&= \sup_{\phi_\sigma^n \in \mathcal{L}} \left\{ \left| \bigvee_{\sigma\in\Sigma} \langle\sigma\rangle \phi_\sigma^n(q_q) - \bigvee_{\sigma\in\Sigma} \langle\sigma\rangle \phi_\sigma^n(q_r) \right| \right\}\\
&= e \cdot \sup_{\phi_\sigma^n \in \mathcal{L}} \{ \sum_{\sigma\in\Sigma} \rho_{\sigma,i}\phi_\sigma^n(q_i) - \sum_{\sigma\in\Sigma} \rho'_{\sigma,j}\phi_\sigma^n(q_j),\\
&\qquad\qquad \sum_{\sigma\in\Sigma} \rho'_{\sigma,j}\phi_\sigma^n(q_j) - \sum_{\sigma\in\Sigma} \rho_{\sigma,i}\phi_\sigma^n(q_i) \}\\
&= e \cdot \sup_{\phi_\sigma^n \in \mathcal{L}} \{ \sum_{\substack{\sigma\in\Sigma\\ 0\le i\le N-1}} \rho_{\sigma,i}\phi_\sigma^n(q_i) - \sum_{\substack{\sigma\in\Sigma\\ 0\le j\le N-1}} \rho'_{\sigma,j}\phi_\sigma^n(q_j),\\
&\qquad\qquad \sum_{\substack{\sigma\in\Sigma\\ 0\le j\le N-1}} \rho'_{\sigma,j}\phi_\sigma^n(q_j) - \sum_{\substack{\sigma\in\Sigma\\ 0\le i\le N-1}} \rho_{\sigma,i}\phi_\sigma^n(q_i) \}\\
&\text{(as } G \text{ is deterministic)}
\end{aligned}$$

where, for any $\sigma, \alpha \in \Sigma$, $|\phi_\sigma^n(q_i) - \phi_\alpha^n(q_j)| \le d_L^n(q_i, q_j) \le ed_{fp}^n(q_i, q_j)$ (by induction hypothesis). The function in (4.2) is a pseudometric (therefore, symmetry holds), and for $a_{\sigma,i} = \phi_\sigma^n(q_i)$, the constraints are satisfied, so $d_L^{n+1}(q_q, q_r) \le ed_{fp}^{n+1}(q_q, q_r)$. □

**Remark 4.3.** *The logic $\mathcal{L}$ can be easily extended such that $d_L = ed_{fp}$ still holds. Therefore, it is easy to make the logic more expressive while preserving the same characterization of our logic. As logic $\mathcal{L}$ is sufficient for the characterization of the metric, and for the sake of simplicity of formalization, the logic was not extended.*

## 4.4 From Logic to Traces

First, $L_p(G)(s)$ is modified to define the *discounted probability of a string* $s$ in $G$, denoted $P_d(G)(s)$.

**Definition 4.9.** *Let $P_d(G) : L(G) \to [0,1]$ be defined as:*

$$P_d(G)(\epsilon) = 1$$

$$P_d(G)(s\sigma) = \begin{cases} e \cdot P_d(G)(s) \cdot p(\delta(q_0, s), \sigma), & \textit{if } \delta(q_0, s)! \\ 0, & \textit{otherwise} \end{cases}$$

*where $s \in L(G)$, $\sigma \in \Sigma$. Then, $P_d(G)(s)$ is the discounted probability of a string $s$ in $G$.*

Informally, the discounted probability of a string is the probability of occurrence of a string discounted by factor $e$ for every event in the string, i.e. $P_d(G)(s) = e^{|s|} L_p(G)(s)$.

Let $G_1$ and $G_2$ be two probabilistic generators. An important result states that there is not a string whose discounted probabilities differ by more than the distance $d_L$ between the corresponding generators.

**Theorem 4.3.**

$$d_L(G_1, G_2) \geq \sup_{s \in \Sigma^*} \{|P_d(G_1)(s) - P_d(G_2)(s)|\} \tag{4.18}$$

*Proof.* Let $t$ be the string for which the supremum in (4.18) is reached. The formula corresponding to this distance is easily constructed. Assume that $t = \sigma_1 \sigma_2 \ldots \sigma_n$. Then, the formula is given as $\phi = \langle \sigma_1 \rangle \langle \sigma_2 \rangle \ldots \langle \sigma_n \rangle 1$. □

Further, it can be shown that distance in the metric $d_L$ between the two systems is also greater than the difference in discounted probabilities of a set of strings such that none of the strings is a substring of another. Let $\Gamma \subseteq \Sigma^*$, such that no string in $\Gamma$ is a prefix of another string in $\Gamma$. Then:

**Theorem 4.4.**

$$d_L(G_1, G_2) \geq \sup_{\Gamma \subseteq \Sigma^*} \left\{ \left| \sum_{s \in \Gamma} P_d(G_1)(s) - \sum_{s \in \Gamma} P_d(G_2)(s) \right| \right\}$$

*Proof.* Similar to Theorem 4.3, by using the disjunction formula. □

Similarly, the correspondence between the discounted probability of strings and formulae in $\mathcal{L}$ can be made for the remaining formulae of Definition 4.5. Therefore, the metric measures not only the difference in probabilities of strings in two languages (discounted for their length), but also the difference in discounted probabilities of a certain set of strings, or some more complicated properties of strings, e.g., whether the discounted probability of a string is greater than a prespecified value.

## 4.5 Choosing the Metric: Justification

Next, we give rationale for choosing the metric in the solution of the OPSCP.

- The metric intuitively matches our notion of the distance between PDES and accounts for all differences between corresponding transition probabilities, as opposed to e.g., that of (Giacalone et al., 1990) that, roughly speaking, considers only the maximum of the differences between the corresponding probabilities.

- As presented in this chapter, the metric has both logical and trace characterization. The logical characterization measures the distance between two systems by a $[0, 1]$-valued formula that distinguishes between the systems the most, while the trace characterization describes the similarity between the probabilistic traces of similar systems. More precisely, the trace characterization shows that the metric measures not only the difference in (appropriately discounted) occurrence probabilities of strings in two systems, but also differences in (appropriately discounted) occurrence probabilities of certain sets of strings as well as complicated properties of strings.

- The metric is suggested for a large class of systems. Therefore, it allows for an extension of our work to e.g., nondeterministic systems.

- The metric discounts the future. The concept of discount has been widely applied in game theory, economics and optimal control. From an engineering point of view, one cares more about an error in the near future than the one in the distant future (de Alfaro et al., 2003).

- There is a simple algorithm to compute distances in this metric for our generative, deterministic model (see Section 4.2).

## 4.6 Summary

This chapter focused solely on the metric that is going to be used in the next chapter to solve the optimal probabilistic supervisory control problem.

The metric is defined on the states of a probabilistic transition system as a fixed point of a function that is given as a linear programming problem. For the case of probabilistic generators, the linear programming problem is solved, and the function is given a closed-form solution. This simplification enables the efficient calculation of the distances in the metric using two different algorithms. The first algorithm reduces to finding the (unique) solution of the system of linear equations. If, e.g., Gaussian elimination is used, the worst-case complexity of the algorithm is $O(|Q|^6)$. The second algorithm approximates the distances with a prespecified accuracy and is iterative. Each iteration takes $O(|Q|^2|\Sigma|)$ time, while the number of iterations sufficient for reaching the accuracy of $\epsilon$ is $\lceil \log_e \epsilon \rceil$. This iterative algorithm will be modified in the solution of the optimal probabilistic supervisory control problem in the next chapter, and its proof of correctness will be partially reused.

Then, this chapter turned to alternative characterizations of the metric: the logical and trace characterizations. First, the metric is characterized through a real-valued logic: the distance in the metric between two systems is measured by a formula that distinguishes between the systems the most. Then, from this logical characterizations follows the trace characterization: systems are similar if the probabilities of their (appropriately discounted) traces, certain sets of traces, and certain properties of traces are similar. The goal of alternative characterizations is to deepen the understanding of what similarity between systems as measured by the metric means in terms of similarities of their logical properties, and similarities of their probabilistic traces.

Finally, the reasoning behind the use of the metric as a measure of behavioural similarity in this thesis is presented. In short, the metric is sensitive enough in the sense that it accounts for more than just maximal differences between corresponding events' probabilities or probabilistic traces. Better intuition of what the metric measures comes from the trace characterization. We observe that the metric, as originally defined, is applicable to more general systems. Further, for the class of systems used in this thesis, efficient algorithms for calculating/approximating the metric have been given.

# Chapter 5

# Optimal Probabilistic Supervisory Control of PDES

In this chapter, the algorithm that solves the OPSCP is presented. All the results in the chapter are applicable for future discount factor $e \in (0,1)$. The results of this chapter have been previously published in (Pantelic and Lawford, 2009) and (Pantelic and Lawford, 2010a).

First, the formulation of the OPSCP is repeated. Assume that the plant is given by the PDES $G_p = (Q_p, \Sigma, \delta_p, q_{p0}, p_p)$, and the requirements specification is given by the PDES $G_r = (Q_r, \Sigma, \delta_r, q_{r0}, p_r)$. (An example is displayed in Figure 5.1.) If there is no probabilistic supervisor $V_p$ such that $L_p(V_p/G_p) = L_p(G_r)$, an optimal solution is sought. The solution is optimal in the following sense. First, it is assumed that the nonprobabilistic language of the requirement is a safety requirement: no other strings are allowed in the plant. Then, it is required that maximal permissible deadlock-free behaviour (in the nonprobabilistic sense) is achieved. Further, in the probabilistic sense, the probabilistic behaviour of the controlled plant should be as close as possible to the requirements specification that is now normalized so that it is constrained to the supremal deadlock-free and controllable sublanguage of $L(G_r)$ with respect to $G_p$. The algorithm to be proposed uses this separation of probabilistic and nonprobabilistic aspects of optimality so that it deals with each aspect separately: the first part handles the "nonprobabilistic optimality", and the second part handles the "probabilistic optimality". This separation is also notable in the conditions (i) and (ii) of the Theorem 3.1 for the existence of a probabilistic supervisor. The first part of both conditions corresponds to controllability as used in classical supervisory control theory (namely, the condition $Pos(q) \cap \Sigma_u = Pos(r) \cap \Sigma_u$ of (i), and $Pos(r) \cap \Sigma_c \subseteq Pos(q) \cap \Sigma_c$ of (ii)). The remaining equations and inequalities correspond to the conditions for probability matching.

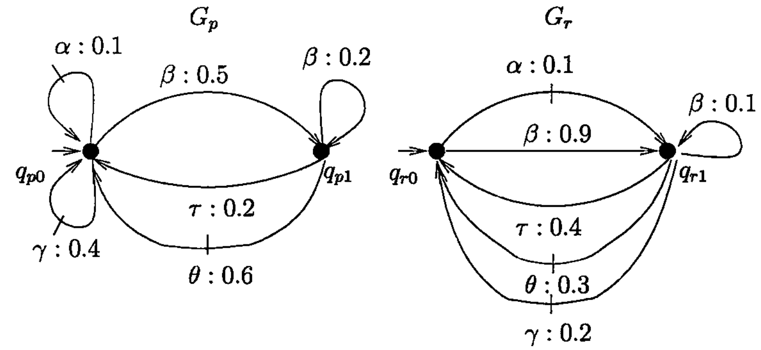

Figure 5.1: An example: Plant $G_p$, and requirements specification $G_r$

Sections 5.1 and 5.2 describe the algorithm that solves the OPSCP problem. The algorithm is summarized and its complexity is analyzed in Section 5.3. Finally, the algorithm is illustrated by an example in Section 5.4.

## 5.1 Algorithm: Part I

Before we start looking for the closest approximation in the sense of probability matching, we resort to the classical supervisory theory of supremal controllable sublanguages. First, the classical controllability condition that corresponds to the first parts of conditions (i) and (ii) of Theorem 3.1 is checked while constructing $L(G_p) \cap L(G_r)$. Then, if the condition is not satisfied, the goal is to find $K$, the supremal deadlock-free and controllable sublanguage of $L(G_r)$ (with respect to $G_p$). The language $K$ is required to be deadlock-free as only nonterminating PDES are considered. Now, let PDES $G_1$ be the modified plant such that its underlying DES represents this language $K$, further equipped with distribution $p_p$ (appropriately normalized). Also, let $G_2$ represent the desired behaviour PDES, such that its underlying DES represents language $K$, equipped with the distribution $p_r$ appropriately normalized. Formally, let the reachable and deadlock-free DES $G_{1k} = (T, \Sigma, \zeta, t_0)$ represent language $K$. We define a PDES $G_1 = (T, \Sigma, \zeta, t_0, p_1)$, where the distribution $p_1 : T \times \Sigma \rightarrow [0, 1]$, for any $q \in T, \sigma \in \Sigma$, is defined as:

$$p_1(q, \sigma) = \frac{p_p(q_p, \sigma)}{\sum\limits_{\sigma \in \{\sigma \in \Sigma | \zeta(q,\sigma)!\}} p_p(q_p, \sigma)}$$

where $q_p = \delta_p(q_{p_0}, s)$ for any $s \in K$ such that $q = \zeta(t_0, s)$.

Similarly, we define a PDES $G_2 = (Q, \Sigma, \delta, q_0, p)$, where $G_2^{np} = (Q, \Sigma, \delta, q_0)$ is a DES isomorphic to $G_{1k}$ (identical up to renaming of states), and, without loss of generality, we assume $T \cap Q = \emptyset$. Obviously, the nonprobabilistic language generated by $G_2^{np}$ is $K$, too. Distribution $p : Q \times \Sigma \rightarrow [0, 1]$ is defined as $(q \in Q, \sigma \in \Sigma)$:

$$p(q, \sigma) = \frac{p_r(q_r, \sigma)}{\sum\limits_{\sigma \in \{\sigma \in \Sigma | \delta(q,\sigma)!\}} p_r(q_r, \sigma)}$$

where $q_r = \delta_r(q_{r_0}, s)$ for any $s \in K$ such that $q = \delta(q_0, s)$. Note that $p_1$ and $p$ are well-defined as no state minimization is performed on the automaton representing language $K$.

## 5.2 Algorithm: Part II

Now, the probability matching equations and inequalities from Theorem 3.1 are checked. If they are not satisfied (i.e., there is no probabilistic supervisor $V_p$ such that $L_p(V_p/G_1) = L_p(G_2)$), the goal is to find $G_2' = (Q', \Sigma, \delta', q_0', p')$ such that there exists a probabilistic supervisor $V_p$ so that $L_p(V_p/G_1) = L_p(G_2')$ holds, and $G_2'$ is closest to $G_2$ in our chosen metric. Without loss of generality, it is assumed that $Q \cap Q' = \emptyset$. Also, without loss of generality, it is assumed that the nonprobabilistic automata underlying $G_2$ and $G_2'$ are isomorphic (with labeling of events being preserved). Therefore, the nonprobabilistic automata underlying $G_2$ and $G_2'$ are identical up to renaming of states. This assumption is not restrictive as there cannot be any string in the desired system that does not belong to $K$, and, therefore, since $K = L(G_2)$, there cannot be any string in the desired system that does not belong to $L(G_2)$. This comes from the fact that $L(G_2)$ is the supremal deadlock-free and controllable sublanguage: if any string not in $L(G_2)$ would be allowed in the controlled plant, either the safety or nontermination requirement would not be met. As our metric is defined on the states of a single system, in order to define distances between the states of different systems, namely $G_2$ and $G_2'$, the union PDES $G_u = (Q \cup Q', \Sigma, \delta_u, q_0, p_u)$ is considered, where for $\sigma \in \Sigma$ and $q \in Q \cup Q'$:

$$\delta_u(q,\sigma) = \begin{cases} \delta(q,\sigma), & \text{if } q \in Q \\ \delta'(q,\sigma), & \text{otherwise,} \end{cases}$$

$$p_u(q,\sigma) = \begin{cases} p(q,\sigma), & \text{if } q \in Q \\ p'(q,\sigma), & \text{otherwise.} \end{cases}$$

Note that the union $G_u$ is just an artificial construct introduced so that it would be possible to overcome the obstacle of defining the distance between the states that belong to different PDES since the metric is defined on states of a single PDES. Generator $G_u$ is merely a PDES consisting of the union of $G_2$ and $G_2'$ with the initial state arbitrarily chosen (between $q_0$ and $q_0'$) to be $q_0$.

Then, $\mathcal{M}$ is the set of 1-bounded pseudometrics on the states of this union system with the same ordering as in (4.1).

First, note that, considering the isomorphism between $G_2^{np}$ and $G_2'^{np}$, only the distances between (probability measures on) states $q \in Q$ of $G_2$ and $q' = f(q) \in Q'$ of $G_2'$ are of interest, where $f$ is the isomorphism between $G_2^{np}$ and $G_2'^{np}$. Also, let $h$ be an isomorphism between $G_2^{np}$ and $G_1^{np}$. It is assumed that PDES $G_2$ is in state $q$ after the occurrence of string $s \in L(G_2)$ $(\delta(q_0, s) = q)$. Then, the closest approximation $G_2'$ is in state $q'$, respectively, where $q' = f(q)$. Let $\rho_q$ be the probability distribution induced by the state $q \in Q$ of PDES $G_2$ and let $\rho'_{q'}$ be the probability distribution induced by the state $q' \in Q'$ of PDES $G_2'$.

Next, a class $\mathcal{A}$ of partial functions $a : Q \times Q' \rightarrow [0,1]$ is defined, such that $\forall q \in Q, q' = f(q) \in Q'$ $a(q,q') = d(q,q')$, where $d \in \mathcal{M}$. Therefore, the class $\mathcal{A}$ is the class of all 1-bounded pseudometrics with domain reduced to $Q \times Q'$, and only distances between $q \in Q$ and $q' = f(q) \in Q'$ defined since the algorithm is independent of the distance between the other states. Next, we define a family $\Delta$ as a set of probability distributions on $\Sigma \times Q'$. Now, for each $\rho' \in Q' \to \Delta$, we define function $\mathcal{D}^{\rho'} : \mathcal{A} \to \mathcal{A}$ as ($q \in Q, q' = f(q) \in Q', d \in \mathcal{A}$):

$$\mathcal{D}^{\rho'}(d)(q,q') = d(\rho_q, \rho'_{q'}) \text{ and } \rho'(q') = \rho'_{q'},$$

where, as before, $d$ is lifted to the metric on distributions, and $d(\rho_q, \rho'_{q'})$ is defined as in (4.6). Also, the reversed ordering on $\mathcal{A}$ is introduced to match the one in (4.1):

$$d_1 \preceq' d_2 \text{ if } \forall q \in Q\ \forall q' \in Q'\ (q' = f(q) \implies d_1(q,q') \geq d_2(q,q')).$$

The fact that $(\mathcal{A}, \preceq')$ is a complete lattice follows from the fact that $(\mathcal{M}, \preceq)$ is a complete lattice. Further, for each $\rho' \in Q' \to \Delta$, we define function $d_{fp}^{\rho'}$ as the greatest fixed point of function $\mathcal{D}^{\rho'}$. The problem of finding the optimal approximation reduces now to finding $\rho'_m \in Q' \to \Delta$ such that $d_{fp}^{\rho'_m}(q_0, q'_0) = \min_{\rho'}\{d_{fp}^{\rho'}(q_0, q'_0) | \rho' \in Q' \to \Delta\}$ and the conditions for the existence of a probabilistic supervisor of Theorem 3.1 are satisfied. It follows straight from the definitions of $\mathcal{D}^{\rho'}$ and $d_{fp}^{\rho'}$ that, for any $\rho' \in Q' \to \Delta$, $q \in Q, q' = f(q) \in Q'$, the distances $d_{fp}^{\rho'}(q,q')$ are distances in our pseudometric.

We assume that $T = \{t_0, t_1, \ldots t_{N-1}\}$, $Q = \{q_0, q_1, \ldots q_{N-1}\}$, and $Q' = \{q'_0, q'_1, \ldots q'_{N-1}\}$, where $q'_i = f(q_i)$, $t_i = h(q_i)$, $i = 0, \ldots, N-1$. Note that, for probability distributions, a different notation will be used than the one used in the previous section. Let $d \in \mathcal{A}$, $0 \leq i \leq N-1$, $\Psi(q_i) = Pos(q_i)$, $\Psi_u(q_i) = Pos(q_i) \cap \Sigma_u$, and $\Psi_c(q_i) = Pos(q_i) \cap \Sigma_c$. Also, we will write $j$ for $j(i,\sigma)$, then $\rho_{q_i,\sigma}$ instead of $\rho_{q_i}(\sigma, q_k)$, and $\rho'_{q'_i,\sigma}$ instead of $\rho'_{q'_i}(\sigma, q'_k)$, $k = 0, 1, \ldots N-1$.

Now, the function $\mathcal{P} : \mathcal{A} \to \mathcal{A}$ is defined as:

$$\mathcal{P}(d)(q_i, q_i') = \underset{\rho'_{q_i,\sigma}}{\text{Minimize}} \sum_{\sigma \in \Psi(q_i)} max(\rho_{q_i,\sigma} - \rho'_{q_i',\sigma} + c_j \rho'_{q_i',\sigma}, c_j \rho_{q_i,\sigma}), \tag{5.1}$$

where $c_j = e \cdot d(q_j, q_j')$ s.t. $q_j = \delta(q_i, \sigma)$

subject to

$$\frac{p_1(t_i,\sigma)}{\sum\limits_{\alpha \in \Psi_u(q_i)} p_1(t_i,\alpha)} = \frac{\rho'_{q_i',\sigma}}{\sum\limits_{\alpha \in \Psi_u(q_i)} \rho'_{q_i',\alpha}}, \qquad \sigma \in \Psi_u(q_i), \tag{5.2}$$

$$\frac{\sum_{\alpha \in \Psi_u(q_i)} p_1(t_i,\alpha)}{p_1(t_i,\sigma)} \rho'_{q_i',\sigma} + \sum_{\alpha \in \Psi_c(q_i)} \rho'_{q_i,\alpha} \leq 1, \quad \sigma \in \Psi_c(q_i), \tag{5.3}$$

$$\sum_{\alpha \in \Psi(q_i)} \rho'_{q_i',\alpha} = 1, \tag{5.4}$$

$$\rho'_{q_i',\sigma} \geq 0, \qquad \sigma \in \Psi(q_i). \tag{5.5}$$

The constraints (5.2) and (5.3) represent the conditions for the existence of probabilistic supervisor given by Theorem 3.1. The function $\mathcal{P}$ is well-defined since, if $\rho'_{q_i',\sigma} = p_1(t_i,\sigma)$ for all $\sigma \in \Sigma$, the constraints (5.2), (5.3), (5.4), (5.5) are satisfied. Therefore, the optimization problem has a feasible origin. Since $\mathcal{A}$ is a complete lattice, and the function $\mathcal{P}$ can be easily shown to be monotone, it has a greatest fixed point. Next, a useful lemma is stated.

**Lemma 5.1.** *Let $(\mathcal{L}, \preceq)$ be a complete lattice, and let $f, g : \mathcal{L} \to \mathcal{L}$ be two monotone functions such that $\forall x \in \mathcal{L} : g(x) \preceq f(x)$. Let gfp($f$) and gfp($g$) denote the greatest fixed point of functions $f$ and $g$, respectively. Then, gfp($g$) $\preceq$ gfp($f$).*

*Proof.* According to Knaster-Tarski theorem, the functions $f$ and $g$ have the greatest fixed points *gfp*($f$) and *gfp*($g$), respectively, where *gfp*($f$) $=$ *sup*($\{x | x \preceq f(x)\}$), and *gfp*($g$) $=$ *sup*($\{x | x \preceq g(x)\}$). Since $\forall x : g(x) \preceq f(x)$, then $\{x | x \preceq g(x)\} \subseteq \{x | x \preceq f(x)\}$; hence *gfp*($g$) $\preceq$ *gfp*($f$). □

Obviously, because of the definition of function $\mathcal{P}$, for any function $\mathcal{D}^{\rho'}$, where $\rho' \in Q' \to \Delta$, it holds that $\forall d \in \mathcal{A}\ \mathcal{D}^{\rho'}(d) \preceq' \mathcal{P}(d)$. Using Lemma 5.1, we conclude that the greatest fixed point of $\mathcal{P}$ is greater than or equal to any $d_{fp}^{\rho'}$, $\rho' \in Q \to \Delta$. This greatest fixed point corresponds to the minimal distance between $q_0$ and $q_0'$ because of the reversed ordering on $\mathcal{A}$. Therefore, the greatest fixed point of function $\mathcal{P}$ corresponds to the distances in our pseudometric where the distance between $q_0$ and $q_0'$ is minimized under the conditions of Theorem 3.1 for the existence of a probabilistic supervisor. Consequently, the values of decision variables $\rho_{q'}'$ for $q' \in Q'$ when the greatest fixed point of $\mathcal{P}$ is reached correspond to the statewise probability distributions of the optimal approximation.

We suggest an iterative algorithm to calculate the minimum achievable distance (i.e. the only fixed point of the function $\mathcal{P}$) up to a desired accuracy and provide the probability distribution of the system's achievable behaviour when this distance is reached. The proof pattern used for the algorithm from Section 4.2.2 is followed. However, as mentioned before, the only relevant distances are the ones between $q \in Q$ and $q' = f(q) \in Q'$.

Let $d \in \mathcal{A}$. Again, we assume that $Q = \{q_0, q_1, \dots q_{N-1}\}$, and $Q' = \{q_0', q_1', \dots q_{N-1}'\}$, where $q_i' = f(q_i)$, $i = 0, \dots, N-1$. Further, let us define function $\hat{\mathcal{V}} : \mathcal{M} \to [0,1]^N$ as:

$$\hat{\mathcal{V}}(d) = (d(q_0, q_0'), d(q_1, q_1'), \dots, d(q_{N-1}, q_{N-1}'))^T.$$

Therefore, $\hat{\mathcal{V}}(d) = (\hat{\mathcal{V}}_1(d), \dots, \hat{\mathcal{V}}_N(d))^T$, where, for $k = 1, \dots, N$:

$$\hat{\mathcal{V}}_k(d) = d(q_{k-1}, q_{k-1}'). \tag{5.6}$$

The function $\mathcal{P}$ is redefined in a natural way as $\mathcal{P}(\hat{\mathcal{V}}(d)) = (\mathcal{P}_1(\hat{\mathcal{V}}(d)), \dots, \mathcal{P}_N(\hat{\mathcal{V}}(d)))^T$, where for any $k \in \{1, \dots, N\}$:

$$\mathcal{P}_k(\hat{\mathcal{V}}(d)) = \mathcal{P}(d)(q_{k-1}, q_{k-1}'),$$

where $q_{k-1}' = f(q_{k-1})$. Also, let $P_0 = \{\hat{\mathcal{V}}(d) | d \in \mathcal{A}\}$.

**Theorem 5.1.** *Function $\mathcal{P}$ is P-contractive on $P_0$.*

*Proof.* Let $d', d'' \in \mathcal{A}$, and $\hat{\mathfrak{d}}' = \hat{\mathcal{V}}(d')$, and $\hat{\mathfrak{d}}'' = \hat{\mathcal{V}}(d'')$. Next, for $q \in Q$, $q' = f(q) \in Q'$, we define set $\Phi(q)$ to be the set of all distributions $\rho_{q'}'$ that satisfy conditions given by (5.2), (5.3), (5.4), and (5.5). Let $k \in \{1, \dots, N\}$. Then,

$\mathcal{P}_k(\hat{\mathfrak{d}}') = \mathcal{P}(d')(q_{k-1}, q'_{k-1})$, and $\mathcal{P}_k(\hat{\mathfrak{d}}'') = \mathcal{P}(d'')(q_{k-1}, q'_{k-1})$. Assume that the minumum of the objective function in (5.1) in function $\mathcal{P}(d')(q_{k-1}, q'_{k-1})$ is reached for $\rho'_{q'} = \mu$ for $\mu \in \Phi(q)$. Further, assume that the minumum of the objective function in (5.1) in function $\mathcal{P}(d'')(q_{k-1}, q'_{k-1})$ is reached for $\rho'_{q'} = \nu$ for $\nu \in \Phi(q)$. Let $\Psi = \Psi(q_{k-1})$. Also, let $j(k,\sigma) = j$ such that $q_j = \delta(q_{k-1}, \sigma)$, and $f(q_j) = q'_j$. Assume that $\mathcal{P}_k(\hat{\mathfrak{d}}') \geq \mathcal{P}_k(\hat{\mathfrak{d}}'')$. Then:

$$
\begin{aligned}
&\left|\mathcal{P}_k(\hat{\mathfrak{d}}') - \mathcal{P}_k(\hat{\mathfrak{d}}'')\right| \\
&= |\sum_{\sigma\in\Psi} max(\rho_{q_{k-1},\sigma} - \mu_{q'_{k-1},\sigma} + ed'(q_j,q'_j)\mu_{q'_{k-1},\sigma}, ed'(q_j,q'_j)\rho_{q_{k-1},\sigma}) \\
&- \sum_{\sigma\in\Psi} max(\rho_{q_{k-1},\sigma} - \nu_{q'_{k-1},\sigma} + ed''(q_j,q'_j)\nu_{q'_{k-1},\sigma}, ed''(q_j,q'_j)\rho_{q_{k-1},\sigma})| \\
&\leq |\sum_{\sigma\in\Psi} max(\rho_{q_{k-1},\sigma} - \nu_{q'_{k-1},\sigma} + ed'(q_j,q'_j)\nu_{q'_{k-1},\sigma}, ed'(q_j,q'_j)\rho_{q_{k-1},\sigma}) \\
&- \sum_{\sigma\in\Psi} max(\rho_{q_{k-1},\sigma} - \nu_{q'_{k-1},\sigma} + ed''(q_j,q'_j)\nu_{q'_{k-1},\sigma}, ed''(q_j,q'_j)\rho_{q_{k-1},\sigma})| \qquad (5.7) \\
&\text{(for } \rho'_{q'_{k-1},\sigma} = \mu_{q'_{k-1},\sigma} \text{ the minimum in } \mathcal{P}_k(\hat{\mathfrak{d}}') \text{ is reached)} \\
&\leq \sum_{\sigma\in\Psi} |max(\rho_{q_{k-1},\sigma} - \nu_{q'_{k-1},\sigma} + ed'(q_j,q'_j)\nu_{q'_{k-1},\sigma}, ed'(q_j,q'_j)\rho_{q_{k-1},\sigma}) \\
&- max(\rho_{q_{k-1},\sigma} - \nu_{q'_{k-1},\sigma} + ed''(q_j,q'_j)\nu_{q'_{k-1},\sigma}, ed''(q_j,q'_j)\rho_{q_{k-1},\sigma})| \qquad (5.8)
\end{aligned}
$$

(Similarly, when $\mathcal{P}_k(\hat{\mathfrak{d}}') \leq \mathcal{P}_k(\hat{\mathfrak{d}}'')$, we get (5.8), with $\mu_{q'_{k-1},\sigma}$ instead of $\nu_{q'_{k-1},\sigma}$.) Since, for any $d \in \mathcal{A}$, (4.7) holds, then every summand in (5.8) has one of the following forms:

$$
\left|\rho_{q_{k-1},\sigma} - \nu_{q'_{k-1},\sigma} + ed'(q_j,q'_j)\nu_{q'_{k-1},\sigma} - (\rho_{q_{k-1},\sigma} - \nu_{q'_{k-1},\sigma} + ed''(q_j,q'_j)\nu_{q'_{k-1},\sigma})\right| \text{ or}
$$
$$
\left|ed'(q_j,q'_j)\rho_{q_{k-1},\sigma} - ed''(q_j,q'_j)\rho_{q_{k-1},\sigma})\right|.
$$

Then:

$$
\begin{aligned}
&\left|\rho_{q_{k-1},\sigma} - \nu_{q'_{k-1},\sigma} + ed'(q_j,q'_j)\nu_{q'_{k-1},\sigma} - (\rho_{q_{k-1},\sigma} - \nu_{q'_{k-1},\sigma} + ed''(q_j,q'_j)\nu_{q'_{k-1},\sigma})\right| \\
&= e\nu_{q'_{k-1},\sigma}\left|d'(q_j,q'_j) - d''(q_j,q'_j)\right| \leq e\rho_{q_{k-1},\sigma}\left|d'(q_j,q'_j) - d''(q_j,q'_j)\right|
\end{aligned}
$$

and

$$
\left|ed'(q_j,q'_j)\rho_{q_{k-1},\sigma} - ed''(q_j,q'_j)\rho_{q_{k-1},\sigma})\right| = e\rho_{q_{k-1},\sigma}\left|d'(q_j,q'_j) - d''(q_j,q'_j)\right|
$$

Hence,

$$\left|\mathcal{P}_k(\hat{\mathfrak{d}}') - \mathcal{P}_k(\hat{\mathfrak{d}}'')\right| \leq \sum_{\sigma\in\Psi} e\rho_{q_{k-1},\sigma} \left|d'(q_j, q_j') - d''(q_j, q_j')\right|$$

Further, using the same reasoning as in the proof of Lemma 4.2, it is straightforward to show that $\mathcal{P}$ is P-contractive. □

**Lemma 5.2.** *Let $d', d'' \in \mathcal{A}$, and $\hat{\mathfrak{d}}' = \hat{\mathcal{V}}(d')$, and $\hat{\mathfrak{d}}'' = \hat{\mathcal{V}}(d'')$. For any $k \in \{1, \ldots, N\}$, there exists $m \in \{1, \ldots, N\}$ such that:*

$$\left|\mathcal{P}_k(\hat{\mathfrak{d}}') - \mathcal{P}_k(\hat{\mathfrak{d}}'')\right| \leq e\left|\hat{\mathfrak{d}}_m' - \hat{\mathfrak{d}}_m''\right|$$

*Proof.* First, use the proof of Theorem 5.1 up to (5.7), and, then, analogous to the proof of Lemma 4.3. □

**Theorem 5.2.** *For any $\hat{\mathfrak{d}}^0 \in P_0$, the sequence*

$$\hat{\mathfrak{d}}^{n+1} = \mathcal{P}(\hat{\mathfrak{d}}^n), \quad n = 0, 1, \ldots$$

*converges to the only fixed point of $\mathcal{P}$ in $P_0$, $\hat{\mathfrak{d}}^*$, and the error can be estimated componentwise ($k \in \{1, \ldots, N\}$) as:*

$$|\hat{\mathfrak{d}}_k^n - \hat{\mathfrak{d}}_k^*| \leq e^n, \quad n = 0, 1, \ldots$$

*Proof.* Analogous to the proof of Theorem 4.1 (with the use of Lemma 5.2 instead of Lemma 4.3). □

**Remark 5.1.** *Analogue to Remark 4.2, the error in the n-th iteration ($n = 1, 2 \ldots$) can also be estimated as ($k \in \{1, \ldots, N\}$)*

$$|\hat{\mathfrak{d}}_k^n - \hat{\mathfrak{d}}_k^*| \leq (1-e)^{-1} e^n \max_i\{|\hat{\mathfrak{d}}_i^1 - \hat{\mathfrak{d}}_i^0|\},$$

*or, as:*

$$|\hat{\mathfrak{d}}_k^n - \hat{\mathfrak{d}}_k^*| \leq (1-e)^{-1} e \max_i\{|\hat{\mathfrak{d}}_i^n - \hat{\mathfrak{d}}_i^{n-1}|\}.$$

The optimization problem of (5.1 – 5.5) is not a linear programming problem, but it is transformable into one by using additional variables $y_{q_i,\sigma}$, and by transforming (5.2) into (5.9):

$$\begin{aligned}
&\text{Minimize } \sum_{\sigma\in\Psi(q_i)} y_{q_i,\sigma}\\
&\text{subject to}\\
&\rho_{q_i,\sigma} - \rho'_{q'_i,\sigma} + c_j\rho'_{q'_i,\sigma} \le y_{q_i,\sigma}, \qquad \sigma\in\Psi(q_i),\\
&c_j\rho_{q_i,\sigma} \le y_{q_i,\sigma}, \qquad \sigma\in\Psi(q_i),\\
&\text{where } c_j = e\cdot d(q_i,q'_i) \text{ s.t. } q_j = \delta(q_i,\sigma),\\
&p_1(t_i,\sigma)\sum_{\alpha\in\Psi_u(q_i)} \rho'_{q_i,\alpha}, = \rho'_{q'_i,\sigma}\sum_{\alpha\in\Psi_u(q_i)} p_1(t_i,\alpha), \quad \sigma\in\Psi_u(q_i),\\
&\frac{\sum_{\alpha\in\Psi_u} p_1(t_i,\alpha)}{p_1(t_i,\sigma)}\rho'_{q'_i,\sigma} + \sum_{\alpha\in\Psi_c(q_i)} \rho'_{q'_i,\alpha} \le 1, \quad \sigma\in\Psi_c(q_i),\\
&\sum_{\alpha\in\Psi(q_i)} \rho'_{q'_i,\alpha} = 1,\\
&\rho'_{q'_i,\sigma} \ge 0, \qquad \sigma\in\Psi(q_i).
\end{aligned} \tag{5.9}$$

It might look as if (5.9) is weaker than (5.2) as it allows the possibility of $\rho'_{q'_i,\sigma} = 0$ for all $\sigma \in \Psi_u(q_i)$, which (5.2) forbids. However, this is not the case. Let $\rho'_{q'_i,\sigma} = 0$ for every $\sigma \in \Psi_u(q_i)$. From (5.3) it follows that:

$$\frac{\sum_{\alpha\in\Psi_u} p_1(t_i,\alpha)}{p_1(t_i,\sigma)}\rho'_{q'_i,\sigma} \le \sum_{\alpha\in\Psi_u(q_i)} \rho'_{q'_i,\alpha} = 0$$

which would mean that $\rho'_{q'_i,\sigma} = 0$ for every $\sigma \in \Psi_c(q_i)$ which contradicts the condition (5.4).

We now present the iterative algorithm for finding the fixed point of function $\mathcal{P}$.

Let $d^0(q_i, q'_i) = 0$, $i = 0, 1, ...N - 1$. The distance $d^n(q_i, q'_i)$ in the $n$-th iteration ($n > 0$) is given as:

$$\begin{aligned}
&\text{Minimize} \sum_{\sigma\in\Psi(q_i)} y_{q_i,\sigma} \qquad (5.10)\\
&\text{subject to}\\
&\rho_{q_i,\sigma} - \rho'_{q'_i,\sigma} + c_j\rho'_{q'_i,\sigma} \le y_{q_i,\sigma}, \qquad \sigma \in \Psi(q_i)\\
&c_j\rho_{q_i,\sigma} \le y_{q_i,\sigma}, \qquad \sigma \in \Psi(q_i)\\
&\text{where } c_j = e \cdot d^{n-1}(q_j, q'_j) \text{ s.t. } q_j = \delta(q_i,\sigma),\\
&p_1(t_i,\sigma)\sum_{\alpha\in\Psi_u(q_i)} \rho'_{q_i,\alpha} = \rho'_{q'_i,\sigma}\sum_{\alpha\in\Psi_u(q_i)} p_1(t_i,\alpha), \quad \sigma \in \Psi_u(q_i),\\
&\frac{\sum_{\alpha\in\Psi_u} p_1(t_i,\alpha)}{p_1(t_i,\sigma)}\rho'_{q'_i,\sigma} + \sum_{\alpha\in\Psi_c(q_i)} \rho'_{q'_i,\alpha} \le 1, \quad \sigma \in \Psi_c(q_i),\\
&\sum_{\alpha\in\Psi(q_i)} \rho'_{q'_i,\alpha} = 1,\\
&\rho'_{q'_i,\sigma} \ge 0, \qquad \sigma \in \Psi(q_i).
\end{aligned}$$

After the $n$-th iteration, the values of decision variables $\rho'_{q'_i,\sigma}$ that represent the unknown transition probabilities, are such that the distance between the (initial states of) systems $G_2$ and $G'_2$ is within $e^n$ of the minimal achievable distance between the two systems (in our pseudometric). Note that the aforementioned results hold for $e \in (0,1)$.

## 5.3 Summarizing the Algorithm

We now summarize the presented algorithm and give a brief complexity analysis.

1) First, the classical algorithm for finding the supremal controllable sublanguage is modified. The automaton $G_s$, the synchronous product of the nonprobabilistic automata underlying $G_p$ and $G_r$, is constructed. While constructing the product, the classical controllability conditions are checked for each state. If the conditions are satisfied for each state of the product, then $G = G_s$, and go to 2). If there is at least one state of the product for which the classical conditions do not hold, the rest of the algorithm for

finding the automaton representing the supremal controllable sublanguage is then applied. The algorithm can easily be modified to exclude deadlock states: these states are considered uncontrollable. Let (reachable and deadlock-free) DES $G = (Q, \Sigma, \delta, q_0)$ represent this supremal deadlock-free and controllable sublanguage.

2) Let $G_1$, $G_2$, and $G_2'$ be defined as previously in this chapter. Check the equalities and inequalities of Theorem 3.1 for each state: if they are satisfied, a supervisor exists, and $G_2$ is the optimal approximation. If not, then let $d^0(q_i, q_i') = 0$ for all $0 \leq i \leq N-1$. The distance $d^n(q_i, q_i')$ in the $n$-th iteration $(n > 0)$ is given by (5.10).

For each of the states of $G_2$ (typically, the number of states of $G_2$ is much smaller than $|Q_p| \cdot |Q_r|$), either the simplex method or an interior point method can be used to solve the linear programming problem (5.10). Depending on what method is used, the running time of the algorithm is either exponential (the simplex method) or polynomial (interior point methods) in the maximal number of events possible from a state of $G_2$. Even the worst-case exponential complexity of the simplex method is not problematic for two reasons: first, the method is very efficient in practice, and second, the number of possible events from a state is small in practical applications. Furthermore, the number of iterations sufficient to reach the accuracy of $\epsilon$ is $\lceil \log_e \epsilon \rceil$. As before, this term is obtained from the fact that the number of iterations $n$ for which an accuracy $\epsilon$ is achieved should be the smallest natural number for which $\epsilon \geq e^n$ is satisfied.

## 5.4 Example

For plant $G_p$ and $G_r$, depicted in Figure 5.1, there does not exist a probabilistic supervisor $V_p$ such that $G(V_p/G_p) = G_r$. Figure 5.2 shows the modified plant $G_1$ and modified specification $G_2$, defined as suggested in Section 5.1. Also, let $G_2'^{np}$ be defined as in Section 5.2. For PDES $G_2$, let $\rho_q$ be the probability distribution induced by the state $q \in Q$ and, for PDES $G_2'$, let $\rho_{q'}'$ be the probability distribution induced by the state $q' \in Q'$. As before, we will write $\rho_{q,\sigma}$ instead of $\rho_q(\sigma, q_i)$, and $\rho_{q',\sigma}'$ instead of $\rho_{q'}'(\sigma, q_j')$, $i, j = 0, 1, 2$.

At the $n$-th iteration, distances $d^n(q_0, q_0'), d^n(q_1, q_1')$, and $d^n(q_2, q_2')$ are calculated as follows:

$d^n(q_0, q_0') = \text{Minimize } (y_{q_0,\alpha} + y_{q_0,\beta})$

subject to

$$\rho_{q_0,\alpha} - \rho'_{q_0',\alpha} + e \cdot d^{n-1}(q_1, q_1')\rho'_{q_0',\alpha} \le y_{q_0,\alpha}, \quad e \cdot d^{n-1}(q_1, q_1')\rho_{q_0,\alpha} \le y_{q_0,\alpha},$$

$$\rho_{q_0,\beta} - \rho'_{q_0',\beta} + e \cdot d^{n-1}(q_2, q_2')\rho'_{q_0',\beta} \le y_{q_0,\beta}, \quad e \cdot d^{n-1}(q_2, q_2')\rho_{q_0,\beta} \le y_{q_0,\beta},$$

$$\frac{p(q_0,\beta)}{p(q_0,\alpha)}\rho'_{q_0,\alpha} + \rho'_{q_0,\alpha} \le 1, \quad \rho'_{q_0',\alpha} + \rho'_{q_0',\beta} = 1, \quad \rho'_{q_0',\alpha} \ge 0, \quad \rho'_{q_0',\beta} \ge 0.$$

$d^n(q_1, q_1') = \text{Minimize } (y_{q_1,\beta} + y_{q_1,\gamma})$

subject to

$$\rho_{q_1,\beta} - \rho'_{q_1',\beta} + e \cdot d^{n-1}(q_2, q_2')\rho'_{q_1',\beta} \le y_{q_1,\beta}, \quad e \cdot d^{n-1}(q_2, q_2')\rho_{q_1,\beta} \le y_{q_1,\beta},$$

$$\rho_{q_1,\gamma} - \rho'_{q_1',\gamma} + e \cdot d^{n-1}(q_0, q_0')\rho'_{q_1',\gamma} \le y_{q_1,\gamma}, \quad e \cdot d^{n-1}(q_0, q_0')\rho_{q_1,\gamma} \le y_{q_1,\gamma},$$

$$\frac{p(q_1,\beta)}{p(q_1,\gamma)}\rho'_{q_0,\gamma} + \rho'_{q_1,\gamma} \le 1, \quad \rho'_{q_1',\beta} + \rho'_{q_1',\gamma} = 1, \quad \rho'_{q_1',\beta} \ge 0, \quad \rho'_{q_1',\gamma} \ge 0.$$

$d^n(q_2, q_2') = \text{Minimize } (y_{q_2,\beta} + y_{q_2,\theta} + y_{q_2,\tau})$

subject to

$$\rho_{q_2,\beta} - \rho'_{q_2',\beta} + e \cdot d^{n-1}(q_2, q_2')\rho'_{q_2',\beta} \le y_{q_2,\beta}, \quad e \cdot d^{n-1}(q_2, q_2')\rho_{q_2,\beta} \le y_{q_2,\beta},$$

$$\rho_{q_2,\theta} - \rho'_{q_2',\theta} + e \cdot d^{n-1}(q_0, q_0')\rho'_{q_2',\theta} \le y_{q_2,\theta}, \quad e \cdot d^{n-1}(q_0, q_0')\rho_{q_2,\theta} \le y_{q_2,\theta},$$

$$\rho_{q_2,\tau} - \rho'_{q_2',\tau} + e \cdot d^{n-1}(q_0, q_0')\rho'_{q_2',\tau} \le y_{q_2,\tau}, \quad e \cdot d^{n-1}(q_0, q_0')\rho_{q_2,\tau} \le y_{q_2,\tau},$$

$$p(q_2,\tau)\left(\rho'_{q_2',\tau} + \rho'_{q_2',\beta}\right) = \rho'_{q_2',\tau}\left(p(q_2,\tau) + p(q_2,\beta)\right),$$

$$\frac{p(q_2,\beta) + p(q_2,\tau)}{p(q_2,\theta)}\rho'_{q_2,\theta} + \rho'_{q_2,\theta} \le 1,$$

$$\rho'_{q_2',\beta} + \rho'_{q_2',\theta} + \rho'_{q_2',\tau} = 1, \quad \rho'_{q_2',\beta} \ge 0, \quad \rho'_{q_2',\theta} \ge 0, \quad \rho'_{q_2',\tau} \ge 0.$$

Note that, for each $d^n(q_0, q_0')$, and $d^n(q_1, q_1')$, the equation that corresponds to the controllability condition for the sole uncontrollable event is missing, as it is trivially satisfied. Also, for $d^n(q_2, q_2')$, the controllability equation was generated only for one of two uncontrollable events, as the two equations can be derived from each other.

For the accuracy $\epsilon = 0.001$, and $e = 0.5$, 10 iterations of the algorithm are sufficient. It is found that the closest behaviour achievable with probabilistic control is as given in Figure 5.3. It took 0.3 seconds on a 2.6GHz dual core Opteron processor with 8GB of RAM running Red Hat Enterprise Linux Server 5.5. In order to find the corresponding probabilistic supervisor, the algorithm of Theorem 3.2 can be used. The supervisor is shown in Figure 5.3.

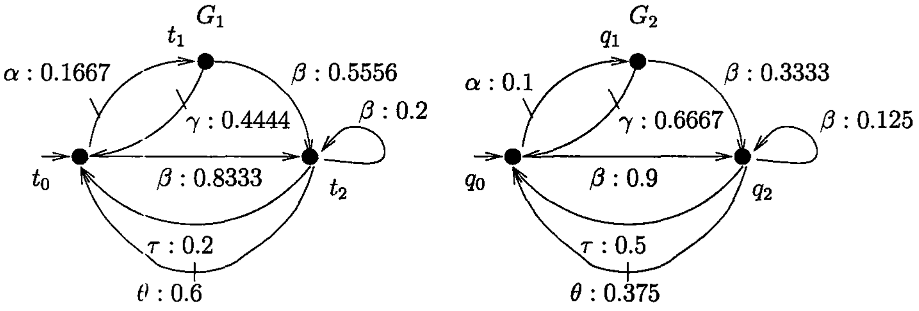

Figure 5.2: Generators $G_1$, and $G_2$

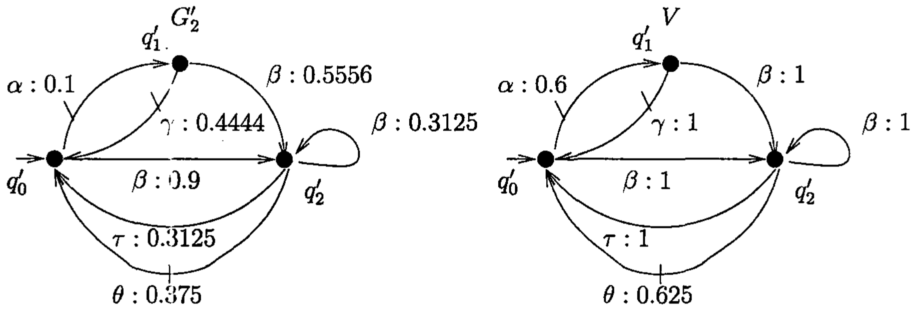

Figure 5.3: Optimal approximation $G'_2$ and probabilistic supervisor $V$ such that $V/G_1 = G'_2$

## 5.5 Summary

This chapter solved the OPSCP problem: an algorithm to approximate the probabilities of the closest approximation was given and its proof of correctness was presented.

The requirement, given as a probabilistic language, and equivalently, represented as a probabilistic generator, is considered a hard safety requirement. Therefore, the supremal deadlock-free and controllable sublanguage of the requirement with respect to the plant is generated as the maximal deadlock-free behaviour of the controlled plant. The requirements specification is constrained to the same sublanguage, with appropriately normalized probabilities. The generators representing achievable behaviour of the controlled plant and the modified requirements specification are isomorphic. The distance between the two is then minimized. The distance is measured by the metric presented in Chapter 4. The algorithm is iterative and works for $e \in (0, 1)$. In each iteration, for every two isomorphic states, a linear programming problem is solved: the distance between two states in the metric is minimized under the controllability conditions of Theorem 3.1. The algorithm iterates until a prespecified accuracy of the distance between the systems is reached. In the next chapter, the algorithm above will be modified for the case when the requirements specification is not revised (see Section 6.3). More precisely, a modification of the presented algorithm can be used to solve the control problem presented in Section 3.3 with criterion (2) changed so that the distance between the controlled plant and the original requirement is minimized.

# Chapter 6

# Probabilistic Model Fitting

In this chapter, the idea of approximating a given probabilistic generator by another probabilistic generator of a prespecified structure is explored, such that the distance between the original generator and the new one is minimized in metric $d_{fp}$. This approximation is called probabilistic model fitting.

Section 6.1 formulates the problem of probabilistic model fitting and presents its solution. Next, Section 6.2 presents the applications of model fitting. Then, in Section 6.3, some of the ideas used in the solution of the model fitting problem are also applied in the solution of the modified OPSCP problem.

## 6.1 Probabilistic Model Fitting: Problem and Solution

First, the probabilistic model fitting problem is introduced. Note that no minimization is done in the construction of the synchronous product of (non-probabilistic) generators as defined by Definition 2.1 in Section 2.2.

*The Probabilistic Model Fitting Problem*: Let $G_1 = (Q_1, \Sigma, \delta_1, q_{01}, p_1)$ be a probabilistic generator. Given a nonprobabilistic generator $G_2^{np} = (Q_2, \Sigma, \delta_2, r_0)$ such that $G_1^{np} \parallel G_2^{np}$ is isomorphic to $G_2^{np}$, find the statewise event probability distribution $p_2$ such that probabilistic generator $G_2 = (Q_2, \Sigma, \delta_2, r_0, p_2)$ is as close as possible to $G_1$ in metric $d_{fp}$.

The idea of solving the problem is as follows. The generator $G_1$ is to be modified to make $G_2^{np}$ isomorphic (identical up to renaming of states) to a subautomaton of modified $G_1^{np}$, while the probabilistic language of $G_1$ is preserved. Then, the distance between $G_1$ and $G_2$ is minimized by minimizing the distance between the modified $G_1$, and $G_2$. This is allowed as the two distances are the same, since $G_1$ and its modified version are probabilistic bisimilar:

**Theorem 6.1.** *Let $G_1$ and $G_2$ be two probabilistic generators. Then, if $L_p(G_1) = L_p(G_2)$, then $d_{fp}(G_1, G_2) = 0$.*

*Proof.* Since $L_p(G_1) = L_p(G_2)$, $G_1$ and $G_2$ are probabilistic trace equivalent in the sense of (Jou and Smolka, 1990). As $G_1$ and $G_2$ are deterministic, probabilistic trace equivalence implies probabilistic bisimulation equivalence. Therefore, $d_{fp}(G_1, G_2) = 0$. □

Next, as previously stated, we seek to represent $L_p(G_1)$ with an automaton $G_{1a}$ such that $G_2^{np}$ is isomorphic to a subautomaton of $G_{1a}^{np}$. Figure 6.1 illustrates an example. The part of $G_{1a}$ drawn by a solid line corresponds to the subautomaton of $G_{1a}^{np}$ isomorphic to $G_2^{np}$. In general, the automaton $G_{1a}$ will represent a non-minimal realization of $L_p(G_1)$ (in the sense that it might have more states than $G_1$, but $L_p(G_1) = L_p(G_{1a})$). Generator $G_{1a}$ can be constructed in the following manner.

1. Self-loop each state of $G_2^{np}$ with events not possible from that state. Formally, $G_{2a}^{np} = (Q_2, \Sigma, \delta_{2a}, r_0)$, where, for $q \in Q_2$, $\sigma \in \Sigma$:

$$\delta_{2a}(q, \sigma) = \begin{cases} \delta_2(q, \sigma), & \text{if } \delta_2(q, \sigma)! \\ q, & \text{otherwise.} \end{cases}$$

2. Next, let $G_{1a}^{np} = (Q_{1a}, \Sigma, \delta_{1a}, q_0) = G_1^{np} \parallel G_{2a}^{np}$.

3. The probabilistic version of $G_{1a}^{np}$ is $G_{1a} = (Q_{1a}, \Sigma, \delta_{1a}, q_0, p_{1a})$, such that, for all $q \in Q_{1a}$, $\sigma \in \Sigma$:

$$p_{1a}(q, \sigma) = p_1(r, \sigma),$$

where $r = \delta_1(q_{01}, s)$ for any $s \in L(G_{1a})$ such that $q = \delta_{1a}(q_0, s)$.

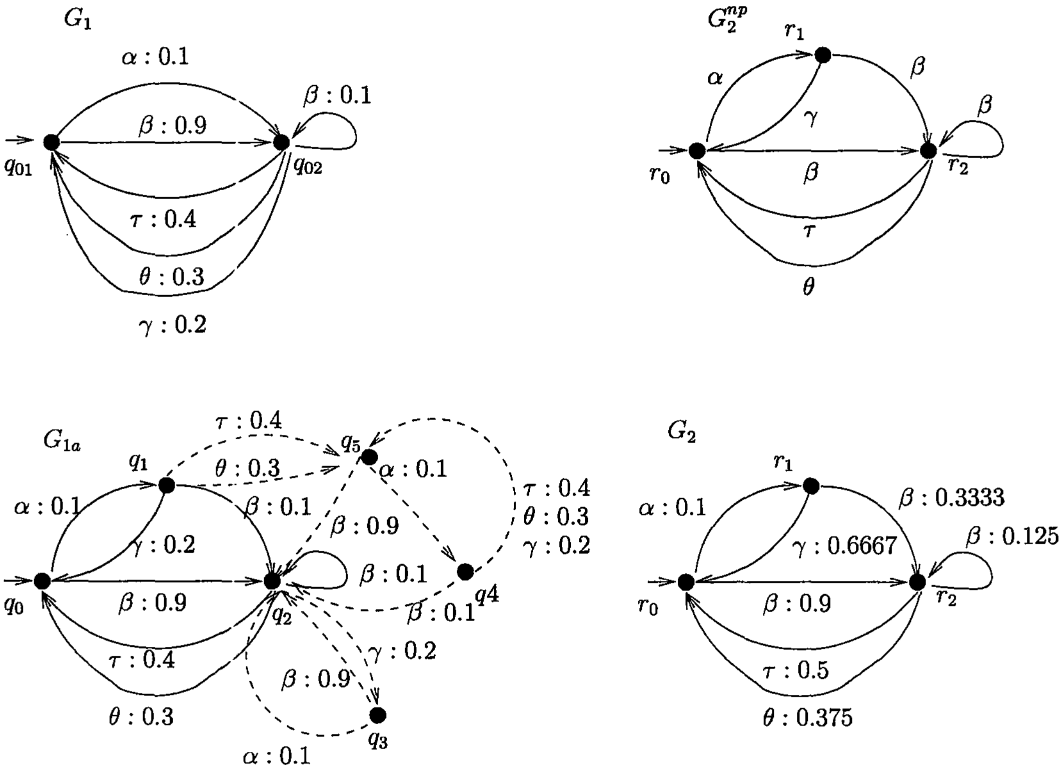

Figure 6.1: Model fitting: an example

**Lemma 6.1.** $L_p(G_1) = L_p(G_{1a})$.

*Proof.* Follows from the construction of $G_{1a}$. □

Now, let $f : Q_2 \rightarrow Q_{1a}$ be an embedding (a monomorphism) of $G_2^{np}$ into $G_{1a}^{np}$, i.e.:

1. $f(r_0) = q_0$,

2. $\forall q \in Q_2 \forall \sigma \in Pos(q) \;\; (f(\delta_2(q,\sigma)) = \delta_{1a}(f(q),\sigma))$.

The function $f$ always exists and is unique. This fact follows from the construction of $G_{1a}$ and the determinism of generators.

Without loss of generality, it is assumed that, $Q_{1a} = \{q_0, \ldots, q_{M-1}\}$, $Q_2 = \{r_0, \ldots, r_{N-1}\}$, and $M \geq N > 0$, $d \in \mathcal{M}$, $q \in Q_2$, where $\mathcal{M}$ is the set of 1-bounded pseudometrics on the states of the system that represents the union of $G_{1a}$ and $G_2$ (see Remark 4.1) with the same ordering as in (4.1). Next, $i(f(q),\sigma) = i$ such that $q_i = \delta_{1a}(f(q),\sigma)$ if $\delta_{1a}(f(q),\sigma)!$, and $i(f(q),\sigma) = 0$, otherwise. Let $j(q,\sigma) = j$ such that $r_j = \delta_2(q,\sigma)$ if $\delta_2(q,\sigma)!$, and $j(q,\sigma) = 0$, otherwise. For readability purposes, we will write $i$ instead of $i(f(q),\sigma)$, and $j$ instead of $j(q,\sigma)$. The distance between $G_{1a}$ and $G_2$ is $d_{fp}(q_0, r_0)$. Also, $f(r_0) = q_0$, and

$$\begin{aligned}
&\mathcal{D}(d)(f(q),q) \\
&= \sum_{\sigma\in\Sigma} max(\rho_{\sigma,i} - \rho'_{\sigma,j} + e\rho'_{\sigma,j} d(q_i, r_j), e\rho_{\sigma,i} d(q_i, r_j)) \\
&= \sum_{\sigma\in Pos(f(q))\setminus Pos(q)} \rho_{\sigma,i} \\
&+ \sum_{\sigma\in Pos(q)} max(\rho_{\sigma,i} - \rho'_{\sigma,j} + e\rho'_{\sigma,j} d(f(r_j), r_j), e\rho_{\sigma,i} d(f(q_j), q_j)) \quad (6.1) \\
&\text{(since } f(r_j) = q_i, \text{ by the definition of } f)
\end{aligned}$$

where $\rho_{f(q)}$ and $\rho_q$ are the distributions on $\Sigma \times Q$ induced by the states $f(q)$ and $q$, respectively, and $\rho_{\sigma,i}$ is written instead of $\rho_{f(q)}(\sigma, q_i)$, and, similarly, $\rho'_{\sigma,j}$ instead of $\rho_q(\sigma, r_j)$.

**Remark 6.1.** *Based on (6.1), it can be concluded that, for $q \in Q_2$, the distance between state $f(q) \in Q_{1a}$ and state $q$ depends only on distances between $f(t)$ and $t$, $t \in Q_2$. In Figure 6.1, the distance between $G_{1a}$ and $G_2$ depends only on distances between states of pairs ($q_0$, $r_0$), ($q_1$, $r_1$), and ($q_2$, $r_2$); states $q_3$, $q_4$, $q_5$ are irrelevant.*

Therefore, in order to calculate the distance between $G_{1a}$ and $G_2$, only the distances $d_{fp}(f(q), q)$, $q \in Q_2$, are of interest. Hence, the distance between $G_{1a}$ and $G_2$, for a fixed $p_2$, can be found by at most $\omega$ iterations given in Definition 4.1, where the domain of $d_{fp}^n$ is restricted to $Q_{1a} \times Q_2$ and only distances between $f(q) \in Q_{1a}$ and $q \in Q_2$ are defined.

This reasoning leads to the solution of the probabilistic model fitting problem as presented next.

**Theorem 6.2.** *Let $G_1 = (Q_1, \Sigma, \delta_1, q_{01}, p_1)$ be a probabilistic generator. For given $G_2^{np} = (Q_2, \Sigma, \delta_2, r_0)$ (such that $G_1^{np} \parallel G_2^{np}$ is isomorphic to $G_2^{np}$), the statewise event probability distribution $p_2$ such that $G_2 = (Q_2, \Sigma, \delta_2, r_0, p_2)$ is as close as possible to $G_1$ in the metric $d_{fp}$ should satisfy, for all $r \in Q_2$, $\sigma \in Pos(r)$:*

$$p_2(r, \sigma) \geq p_1(q, \sigma) \tag{6.2}$$

*where $q = \delta_1(q_{01}, s)$ for any $s \in L(G_2)$ such that $r = \delta_2(r_0, s)$.*

*Proof.* Let $G_2 = (Q_2, \Sigma, \delta_2, r_0, p_2)$, where $p_2$ satisfies (6.2). Also, let $G_2' = (Q_2, \Sigma, \delta_2, r_0, p_2')$ be a probabilistic generator with an arbitrary probability distribution $p_2'$. We use induction to show that $d_{fp}(G_{1a}, G_2') \geq d_{fp}(G_{1a}, G_2)$ by showing that $d_{fp}^n(G_{1a}, G_2') \geq d_{fp}^n(G_{1a}, G_2)$, $n \in \mathbb{N}$. For $q \in Q_2$, let $d_{fp}^n(f(q), q)$ be the distance between the states $f(q)$ of $G_{1a}$ and $q$ of $G_2$, and $d_{fp}'^n(f(q), q)$ be the distance between $f(q)$ of $G_{1a}$ and $q$ of $G_2'$. The base case is trivially satisfied. Next, assume that, for each $q \in Q_2$, $d_{fp}'^n(f(q), q) \geq d_{fp}^n(f(q), q)$. The functions $i$ and $j$ are defined as for (6.1), and, for $q \in Q_2$, $k(q, \sigma) = k$ such that $r_k = \delta_2(q, \sigma)$ if $\delta_2(q, \sigma)!$, and $k(r, \sigma) = 0$, otherwise. The shorthand notation $k$ will be used. For $q \in Q_2$, let $\rho_{f(q)}$, $\nu_q$ and $\nu_q'$ be the distributions induced by the states $f(q)$ of $G_{1a}$, $q$ of $G_2$ and $q$ of $G_2'$, respectively. Also, for $q \in Q_2$, let $\rho_{\sigma,i}$ be used instead of $\rho_{f(q)}(\sigma, q_i)$, and, similarly, $\nu_{\sigma,j}$ instead of $\nu_q(\sigma, r_j)$ and

$\nu'_{\sigma,k}$ instead of $\nu'_q(\sigma, r_k)$. Then:

$$\begin{aligned}
& d'^{n+1}_{fp}(f(q), q) \\
= & \sum_{\sigma\in\{\sigma\in\Sigma|\rho_{\sigma,i}\geq\nu'_{\sigma,k}\}} \left(\rho_{\sigma,i} - \nu'_{\sigma,k} + e\nu'_{\sigma,k} d'^{n}_{fp}(q_i, r_k)\right) + \sum_{\sigma\in\{\sigma\in\Sigma|\rho_{\sigma,q_i}<\nu'_{\sigma,k}\}} e\rho_{\sigma,i} d'^{n}_{fp}(q_i, r_k) \\
= & \sum_{\sigma\in Pos(f(q))\setminus Pos(q)} \rho_{\sigma,i} + \sum_{\sigma\in\{\sigma\in Pos(q)|\rho_{\sigma,i}\geq\nu'_{\sigma,k}\}} \left(\rho_{\sigma,i} - \nu'_{\sigma,k} + e\nu'_{\sigma,k} d'^{n}_{fp}(q_i, r_k)\right) \\
+ & \sum_{\sigma\in\{\sigma\in Pos(q)|\rho_{\sigma,i}<\nu'_{\sigma,k}\}} e\rho_{\sigma,i} d'^{n}_{fp}(q_i, r_k) \\
\geq & \sum_{\sigma\in Pos(f(q))\setminus Pos(q)} \rho_{\sigma,i} + \sum_{\sigma\in Pos(q)} e\rho_{\sigma,i} d'^{n}_{fp}(q_i, r_k) \\
\geq & \sum_{\sigma\in Pos(f(q))\setminus Pos(q)} \rho_{\sigma,i} + \sum_{\sigma\in Pos(q)} e\rho_{\sigma,i} d^{n}_{fp}(q_i, r_k) \\
& \text{(because of induction hypothesis, since } q_i = f(r_k)\text{)} \\
= & \sum_{\sigma\in Pos(f(q))\setminus Pos(q)} \rho_{\sigma,i} \\
+ & \sum_{\sigma\in Pos(q)} max(\rho_{\sigma,i} - \nu_{\sigma,j} + e\nu_{\sigma,j} d^{n}_{fp}(q_i, r_j), e\rho_{\sigma,i} d^{n}_{fp}(q_i, r_j)) \\
& \text{(since } \nu_{\sigma,j} \geq \rho_{\sigma,i} \text{ for every } \sigma \in Pos(q)\text{)} \\
& = d^{n+1}_{fp}(f(q), q)
\end{aligned}$$

□

Therefore, the new model is not unique: as long as the probabilities of the events possible in the new model do not decrease, the new model is as close as possible to the original one. For the example from Figure 6.1, one of the possible solutions is represented by the generator at the bottom right corner of the figure. In another possible solution, the probabilities of occurrence of $\beta$ and $\gamma$ at state $r_1$ would be 0.2 and 0.8, respectively. Therefore, the fitting can be performed by any redistribution of the probabilities of events that are not possible anymore over the possible ones. Hence, model fitting can accommodate some further requirements on $p_2$.

## 6.2 Some Applications of Model Fitting

Other than the obvious use of the presented fitting to simplify and reduce the state space of probabilistic systems, the fitting has much more significant control implications.

As mentioned before, it is possible to choose probabilities of events in the new system to a certain extent: as long as they are greater than or equal to the original ones. However, some of the further requirements on $p_2$ cannot be accommodated by Theorem 6.2 (e.g., an obvious one would be that the probability of an event still possible in the new system should be smaller than in the original system). If the restrictions are given on probabilities of events, statewise, a straightforward modification of the OPSCP algorithm of Chapter 5 for $e \in (0,1)$ would suffice. An example of such an additional requirement would be that the probability of a certain event from a state is less than a specified value, that is, in turn, smaller than the original one.

Further, in the solution of the OPSCP presented in Chapter 5, in order for the first criterion as presented in Section 3.3 to be satisfied, the supremal deadlock-free and controllable sublanguage of $L(G_r)$ with respect to $G_p$ is generated. Then, the distance between the controlled plant, and the probabilistic requirement now restricted to the sublanguage, with normalized probabilities, is minimized. Intuitively, after satisfying the nonprobabilistic requirement, and before the probabilistic part is handled, it makes sense for a designer to modify the original requirement so that its nonprobabilistic behaviour matches the one achievable Then, the probabilities are revised accordingly: probabilities of the events that are inadmissible because they do not satisfy the nonprobabilistic requirement, are redistributed over the admissible ones. The redistribution is such that the probability of an event in the new system is proportional to its original probability. Theorem 6.2 proves that this normalization is justified in a strict mathematical sense, as the new model that is normalized is as close as possible to the original one in the metric $d_{fp}$. However, a revised specification is going to be at a minimal distance from the original one, as long as the probabilities of its remaining events are greater than or equal to the original ones: a designer has a freedom to choose how to redistribute the probabilities over the events that are still possible.

Further, the transformation of $G_1$ into $G_{1a}$ presented here can be used in a modification of the OPSCP algorithm to solve the OPSCP (as presented in Section 3.3) with criterion (2) changed so that the controlled plant is "as close as possible" to the unmodified requirement. More precisely, the probabilistic language $L_p(G_r)$ of the requirements specification $G_r$ can be exactly represented by a probabilistic generator $G_2$ with nonprobabilistic automaton $G_2^{np}$ that has a subautomaton that is isomorphic to automaton $G_{1k}$ representing the supremal deadlock-free and controllable sublanguage of the controlled plant (see Section 5.1). Then, using the reasoning of Remark 6.1, the distance between the requirement and the controlled plant depends only on the distances between isomorphic states of the subautomaton of $G_2^{np}$ and $G_{1k}$. Hence, the OPSCP algorithm can be modified to minimize the distance between the two systems under the probabilistic controllability conditions of Theorem 3.1. This modification is shown in the next section.

## 6.3 Model Fitting and Closest Approximation: Problem Revisited

In Section 3.3, after satisfying the nonprobabilistic criterion (the first part of OPSCP of Section 3.3), a designer revises the requirements specification before satisfying the probabilistic criterion (the second part of OPSCP of Section 3.3). In this section, after satisfying the nonprobabilistic requirement, the distance between the achievable behaviour of the plant under control and the original requirements specification is minimized.

Let $e \in (0,1)$. As before, assume that the plant is given as PDES $G_p = (Q_p, \Sigma, \delta_p, q_{p0}, p_p)$, and the requirements specification is given as $G_r = (Q_r, \Sigma, \delta_r, q_{r0}, p_r)$. Formally, let the reachable and deadlock-free DES $G_{1k} = (T, \Sigma, \zeta, t_0)$ represent language $K$, the supremal deadlock-free and controllable sublanguage as defined in Section 5.1. Then, $G_1 = (T, \Sigma, \zeta, t_0, p_1)$ is defined in the same manner as in Section 5.1 – it is the probabilistic automaton corresponding to the restriction of the plant $G_p$ to $K$. Next, the requirement is not normalized as before, but, instead, the language $L_p(G_r)$ is represented using the generator $G_2 = (Q, \Sigma, \delta, q_0, p)$, such that a subautomaton of $G_2^{np}$ is isomorphic to $G_{1k}$; hence, isomorphic to $G_1^{np}$, too (see Figure 6.2 for an illustration). The part of $G_2$ drawn by a solid line corresponds to the subautomaton of $G_2^{np}$ isomorphic to $G_1^{np}$. As before, we should find $p'$ in $G_2' = (Q', \Sigma, \delta', q_0', p')$, such that $L_p(V_p/G_1) = L_p(G_2')$ holds, and $G_2'$ is closest to $G_2$ in our chosen metric. Also, $G_2'^{np}$ is such that $G_2'^{np}$ is isomorphic to $G_1^{np}$. This comes from the fact that there cannot be any string in the desired system that does not belong to $K$, and, therefore, there cannot be any string in the desired system that does not belong to $L(G_1)$ (as explained in Section 5.2). It follows from Lemma 6.1 that minimizing the distance between the $G_r$ and $G_2'$ is the same as minimizing the distance between $G_2$ and $G_2'$. Also, generator $G_2$ can be constructed in the same manner as $G_{1a}$ in Section 6.1, and, according to the results of Section 6.1, the construction is possible, as $G_r^{np} \parallel G_1^{np}$ is isomorphic to $G_1^{np}$. Now, given the definitions of $G_2$ and $G_2'$, there exists an embedding $f : Q' \to Q$ of $G_2'^{np}$ to $G_2^{np}$, i.e.:

1. $f(q_0') = q_0$,

2. $\forall q \in Q' \, \forall \sigma \in Pos(q) \, (f(\delta'(q, \sigma)) = \delta(f(q), \sigma))$.

We assume that $T = \{t_0, t_1, \dots t_{N-1}\}$, $Q = \{q_0, q_1, \dots q_{M-1}\}$, and $Q' = \{q_0', q_1', \dots q_{N-1}'\}$, where $q_i = f(q_i')$, $t_i = h(q_i')$, $i = 0, \dots, N-1$, where $M \geq N > 0$, and $h$ is the isomorphism between $G_2'^{np}$ and $G_1^{np}$. Let $\rho_{q_i}$ be the probability distribution induced by the state $q_i \in Q$ of PDES $G_2$ and let $\rho'_{q_i'}$ be the probability distribution induced by the state $q_i' \in Q'$ of PDES $G_2'$. Also, we will write $j$ for $j(i, \sigma)$, then $\rho_{q_i,\sigma}$ instead of $\rho_{q_i}(\sigma, q_k)$, and $\rho'_{q_i',\sigma}$ instead of $\rho'_{q_i'}(\sigma, q_k')$, $k = 0, 1, \dots, N-1$. Let $\mathcal{A}$ be the class of all 1-bounded pseudometrics on the states of the system that represents the union of $G_2$ and $G_2'$, with domain reduced to $Q \times Q'$, and only distances between $q = f(q') \in Q$ and $q' \in Q'$ defined (similar to Section 5.2). Let $d \in \mathcal{A}$, $0 \leq i \leq N-1$, $\Psi(q_i) = Pos(q_i)$, $\Psi(q_i') = Pos(q_i')$, $\Psi_u(q_i') = Pos(q_i') \cap \Sigma_u$, and $\Psi_c(q_i') = Pos(q_i') \cap \Sigma_c$.

Let $c_j = e \cdot d(q_j, q_j')$ such that $q_j = \delta(q_i, \sigma)$. Note that, since,

$$\mathcal{D}(d)(q_i, q_i') = \sum_{\sigma \in \Psi(q_i) \setminus \Psi(q_i')} \rho_{q_i,\sigma} + \sum_{\sigma \in \Psi(q_i')} max(\rho_{q_i,\sigma} - \rho'_{q_i',\sigma} + c_j \rho'_{q_i',\sigma}, c_j \rho_{q_i,\sigma}), \tag{6.3}$$

the distance between $G_2$ and $G_2'$ is going to depend only on distances between the isomorphic states. E.g., in Figure 6.2, the distance between $G_2$ and $G_2'$ depends only on distances between states of pairs ($q_0$, $q_0'$), ($q_1$, $q_1'$), and ($q_2$, $q_2'$); states $q_3$, $q_4$, $q_5$ are irrelevant.

**Theorem 6.3.** *Let $d^0(q_i, q_i') = 0$. The distance $d^n(q_i, q_i')$ in the n-th iteration ($n > 0$) is given as:*

$$Minimize \sum_{\sigma \in \Psi(q_i) \setminus \Psi(q_i')} \rho_{q_i,\sigma} + \sum_{\sigma \in \Psi(q_i')} y_{q_i,\sigma} \tag{6.4}$$

*subject to*

$$\rho_{q_i,\sigma} - \rho'_{q_i',\sigma} + c_j \rho'_{q_i',\sigma} \leq y_{q_i,\sigma}, \qquad \sigma \in \Psi(q_i'),$$

$$c_j \rho_{q_i,\sigma} \leq y_{q_i,\sigma}, \qquad \sigma \in \Psi(q_i'),$$

$$where\ c_j = e \cdot d^{n-1}(q_j, q_j')\ s.t.\ q_j = \delta(q_i, \sigma),$$

$$p_1(t_i, \sigma) \sum_{\alpha \in \Psi_u(q_i)} \rho'_{q_i,\alpha} = \rho'_{q_i',\sigma} \sum_{\alpha \in \Psi_u(q_i)} p_1(t_i, \alpha), \quad \sigma \in \Psi_u(q_i'), \tag{6.5}$$

$$\frac{\sum_{\alpha \in \Psi_u} p_1(t_i, \alpha)}{p_1(t_i, \sigma)} \rho'_{q_i',\sigma} + \sum_{\alpha \in \Psi_c(q_i')} \rho'_{q_i',\alpha} \leq 1, \quad \sigma \in \Psi_c(q_i'),$$

$$0 \leq \rho'_{q_i',\sigma} \leq 1, \qquad \sigma \in \Psi(q_i'),$$

$$\sum_{\alpha \in \Psi(q_i')} \rho'_{q_i',\alpha} = 1.$$

*After the n-th iteration, the values of decision variables $\rho'_{q_i',\sigma}$ that represent the unknown transition probabilities, are such that the distance between the (initial states of) systems $G_2$ and $G_2'$ is within $e^n$ of the minimal achievable distance between the two systems (in our pseudometric). Note that the aforementioned results hold for $e \in (0, 1)$.*

*Proof.* The proof follows from (6.3) and the proof of the algorithm of Section 5.2. □

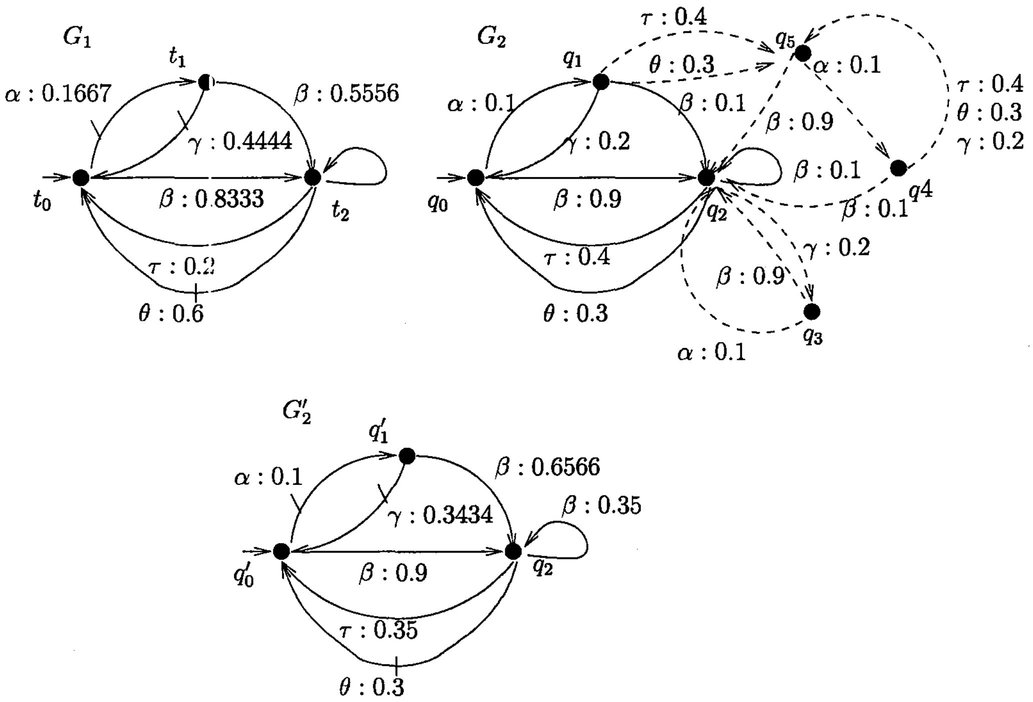

Figure 6.2: Generators $G_1$, $G_2$, and the closest approximation $G'_2$ in the revisited problem

For the example from Section 5.4, the closest approximation $G_2'$ (for 10 iterations, $e = 0.5$) is given in Figure 6.2.

## 6.4 Summary

This chapter focused on the probabilistic model fitting problem: a transformation (under certain conditions) of a probabilistic generator to another probabilistic generator of a prespecified graph. The new probabilistic generator is at the smallest possible distance from the original probabilistic generator in the metric of Chapter 4. As it turns out, the solution of the fitting problem is rather simple: the probability of a transition in the new probabilistic generator can have any value greater than or equal to the original probability of that transition. The proof of this claim consists of two parts. In the first part, the probabilistic language generated by the original probabilistic generator is exactly represented using another generator such that the new generator has a subgraph isomorphic to the prespecified graph (this is always possible under the conditions given in the formulation of the problem). In the second part, the distance between the two generators is minimized. From this point, it is easily shown that any redistribution of the probabilities of events not possible anymore over the probabilities of the events that are still possible, would result in the generator that is at the minimal distance from the original one.

Model fitting has a number of applications. A trivial state space reduction is one of them. Also, the solution to the fitting problem can serve to show that the normalized requirements specification as used in criterion (2) of the OPSCP problem is mathematically sound. Most notably, some intermediate results reached while solving the probabilistic model fitting problem are used in order to solve a modified version of the OPSCP problem. More concretely, the OPSCP problem is modified such that criterion (2) now states that the distance between the controlled plant and the original (unmodified) requirement should be minimized. It has been shown that the algorithm from Chapter 5 can be reused in a straightforward manner, without any change in complexity.

# Chapter 7

# Conclusions

The research presented in this thesis focuses on establishing a framework for reasoning about probabilistic supervisory control of probabilistic discrete event systems. The systems are modeled using probabilistic generators, a straightforward extension of generators used in standard supervisory theory. The control used is probabilistic as it allows for greater flexibility in design. The main control goal is to match the probabilistic language generated by a plant to the probabilistic language of a requirements specification. In this thesis, the solution of the problem as already existing in the literature is completed with the solution of a special case and complexity analysis. Another standard problem is solved: if there does not exist a probabilistic supervisor to match the two languages, what is the optimal solution? The problem is called the optimal probabilistic supervisory control problem (OPSCP). The nonprobabilistic language of the requirement is considered a safety requirement: the plant cannot leave the required (nonprobabilistic) language even with the smallest of probabilities. Further, the requirement of maximal permissiveness in nonprobabilistic sense is imposed. Also, the controlled plant should be deadlock-free as only nonterminating generators are considered. In the probabilistic sense, on the other hand, it is required that the controlled plant is as close as possible to the requirement. As a measure of proximity, a metric on the states of probabilistic generators is chosen.

The metric – adopted from the literature – is defined on the states of a large class of probabilistic transition systems and is given as a greatest fixed point of a monotone function. In the case of our probabilistic generators, this function can be significantly simplified. This simplification then permits the derivation of two efficient algorithm for the calculation of distances between the states of a probabilistic generator. The distance between two generators is defined as the distance between their initial states.

An algorithm to solve the OPSCP is described. First, the algorithm finds the supremal deadlock-free and controllable sublanguage of the nonprobabilistic part of the requirements specification with respect to the plant. This sublanguage represents the most permissible deadlock-free (nonprobabilistic) behaviour of the controlled plant. As the requirements specification is also constrained to the sublanguage, the underlying nonprobabilistic generators of the controlled plant and the modified requirements specification are isomorphic. Probabilities of the transitions in the generator representing the controlled plant are approximated through an iterative process: in each iteration, for each state of the controlled plant, the distance between the state and its corresponding isomorphic state of the modified requirements specification is minimized by solving a linear programming problem. The algorithm runs in time linear in the number of states of both generators representing the plant and the requirement. Depending on what method is used for solving linear programming problems as a part of algorithm, the worst-case running time of algorithm is either exponential (simplex method) or polynomial (interior point methods) in the maximal number of events possible from a state of the supremal deadlock-free and controllable sublanguage of the specification (with respect to the plant). Even the worst-case exponential complexity of the simplex method is not typically problematic for two reasons: first, the method is very efficient in practice, and second, the number of possible events from a state is often small in practical applications.

The metric used in the solution of the optimal probabilistic supervisory control problem for PDES is based on the well-researched Kantorovich metric that has been widely used for reactive systems. As mentioned, the metric is initially given a fixed point characterization. This thesis further expands the understanding of the metric by giving it an alternative, logical characterization. Logical characterization measures how close the systems satisfy formulae of a real-valued logic. Next, this logical characterization is used to reason about similarity of probabilistic strings of systems: the distance in the metric is viewed through differences of appropriately discounted probabilities of traces (strings) and sets of traces of the systems, as well as some more complicated properties of traces.

The same metric can be used in the approximation of one probabilistic generator with another one with a prespecified underlying automaton, where the distance in the metric between the original and the new model is minimized. First, the probabilistic language of the original probabilistic generator is represented using another probabilistic generator such that a subautomaton of its underlying nonprobabilistic generator is isomorphic to the prespecified automaton's. From here, it can be shown that the minimal distance between the original generator and the new one can be achieved by any distribution of probabilities of events not possible from a state in the new generator over remaining events. The approximation can be used in model order reduction, but its ideas have a more significant application in the solution of the modified OPSCP. More concretely, the requirements specification can be represented using a probabilistic generator such that a subautomaton of its underlying nonprobabilistic generator is isomorphic to the automaton representing the supremal deadlock-free and controllable sublanguage of the requirements specification. Then, the iterative part of the OPSCP algorithm can be modified in a straightforward manner so that the resulting closed loop system is optimally close to the original requirements specification.

## 7.1 Future Research

In the OPSCP algorithm, uniqueness of the closest approximation has not been researched. The problem would reduce to the uniqueness of values of decision variables in the solution of the linear programming problem of Equation (6.4). The same analysis, we believe, could give the answer to the question of whether the nonprobabilistic language generated by the closest approximation is exactly equal to the supremal deadlock-free and controllable sublanguage of requirements specification with respect to plant. Equivalently: in the solution of the linear programming problem of (6.4) can it happen that one of the variable $\rho_\sigma$ is 0?

The systems considered in this thesis are completely observable. However, it is often the case that some of the events generated by a plant are not observable by a supervisor. The work of (Desharnais et al., 2002) suggests a metric analogue to ours for a special kind of probabilistic systems (labelled concurrent Markov chain), with hidden internal events. Also, partially observed MDPs (Aström, 1965; Drake, 1962; Dynkin, 1965) have been a focus of much of research. Partially observed probabilistic transition systems modeled as probabilistic generators in the context of the infinite horizon decision problem are considered in (Chattopadhyay and Ray, 2010).

A simple application of the research to a real-world system was presented in this thesis. More applications should also be found in the field of robotics as probabilistic generators have been used to model systems in the problems of control of robot systems (Mallapragada et al., 2009; Chattopadhyay et al., 2009). Further, the use of probabilistic generators in the modeling of systems in human sequence prediction (Feldman and Hanna, 1966) might be a starting point for the introduction of control in similar systems. Also, the application in QoS (Quality of Service) should be investigated. One of the routes to explore is the use of our research in the generation of test cases (adversaries) for MDPs. More precisely, a probabilistic generator can be viewed as a supervisor for MDPs (see Section 3.2). On the other hand, a probabilistic supervisor as defined in our framework can be represented as an MDP (also shown in Section 3.2). This duality of systems to be controlled and probabilistic supervisors in the two frameworks might prove fruitful in the search for an application of the theory.

Operators on probabilistic generators remain to be defined (prefixing, choice operators, parallel composition, etc.). The desired property of non-expansiveness of operators with the respect to the metric merits further study. The property of non-expansiveness would provide for compositional reasoning about complex systems made of modules.

The nonprobabilistic behaviour of requirements specification is considered a safety requirement: a plant is not allowed to execute any trace not in the requirement. An interesting problem to solve would be relaxing this requirement such that, after a string has been observed, an event that is otherwise illegal, is allowed to occur with a small prespecified probability (similar to (Mortazavian, 1993)).

# Bibliography

Arapostathis, A., Borkar, V. S., Fernández-Gaucherand, E., Ghosh, M. K., and Marcus, S. I. (1993). Discrete-time controlled Markov processes with average cost criterion: a survey. SIAM Journal on Control and Optimization, 31(2):282–344.

Arapostathis, A., Kumar, R., and Hsu, S.-P. (2005). Control of Markov chains with safety bounds. Automation Science and Engineering, IEEE Transactions on, 2(4):333 – 343.

Arapostathis, A., Kumar, R., and Tangirala, S. (2003). Controlled Markov chains with safety upper bound. Automatic Control, IEEE Transactions on, 48(7):1230 – 1234.

Arnold, A. (1994). Finite Transition Systems. Prentice Hall.

Aström, K. J. (1965). Optimal control of Markov processes with incomplete state information. Journal of Mathematical Analysis and Applications, 10:174–205.

Baier, C., Größer, M., Leucker, M., Bollig, B., and Ciesinski, F. (2004). Controller synthesis for probabilistic systems. In Lévy, J.-J., Mayr, E. W., and Mitchell, J. C., editors, Proceedings of the IFIP International Conference on Theoretical Computer Science, pages 493–506. Kluwer.

Baier, C. and Kwiatkowska, M. (2000). Domain equations for probabilistic processes. Mathematical Structures in Computer Science, 10(6):665–717.

Barrett, G. and Lafortune, S. (1997). Using bisimulation to solve discrete event control problems. In Proceedings of the 1997 American Control Conference, pages 2337–2341, Albuquerque, NM.

Beauquier, D., Burago, D., and Slissenko, A. (1995). On the complexity of finite memory policies for Markov decision processes. In Proceedings of the

20th International Symposium on Mathematical Foundations of Computer Science, pages 191–200, London, UK. Springer-Verlag.

Bellman, R. (1957). Dynamic Programming. Princeton University Press, Princeton, NJ.

Bertsekas, D. P. (1987). Dynamic Programming and Optimal Control. Prentice-Hall, Englewood Cliffs, NJ.

Blackwell, D. (1962). Discrete dynamic programming. Annals of Mathematical Statistics, 33(2):719–726.

Borkar, V. S. (1991). Topics in controlled Markov chains. Wiley, New York.

Canny, J. (1988). Some algebraic and geometric computations in PSPACE. In Proceedings of 20th annual ACM symposium on Theory of computing, pages 460–469, New York, NY, USA. ACM.

Cassandras, C. G. (1993). Discrete Event Systems: Modeling and Performance Analysis. Richard D. Irwin, Inc., and Aksen Associates, Inc., Homewood, IL, USA.

Chattopadhyay, I., Mallapragada, G., and Ray, A. (2009). $\nu^\star$: A robot path planning algorithm based on renormalized measure of probabilistic regular languages. International Journal of Control, 82(5):849–867.

Chattopadhyay, I. and Ray, A. (2007a). Language-measure-theoretic optimal control of probabilistic finite state systems. In Proceedings of 46th IEEE Conference on Decision and Control, pages 5930–5935, New Orleans, LA, USA.

Chattopadhyay, I. and Ray, A. (2007b). Language-measure-theoretic optimal control of probabilistic finite-state systems. International Journal of Control, 80(8):1271–1290.

Chattopadhyay, I. and Ray, A. (2008). Structural transformations of probabilistic finite state machines. International Journal of Control, 81(5):820–835.

Chattopadhyay, I. and Ray, A. (2010). Optimal control of infinite horizon partially observable decision processes modelled as generators of probabilistic regular languages. International Journal of Control, 83(3):457–483.

de Alfaro, L., Henzinger, T. A., and Majumdar, R. (2003). Discounting the future in systems theory. In Baeten, J. C. M., Lenstra, J. K., Parrow, J., and Woeginger, G. J., editors, Proceedings of International Colloquium on Automata, Languages and Programming, volume 2719 of Lecture Notes in Computer Science, pages 1022–1037. Springer.

de Vink, E. P. and Rutten, J. J. M. M. (1999). Bisimulation for probabilistic transition systems: a coalgebraic approach. Theoretical Computer Science, 221(1-2):271–293.

den Hartog, J. (1998). Comparative semantics for a process language with probabilistic choice and non-determinism. Technical Report IR-445, Free University, Amsterdam, Netherlands.

Deng, Y., Chothia, T., Palamidessi, C., and Pang, J. (2006). Metrics for action-labelled quantitative transition systems. Electronic Notes in Theoretical Computer Science, 153(2):79–96.

Desharnais, J., Gupta, V., Jagadeesan, R., and Panangaden, P. (1999). Metrics for labeled Markov systems. In Baeten, J. C. M. and Mauw, S., editors, Proceedings of the 10th International Conference on Concurrency Theory, volume 1664 of Lecture Notes in Computer Science, pages 258–273. Springer.

Desharnais, J., Gupta, V., Jagadeesan, R., and Panangaden, P. (2004). Metrics for labelled Markov processes. Theoretical Computer Science, 318(3):323–354.

Desharnais, J., Jagadeesan, R., Gupta, V., and Panangaden, P. (2002). The metric analogue of weak bisimulation for probabilistic processes. In Proceedings of the 17th Annual IEEE Symposium on Logic in Computer Science, pages 413–422, Washington, DC, USA. IEEE Computer Society.

Drake, A. W. (1962). Observation of a Markov Process Through a Noisy Channel. PhD thesis, Department of Electrical Engineering, MIT, Cambridge, MA.

Dynkin, E. B. (1965). Controlled random sequences. Theory of Probability and its Applications, 10(1):1–14.

Etessami, K. and Yannakakis, M. (2009). Recursive Markov chains, stochastic grammars, and monotone systems of nonlinear equations. Journal of ACM, 56(1):1–66.

Fan-Orzechowski, X. and Feinberg, E. A. (2007). Optimality of randomized trunk reservation for a problem with multiple constraints. Probability in Engineering and Informational Sciences, 21(2):189–200.

Feinberg, E. A. and Reiman, M. I. (1994). Optimality of randomized trunk reservation. Probability in Engineering and Informational Sciences, 8(4):463–489.

Feldman, J. and Hanna, J. F. (1966). The structure of responses to a sequence of binary events. References Journal of Mathematical Psychology, 3(2):371–387.

Ferns, N., Castro, P. S., Precup, D., and Panangaden, P. (2006). Methods for computing state similarity in Markov Decision Processes. In Proceedings of the 22nd Conference on Uncertainty in Artificial intelligence, Cambridge, MA, USA,July 13-16, 2006, pages 174–181, Cambridge, MA, USA. AUAI Press.

Ferns, N., Panangaden, P., and Precup, D. (2004). Metrics for finite Markov Decision Processes. In Proceedings of the 20th Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-04), Banff, Canada, July 07-11, 2004, pages 162–169, Arlington, Virginia. AUAI Press.

Ferns, N., Panangaden, P., and Precup, D. (2005). Metrics for Markov Decision Processes with infinite state spaces. In Proceedings of the 21st Conference in Uncertainty in Artificial Intelligence, July 26-29, 2005, Edinburgh, Scotland, pages 201–208, Cambridge, MA, USA. AUAI Press.

Garg, V. (1992a). An algebraic approach to modeling probabilistic discrete event systems. In Proceedings of 31st IEEE Conference on Decision and Control, pages 2348–2353, Tucson, AZ, USA.

Garg, V. (1992b). Probabilistic languages for modeling of DEDS. In Proceedings of 26th Conference on Information Sciences and Systems, volume 1, pages 198–203, Princeton, NJ.

Garg, V., Kumar, R., and Marcus, S. (1999). Probabilistic language formalism for stochastic discrete event systems. IEEE Transactions on Automatic Control, 44(2):280–293.

Giacalone, A., Jou, C., and Smolka, S. (1990). Algebraic reasoning for probabilistic concurrent systems. In M. Broy and C.B. Jones, eds, Proceedings of the Working Conference on Programming Concepts and Methods, pages 443–458, Sea of Gallilee, Israel. North-Holland.

Gihman, I. and Skorohod, A. (1979). Controlled Stochastic Processes. Springer-Verlag, New York.

Glabbeek, R. J. V., Smolka, S. A., and Steffen, B. (1995). Reactive, generative and stratified models of probabilistic processes. Information and Computation, 121(1):59–80.

Hennessy, M. and Milner, R. (1985). Algebraic laws for nondeterminism and concurrency. Journal of the ACM, 32(1):137–161.

Hernández-Lerma, O. and Lasserre, J. B. (1996). Discrete-Time Markov Control Processes: Basic Optimality Criteria. Springer-Verlag, New York.

Howard, R. A. (1960). Dynamic Programming and Markov processes. The MIT Press, Cambridge, Massachusetts.

Hutchinson, J. E. (1981). Fractals and self-similarity. Indiana University Mathematics Journal, 30(5):713–747.

Huth, M. and Kwiatkowska, M. (1998). Comparing CTL and PCTL on labeled Markov chains. In Proc. PROCOMET'98. IFIP, Chapman & Hall.

Jou, C.-C. and Smolka, S. A. (1990). Equivalences, congruences, and complete axiomatizations for probabilistic processes. In Baeten, J. C. M. and Klop, J. W., editors, Proceedings of International Conference on Concurrency Theory, volume 458 of Lecture Notes in Computer Science, pages 367–383. Springer.

Kalai, E. and Solan, E. (2003). Randomization and simplification in dynamic decision-making. Journal of Economic Theory, 111(2):251–264.

Kantorovich, L. (1942). On the transfer of masses (in Russian). Doklady Akademii Nauk, 37(2):227–229. Translated in Management Science, 5:(1-4), 1959.

Kozen, D. (1985). A probabilistic PDL. Journal of Computer and System Sciences, 30(2):162–178.

Kreinovich, V., Lakeyev, A., Rohn, J., and Kahl, P. (1998). Computational complexity and feasibility of data processing and interval computations. Kluwer Academic Publishers, Dordrecht, The Netherlands.

Kumar, R. and Garg, V. (1998a). Control of stochastic discrete event systems:

Existence. In Proceedings of 1998 International Workshop on Discrete Event Systems, pages 24–29, Cagliari, Italy.

Kumar, R. and Garg, V. (1998b). Control of stochastic discrete event systems: Synthesis. In Proceedings of 37th IEEE Conference on Decision and Control, pages 593–606, Tampa, FL.

Kumar, R. and Garg, V. (2001). Control of stochastic discrete event systems modeled by probabilistic languages. IEEE Transactions on Automatic Control, 46(4):593–606.

Kučera, A. and Stražovský, O. (2008). On the controller synthesis for finite-state Markov decision processes. Fundamenta Informaticae, 82(1-2):141–153.

Kwiatkowska, M., Norman, G., and Parker, D. (2007). Stochastic model checking. In Bernardo, M. and Hillston, J., editors, Proceedings of Formal Methods for the Design of Computer, Communication and Software Systems: Performance Evaluation, volume 4486 of Lecture Notes in Computer Science (Tutorial Volume), pages 220–270. Springer.

Larsen, K. G. and Skou, A. (1991). Bisimulation through probabilistic testing. Information and Computation, 94(1):1–28.

Lawford, M. and Wonham, W. (1993). Supervisory control of probabilistic discrete event systems. In Proceedings of the 36th IEEE Midwest Symposium on Circuits and Systems, volume 1, pages 327–331. IEEE.

Lee, E. and Zadeh, L. (1969). Note on fuzzy languages. Information Sciences, 1(4):421–434.

Li, Y., Lin, F., and Lin, Z. H. (1998). Supervisory control of probabilistic discrete event systems with recovery. IEEE Transactions on Automatic Control, 44(10):1971–1975.

Mallapragada, G., Chattopadhyay, I., and Ray, A. (2009). Autonomous robot navigation using optimal control of probabilistic regular languages. International Journal of Control, 82(1):13–26.

Molloy, M. (1982). Performance analysis using stochastic Petri Nets. IEEE Transactions on Computers, 31(9):913–917.

Mortazavian, H. (1993). Controlled stochastic languages. In Proceedings of 31st Annual Allerton Conference on Communications, Control, and

Computing, pages 938–947, Urbana, Illinois.

Norman, G. J. (1997). Metric semantics for reactive probabilistic processes. PhD thesis, University of Birmingham.

Ortega, J. M. and Rheinboldt, W. C. (1970). Iterative solution of nonlinear equations in several variables. Academic Press, Inc., New York, New York, USA.

Pantelic, V. and Lawford, M. (2009). Towards optimal supervisory control of probabilistic discrete event systems. In Proceedings of 2nd IFAC Workshop on Dependable Control of Discrete Systems, pages 85–90, Bari, Italy.

Pantelic, V. and Lawford, M. (2010a). Optimal supervisory control of probabilistic discrete event systems. IEEE Transactions on Automatic Control. Accepted subject to revision, December 2010.

Pantelic, V. and Lawford, M. (2010b). Use of a metric in supervisory control of probabilistic discrete event systems. In Proceedings of the 10th International Workshop on Discrete Event Systems, pages 227–232, Berlin, Germany. Selected for possible publication in Discrete Event Dynamic Systems, Special issue on WODES'10.

Pantelic, V. and Lawford, M. (2011). Pseudometric in supervisory control of probabilistic discrete event systems. Submitted March 11th 2011 to Discrete Event Dynamic Systems, Special issue on WODES 2010.

Pantelic, V., Postma, S., and Lawford, M. (2008). Supervisory control of probabilistic discrete event systems. Technical Report 21, Software Quality Research Lab, McMaster University, Hamilton, ON, Canada.

Pantelic, V., Postma, S., and Lawford, M. (2009). Probabilistic supervisory control of probabilistic discrete event systems. IEEE Transactions on Automatic Control, 54(8):2013–2018.

Postma, S. and Lawford, M. (2004). Computation of probabilistic supervisory controllers for model matching. In Venu Veeravalli and Geir Dullerud, editors, Proceedings of Allerton Conference on Communications, Control, and Computing, Monticello, Illinois.

Rabin, M. O. (1963). Probabilistic automata. Information and Control, 6(3):230–245.

Ramadge, P. and Wonham, W. (1987). On the supremal controllable sublanguage of a given language. SIAM Journal on Control and Optimization, 25(3):637–659.

Renegar, J. (1992). On the computational complexity and geometry of the first-order theory of the reals, part I-III. Journal of Symbolic Computation, 13(3):255–352.

Rosenberg, D., Solan, E., and Vieille, N. (2000). Blackwell optimality in Markov decision processes with partial observation. Discussion Papers 1292, Northwestern University, Center for Mathematical Studies in Economics and Management Science.

Rutten, J., Kwiatkowska, M., Norman, G., and Parker, D. (2004). Mathematical Techniques for Analyzing Concurrent and Probabilistic Systems, P. Panangaden and F. van Breugel (eds.), volume 23 of CRM Monograph Series. American Mathematical Society.

Salomaa, A. (1990). Formal language and power series. In Handbook of Theoretical Computer Science, Volume B: Formal Models and Sematics (B), pages 103–132.

Schröder, L. and Mateus, P. (2002). Universal aspects of probabilistic automata. Mathematical Structures in Computer Science, 12(4):481–512.

van Breugel, F., Hermida, C., Makkai, M., and Worrell, J. (2005). An accessible approach to behavioural pseudometrics. In Caires, L., Italiano, G., Monteiro, L., Palamidessi, C., and Yung, M., editors, Automata, Languages and Programming, volume 3580 of Lecture Notes in Computer Science, pages 1018–1030. Springer Berlin / Heidelberg.

van Breugel, F., Sharma, B., and Worrell, J. (2008). Approximating a behavioural pseudometric without discount for probabilistic systems. Logical Methods in Computer Science, 4(2:2):1–23.

van Breugel, F. and Worrell, J. (2001a). An algorithm for quantitative verification of probabilistic transition systems. In Larsen, K. G. and Nielsen, M., editors, Proceedings of International Conference on Concurrency Theory, volume 2154 of Lecture Notes in Computer Science, pages 336–350. Springer.

van Breugel, F. and Worrell, J. (2001b). Towards quantitative verification of probabilistic transition systems. In Orejas, F., Spirakis, P. G., and van

Leeuwen, J., editors, International Colloquium on Automata, Languages and Programming, volume 2076 of Lecture Notes in Computer Science, pages 421–432. Springer.

van Breugel, F. and Worrell, J. (2005). A behavioural pseudometric for probabilistic transition systems. Theoretical Computer Science, 331(1):115–142.

van Breugel, F. and Worrell, J. (2006). Approximating and computing behavioural distances in probabilistic transition systems. Theoretical Computer Science, 360(1-3):373–385.

Wasserstein, L. (1969). Markov processes over denumerable products of spaces describing large systems of automata. Problems in Information Transmission, 5(3):47–52.